Bootstrapping Vehicles: a Formal Approach to Unsupervised Sensorimotor Learning Based on Invariance

Thesis by

Andrea Censi

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2013

(Defended June 27th, 2012)

© 2013

Andrea Censi

All Rights Reserved

Dedication

I wish to dedicate this work to my parents, Rita and Pino, as a partial repayment of their sizable investment in Lego pieces 30 years ago, which ultimately led to my interest in robotics; and to my brother Marco, who bore the burden of being my first office mate. I am deeply indebted to my parents as they taught me to ask questions, and to question answers. They are a model of moral integrity and hushed selflessness that I hope I will have the strength to emulate.

Acknowledgments

These years at Caltech have been an exciting roller-coaster, which began slowly, went through frenetic ups and downs, and ended much too quickly. I wish I could buy a ticket for another ride.

I will always be indebted to my advisor Richard Murray, who was the main reason for this to be a relatively pleasant and fulfilling experience. He has been the last of a long series of people whom I was lucky to meet from high school to graduate school, who taught me the value of intellectual honesty, how to respect myself and others, and, of equal importance, how to get things done, without compromise. They encouraged me to explore the road less traveled, they guided me through the clouded waters of failure and uncertainty, and helped me enjoy the eventual successes. Their advice has been heard and preciously guarded, even when I did not follow it. I will always feel their eyes on me, and, as I walk alone, their memory will keep my chin up and my path straight. My repayment to them is the commitment to put the same effort and dedication towards guiding the people who will come after me.

During my years at Caltech, I had the privilege of working with Michael Dickinson and Andrew Straw, who made me realize that, after all, biologists might be smarter than engineers, and with Stefano Soatto, whom I thank for his frank advice and honest critiques. I thank Joel Burdick and Yaser Abu-Mostafa for their advice and for serving on my thesis committee. I thank my geographically uniformly distributed co-authors Shuo Han, Sawyer Fuller, Davide Scaramuzza, Paloma De La Puente, Stefano Carpin, Antonio Franchi, Luca Marchionni, Giuseppe Oriolo, and Luca Carlone, for dealing with my (now long gone) perfectionism and the occasional idea which was, well, just a bit far-fetched. I thank my summer students Magnus Håkansson and Adam Nilsson for contributing data and code used in the experiments. I thank Benjamin Kuipers and Daniel Naftalovich for

insightful feedback on this dissertation and on my previous works. I also wish to thank the many people at Caltech who made day-to-day life in the bubble enjoyable, or, at least, bearable, especially those who routinely indulged my inclination for late-night conversations.

I thank the fellow roboticists who made their data and software available to the community, in particular the people who contributed to Radish, OpenSLAM, Rawseeds, and ROS. Finally, I wish to thank the people who created the free/open source software on which my work so heavily relies (the GNU project, Linux, Debian, LaTeX, LyX, Python, Scipy, PyTables, and many others): they are the unsung heroes of science.

Abstract

Could a "brain in a jar" be able to control an unknown robotic body to which it is connected, and use it to achieve useful tasks, without any prior assumptions on the body's sensors and actuators? Other than of purely intellectual interest, this question is relevant to the medium-term challenges of robotics: as the complexity of robotics applications grows, automated learning techniques might reduce design effort and increase the robustness and reliability of the solutions. In this work, the problem of "bootstrapping" is studied in the context of the Vehicles universe, which is an idealization of simple mobile robots, after the work of Braitenberg. The first thread of results consists in analyzing such simple sensorimotor cascades and proposing models of varying complexity that can be learned from data. The second thread regards how to properly formalize the notions of "absence of assumptions", as a particular form of invariance that the bootstrapping agent must satisfy, and proposes some invariance-based design techniques.

Contents

CHAPTER 1

Introduction

We are late! At least, according to Asimov. His novels popularized the idea of robots as useful mechanical servants for mankind. As a scientist himself, writing in the fifties, he set the scene quite far away in time. In 2015, a young Susan Calvin, is a promising graduate student who is already interacting with robots with cognitive capabilities. Susan completes her degree and eventually becomes the head robopsychologist for the US Robots & Mechanical Men Inc., which, at the time, was already producing humanoid robots for general use [1].

It is now 2012. Information and communication technology flourished: it is many decades that the word "computer" does not refer to a patient human being [2]. But "intelligent" robotics failed the expectations. (There exists a US Robotics company, but it produces network appliances.) Robots are widespread in industry, but they are relatively simple artifacts with no presumption of intelligence, used to substitute for human workers in rote tasks [3]. Artificial intelligence produced a wealth of results in the last decades,

---

Isaac Asimov: (1919–1992) Russian–American biochemist and prolific author of science-fiction stories. He invented the word "robotics".

Susan Calvin: (1982–2064) Head robopsychologist for the US Robots & Mechanical Men incorporated. One of the writer's favorite fictional characters.

robopsychology: In Asimov's novels, robopsychology is the fictional study of the personalities of intelligent machines.

artificial intelligence: The set of problems that humans can solve, but don't know *how*.

but failed in creating an embodied intelligence [4]. Today, robots are not smart enough to need a robopsychologist, while roboticists do need therapists for handling the frustration of working with robots.

We do have reasons to be optimistic: after decades of a steady, but slow progress, we have witnessed very rapid advancements, and the first commercial deployments of "intelligent" robotics, both in industry and in consumer products. This chapter looks at the recent history of robotics, at the obstacles that have been overcome, and at those that remain on our path.

1.1. The State of Robotics

Robotic manipulators have been used in the manufacturing and packaging industry for several decades. In addition to improving the safety of workers, robotic solutions often improve the precision, manufacturing time, and cost [3]. The largest robot manufacturers for industrial applications are in Europe and Japan. Robotic technology has found application also in nontraditional fields, such as robot-assisted surgery, surveillance, search and rescue, prosthetics, and artificial affective companions.

The more visible advances of the the last decade were in the area of autonomous mobile robotics. A scientific success for robotics were the Mars rovers (*Spirit* and *Opportunity*), which explored the surface of Mars for more than 10 cumulative years. Autonomous operation was necessary due to the 3–22 minutes long round-trip communication delay.

It was easier for robots to land on Mars than to arrive in consumers houses. *Roomba* [5], produced by iRobot, is a blind robot that navigates using random movements (more than 5 million have been sold to date). Evolution Robotics' *Mint* [6] performs localization and mapping aided by external infrared markers. While these robots use relatively simple technology compared to the state of the art, they are historically significant because they have found a combination of price point/features/ease of use that made them the first

practical robots to be used in households.

Robotics also evolved in industry beyond manufacturing and packaging. In Australia, a whole waterfront operations system operates autonomously by loading and unloading cargo with no human intervention [7], and autonomous mines are currently being developed [8].

Robotics revolutionized large stockroom operations. In the robotic warehouse developed by Kiva Systems [9], employees wait patiently at their stations, while a swarm of robots brings them the shelves from which they pick up products. This allows to cut costs and makes it possible to have a more diversified inventory. Amazon acquired the company in March 2012 for $775M [10].

Military applications of robotics are pioneered by large American contractors. "Drones", such as the General Atomics MQ-9 *Reaper*, are technically robots, but their genesis is in the guidance & control field of aeronautics, rather than in the robotics/automation community. Drones are currently used in autonomous mode for surveillance, and in semi-autonomous mode for target elimination. Drones are being introduced in several American states for civilian use by police forces. As of today, there are (officially) no robots implementing lethal behavior on the basis of autonomous decisions, but there have been studies about what deliberation system would support such a function [11]. In any case, manuals for surviving the coming robot uprising are readily available [12].

Autonomous cars are a promising application of robotics, which was given large publicity by the 2004-2007 DARPA Grand Challenges. The technology has matured rapidly. In 2010, a squadron of autonomous cars drove from Italy to China in a 15,926 km long

semi-autonomous: A human operator is constantly supervising the robot and manually activates certain functions.

trip [13]. Outside of the academic environment, Google's autonomous cars have reportedly logged many thousands of miles of autonomous driving. In 2011, Nevada introduced laws to regulate testing of autonomous cars [14].

Much of this innovation was enabled by the improvements of exteroception capabilities. Until about a decade ago, mobile robots could rely only on imprecise and unreliable exteroceptive sensors [15]. The introduction of more reliable sensors, such as laser range finders, allowed to develop robust solutions for autonomous mapping and navigation in large environments; this went hand in hand with the development of probabilistic techniques for handling perception data [16]. It is thought that affordable three-dimensional range finders such as the Kinect will provide similar gains for the area of manipulation. Perception is still thought to be an issue for certain applications. For example, very agile micro air vehicles have been demonstrated, but they are "blind" and must guided by external localization signals such as the Vicon system.

Standardization of software infrastructure was considered a large impediment to the progress of the field, and it is now being solved by pooling the efforts into few large open source projects. Two such projects are Orocos, oriented towards low-level control, and ROS , developed by Willow Garage, oriented towards higher-level applications. They provide a middleware for distributed applications, a basic library of data types primitive, and common components for robotics.

Kinect: An RGBD sensor introduced by Microsoft in 2010.

ROS: (Robot Operating System) A middleware oriented to robotics applications together with many modules implementing basic robotic tasks.

1.2. What Makes Robotics Difficult

At the current state of the art, it is clear that we are able to create much better robotic bodies than robotic minds. The performance of robots is not limited by the mechanical body's physical limitations, or their sensor capabilities, but rather by our incapability of writing suitable control software. This can be seen in rescue robotics, where an operator can use the robot sensors and actuators to achieve tasks that cannot be done automatically.

A robot that moves and manipulates cannot just execute a fixed set of movements, like industrial manipulators. Rather, it must use its exteroceptive sensors to acquire observations from the environment (e.g., range finder data), and process the data to infer a model of the environment (e.g., a map of the environment), and then use such model to plan its actions in the future.

A favorite trope of pop-scientists is to blame the lack of progress on limited computational resources*, but computation does not seem to be the main issue. Technologies like GPU s allow lots of computation power on relatively small systems. A recent trend in computing is to offload computation to cloud components; an example of a deployed system is Apple's Siri, which uploads the user's speech to central servers for processing. On the other hand, it is true that computation is a limiting factor for applications such as micro air vehicles.

*This is often followed by a summary of the numbers of synapses of the brain, an extrapolated plot of Moore's law, and a warning on the impending enslavement of humanity by post singularity machines.

GPU: (Graphics Processing Units) Specialized processors originally developed for vectorized processing 3D rendering. Their design makes them useful for a wide range of tasks that can be efficiently parallelized.

SIRI: (Speech Interpretation and Recognition Interface) A voice-activated personal assistant software included in Apple cell phones. To decrease the load on the device CPU, the speech is uploaded on a remote server for processing.

In robotics, *the easy problems are hard and the hard problems are easy* [17]. One must deal with the uncertainty of the observations, unstructured environments that cannot be fully modeled, and tasks that would be difficult enough even in the absence of any uncertainty. Roboticists must invest tremendous design effort to have robots do tasks that appear trivial to the eye of the non-expert. Robotics application are typically fragile with respect to deviations of the world from the designer assumption.[†]

There is no consensus either on the ultimate causes of these problems [18]. One possible explanation is that robots are fragile because the development of robotic applications follows a *constructive*, rather than *deductive,* methodology. One does not start from a formalization of the problem to deduce a solution through a formal synthesis procedure; rather, the solution is assembled from pieces, and it is validated empirically. A slight change of the operating conditions can confuse the robot in ways that the designers did not anticipate.

The problem of fragility goes hand in hand with the problem of complexity: more complex solutions are inherently more fragile. What makes robotics a unique discipline is that robots need to have incredibly rich models for the world. It is likely that robotic applications will be among the most complex engineered systems: if it takes hundreds of thousands of lines of code to write the simplest of word processors, how many are needed to create a robotic butler?

Robotic software development is hard because robotics does not have the "compositional" properties of other disciplines, which allows creating a more complex object from

[†]The Three Laws of Roboticists:

(1) Never give a robotic demo.
(2) Follow your adviser's advice, except when it contrasts with the first Law.
(3) When the demo fails, blame it on the batteries.

simple parts, using a divide-and-conquer approach. In mathematics, one can prove lemmas, and by those lemmas prove a larger theorem. Being used as part of a theorem does not put any stress on the lemmas. In electronics, components do interact physically among each other, but careful design allows ignoring analog interactions and only consider digital behaviors. Unfortunately, robotics does not work that way. The building blocks of robotics should be sense-plan-act loops implementing a certain task, but putting two in parallel does not guarantee that the resulting behavior is coherent.

The recent coagulation of software infrastructure efforts removed some of the problems and allowed to design shared components. However, the shared interface is relatively low level (images, poses, etc.); there is no shared "semantic" structure. Consequently, for each project most of the effort goes towards writing fragile ad hoc code, whose complexity grows nonlinearly with the complexity of the applications.

1.3. Learning and Adaptivity

As robotics applications become more complicated and robot inhabit unpredictable and unstructured environments, it is reasonable to predict that the scarce resource for robotics in the future will be *design effort*. Dijkstra liked to say, "a programmer should be well aware of the limited size of their skull" [19]. It will not be possible to predict everything. Learning and adaptivity will play an ever growing role in robotics because they allow reducing design effort.

1.3.1. *Approaches to learning and adaptivity*

Let us broadly consider "learning" any design methodology such that a relevant part of the final design depends on the interaction of the system with the world, rather than on

Edsger W. Dijkstra: (1930–2002) Dutch computer scientist who advocated for a disciplined approach to computer programming.

the decisions of the designer.

Across different fields, there are many approaches to learning, often inspired by nature. The first fundamental classification is between phylogenetic learning, which is evolution across generations, and ontogenic learning, which refers to learning from the individual experience.

Genetic algorithms mimic aspects of evolution and natural selection. A genetic algorithm is "blind": the only feedback from the world is the fitness of the individuals. For this reason, genetic algorithms are studied independently of the domain, and are considered by most akin to a generic optimization scheme. Such algorithms have been demonstrated in robotics [20, 21].

It has long been shown that the changes over time of the culture of a social population of agents can be regarded as the evolution of a population of *memes* [22]. Memetic evolution has been demonstrated in robotics for distributed optimization problems (e.g., [23]).

System identification [24, 25] and *adaptive control* [26] are subfields of control theory that are concerned with designing controllers that can adapt to a plant that is partially unknown. System identification is limited to only the inference problem, while adaptive control focuses on the closed-loop performance of a controller based on the identified model. These techniques have been industrially deployed since the 1980s with the introduction of digital control, and are the roots for deployed parameter identification and self-calibration techniques in robotics. Control theory gives strong results, but for limited classes of systems. Recent approaches to system identification from computer science (e.g., [27]) have mainly an empirical validation but describe much more expressive classes of models.

Supervised learning is the problem of making educated guesses about data that has not

been seen yet, based on the data that has been observed.[‡] Typical questions of the discipline are how much data is needed to obtain guarantees on the inference, to what degree it is possible to generalize over data that has not been observed, and understanding how a particular choice of prior influences the final results. *Reinforcement learning* is a particular form to supervised learning in an "interactive" setting in which the agent interacts with world [31]. Many works in robotics employ supervised learning (e.g., [32]).

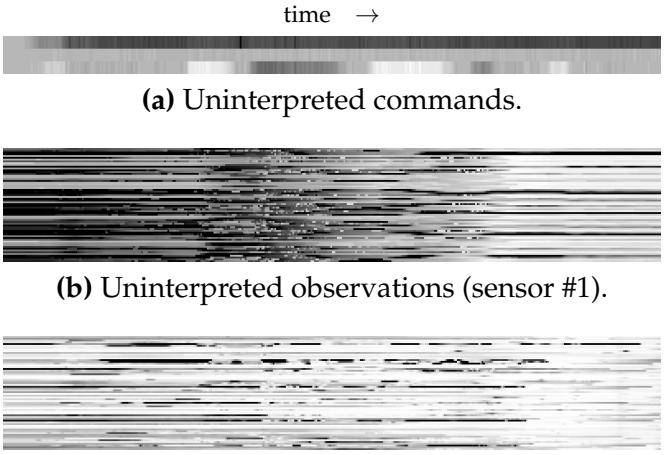
Unsupervised learning is a much more loosely defined set of problems which are concerned with understanding the relations in the data, rather than fitting a particular input/output relation [33, 34]. *Neural networks* were intensively studied as black-box approximators until the 1990s, but more recently, so called "deep" networks have shown surprising properties in finding interesting features from pixel-level data [35, 36].

Developmental robotics [37, 38] is concerned with mimicking the successive stages of learning observed in humans and other primates as studied in developmental psychology, such as in the work of Piaget. The goal is either to obtain better robots, or simply to use robots as a test bench to validate the formalization of a biological theory (e.g., [39]).

1.3.2. *Bootstrapping: starting from scratch*

A rather extreme, yet concrete, version of the learning problem has been put forward by Pierce and Kuipers [40] in the first of a long series of papers. In the "bootstrapping" scenario, an agent starts its life with no prior information about its sensors and actuators. It can read the sensors output as a sequence of values, but no semantics is associated to them (figure 1.1). Likewise, the agent does not know how its commands affect the world. The bootstrapping problem concerns creating models for its sensorimotor cascade from scratch, and using it to achieve useful tasks.

[‡]Vapnik [28] is the manifesto for statistical learning theory. Hastie, Tibshirani, Friedman [29] is a modern reference. For a tutorial introduction, see Poggio and Smale [30].
.



(a) Uninterpreted commands.

(b) Uninterpreted observations (sensor #1).

(c) Uninterpreted observations (sensor #2).

Figure 1.1. A bootstrapping agent must learn to use uninterpreted streams of observations and commands. This figure shows how such streams appear for an actual robotic platform (a planar, differential-drive robot). The commands are the linear and angular velocities. The two observations streams corresponds to 64 randomly sampled *sensels* of a camera and a range-finder, during one minute of operation. Can the reader guess which sensor is which?

Bootstrapping would reproduce two features of natural intelligence that we are far from emulating in artificial systems: adaptiveness and generality. The human neocortex is highly uniform and its parts can be repurposed; for example, the visual cortex is repurposed to process tactile information in blind subjects [41, 42]. The bootstrapping problem also gives a concrete context for arguing over fuzzier topics, such as the nature of consciousness [43], in contrast to more vacuous approaches (e.g., [44]).

Bootstrapping can be seen as an extreme form of system identification/calibration. Currently, there exist autocalibration techniques that can estimate parametric models of the dynamics (for example, the odometric parameters) or the extrinsic sensor configuration, (although the solutions, rather than general, tend to be tailored to specific sensors or groups of sensors), but always the type of sensors/actuators being calibrated is known a priori. Can a robot learn to use an unknown sensor and unknown actuators? Can the *same* learning algorithm work for a range-finder and a camera? What can the agent learn on its own? What needs to be encoded a priori?

These are, at the moment, open questions. They are important for practical robotics applications: it would be extremely convenient if we could just attach any sensor to a robot and the robot would learn how to use it, without any tedious programming. More in general, robotic systems are the perfect benchmark for supposedly "universal learning agents," which so far have been studied for only perception/classification tasks [45], or as bodyless agents [46].

1.4. Approach

Current applications of learning to robotics are more an art than a science: the approach is usually algorithmic and relies on empiric evaluation. One of the strong points of control theory is that the theoretical results can be converted to "recipes" that can be applied (sometimes automatically by a software package) to obtain a valid controller, without the final user needing to understand much of the underlying theory. In the same spirit, the vision is that we should be able to understand, for each learning agent, exactly what subset of what robots it can work with, and what tasks it can accomplish.

The main methodological difference is using a "problem-based" rather than an "algorithm-based" approach. In machine learning research, one typically elaborates a new method/model, and then finds applications that validate the method/model. In control theory, one first defines a class of systems, and then finds a method that can work with that class of system. For bootstrapping, the philosophy is the same as control theory, but there are a few special things to note. Let us be clear: bootstrapping for a general system is either impossible or very impractical. The "set of all robots", is much larger than what is manageable using a control theory approach. Bootstrapping is all about being careful about the assumptions of an agent about the system. Characterizing the agents "assumptions" is the leitmotif of this work.

1.4.1. *Embodied approach*

This work formulates some general principles for learning agents, but insofar as the discussion gets specific, the focus is for agents embodied in robotic bodies.

To this end, we shall use the *Vehicles universe* as an idealization of mobile robotics. This is inspired by the work of Valentino Braitenberg. In a famous book [47] he describes these simple machines, now called *Braitenberg Vehicles*, as idealization of biological creatures. The basic message is that complex behaviors can arise from simple controllers.

Braitenberg vehicles are slightly updated. The Vehicles universe is composed by all possible combinations of a set of "canonical" mobile robot sensors and dynamics (Figure 1.1). The canonical dynamics considered are the usual ones (differential drive, car-like, etc.). The canonical sensors considered are range-finders, cameras, field-samplers, and other formally equivalent sensors.

The combinations of these sensors and dynamics represent most phenomena of interest for mobile robots, but it does not capture phenomena related to manipulation (proprioception, touch, articulated bodies, etc.). This is part of future work. After all, animals first evolved to move, and then eventually to manipulate the environment.

1.4.2. *Tasks for bootstrapping agents*

It is important to study agents in the context of a specific goal, because it provides a falsifiable context. There are several approaches to encode the goal of an agent. In reinforcement learning, the agent receives a *reward* signal from the world that should be maximized. In developmental robotics, *intrinsic motivation* (e.g., [48–50]) describes heuristics that can guide the exploration phase of an agent.

Here, the focus is on defining *explicit tasks*. "Information tasks" (such as predicting the next observations) allow for checking that the agent has acquired a generative model for

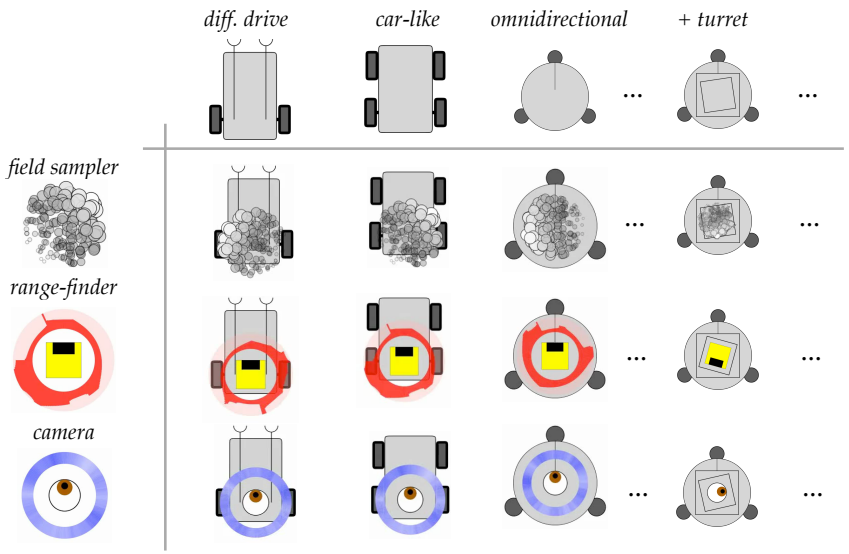


Figure 1.1. The *Vehicles* universe.

the data. "Spatial tasks" (such as *servoing*) allow for checking that the agent is powerful enough to solve what an animal must do. The idea is that a hierarchy of such tasks is better suited to understand the capabilities of an agent, than minimizing an opaque reward.

In any case, these three approaches (extrinsic rewards, intrinsic motivation, explicit tasks) are not mutually exclusive. In an ideal architecture, intrinsic motivation is what guides the agent's exploration toward acquiring models/skills for solving explicit tasks, which can be used toward a goal specified by extrinsic rewards.

The interaction between agent and world follows a two-stage approach (Figure 1.2), in which the learning (exploration) and the action (exploitation) phases are separate. An "agent" is defined by two objects: an exploration strategy ("expl") that interacts with the world with the goal of learning a "model" ("$\boldsymbol{m}$"), and a strategy ("act") that on the basis of the model instantiates the behavior that interacts with the world during the acting stage. This two-stage approach does not allow to explore the trade-offs of exploration *vs* exploitation, but simplifies some aspects of the analysis.

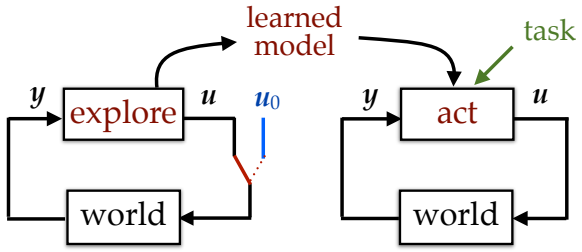


Figure 1.2. In the learning stage, the agent interacts with the world with the goal of estimating a "model." In the acting phase, the agent instantiates a behavior which is a function of the model.

1.5. Summary of Main Results

1.5.1. *The statistics of the observations allow accurate reconstruction of the sensor geometry*

Previous work has shown that the statistics of the sensel values allow to obtain information about the sensor geometry, based on the assumption that the signals of two sensels is more similar if the sensels are closer to each other. So far, it has been assumed that it is possible to reconstruct accurately only the topological information (which sensel is close to which) but not the metric information, such as the field of view of a camera. A careful analysis shows that the metric information is actually observable (Chapter 10). An engineering application is the calibration of camera sensors with arbitrary optics.

1.5.2. *Canonical robotic sensors have relatively similar dynamics at the sensel level*

While it is common for robotic applications to work with heterogeneous sensors, the actual "sensor fusion" happens after the data from each sensor has already been processed by sensor-specific routines, the assumptions being that separate sensors need separate treatment.

We will see that, at the sensel level, the dynamics of the canonical robot sensors considered (range-finder, camera, field sampler) are relatively similar. Table 1.1 shows the ordinary differential equations satisfied by the observations y as a function of the kinematic velocities ω and v. The observations y are the pixel luminance in a camera, the readings of

a range-finder, the intensity sampled by the field-sampler. The field-sampler dynamics is purely bilinear in the observations and the velocities; the range-finder dynamics is bilinear up to a nonlinearity; the camera dynamics is bilinear up to a hidden state (the nearness μ).

Table 1.1. Sensel-level dynamics of canonical robot sensors

	$\mathcal{S}$	*meaning of* $y(s)$	*dynamics*
field sampler	$\mathbb{R}^3$	intensity of a field	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \nabla_i y^s \boldsymbol{v}^i$
camera	$\mathbb{S}^2$	luminance of a sensel	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \mu^s \nabla_i y^s \boldsymbol{v}^i$
range-finder	$\mathbb{S}^2$	distance readings	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + (\nabla_i \log y^s - s_i^*) \boldsymbol{v}^i$

y^s is the value returned by the sensel s (field intensity, luminance, range readings). The time variable is implicit. These formulas are valid far from occlusions. Near occlusions, the dynamics is discontinuous and cannot be represented by an ODE.

1.5.3. *Simple models approximate the sensors dynamics*

The fact that disparate sensors have similar dynamics allows for designing agents that can work without any prior knowledge of the sensors. Several generative models that can approximate the sensor models will be described (Table 1.2).

- The BDS model (Chapter 11) assumes that the observations are vectors of real numbers, and there is a bilinear relation between $\dot{\boldsymbol{y}}$, $\boldsymbol{y}$, and the commands $\boldsymbol{u}$.
- The BGDS model (Chapter 12) assumes that the observations are a function on a manifold $\mathcal{S}$. The dynamics is a bilinear flow on $\mathcal{S}$ that depends explicitly on the gradient of $\boldsymbol{y}$.
- The DDS model (Chapter 13) models the dynamics as diffeomorphisms of the observations space.

---

differentiable manifold: A rigorous generalization of the intuitive concept of "surface". See Definition E.16.

diffeomorphism: An invertible and differentiable transformation between two manifolds whose inverse is differentiable. See Definition E.20.

These models do not exactly represent the real sensor dynamics, but they are sufficient to perform simple tasks, such as servoing.

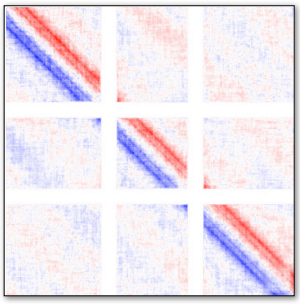
Table 1.2. Three models for low-level sensorimotor learning

	model	*assumption on format*	*dynamics*
BDS	bilinear dynamics	$\boldsymbol{y} \in \mathbb{R}^{n_y}$	$\dot{y}^s = \sum_{i,v} \mathsf{M}^s_{vi} y^v u^i$
BGDS	bilinear flows	$\boldsymbol{y} : \mathcal{S} \to \mathbb{R}$	$\dot{y}(s) = \sum_i (\mathsf{G}^d_i(s) \nabla_d y(s) + \mathsf{B}_i(s)) u^i$
DDS	diffeomorphisms	$\boldsymbol{y} : \mathcal{S} \to \mathbb{R}$	$y_{k+1}(s) = y_k(\varphi_j(s))$

1.5.4. *These models are reasonably robust to be applied to raw sensor data*

These models can be used to fit real data coming from a robotic platform with zero or minimal preprocessing. One reason is that the learning procedure, based on streaming data, is a form of Hebbian learning that is robust to occasional noise and slight deviations from the idealized model. At the same time, these models are powerful enough to discriminate several kinds of low-level sensorimotor faults (Figure 1.1).

(a) Effect of populated environments.



(b) Effect of occlusions.

Figure 1.1. The models we use are quite robust to deviations from nominal models. The pictures show a graphical representation for a BDS model corresponding to the rotation command (see Chapter 11). On the left, the tensor learned for a range finder in a populated environment. On the right, the tensor learned for a robot that has some fixed occlusions. The white strips correspond to the occlusions due to the fixtures.

1.5.5. *Describing an agent's assumptions*

The main conceptual contribution of this work is to classify the assumptions of the agents in three classes:

(1) The assumptions about the "format" of the data;

(2) The assumptions about the "physical system";

(3) The assumptions about the "representation" of the data.

The assumptions about the "format" of the data are the simplest to describe. If the agent must interact with the world, there needs to be a common interface between the two that specifies how observations and commands are encoded. Call $\mathcal{Y}$ the set to which the observations belong. Some possible choices of $\mathcal{Y}$ are

$\mathcal{Y} = \{\square, \boxdot\}$ The observations consists of one uninterpreted bit; here $\square, \boxdot$ represent the two possible states of the bit.

$\mathcal{Y} = \mathbb{R}$ The observations consist of one real number.

$\mathcal{Y} = \mathsf{Differentiable}(\mathbb{R}; \mathcal{S})$ The observations are a differentiable function from a manifold $\mathcal{S}$ to $\mathbb{R}$.

Fix a format for the observations (the *observation space* $\mathcal{Y}$) and for the commands (the commands space $\mathcal{U}$), call $\mathcal{D}(\mathcal{Y}; \mathcal{U})$ the set of all such systems.

The assumptions about the *representation* are different than the assumptions about the format. Pick a system $\boldsymbol{D} \in \mathcal{D}(\mathcal{Y}; \mathcal{U})$. Consider an invertible function $f : \mathcal{Y} \to \mathcal{Y}$ and let $f \cdot \boldsymbol{D}$ denote the system obtained by filtering the observations of $\boldsymbol{D}$ with the function f. This transformation does not lose any information, because one can always apply the inverse transformation f^{-1} to obtain back $\boldsymbol{D}$. The systems $f \cdot \boldsymbol{D}$ and $\boldsymbol{D}$ are effectively the same system with a different representation. The same construction can be done for the commands. As a further generalization, one can consider, instead of an instantaneous function, any causally invertible system. The set of all causally invertible systems is called the set of

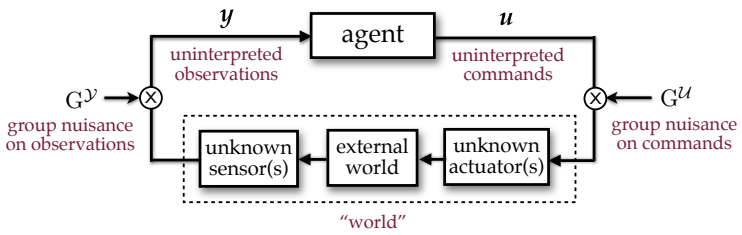


Figure 1.2. *Representations nuisances* are fixed, invertible transformations of the observations and commands, which affect the "representation" of the system while preserving all intrinsic properties (observability, controllability). Because the nuisances are invertible, an agent should be able to compensate for them, so that the dynamics of the world remains the same. The set of nuisances to which an agent is invariant can be understood in terms of the semantic assumptions on the data.

representation nuisances and it is denoted $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$.

These representation nuisances can be used to model the assumptions of the agent about the representation of observations and commands. Imagine an adversarial setting (Figure 1.2), in which the nuisances act on observations and commands in between the agent and the world. An adversary can choose the nuisances once, before the interaction between agent and world starts. Assuming that the performance of the agent does not depend on the representation of the data, but rather on some hidden state of the system, the nuisances do not change the achievable performance in a task, because it is always possible for the agent to compensate for them.

Clearly no agent will be able to compensate for all nuisances. The set of nuisances for which the agent can compensate encode the assumptions of the agent about the representation.

One can define an equivalence relation of the space $\mathcal{D}(\mathcal{Y};\mathcal{U})$, in which two systems are equivalent if they can be transformed into each other by a representation nuisance The set of all systems $\mathcal{D}(\mathcal{Y};\mathcal{U})$ can be factorized as $\mathcal{D}(\mathcal{Y};\mathcal{U}) = \mathcal{D}^\star(\mathcal{Y};\mathcal{U}) \times \mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$, where $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$ is the set of equivalence classes (systems up to representation). This factorization allows to distinguish two types of assumptions of an agent:

(1) The assumptions about the properties of the system that are invariant to the representation (here called "physical" properties). This is a subset of $\mathcal{D}^{\circ}(\mathcal{Y};\mathcal{U})$.

(2) The assumptions about the representation of the data. These are specified by the subset of the representation nuisances $\mathcal{D}^{\star}(\mathcal{Y};\mathcal{U})$ that the agent can tolerate.

1.5.6. *Symmetries of the task*

The "task," defined by a reward function, an error function, or other similar performance measure, is usually considered part of the problem formalization, and beyond any critique. From the bootstrapping perspective, one can show that the symmetries of the task (defined formally as the symmetries of a certain stochastic process derived from the agent-world interaction) are equivalent to semantic assumptions from which the agent will suffer.

The same task can be encoded by slightly different error function (Table 1.3). Each error function can be associated to a symmetry group with respect it is invariant; and each symmetry group corresponds to a certain class of semantic assumptions.

Table 1.3. Examples of symmetries for error functions

error function	*symmetries*	
	for $\mathcal{Y}$	*for* $\mathcal{U}$
$\int \|\boldsymbol{y}\|_2 \,\mathrm{d}t$	$\mathsf{O}(n_{\boldsymbol{y}})$	$\mathsf{Aut}(\mathcal{U})$
$\int \|\boldsymbol{y}\|_1 \,\mathrm{d}t$	$\mathsf{D}_{\pm}(n_{\boldsymbol{y}}) \times \mathsf{Perm}(n_{\boldsymbol{y}})$	$\mathsf{Aut}(\mathcal{U})$
$\int \|\boldsymbol{y}\|_\infty \,\mathrm{d}t$	$\mathsf{D}_{\pm}(n_{\boldsymbol{y}}) \times \mathsf{Perm}(n_{\boldsymbol{y}})$	$\mathsf{Aut}(\mathcal{U})$

Each error function can be mapped to its symmetry group. In the bootstrapping scenario, an error function that has a large symmetry group is better because it makes less assumptions about the system.

1.5.7. *Bootstrapping as a canonization problem*

A large part of the complexity of bootstrapping (and other learning problems) is in being invariant to the representation nuisances. The same physical system (an element

of $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$) can appear radically different depending on the way its observations and commands are represented.

Many other estimation problems have the same structure, in which a relatively low-dimensional object produces a high-dimensional data distribution mainly because of a group that acts as a nuisance (Figure 1.3). An example is object classification, where the variability of the observations is mostly due to the nuisances on the data rather than class variability [51].

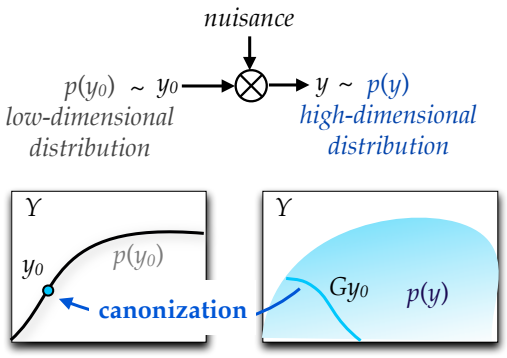


Figure 1.3. Bootstrapping belongs to a large class of problems in which the variability of the observed data is mainly due to an invertible nuisance acting on the data. *Canonization* is one approach to create agents that are invariant to such nuisances.

A general approach to deal with such a nuisance is finding a *canonization operator* that maps the observed data to a *canonical representation.*

A "pontifical feature" [52] is a function of the data that allows to define a canonical representation for the action of a group. While it is always possible to define such a pontifical feature for the action of a group, obtaining the corresponding canonization operator implies solving a difficult optimization problem.

Relaxing some of the assumptions leads to studying "weaker" pontifical features that can canonize only a subgroup of the group acting as a nuisance. The canonization operators of weak pontifical features can be composed together to obtain a canonization operator for larger groups, but only if the subgroups "play well" together.

Part 1

A Formal Approach to Bootstrapping

CHAPTER 2

A Quick Group Theory Tutorial

This chapter is a quick tutorial to groups, groups actions, and related concepts, for the reader who does not know the meaning of "homomorphism". An alternative to this chapter, which leaves out many of the things that make group theory interesting, is spending a couple of afternoons with the first chapters of a text such as Rotman [53].

2.1. Automorphisms

An *automorphism* of a set $\mathcal{X}$ is an invertible function from the set to itself. For example, if $\mathcal{X} = \mathbb{R}^n$, the map $v \mapsto 2v$ is an automorphism, because it maps $\mathbb{R}^n$ to itself, and it has an inverse $v \mapsto \frac{1}{2}v$. Another automorphism is the map $v \mapsto v + \mathbf{1}_n$, with inverse $v \mapsto v - \mathbf{1}_n$.

$\mathsf{Aut}(\mathcal{X})$ is the set of all automorphisms of $\mathcal{X}$. This set is huge if $\mathcal{X}$ is a continuous space, but if the set $\mathcal{X}$ is finite, then one can enumerate all automorphisms. Suppose there are n elements in $\mathcal{X}$ and give an index $i \in \{1, 2, \ldots, n\}$ to each element. An automorphism of the set is described by a map between indices, thus by a vector $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_n\} \in \{1, 2, \ldots, n\}^n$, where the element π_i describes the index of the element in which the i-th object is transformed. Because the map must be invertible, there cannot be repetitions ($\pi_i \neq \pi_j$ if $i \neq j$). In other words, this vector describes a *permutation* of the indices $\{1, 2, \ldots, n\}$. This means that there are exactly $n!$ automorphisms in a discrete set, corresponding to all possible permutations of the n elements.

2.2. Groups

A set is a *group,* if it is equipped with a binary operation called *group operation* (here denoted by "$\circ$"), a unary operation, inversion, and an identity element, denoted e. The group operation must be *closed* on the set (for any two elements g, h, the element $g \circ h$ must belong to the set as well) and associative, in the sense that $g \circ (h \circ k) = (g \circ h) \circ k$. The inversion operation maps each element g to its inverse g^{-1}, such that their product is the group identity: $g \circ g^{-1} = g^{-1} \circ g = e$.

These conditions are verified for automorphisms, considering function composition as the group operation. By definition, all elements $f \in \mathsf{Aut}(\mathcal{X})$ have an inverse f^{-1}, contained in $\mathsf{Aut}(\mathcal{X})$. The set is also closed with respect to the composition operation. If two maps $g, h \in \mathsf{Aut}(\mathcal{X})$, then $g \circ h$, the composition of the two functions, is again an element of $\mathsf{Aut}(\mathcal{X})$, because it is closed on $\mathcal{X}$ and it is invertible, its inverse being $h^{-1} \circ g^{-1}$. This makes $(\mathsf{Aut}(\mathcal{X}), \circ)$ a group.

Most of the time, the group operation is not indicated explicitly, but, in principle, one should specify it, because a set can be a group under different operations. Consider the operation $\diamond : \mathsf{Aut}(\mathcal{X}) \times \mathsf{Aut}(\mathcal{X}) \to \mathsf{Aut}(\mathcal{X})$, defined as $g \diamond h = h \circ g$. This is just function composition, in the "wrong" order, and still satisfies associativity: $g \diamond (h \diamond k) = (k \circ h) \circ g = k \circ (h \circ g) = (g \diamond h) \diamond k$. Hence $(\mathsf{Aut}(\mathcal{X}), \diamond)$ is a group, different from $(\mathsf{Aut}(\mathcal{X}), \circ)$. While it is important to specify the composition operator, it is not important to specify the inversion operation because it is uniquely defined given the group operation.

Group theory uses minimal notation. Because the group operation is associative, parentheses are unnecessary, so there is no ambiguity in writing $g \circ h \circ k$. It is common to use the multiplicative notation, whereas the group operation is represented by an implicit multiplication. For example, when one writes an expression like "$ax + b$", it is implicit that

there is a multiplication between "a" and "x". Using this convention, one writes simply $g\,h\,k$ instead of $g \circ h \circ k$. Contrary to multiplication, in general, group operations are *not* commutative, which means that $g\,h\,k$ is not necessarily equal to $k\,h\,g$.

2.3. Some Commonly Used Groups

$(\mathbb{R}_\circ, \times)$ The nonzero reals with multiplication as the operation.

$(\mathbb{R}, +)$ The reals with addition as the operation.

$(\pm 1, \times)$ The set $\{+1, -1\}$ with multiplication.

$(\mathbb{R}^+_\circ, \times)$ The positive reals with multiplication as the operation.

The matrix groups represent affine transformations of Euclidean space. The relation among these can be seen in Figure E.2.

$\mathsf{Aff}(\mathbb{R}^n)$ Affine transformations (Definition D.3)

$\mathsf{E}(n)$ Euclidean transformations (Definition D.3)

$\mathsf{SE}(n)$ Orientation-preserving euclidean transformations (Definition D.3)

$\mathsf{GL}(n)$ Linear transformations (Definition D.2)

$\mathsf{O}(n)$ Orthogonal transformations (rotations and reflections) (Definition D.2)

$\mathsf{SO}(n)$ Rotations (Definition D.2)

For $\mathcal{M}$ a generic differentiable manifold, the following sets of transformations are groups:

$\mathsf{Homeo}(\mathcal{M})$ Homeomorphisms (invertible and continuous) (Definition E.5)

$\mathsf{Diff}(\mathcal{M})$ Diffeomorphisms (invertible and differentiable) (Definition E.20)

$\mathsf{Isom}(\mathcal{M})$ Isomorphisms (Definition E.3)

The relations among these groups can be seen in Figure E.3.

2.4. Subgroups

A *subgroup* (Definition C.5) is a subset of a group that is closed with respect to the group operation and to inversion.

EXAMPLE 2.1. A subgroup of $\mathsf{Aut}(\mathbb{R}^2)$ is the set of uniform scaling maps

$$\mathsf{Sc}(2) = \{g_\alpha \mid \alpha \in \mathbb{R},\, \alpha \neq 0\}, \tag{2.1}$$

where each element g_α of $\mathsf{Sc}(2)$ is a map that scales vectors uniformly by a constant:

$$\begin{aligned} g_\alpha : \mathbb{R}^2 &\rightarrow \mathbb{R}^2, \\ v &\mapsto \alpha v. \end{aligned}$$

$\mathsf{Sc}(2)$ is closed with respect to the group operation, because for any two elements g_α, g_β, the composition $g_\alpha \circ g_\beta = g_{(\alpha\beta)}$ is still an element of $\mathsf{Sc}(2)$, and to inversion, because the inverse of g_α is $g_{\alpha^{-1}}$.

The notation "$G \leq H$" means that G is a subgroup of H. The "$\leq$" comparison is a partial order on the set of subgroups of a given group. This means that $G \leq H$ and $H \leq K$ implies $G \leq K$ (transitivity), but it is not true that necessarily $A \leq B$ or $B \leq A$.

A *normal subgroup* (Definition C.11) is a special kind of subgroup. Most of the difficulty with dealing with groups is that the group operation is not commutative. A subgroup N of a group G is *normal* (written "$N \triangleleft G$") if it commutes with all elements of the groups, in the

---

partial order: An antisymmetric, transitive, and reflexive binary relation. In a partial order it is not guaranteed that all elements are comparable. For example, the set of subsets equipped with the subset inclusion relation $\subset$ is a partial order. Another partial order is the set of first-order logic statements using implication as the relation. See Definition A.8.

sense that $gN = Ng$ for all elements of G. The set gN is the set obtained by multiplying g with all elements of N: $gN = \{ gn \mid n \in N \}$. Usually one must be careful when dealing with groups: $gN = Ng$ does not imply that $gn = ng$ for every $n \in N$.

2.5. Homomorphisms

A *homomorphism* (Definition C.17) is a map between two groups that preserves the group operation. Let $(G, \circ)$ and $(H, \diamond)$ be two groups. A map $\varphi : G \to H$ is a homomorphism if for any $g, h \in G$ it holds that

$$\varphi(g \circ h) = \varphi(g) \diamond \varphi(h).$$

REMARK. A "homomorphism" should not be confused with a "homeomorphism".

Two groups are said to be *isomorphic* (written $G \cong H$) if there is an isomorphism between the two.

EXAMPLE. Because there is a homeomorphism between $\mathbb{R}$ and $[0, 1]$, the homeomorphisms of the real line and the homeomorphisms of an interval are isomorphic:

$$\mathsf{Homeo}(\mathbb{R}) \cong \mathsf{Homeo}([0, 1]).$$

2.6. Group Actions

The *action* of a group $(G, \circ)$ on a set $\mathcal{X}$, indicated by a dot "$\cdot$", is a map $\cdot : G \times \mathcal{X} \to \mathcal{X}$ that satisfies two properties (Definition C.35):

---

homomorphism: A homomorphism between is a map between two groups compatible with the group operation. See Definition C.17.

homeomorphism: A homeomorphism (different from a *homomorphism*) is a continuous invertible transformation. See Definition E.5.

Identity. The action of the identity e leaves all elements $x \in \mathcal{X}$ fixed:

$$e \cdot x = x.$$

Associativity. The action of an element g, followed by the action of h, is equivalent to the action of $h \circ g$:

$$h \cdot (g \cdot x) \quad = \quad (h \circ g) \cdot x.$$

EXAMPLE 2.2. The group $(\mathsf{Aut}(\mathcal{X}), \circ)$ acts on $\mathcal{X}$ by defining $f \cdot x$ as simply the evaluation of the function $f \in \mathsf{Aut}(\mathcal{X})$ on the element x:

$$f \cdot x \triangleq f(x).$$

Identity. The identity of $\mathsf{Aut}(\mathcal{X})$ is $\mathsf{Id}_{\mathcal{X}}$ and it satisfies

$$\mathsf{Id}_{\mathcal{X}} \cdot x = x.$$

Associativity. Given two functions $f, g \in \mathsf{Aut}(\mathcal{X})$, it holds that

$$f \cdot (g \cdot x) = (f \circ g) \cdot x.$$

2.7. Orbits and Equivalence Classes

The *orbit* of a point $x \in \mathcal{X}$ for the action of a group G is the subset $G \cdot x \subseteq \mathcal{X}$ corresponding to all the images of x produced by the action of all elements of G:

$$G \cdot x = \{\, g \cdot x \mid g \in G \}.$$

The action of a group on a set induces an equivalence relation on the set, which is a partition of $\mathcal{X}$ in disjoint orbits.

EXAMPLE 2.3. Consider the action of the rotation group $G = \mathsf{SO}(2)$ on $\mathbb{R}^2$. For the point $x = (1,0) \in \mathbb{R}^2$, the orbit $G \cdot x$ is the unit circle. For the point $y = (2,0)$, the orbit $G \cdot y$ is the circle of radius 2. For the point $z = (0,1)$, the orbit is again the unit circle, so in this case $G \cdot x = G \cdot z$, and the two points belong to the same orbit. For the origin, the orbit $G \cdot 0$ is just $\{0\}$.

2.8. Invariance of Functions

Let G be a group acting on the set $\mathcal{X}$. A function $f : \mathcal{X} \to \mathcal{Y}$ is *G-invariant* (Definition C.39) to the action of G if

$$f(g \cdot x) = f(x).$$

A G-invariant function is constant on the orbits of G (Figure 2.1); this means that the preimage of a value is a G-orbit, or the union of several orbits. Some invariant functions are trivial: for example, 0 is an invariant function.

---

orbit: The image of the action of an entire group applied to one base point. See Definition C.36.

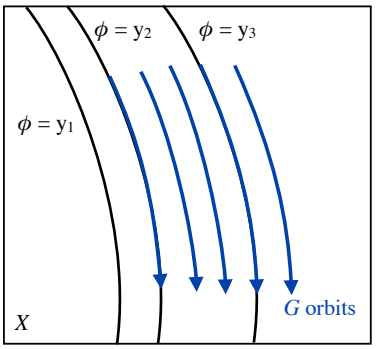


Figure 2.1. A G-invariant function $y = \phi(x)$ is constant on the orbits of G.

Sometimes a function "plays well" with the action of a group even though it is not invariant.

An *equivariant* function (Definition C.41) is a function from $\mathcal{X}$ to itself for which

$$f(g \cdot x) = g \cdot f(x).$$

A *contravariant* function (Definition C.42) is one for which

$$f(g \cdot x) = g^{-1} \cdot f(x).$$

2.9. Symmetries of Sets and Functions

Informally, the "symmetries" of an object are the set of group actions that preserve some specific property of the object. Finding the symmetries of an object is an important operation that is given different names for different kinds of objects.

2.9.1. *Symmetries of sets*

Let G be a group acting on $\mathcal{X}$, and let $\mathcal{A}$ be a subset of $\mathcal{X}$. The *stabilizer* $\mathsf{stab}^G(\mathcal{A})$ is the largest subgroup of G that leaves the set $\mathcal{A}$ invariant, in the sense that, for each element $a \in \mathcal{A}$ and each element $g \in \mathsf{stab}^G(\mathcal{A})$, the transformed element $g \cdot a$ still belongs to $\mathcal{A}$ (Definition C.39).

EXAMPLE 2.4. Take as the group the set of all euclidean motions $\mathsf{SE}(2)$ on the plane (Definition D.3), and as the set the unit circle $\mathbb{S}^1$. Then the stabilizer is the group of rotations:

$$\mathsf{stab}^{\mathsf{SE}(2)}(\mathbb{S}^1) = \mathsf{SO}(2).$$

Take as the set the horizontal axis $\mathcal{A} = \{(a, 0) \mid a \in \mathbb{R}\} \subset \mathbb{R}^2$. Then the stabilizer is the group of translations in the horizontal direction, as well as reflections about the vertical

axis.

2.9.2. *Symmetries of functions*

A similar construction can be done for functions. Let G be a group acting on $\mathcal{X}$, and consider a function f on $\mathcal{X}$. The set $\mathsf{Sym}^G(f)$ is the largest subgroup of G to which f is invariant, in the sense that if $g \in \mathsf{Sym}^G(f)$, $f(g \cdot x) = f(x)$ for all x (Definition C.40).

EXAMPLE 2.5. Consider the function $f : \mathbb{R}^2 \to \mathbb{R}^+_\bullet$ that computes the norm of a vector: $f(\boldsymbol{v}) = \|\boldsymbol{v}\|_2$. Then the symmetries are the set of orthogonal transformations (Definition D.2):

$$\mathsf{Sym}^{\mathsf{E}(2)}(f) = \mathsf{O}(2).$$

2.10. Group Products

The set $\mathsf{Sc}(2)$ of uniform scalings in (2.1) is a subgroup of $\mathsf{Aut}(\mathbb{R}^2)$. Another subgroup is the set of rotation of the plane:

$$\mathsf{SO}(2) = \{\, r_\theta \mid \theta \in [0, 2\pi) \,\},$$

with each element being a rotation associated to a particular angle:

$$\begin{array}{rcl} r_\theta : \mathbb{R}^2 & \to & \mathbb{R}^2, \\ \boldsymbol{v} & \mapsto & \left(\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right) \boldsymbol{v}. \end{array}$$

Given two subgroups $G, H \leq \mathsf{Aut}(\mathcal{X})$ the set "$G \vee H$" is defined as the set of all possible combinations of arbitrary lengths of elements in G and H (Definition C.9). The result is yet another subgroup of $\mathsf{Aut}(\mathcal{X})$, often much larger than either two.

A *direct product* (Definition C.26) $G \times H$ is the product of two disjoint subgroups that commute with each other. In this case, rotation and scalings do commute. Therefore, any

arbitrary sequence of rotations and scalings, however intertwined, can be always written in a unique way as exactly one rotation followed by one scaling, or vice versa. In this case, these groups play very well together.

Let $\mathsf{T}(2)$ be the group of translations of the plane:

$$\mathsf{T}(2) = \{\, t_{\boldsymbol{x}} \mid \boldsymbol{x} \in \mathbb{R}^2 \,\},$$

$$\begin{aligned} t_{\boldsymbol{x}} : \mathbb{R}^2 &\rightarrow \mathbb{R}^2, \\ \boldsymbol{v} &\mapsto \boldsymbol{v} + \boldsymbol{x}. \end{aligned}$$

A translation does not commute with scaling and rotations. Given a scaling $g_\alpha \in \mathsf{Sc}$ and a translation $t_{\boldsymbol{x}} \in \mathsf{T}(2)$, the product $g_\alpha \circ t_{\boldsymbol{x}}$ is not equal to $t_{\boldsymbol{x}} \circ g_\alpha$. In fact, it holds that

$$g_\alpha \circ t_{\boldsymbol{x}} = t_{\alpha \boldsymbol{x}} \circ g_\alpha.$$

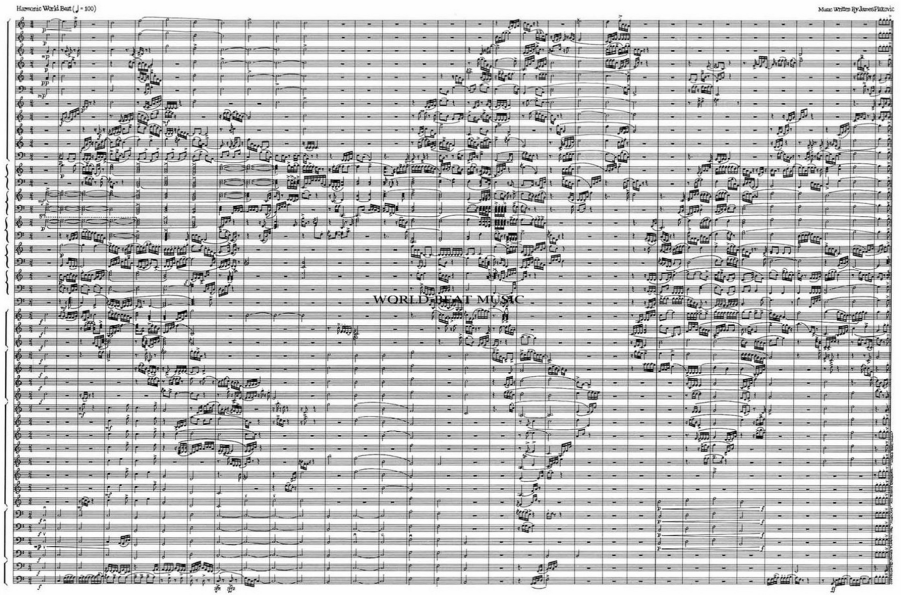
Similarly, given a rotation $r_\theta \in \mathsf{SO}(2)$ and a translation $t_{\boldsymbol{x}} \in \mathsf{T}(2)$, it holds that

$$r_\theta \circ t_{\boldsymbol{x}} = t_{r_\theta \boldsymbol{x}} \circ r_\theta.$$

In these two cases, there is a "partial" commutation, because the elements r_θ and g_α could move from the left to the right of $t_{\boldsymbol{x}}$, which was affected by them. The union of the two groups is $\mathsf{SE}(2) = \mathsf{SO}(2) \vee \mathsf{T}(2)$. $\mathsf{T}(2)$ is a normal subgroup of $\mathsf{SE}(2)$, because, for each r_θ, $r_\theta \mathsf{T}(2) = \mathsf{T}(2) r_\theta$.

The product of two disjoint subgroups $N, H \leq G$ where one of them is normal ($N \triangleleft G$) is called a *semidirect product* (Definition C.27), and it is written as $N \rtimes H$.

(a) On the left, a 25×25 bitmap picture of a rose. On the right, a 25×25 bitmap corresponding to the ASCII encoding of the sentence *"What's in a name? That which we call a rose, by any other representation would smell as sweet"*.



(b) "World Beat Music": a piece composed by musician James Plakovic.

Figure 2.1. Meditation materials for thinking about the relation between representation and semantics.

CHAPTER 3

Bootstrapping and Semantic Assumptions

We want to create agents that can work with the largest class of systems possible. In other words, they must have as few "assumptions" as possible. There are two kinds of assumptions: those about the physical system (which sensors/actuators the body has) and those about the representation of observations and commands. This goes beyond specifying the format *of the data, but, rather, it implies specifying what is the implied "semantic" of the data. This chapter makes the connection between the invariance properties of the agent, which can be described concretely with particular group actions, and their interpretation as proxies for the "assumptions" of the agent about the "semantic" of the data.*

3.1. Do Not Trust Strangers in the Streets

It is a serene night in the magnificent historical city that hosts the annual robotics conference. After the conference banquet, you and I are walking back to our hotel. We are talking about the many merits of choosing a conservative thesis topic in a well-defined, established field. We then talk about bootstrapping, and how much it looks like an impossible problem.

Suddenly, a voice comes from a shady alley: "Bootstrapping, uh? You fine scholars might be interested in this: the ultimate bootstrapping agent,". The man produces one from his pocket. It is "an agent that has no assumptions," the stranger says, and "can work with any system," he continues, ignoring our raised eyebrows. He asks a reasonable price; at least, it is cheaper than what we paid last year for "the Newton variant that always

finds the local minimum."

"How does it work?" "See this port, here on the side, where it says *observations*? You connect this to the output of your black box. This other port, where it says *commands*, should be connected to the input of your black box. There is a keyboard on the top, which you used to define the task. Then turn it on, and it is ready to go!"

"What can this agent do?" "Any task you want! But you should be able to describe it with uninterpreted signals. It's all explained in the manual. For a quick start, it also comes with a default set of intrinsic motivation policies. This model has even an additional port for a reward signal, right here under this flap. You can get it working on any system in less than five minutes."

"*Any* system? Without *any* assumptions? That seems a bit far-fetched." "Oh yes, no assumptions at all. And let me be clear: this bootstrapping agent does not do some simple policy search, but actually learns a model of the world on which it does explicit inference. Do you want to see a demo?"

As it happens, I always keep a black box in my bag. This one is connected to a robot in our lab, but I do not remember which one. "Wait!" you say "Back at the banquet, we just saw that magician doing physically impossible things with that deck of cards. I am not sure I can trust any demo tonight. Even so, you only have one black box—how can one demo show anything?..." I stop you before you start citing Popper and Dijkstra.

We are skeptical about the stranger's claims. Clearly this agent has *some* assumptions about its observations and commands. If we find a way to convert these vague "no assumptions" claims into a falsifiable statement, then we have some ground to haggle down the price.

Karl Popper: (1902–1994) Austrian philosopher of science who popularized falsificationism as opposed to inductionism.

"Wait a minute, let us think." The stranger does not seem to be in a hurry, so we have time to think about this in a principled way.

3.2. Format vs Semantics

The black box that I carry everywhere is travel sized. TSA requirements limit its output to only one real number.

"Does your agent work with real numbers?" "Not natively—it actually assumes that the signals are infinite bit strings. But, for a small price, we can give you a universal adapter that transforms every data to bit strings. It comes with a ROS interface."

Even before the interaction begins, there needs be some interface between agent and world, some agreement on the domains for observations and commands. By calling the observations $\boldsymbol{y} \in \mathcal{Y}$, and the commands $\boldsymbol{u} \in \mathcal{U}$, the agent and the world should agree on the *observations space* $\mathcal{Y}$ and the *command space* $\mathcal{U}$. Possible choices for the observations space $\mathcal{Y}$ could be the set of real numbers, vectors of natural numbers, grayscale images, string dictionaries. This is called the "format" of the data.

An alternate view is assuming that the signals of all worlds belong to some large fixed space, such as binary strings. Every data can have a suitable representation as a string of bits, therefore we could take $\mathcal{Y}$ and $\mathcal{U}$ to be $\{\square, \boxdot\}^n$, for some large n. This is what actually happens in a computer: while we can talk mathematically about agents reading and writing continuous data (e.g., $\boldsymbol{y} \in \mathbb{R}$), in practice the algorithmic implementation is run on a digital computer, where operands are bits. In this view, the format of the data corresponds to the syntax that specifies which are the valid messages from world to agent and vice versa.

A central point is trying to reason about the implied *semantics*, which is beyond the choice of format. Assume the observations are strings in $\{\square, \boxdot\}^n$. We purposefully use

the symbols $\{\square, \boxdot\}$ instead of $\{0, 1\}$ to underline that these are just symbols, that can have meaning only when a suitable semantic is attached to it. But what are the semantic assumptions on those bits? For example: bits are grouped in octets, sometimes four to eight of those octets represent a floating point number, $\boxdot$ is "more" than $\square$, etc.

3.3. Formalizing Semantic Assumptions

We stare at the travel-sizes black box. We are convinced that data format is not the main obstacle toward understanding the agent's "assumptions."

We know that there are many reasonable assumptions about the *physical* properties of the system: the system is causal; it obeys the laws of physics; the observations give some information about some internal state of the system, which in turn is influenced by the commands. But here we are interested in the "assumptions" about the "semantics" of the *values* that we observe coming out of the black box. We realize that even we have some predefined "assumptions" on the "semantics" of the data. We also realize that so far we have used the words "assumptions" and "semantics" with quotes because we have not defined them, and doing all these air quotes is starting to be tiring.

When we turn on the black box, we shall see numbers flashing before us. For example, "1" and "2" are both elements of $\mathbb{R}$ and thus possible observations of the black box. Those are just symbols, and symbols have no meaning before a proper semantics is attached to them. If we saw the observations to be 1 instead of 2, we would conclude something different about the state of the system. Hence, a first semantic assumption is that different values have different meaning. Having established exactly in what way $1 \neq 2$, we can rejoice in our best academic chuckle.

Consider other two symbols: "0" and "2π." If the observations are supposed to represent angles, and if those angles are represented using radians, as opposed to degrees, then the two symbols have the same meaning. Sometimes, $1 = 2$.

Consider other two symbols: "1" and "1.01". Would you say that they are "close"? If you would not commit to such a strong assertion—"How close is 'close'," you say—would you at least say that "1" is "closer" to "1.01" than "1.01" is to "2"?

All these statements might be part of the assumptions of the agent when processing the observations. These statements can be formalized as a set of relations, which can be imagined as part of the axioms of an inference engine used by the agent.

Let d be a small positive value. (If you know nonstandard analysis, the statements below have formal validity assuming d to be an infinitesimal.) Consider the statement:

$$\text{For all } y \in \mathbb{R},\ y \text{ is "close" to } y + d \text{ (and vice versa).} \tag{3.1}$$

This statement, or rather, because of the qualifier "for all", this *set* of statements, describes some structure on the set $\mathcal{Y} = \mathbb{R}$, namely, the standard topology. The word "close" in (3.1) is distracting, because we already have a semantic interpretation for it in our head. For this reason, we rewrite the relation using the Chinese character "近", in the same way as we used the symbols □ and ⊡ in place of 1 and 0:

$$\text{For all } y \in \mathbb{R},\ (y \text{ 近 } y + d) \wedge (y \text{ 近 } y - d)\ . \tag{3.2}$$

The statement above contains the "$+$" operation, but that is just a shortcut for not enumerating each value of y.

A binary relation (Definition A.5) over elements of $\mathbb{R}$ can be described either as a subset of $\mathbb{R} \times \mathbb{R}$ (i.e., 近 $\subset \mathbb{R} \times \mathbb{R}$), or, equivalently, by a binary function on $\mathbb{R} \times \mathbb{R}$ (i.e., 近 :

---

relation: A binary relation is formally defined as a subset of the cartesian product of two sets. See Definition A.5.

topological space: A topological space is a set endowed with a topology. See Definition A.3.

$\mathbb{R} \times \mathbb{R} \to \{\square, \boxdot\}$).

Some semantic assumptions correspond to relations with more than two terms, and consequently to functions on $\mathcal{Y} \times \cdots \times \mathcal{Y} = \mathcal{Y}^m$, where m is the order of the relation. These relations can be represented graphically if some poetic license is allowed (Figure 3.1).

An agent that assumes that the observations are angle expressed in radians would use a suitably modified relation:

$$\text{For all } y \in \mathbb{R},\ (y \text{ 近 } y + d) \wedge (y \text{ 近 } y - d) \wedge (y \text{ 近 } y + 2\pi). \tag{3.3}$$

Only the partition induced by these relations is significant. For maximum cognitive

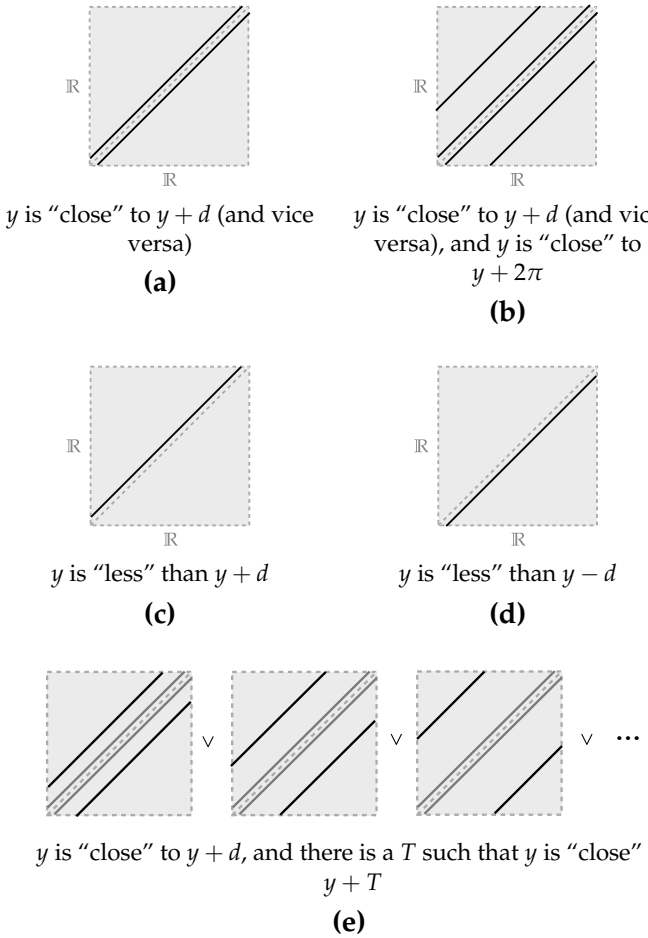


Figure 3.1. Some semantic assumptions can be described by a relation on $\mathcal{Y}^k$, where k is the order of the relation.

dissonance, let us return to English words for describing relations. Consider the following two statements:

$$\text{For all } y \in \mathbb{R},\ y \text{ is "less" than } y + d. \tag{3.4}$$

$$\text{For all } y \in \mathbb{R},\ y \text{ is "less" than } y - d. \tag{3.5}$$

The partition of $\mathbb{R} \times \mathbb{R}$ induced by these relations is the same (Figure 3.1c, Figure 3.1d), in the sense that the mapping $y \mapsto -y$ transforms one into the other.

This is an example of a relation that involves more than two values:

$$\text{For all } y \in \mathbb{R},\ y \text{ is "as close" to } y + d \text{ as it is to } y - d. \tag{3.6}$$

This statement concerns the two couples of numbers $(y, y + d)$ and $(y, y - d)$. This relation is not pictured in the figure because we would need a 4D space to display.

Also *operations* on y that the agent performs on the data can be interpreted as relations. For example, "$1 + 1 = 2$" means "the tuple $(1, 1, 2)$ belongs to the '$+$' relation."* The order of this relation is 3, so we abstain from drawing it, but you can picture it as the $x + y = z$ plane in the (x, y, z) space.

3.4. Symmetries of Semantic Assumptions

If we accept that semantic assumptions can be described by these relations, the next natural question is how to compare assumptions across agents, so that we can find agents with the "least" assumptions.

One way to proceed would be to study the semantic assumptions as known mathematical objects or structures attached to the domain $\mathcal{Y}$. It was already noted that the assumption (3.1) implies that the agent takes the real line $\mathbb{R}$ to have the usual topology. The

*You might remember from your first undergraduate analysis class (if you took it in a Bourbaki-infected country) that a *function* is a particular kind of relation. See Definition A.13.

assumption (3.3) implies that the topology is that of the unit circle $\mathbb{S}^1$. Assumptions (3.4) and (3.5) imply that the real line is assumed to be the ordered set $(\mathbb{R}, <)$.

However, this approach would be unsatisfactory, for two reasons. First, we are not content with stating facts such has "the agent considers $\mathbb{R}$ as a topological space" or "an ordered set," because it will not be possible to make the statements "considers," "knows," "assumes" *falsifiable*. Second, sometimes there is no preexisting mathematical structure to use. Consider the assumption

$$\text{For all } y,\ y \text{ is "close" to } y + d, \text{ and there is a } T \text{ such that } y \text{ is "close" to } y + T. \tag{3.1}$$

This is the point of view of an agent which assumes to deal with angles, but does not know the measurements units *a priori*.

An alternative approach is characterizing the *symmetries* of these semantic assumptions. Each relation can be assigned to the set of invertible transformations of $\mathcal{Y}$ that preserve it. We limit this search to the invertible transformations (called "automorphisms" of $\mathcal{Y}$) because we do not want to lose information.

Consider, for example, the assumption (3.1). In this case, the set of invertible functions $f : \mathcal{Y} \to \mathcal{Y}$ such that

$$f(y) \text{ is "close" to } f(y + d) \text{ (and vice versa } \Leftrightarrow$$

$$y \text{ is "close" to } y + d \text{ (and vice versa).}$$

is the set of *homeomorphisms* $\mathsf{Homeo}(\mathbb{R})$, which are the invertible and continuous functions from $\mathbb{R}$ to itself.[†]

ordered set: A set equipped with a total order relation. See Definition A.9.

[†]Continuous functions preserve the topology by definition, if one uses the topological

This exercise can be repeated for the other assumptions (Table 3.1).

The transformations that preserve (3.3) are necessarily homeomorphisms, with the extra condition that the periodicity is preserved:

$$\{\, f \in \mathsf{Homeo}(\mathbb{R}) \mid f(y + 2\pi) = f(y) + 2\pi \,\}. \tag{3.2}$$

The assumption (3.1) is less restrictive, because the period T is allowed to vary, and therefore the set of symmetries is larger:

$$\{\, f \in \mathsf{Homeo}(\mathbb{R}) \mid \exists\, T :\ f(y + T) = f(y) + T \,\}. \tag{3.3}$$

As the semantic assumptions become more complicated, the description of these sets of functions becomes more convoluted. At this point, it is useful to switch to the language of group theory.

Rusty in group theory? Go read the first sections of Chapter 2 and then come back.

Using the language of group theory has two immediate advantages. We can compress down the description of large set of transformations, such as (3.2), by using constructions such as the direct product of groups. We can easily compare which sets of transformations are larger, using isomorphisms properties. This allows to compare the assumptions of different agents.

If the semantic assumption is represented as a function $\mathcal{R} : \mathcal{Y}^m \to \{\square, \boxdot\}$, then we

definition of continuous function (Definition A.4). The $\epsilon - \delta$ definition of continuous function (Definition E.4) is applicable only to metric spaces. If the topology is generated by the metric, the two definitions agree.

group: A set equipped with an operation that satisfies the properties of closure and associativity, and in which every element has an inverse. See Definition C.4.

group isomorphism: An invertible map between two groups that is compatible with the group operation. See Definition C.18.

would say that we are looking at the set of *symmetries* of $\mathcal{R}$, meaning for what invertible functions f it holds that

$$\mathcal{R}(f(y_1), \ldots, f(y_m)) = \mathcal{R}(y_1, \ldots, y_m).$$

Let $\mathsf{Sym}(\mathcal{R})$ be the set of all such functions:

$$\mathsf{Sym}(\mathcal{R}) = \{\, f \in \mathsf{Aut}(\mathcal{Y}) \mid \mathcal{R}(f(y_1), \ldots, f(y_m)) = \mathcal{R}(y_1, \ldots, y_m)\}.$$

This set is a *group* under the operation of function composition $\circ$. If f belong to $\mathsf{Sym}(\mathcal{R})$, then also its inverse f^{-1} belongs to $\mathsf{Sym}(\mathcal{R})$. Moreover, if both functions f, g belong to $\mathsf{Sym}(\mathcal{R})$, then their composition $f \circ g$ belongs to $\mathsf{Sym}(\mathcal{R})$ as well.

The set of homeomorphisms $\mathsf{Homeo}(\mathbb{R})$ mentioned earlier is a group. For the set (3.2), it takes a few steps to see that this set is isomorphic to the homeomorphisms of the unit circle $\mathsf{Homeo}(\mathbb{S}^1)$. For the set (3.3), there is an extra scale invariance, which makes the set isomorphic to the direct product $\mathsf{Homeo}(\mathbb{S}^1) \times (\mathbb{R}^+_\circ, \times)$. Regarding the assumptions (3.4) and (3.5), the associated group is the set of monotonic homeomorphisms $\mathsf{Homeo}_+(\mathbb{R})$. The results are summarized in Table 3.1.

---

monotonic function: A function preserving the total order of the real line.

Table 3.1. Some semantic assumptions on the data and relative symmetry groups preserving the assumptions.

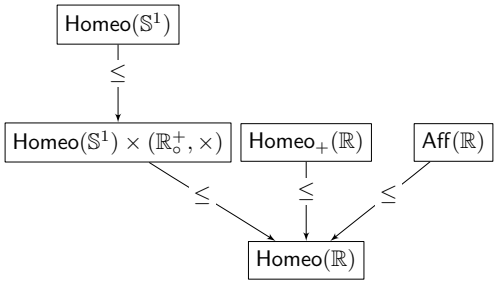
semantic assumption $\mathcal{R}$	$\mathsf{Sym}(\mathcal{R})$
y is "close" to $y+d$ (and vice versa)	$\mathsf{Homeo}(\mathbb{R})$
y is "close" to $y+d$ (and vice versa), and y is "close" to $y+2\pi$	$\mathsf{Homeo}(\mathbb{S}^1)$
y is "close" to $y+d$, and there is a T such that y is "close" to $y+T$	$\mathsf{Homeo}(\mathbb{S}^1)\times(\mathbb{R}^+_\circ,\times)$
y is "less" than $y+d$	$\mathsf{Homeo}_+(\mathbb{R})$
y is "less" than $y-d$	$\mathsf{Homeo}_+(\mathbb{R})$
y is "as close" to $y+d$ as it is to $y-d$	$\mathsf{Aff}(\mathbb{R})$

3.5. Ordering Assumptions

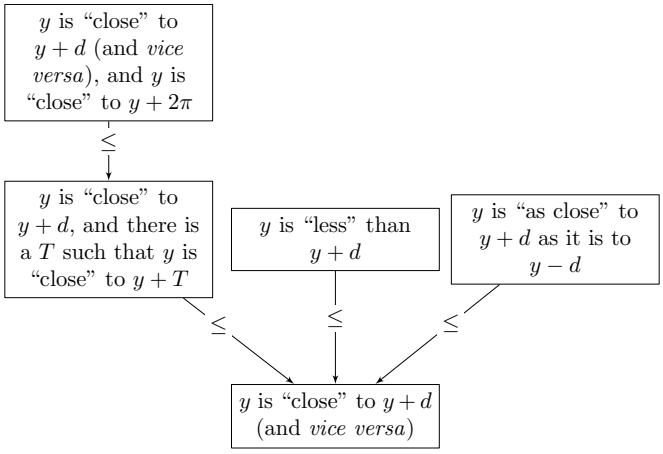
Once each semantic assumption has been assigned its symmetry group, the assumptions can be given a partial order by using the groups as proxies.

The partial order on groups is given by the subgroup relation "$\leq$" (Definition C.5). This partial order is shown in Figure 3.1a for the groups in the last example. Figure 3.1b shows the semantic assumptions ordered according to the groups partial order.

For some of these assumptions, the partial order on the group is the same as the "implication" in first-order logic. However, the groups are actually simpler to consider, because, while relations can have arbitrary order (e.g., (3.1) is a four-way relation, thus a subset on $\mathcal{Y}^4$), the symmetry groups only apply to the observations space $\mathcal{Y}$, even for higher-order relations.



(a) Partial orders on automorphisms given by the subgroup relation.



(b) Induced partial order on semantic assumptions.

Figure 3.1. Partial ordering of the assumptions.

This partial order on semantic assumptions induces a partial order on agents as well, if we can find a way to associate to each agent its semantic assumptions.

3.6. Starting from the Agent

"We think we got it," we tell the stranger, who has patiently been waiting for our elucubrations. "Could you pop your agent open? We need to have a look at its semantic assumptions."

Inside the agent, we expected to see an ordered shelf of rulebooks, possibly written in Chinese, describing the agent's axioms and rules, and some generic inference engine. Instead, the agent's internals look like an endless series of leaky pipes, from which many whirring electric components protrude. In a corner, a squirrel is busy exercising in a running wheel. "The squirrel just provides energy to the system. It is part of our company's commitment to more ecologically friendly bootstrapping agents. Sometimes it stops unpredictably, and you need to be patient and give it a few minutes. If it stops for more than five minutes, you might have to replace the squirrel."

This is not a problem, I think, because at Caltech we do have a large supply of squirrels. "We can live with the biological component,"—catching squirrels will be a much more productive endeavor than the average project for Caltech undergrads—"but where can we find this agent's semantic assumptions?" "I am sorry, but this agent has not been programmed with explicit semantic assumptions, nor it does explicit manipulation of symbols. It's more like a neural network—but not exactly so. . . "

Suppose that we have an agent, described algorithmically. Can we backtrack from the agent's algorithmic description to the agent's semantic assumptions?

As an example, consider the task of learning the sensor geometry, which is a basic task

---

sensor geometry: The metric or topological arrangement of a sensor's sensels in space.

for bootstrapping agents. It is assumed that the observations come from the sampling of a *spatial field* on some manifold. For example, the pixels of a camera sample the plenoptic function (the field) on the visual sphere (the manifold). One approach to sensor geometry reconstruction consists in finding a measure of similarity between each pair of sensels, and then use an embedding algorithm which uses the similarities as input. The basic idea is that if sensels are close in space, then their measured values will be more similar.

There are many possible choices for the similarity measures, with various tradeoffs in space/computation requirements and robustness to noise. Here is a representative samples of plausible similarities measures that an agent might use. They are written as functions $R(y^i, y^j)$, with the understanding that they are all statistics of the observed sensel values over some time period.

- The simplest choice is using the sample correlation:

$$R_1(y^i, y^j) = \mathsf{corr}(y^i, y^j). \tag{3.1}$$

- Some agent might think it is useful to use the absolute value of the correlation:

$$R_2(y^i, y^j) = |\mathsf{corr}(y^i, y^j)|, \tag{3.2}$$

  reasoning that, if one consistently observes $y^i = -y^j$, the sensels should be considered spatially close because they are observing the same signal, just with the opposite sign.

- An agent, if it had taken more than just an introductory statistics class, could use

---

embedding problem: The problem of reconstructing the positions of a set of points in a metric space given their distances.

the Spearman correlation as the similarity measure:

$$R_3(y^i, y^j) = |\mathsf{spear}(y^i, y^j)|. \tag{3.3}$$

- An agent with more computation at his disposal might use the *normalized variation of information* $\mathcal{V}_1(y^i; y^j)$ (Definition B.13):

$$R_4(y^i, y^j) = 1 - \mathcal{V}_1(y^i; y^j). \tag{3.4}$$

This is a proper metric for random variables, derived from the mutual information, which satisfies $0 \leq \mathcal{V}_1(y^i; y^j) \leq 1$.

One can associate a group of symmetries to each of these similarity measures. For example, each of the similarity measures R_k would be unchanged if all the data were to be multiplied by -1:

$$R_k(-y^i, -y^j) = R_k(y^i, y^j).$$

Formally, one says that all similarity measures are invariant to the action of the group $(\pm 1, \times)$. Here is the rest of the analysis for the other measures.

- The absolute value of correlation (3.2) is invariant to the action of the group

$$\mathsf{Aff}(\mathbb{R})^n = \mathsf{Aff}(\mathbb{R}) \times \cdots \times \mathsf{Aff}(\mathbb{R}),$$

which corresponds to an affine map acting independently on each sensels.

---

invariant function: A function that is preserved by the action of a group ($f(g \cdot x) = f(x)$). See Definition C.39.

affine map: Geometrically, an affine map preserves straight lines, but not angles or lengths. See Definition D.3.

- The simple correlation (3.1) is invariant only to orientation-preserving affine maps acting separately on each variable:

$$\mathsf{Aff}_+(\mathbb{R})^n = \mathsf{Aff}_+(\mathbb{R}) \times \cdots \times \mathsf{Aff}_+(\mathbb{R}).$$

- The Spearman correlation is invariant to all monotonic homeomorphisms $\mathsf{Homeo}_+(\mathbb{R})^n$ (Lemma B.11). Taking the absolute value of the Spearman correlation makes the similarity measure invariant to all homeomorphisms.
- The similarity measure $1 - \mathcal{V}_1(y^i; y^j)$ is invariant to all invertible piecewise-continuous maps $\mathsf{PieceHomeo}(\mathbb{R})^n$ (Remark B.12).

These results are summarized in Table 3.2.

Just like the previous example, we can use the subgroup partial order to order the groups into a lattice (Figure 3.1a), and this induces a partial order on the similarity measures (Figure 3.1b). Moreover, we can go from the groups to the semantic assumptions, by using the information in Table 3.1.

---

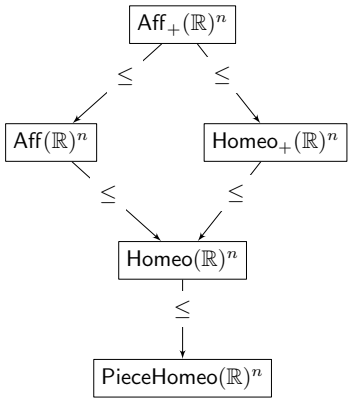
lattice: A partially ordered set with a "top" and a "bottom" elements which are comparable to all elements of the set. See Definition A.11.

Table 3.2. Some similarity measures and relative symmetry groups.

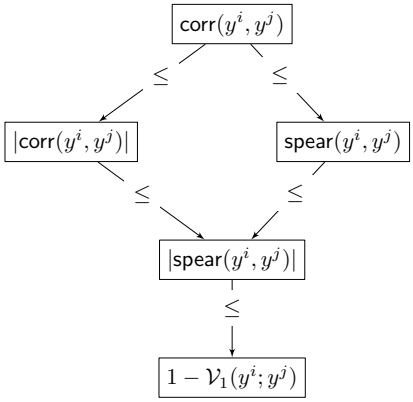
similarity measure	*symmetry*
$\lvert\mathsf{corr}(y^i, y^j)\rvert$	$\mathsf{Aff}(\mathbb{R})^n$
$\mathsf{corr}(y^i, y^j)$	$\mathsf{Aff}_+(\mathbb{R})^n$
$\mathsf{spear}(y^i, y^j)$	$\mathsf{Homeo}_+(\mathbb{R})^n$
$\lvert\mathsf{spear}(y^i, y^j)\rvert$	$\mathsf{Homeo}(\mathbb{R})^n$
$1 - \mathcal{V}_1(y^i; y^j)$	$\mathsf{PieceHomeo}(\mathbb{R})^n$

For each similarity measure among sensels (left column) we can associate its invariance group (right column). In addition to the groups displayed, every measure is invariant to the reflection $(y^i, y^j) \mapsto (-y^i, -y^j)$.

Thus, the symmetries of the algorithm used by the agent ultimately define which semantic assumptions the agent is making about the data.



(a) Partial orders on groups given by the subgroup relation.



(b) Induced partial order on similarities measures.

3.7. The adversarial view

"Could you give us an algorithmic description of this agent? We need to compute some symmetries to check its semantic assumptions." "I am sorry, our lawyers tell me I cannot show anybody the plans for this agent. Truth to be told, there was also an evolutionary stage, in which some of the original components were randomly replaced. For some reason, we found out that squirrels worked better than hamsters, even though they were only supposed to produce power. We believe that the hamster smell somehow distracted the cat ganglia simulated in those steam pipes. So, sorry, no blueprints are available. But you can absolutely play with the agent as much as you want! Let me put in a fresh squirrel."

Fortunately, we just need to look at the input-output behavior of an agent to characterize its invariance properties. It is not necessary to have an algorithmic description of the agent's internals or to access its internal states.

Imagine an adversarial setting (Figure 3.1), where we play the role of the adversaries, disturbing the communication between agent and world, by choosing a "nuisance" acting on the observations between agent and world. The term *nuisance*, as opposed to *noise*, describes something acting on the data that, in principle, can be neutralized a posteriori.

Representation nuisances are defined as fixed, invertible transformations of the observations. "Fixed" means that we, as the adversaries to the agent, can only choose one transformation at the beginning of the interaction, and the transformation will not change during the agent's experience. Because the transformations are invertible, they do not change the informative content of the data, but only the representation.

We can test the agent with different classes of representation nuisances, and check for which ones the agent's behavior is invariant. As per the previous discussion, one can interpret the invariance properties as semantic assumptions of the agents.

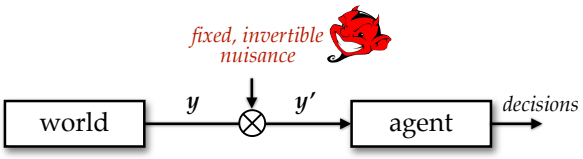


Figure 3.1. To understand the agent's assumptions about the representation of the data, we ask what fixed, invertible transformations an agent can tolerate.

3.8. Tolerance to Nuisances Acting on the Commands

The same reasoning can be followed for the commands. We close the agent-world loop (Figure 3.1), and ask what would be the effect of a representation nuisance acting on the commands.

There is a twist: instead of being *invariant* to a nuisance acting on the commands, the agent must be able to precompensate it. The reason is that the representation nuisance acts on the commands *after* the agents has chosen them. Therefore, to allow the original signal to reach the world, the agent must transform the commands by the inverse transformation applied. The agent must be invariant to the observations nuisances, but contravariant to the commands nuisances. This does not necessarily imply that the agent is separately estimating this transformation—all of these properties regard the input-output behavior, and imply nothing about the design or the agent states.

Just like the observations, the representations nuisances on the commands that the agent can compensate say something about the assumed semantic. For example, suppose that the commands space $\mathcal{U}$ is the interval $[0, 1]$. During the interaction with the world, the agent might try to learn about the effect of each command. Perhaps the interval $[0, 1]$ will be discretized to a set of 10 commands, with cells of width 0.1. Is this discretization fine enough? That depends on the properties of the world as well as the task the agent is supposed to do. However, we can already say something about the agent assumptions by looking at the invariance of the internal representation. In this case, the agent would be invariant to any transformation that maps cells to cells. The cells can be permuted, plus each of the cells interval can be reparametrized, independently by the others. Assuming that we limit ourselves to continuous transformations, this discretized representation is

---

contravariant function: $f(g \cdot x) = g^{-}1 \cdot f(x)$. See Definition C.42.

invariant to the action of the group which is isomorphic to $\mathsf{Homeo}([0,0.1])^{10} \times \mathsf{Perm}(10)$. The results on the invariance for the internal representation put an upper bound on the agent's input-output invariance.

"Wait a minute, I can see a problem." It is not possible to impose that the agent is invariant to the representation commands for all times. At time $t = 1$, when the agent chooses the first command, it has only seen one observation, from which it cannot estimate the commands transformation.

We decide that it is indeed too much to ask for the agent to be invariant at each time step from the beginning. We decide on a two-stage protocol. In the first stage, the agent is free to interact in closed-loop with the world. The output of this stage is some statistics computed by the agent, which could be interpreted as a learned model. Given the model, the agent instantiates some behavior to perform the task.

The formal definition of a "bootstrapping agent" is a tuple containing one dynamical system for the exploration stage, whose output is the model, and then a function that maps this model to another dynamical system (a "behavior") used for interacting with the world

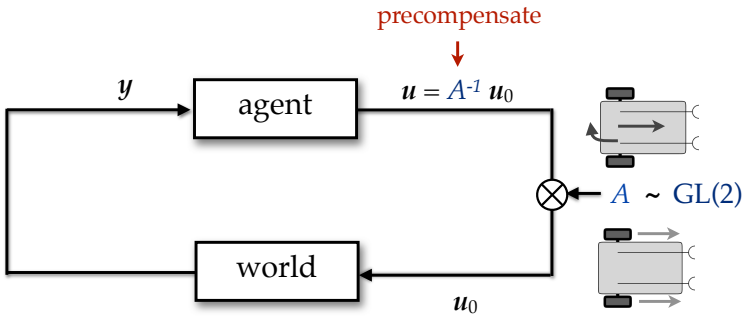


Figure 3.1. We expect that the closed-loop agent-world system to be invariant with respect to the representation nuisances. This implies that the agent's commands must be invariant with respect to the representation nuisances acting on the observations. The situation is not symmetric. Because the representation nuisance acting on the commands acts *after* the agent has chosen the commands, the agent must be able to pre-compensate those nuisances. In this figure, the linear operator **A** changes a differential-drive commanded in angular/linear velocity into one commanded with left/right wheel velocity.

during the acting stage.

3.9. Symmetries of the task

Defining the task of a bootstrapping agent can be difficult, because the task should be well defined for the agent before it even looks at the system. Moreover, just the act of specifying the task might be equivalent to making some assumptions about the system.

Let us consider tasks specified by an error function which is known to the agent, in the style of control theory. This is different than a reward, not only because the error function must be minimized,[‡] but because the agent knows explicitly how the error function depends on the observations.

In general, there is a certain freedom of the designer in choosing an error function for a given physical task, with the choice depending on a trade-off between tractability and the results in the particular testing condition. Once the designer has made their choice, the error function is considered part of the mathematical formalization of the problem. In bootstrapping, the choice of the error function can be criticized, just like an agent can be criticized for its invariance properties.

Assume that the observations $\boldsymbol{y}$ are a real vector ($\mathcal{Y} = \mathbb{R}^{n_y}$). Consider the error function

$$\int \|\boldsymbol{y}\|_2 \, \mathrm{d}t. \tag{3.1}$$

This error function is invariant to the orthogonal group $\mathsf{O}(n_y)$ acting on $\boldsymbol{y}$. This means that the choice of coordinates for $\boldsymbol{y}$ matters, because if $\boldsymbol{y}$ goes through a linear coordinated $\boldsymbol{y} \mapsto \mathbf{A}\boldsymbol{y}$, the resulting error function is not preserved, but it has the form

$$\int \|\mathbf{A}\boldsymbol{y}\|_2 \, \mathrm{d}t.$$

This new function is not invariant to $\mathsf{O}(n_y)$ anymore, but it is invariant to the conjugated

[‡]Control theorists are pessimists (bordering on the paranoid), so they want to *minimize errors*; computer scientists are optimists, so they *maximize rewards*.

group $\mathsf{conj}_{\mathbf{A}}(\mathsf{O}(n_y))$, which is isomorphic to $\mathsf{O}(n_y)$ (Lemma D.5).

The choice of norm is also very important. An important branch of control theory is concerned with minimizing error functions containing the infinity norm

$$\int \|\boldsymbol{y}\|_\infty \, \mathrm{d}t,$$

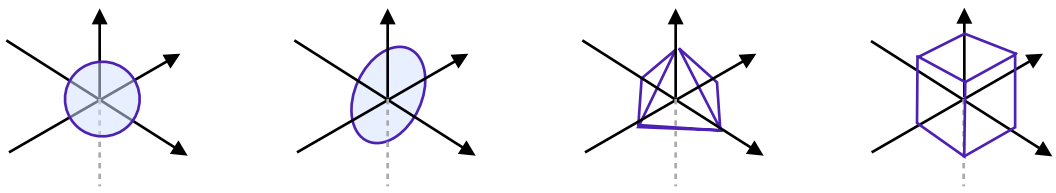
which is, roughly speaking, equivalent to a "worst-case" analysis. On the other end of the spectrum, minimizing a function of the 1-norm, such as

$$\int \|\boldsymbol{y}\|_1 \, \mathrm{d}t,$$

would correspond to being very tolerant to large deviations from the desired sate. It is possible to show (Lemma D.16) that these two norms have the same symmetries, namely they are invariant to permutations of the entries of $\boldsymbol{y}$, and any change of sign of a single entry. The invariance group is $\mathsf{D}_\pm(n_y) \times \mathsf{Perm}(n_y)$, where $\mathsf{D}_\pm(n)$ is the set of diagonal matrices with entries in $\{+1, -1\}$ (Definition D.1).

Note that this group is strictly contained in $\mathsf{O}(n_y)$: these norms are less invariant than the 2-norm. Visualizing the level sets gives a clear intuition of this fact (Figure 3.1).

It is also common to put some penalty on the commands used, representing the energy spent. Assuming that also the commands are real numbers ($\mathcal{U} = \mathbb{R}^{n_u}$), we could use $\alpha\|\boldsymbol{u}\|_2$,



(a) $\|\boldsymbol{y}\|_2$ **(b)** $\|\mathbf{A}\boldsymbol{y}\|_2$ **(c)** $\|\boldsymbol{y}\|_1$ **(d)** $\|\boldsymbol{y}\|_\infty$

Figure 3.1. The level sets of the norms considered give a geometric intuition of the relative symmetry groups. Which one has more symmetries?

where α weights the trade-off between the two terms:

$$\int \|\boldsymbol{y}\|_2 + \alpha\|\boldsymbol{u}\|_2 \,\mathrm{d}t. \tag{3.2}$$

In this case, there are several symmetries to consider. Clearly there is the invariance $\mathsf{O}(n_{\boldsymbol{y}})$ for $\boldsymbol{y}$, and $\mathsf{O}(n_{\boldsymbol{u}})$ for $\boldsymbol{u}$. These group actions act independently on observations of commands.

It is possible to find error functions which have other symmetries. For example, the error function

$$\int \log(\|\boldsymbol{y}\|_2) + \alpha \log(\|\boldsymbol{u}\|_2) \,\mathrm{d}t \tag{3.3}$$

is invariant to the joint transformation

$$\boldsymbol{u}\boldsymbol{y} \mapsto \beta\boldsymbol{y}, \qquad \boldsymbol{u} \mapsto \frac{1}{\beta}\boldsymbol{u},$$

for any scalar $\beta > 0$. (This is different from changing α in (3.3), which is a fixed constant). This transformation group is isomorphic to the multiplication group $(\mathbb{R}^+_\circ, \times)$. The total symmetry group for (3.3) is isomorphic to $\mathsf{O}(n_{\boldsymbol{y}}) \times \mathsf{O}(n_{\boldsymbol{u}}) \times (\mathbb{R}^+_\circ, \times)$.

Consider another unusual error function

$$\int \|\dot{\boldsymbol{y}}\|_2 \,\mathrm{d}t.$$

This might make sense in some situations; for example, if $\boldsymbol{y}$ was the output of a camera mounted on a robot, requiring $\dot{\boldsymbol{y}}$ implies requiring that the robot has stopped. With respect to (3.1), this error function is not only invariant to $\mathsf{O}(n_{\boldsymbol{y}})$, but also to a translation of $\boldsymbol{y}$:

$$\boldsymbol{y} \mapsto \boldsymbol{y} + \boldsymbol{v},$$

Table 3.3. Symmetry groups of several error functions for a stabilization task.

	symmetries		
error function	*for* $\mathcal{Y}$	*for* $\mathcal{U}$	*joint*
$\int \|\boldsymbol{y}\|_2 \, \mathrm{d}t$	$\mathsf{O}(n_y)$	$\mathsf{Aut}(\mathcal{U})$	*none*
$\int \|\mathbf{A}\boldsymbol{y}\|_2 \, \mathrm{d}t$	$\mathsf{conj}_{\mathbf{A}}(\mathsf{O}(n_y))$	$\mathsf{Aut}(\mathcal{U})$	*none*
$\int \|\boldsymbol{y}\|_1 \, \mathrm{d}t$	$\mathsf{D}_{\pm}(n_y) \times \mathsf{Perm}(n_y)$	$\mathsf{Aut}(\mathcal{U})$	*none*
$\int \|\boldsymbol{y}\|_\infty \, \mathrm{d}t$	$\mathsf{D}_{\pm}(n_y) \times \mathsf{Perm}(n_y)$	$\mathsf{Aut}(\mathcal{U})$	*none*
$\int \|\dot{\boldsymbol{y}}\|_2 \, \mathrm{d}t$	$\mathsf{O}(n_y) \times \mathsf{T}(n_y)$	$\mathsf{Aut}(\mathcal{U})$	*none*
$\int \|\boldsymbol{y}\|_2 + \alpha \|\boldsymbol{u}\|_2 \, \mathrm{d}t$	$\mathsf{O}(n_y)$	$\mathsf{O}(n_u)$	*none*
$\int \log(\|\boldsymbol{y}\|_2) + \alpha \log(\|\boldsymbol{u}\|_2) \, \mathrm{d}t$	$\mathsf{O}(n_y)$	$\mathsf{O}(n_u)$	$(\mathbb{R}^+_\circ, \times)$

Each error function (first column) has associated its groups of symmetries (second column), and consequently a set of semantic assumptions. For all of these it is assumed that $\mathcal{Y} = \mathbb{R}^{n_y}$ and $\mathcal{U} = \mathbb{R}^{n_u}$. See Lemma D.5, Lemma D.16, Lemma D.17 for some supporting results.

for a fixed vector $\boldsymbol{v} \in \mathbb{R}^n$. This is the action of the translation group $\mathsf{T}(n_y)$ (Definition D.3).

These results are summarized in Table 3.3.

Like we did with the similarity measures, the partial order of the symmetry groups (Figure 3.2a) induces a partial order on the error functions (Figure 3.2b).

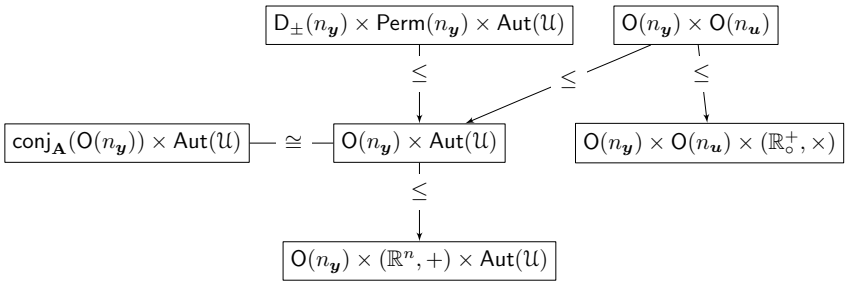
Also we can backtrack the symmetries to some *semantic* assumption. The invariance groups corresponding to the various norms of $\|\boldsymbol{y}\|_p$, $p \in \{1, 2, \infty\}$ are $\mathsf{O}(n_y)$ or its subgroup $\mathsf{D}_{\pm}(n_y) \times \mathsf{Perm}(n_y)$. Their actions all leave the point $\boldsymbol{y} = \mathbf{0}$ fixed. This means that, when formulating these error functions, we have already chosen $\boldsymbol{y} = \mathbf{0}$ as a "special" point. The error function $\int \|\dot{\boldsymbol{y}}\|_2 \, \mathrm{d}t$, instead, does not make this assumption. In fact, its symmetry group is $\mathsf{O}(n_y) \times \mathsf{T}(n_y)$, where the action of $\mathsf{T}(n_y)$ corresponds to shifting the origin.

The differences can be summarized as follows:

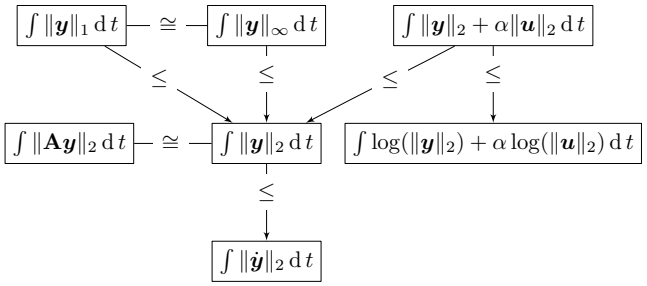
- Using $\int \|\dot{\boldsymbol{y}}\|_2 \, \mathrm{d}t$ assumes that we "know" the metric used in the vector space, and that it is radially symmetric.
- Using $\int \|\boldsymbol{y}\|_2 \, \mathrm{d}t$ assumes that we "know" the metric, and also the origin of the vector space.

- Using $\int \|\boldsymbol{y}\|_\infty \, \mathrm{d}t$ or $\int \|\boldsymbol{y}\|_1 \, \mathrm{d}t$ assumes that we "know" the metric, the origin, and that a particular choice of orthogonal axes.

In summary, in a bootstrapping scenario we must be careful also about defining the task. For example, if we choose an error function, and then we derive an agent optimal with respect to that particular error function, the agent will be invariant only to the symmetries of that particular error function. In particular, it will suffer from all the semantic assumptions that the particular error function implies.



(a) Partial orders on groups given by the subgroup relation.



(b) Induced partial order on the error functions.

Figure 3.2. A partial order on error functions can be defined by using the partial order of their symmetry groups.

3.10. What Comes Next

We are tired. The stranger left a long time ago, probably looking for easier. What can we conclude at the end of this night?

One firm conclusion that we have is that using groups makes many of the concepts more precise, and so it is worth learning some basic notions about them, perhaps after a good night's sleep. Tomorrow, we should also describe better these black boxes (Chapter 4) and give a formal definition of bootstrapping agents (Chapter 5). The last discussion about tasks is troubling as well. What tasks can we think for bootstrapping agents (Chapter 8)?

As we make our way to our hotel rooms, we talk about things more on the horizon. For example, we only really care about embodied agents which have a robotic body. Can we design agents that work for all robots (Part 2)? And how can we design more complicated agents from simpler ones (Part 3)?

As you close your eyes, after a very long day, you definitely think that one PhD would not be enough to solve this bootstrapping problem, but it is worth starting anyway…

CHAPTER 4

Black Boxes and Representation Nuisances

This chapter gives a definition of "black box" system that makes minimal assumptions on the internal representation, and just assumes a causal relation between input and output. The definition of black box is quite abstract, but it allows to treat different objects under the same interface. For example, in addition to the "world" being a black box systems, also the agent's "behavior" is a black box, as well as what will be called "representation nuisances."

Table 4.1. Symbols used in this chapter

Preliminaries		
$\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$		Generic sets for input and output signals.
$\mathsf{Sequences}(\mathcal{A})$		Sequences of all lengths on $\mathcal{A}$.
$\mathsf{ProbMeasures}(\mathcal{A})$		All probability measures on the set $\mathcal{A}$.
$\mathsf{StocProcesses}(\mathcal{A})$		All stochastic processes on the set $\mathcal{A}$.
Black box systems		
$\mathcal{D}(\mathcal{B};\mathcal{A})$		Systems with input in $\mathcal{A}$ and output in $\mathcal{B}$.
$\mathsf{Loop}(\boldsymbol{D})$		System obtained by closing the loop around the system $\boldsymbol{D}$.
Special systems (see also Figure E.4)		
$\mathsf{IdSys}_{\mathcal{A}}$	$\subset \mathcal{D}(\mathcal{A};\mathcal{A})$	The identity system on $\mathcal{A}$.
$\mathbb{III}$	$\subset \mathcal{D}(\mathcal{A};\mathcal{A})$	Accumulator system.
Δ	$\subset \mathcal{D}(\mathcal{A};\mathcal{A})$	Unit delay system.
$\mathcal{D}_{\mathsf{inst}}(\mathcal{B};\mathcal{A})$	$\subset \mathcal{D}(\mathcal{B};\mathcal{A})$	Instantaneous systems.
$\mathcal{D}_{\mathsf{fm}}(\mathcal{B};\mathcal{A})$	$\subset \mathcal{D}(\mathcal{B};\mathcal{A})$	Systems with finite memory.
$\mathcal{D}_{\mathsf{det}}(\mathcal{B};\mathcal{A})$	$\subset \mathcal{D}(\mathcal{B};\mathcal{A})$	Deterministic systems.
Representation nuisances		
$\boldsymbol{D}^R$		Right-inverse of the system $\boldsymbol{D}$.
$\boldsymbol{D}^L$		Left-inverse of the system $\boldsymbol{D}$.
$\mathcal{D}^\star(\mathcal{A})$	$\subset \mathcal{D}_{\mathsf{det}}(\mathcal{A};\mathcal{A})$	Systems that are left- and right-invertible.
$\mathcal{D}^\star_{\mathsf{inst}}(\mathcal{A})$	$\leq \mathcal{D}^\star(\mathcal{A})$	Systems that are instantaneously invertible.
$\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{A})$	$\leq \mathcal{D}^\star(\mathcal{A})$	Systems that are invertible and with finite memory.
$\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$	$= \mathcal{D}^\star(\mathcal{Y}) \times \mathcal{D}^\star(\mathcal{U})$	"Representation nuisances" of observations and commands.
$\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$	$= \mathcal{D}(\mathcal{Y};\mathcal{U})/\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$	Systems up to representation.

4.1. Preliminaries

Curly symbols ($\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$) are used for sets. Bold symbols ($\boldsymbol{a}, \boldsymbol{b}, \dots$) denote sequences, and indexing start at 0:

$$\boldsymbol{a} = \langle a_0, a_1, a_2, \dots \rangle \in \mathcal{A}^{\mathbb{N}}.$$

The notation "$\boldsymbol{a}_{:k}$" denotes subsequences of $\boldsymbol{a}$ up to and including the k-th term: $\boldsymbol{a}_{:k} = \langle a_0, \dots, a_k \rangle$. $\mathsf{Sequences}(\mathcal{A})$ is the set of all sequences of all lengths of elements in $\mathcal{A}$, including the empty sequence:

$$\mathsf{Sequences}(\mathcal{A}) = \emptyset \cup \mathcal{A} \cup \mathcal{A}^2 \cup \mathcal{A}^3 \cup \dots .$$

$\mathsf{ProbMeasures}(\mathcal{A})$ is the set of probability measures on $\mathcal{A}$ (Definition B.1). "$x \sim \mu$" means that x is a random variable and μ is its probability distribution. $\mathsf{Conditional}(\mathcal{B}; \mathcal{A})$ is the set of conditional distributions from $\mathcal{A}$ to $\mathcal{B}$ (Definition B.3) A stochastic process is a particular type of probability distribution on infinite sequences (Definition B.7). $\mathsf{StocProcesses}(\mathcal{A})$ is the set of all stochastic processes with values in $\mathcal{A}$.

4.2. Describing Black Boxes

DEFINITION 4.1 (Black box system). Given an input space $\mathcal{A}$ and an output space $\mathcal{B}$, a *black-box* system $\boldsymbol{D}$ is a function that assigns a probability distribution for the next output as a function of the previous history of input and output:

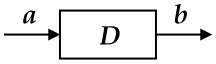
$$\boldsymbol{D} : \mathsf{Sequences}(\mathcal{B} \times \mathcal{A}) \to \mathsf{ProbMeasures}(\mathcal{B}), \tag{4.1}$$

such that the distribution of the output $\boldsymbol{b}_k$, after the input $\boldsymbol{a}_{:k} = \langle a_0, \ldots, a_k \rangle$ has been given, and the past output $\boldsymbol{b}_{:k-1} = \langle b_0, \ldots, b_{k-1} \rangle$ has been decided, is given by $\boldsymbol{D}(\boldsymbol{b}_{:k-1}, \boldsymbol{a}_{:k})$:

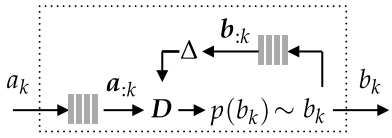
$$b_k \sim \boldsymbol{D}(\boldsymbol{b}_{:k-1}, \boldsymbol{a}_{:k}). \tag{4.2}$$

(See Figure 4.1b.)

$\mathcal{D}(\mathcal{B}; \mathcal{A})$ denotes the set of all black box systems with output in $\mathcal{B}$ and input in $\mathcal{A}$.



(a) Graphical notation for a black box systems $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ between two signals $\boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A})$ and $\boldsymbol{b} \in \mathsf{Sequences}(\mathcal{B})$.



(b) Realization of a black-box as a map $\boldsymbol{D} : \mathsf{Sequences}(\mathcal{B} \times \mathcal{A}) \to \mathsf{ProbMeasures}(\mathcal{B})$ that maps histories to the probability distribution of the next output. The symbol ||| refers to a buffer accumulating previous values (Definition 4.22) and Δ is a unit delay (Definition 4.21).

Figure 4.1. The "black boxes" described in this chapter are possibly stateful systems, but the internal state is not modeled explicitly. Rather, the system is defined only by the recursive distribution of the observations given the history of observations and commands.

REMARK 4.2. (*Black box systems include instantaneous relations.*) The output b_k depends on the input up to instant k included. While an alternative definition would have been $b_{k+1} \sim \boldsymbol{D}(\boldsymbol{b}_{:k}, \boldsymbol{a}_{:k})$, it is useful to include in this class of systems also instantaneous transformations of the input, in which b_k is a function of a_k.

REMARK 4.3. (*Conventions are compatible with left composition.*) In the definition "$\mathcal{D}(\mathcal{B};\mathcal{A})$", the set $\mathcal{B}$ is the output space, and $\mathcal{A}$ is the input space. This will prove to be the most coherent with the definition of series as left composition, where $\boldsymbol{ED} \in \mathcal{D}(\mathcal{C};\mathcal{A})$ indicates the series of $\boldsymbol{E} \in \mathcal{D}(\mathcal{C};\mathcal{B})$ following $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$.

REMARK 4.4. (*Priors are explicit.*) This formalization includes *priors* on the state as an integral part of the system; that is, two systems $\boldsymbol{D}$ and $\boldsymbol{D}'$ which have the same dynamics but different priors are considered different points of the space $\mathcal{D}(\mathcal{B};\mathcal{A})$. To see this, note that the domain of the maps $\boldsymbol{D}$ is $\mathsf{Sequences}(\mathcal{B} \times \mathcal{A})$, which includes sequences of all lengths. The output of the systems b_0 and b'_0 for $k = 0$ is a function of only the input a_0:

$$b_0 b'_0 \sim \boldsymbol{D}(a_0), \qquad \sim \boldsymbol{D}'(a_0).$$

The formalization forces the output b_0 to be well defined even if one just started interacting with the system. In other words, it coincides with some idea of "prior" for the system.

REMARK 4.5. (*States are implicit.*) The formalization does not mention an internal state space explicitly. The state is implicit in the fact that the map depends on the complete history of commands and observations. This is, of course, the traditional way to define a state from an input-output relation, as a set of equivalence classes between sequences [24]. This view is very close to the epistemological perspective of an agent that starts with zero information about the world: the knowledge of the state space is inaccessible, and uncertainty has a dominant role.

REMARK 4.6. (*Continuous-time modeling*) This formalization can be extended to continuous-time. This allows extending most of the definitions in this chapter to be valid for the continuous time setting.

First, replace the discrete $\mathsf{Sequences}(\mathcal{B} \times \mathcal{A})$ with the set of continuous sequences $\mathsf{ContSequences}(\mathcal{B} \times \mathcal{A})$ (Definition A.30). The dependence of (4.2) on $\boldsymbol{a}_{:k}$ is replaced by the dependence on $\boldsymbol{a}_{[0,t]}$ (t included), and $\boldsymbol{b}_{:k}$ is replaced by $\boldsymbol{b}_{[0,t)}$ (t excluded).

DEFINITION 4.7 (Black box system, continuous time definition.). Given an input space $\mathcal{A}$ and an output space $\mathcal{B}$, a *black-box* system $\boldsymbol{D}$ is a function that assigns a probability distribution for the next output as a function of the previous history of input and output:

$$\boldsymbol{D} : \mathsf{ContSequences}(\mathcal{B} \times \mathcal{A}) \to \mathsf{ProbMeasures}(\mathcal{B}) \tag{4.3}$$

such that the distribution of the output $\boldsymbol{b}_t$, after the inputs $\boldsymbol{a}_{[0,t]}$ have been given, and the past output $\boldsymbol{b}_{[0,t)}$ have been observed, is given by $\boldsymbol{D}(\boldsymbol{b}_{[0,t)}, \boldsymbol{a}_{[0,t]})$:

$$b_t \sim \boldsymbol{D}(\boldsymbol{b}_{[0,t)}, \boldsymbol{a}_{[0,t]}).$$

EXAMPLE 4.8 (Instantaneous transformations). Any map $d : \mathcal{A} \to \mathcal{B}$ induces an instantaneous, deterministic black box $\boldsymbol{D}_d \in \mathcal{D}(\mathcal{B}; \mathcal{A})$ defined by $b_k \sim \delta_{d(a_k)}$, where $\delta_{d(a_k)}$ is an impulse centered at $d(a_k)$.

EXAMPLE 4.9 (Deterministic systems with hidden state space, and fixed initial state). A discrete-time dynamical system is usually defined as a tuple $\langle \mathcal{A}, \mathcal{B}, \mathcal{X}, f, h, \rangle$ with $f : \mathcal{X} \times \mathcal{A} \to \mathcal{X}$ and $h : \mathcal{X} \times \mathcal{A} \to \mathcal{B}$, such that

$$\begin{aligned} x_k &= f(x_{k-1}, a_k), \\ b_k &= h(x_k, a_k). \end{aligned}$$

To be converted to the representation we use, it is necessary to specify also the initial state (see Remark 4.4). Let $x_* \in \mathcal{X}$ be the initial state. The complete dynamics is

$$\begin{aligned} x_0 &= x_*, \\ x_k &= f(x_{k-1}, a_k), \qquad \text{for } k \geq 1, \\ b_k &= h(x_k, a_k), \end{aligned}$$

which can be put in the form (4.1) with the following technical construction.

This kind of deterministic system induces a deterministic map $F : \mathsf{Sequences}(\mathcal{A}) \to \mathcal{B}$, constructed as follows. First define the evolution of the state as a map $G : \mathsf{Sequences}(\mathcal{A}) \to \mathcal{X}$ as

$$\begin{aligned} G_{f,x_*}(a_0) &= x_*, \\ G_{f,x_*}(a_{:k}) &= f(G_{f,x_*}(a_{:k-1}), a_k)). \end{aligned}$$

Then the map $F : \mathsf{Sequences}(\mathcal{A}) \to \mathcal{B}$ is defined recursively as follows:

$$F_{\langle \mathcal{A},\mathcal{B},\mathcal{X},f,h,x_* \rangle}(a_{:k}) \quad = \quad h(G(a_{:k}), a_k).$$

The induced black box is given by an impulse distribution centered at $F_{\langle \mathcal{A},\mathcal{B},\mathcal{X},f,h,x_* \rangle}(a_{:k})$:

$$\boldsymbol{D}_{\langle \mathcal{A},\mathcal{B},\mathcal{X},f,h,x_* \rangle}(\boldsymbol{b}_{:k-1}, \boldsymbol{a}_{:k}) = \delta_{F_{\langle \mathcal{A},\mathcal{B},\mathcal{X},f,h,x_* \rangle}(\boldsymbol{a}_{:k})}.$$

EXAMPLE 4.10 (Stochastic systems with state space). In the case of a discrete-time stochastic system with observations history $y_{:k-1}$ and command history $u_{:k}$, the measure $\boldsymbol{D}(\langle y_{:k-1}, u_{:k} \rangle)$ is simply the posterior distribution $p(y_t|y_{0:t-1}, u_{0:t})$, which can be written as a function of the observation model $p(y|x)$, the transition model $p(x_t|x_{t-1}, u_t)$, and the prior $p(x_0)$.

4.3. Series

If two systems $\boldsymbol{D}$ and $\boldsymbol{E}$ have compatible input and output spaces, then they can be composed in a series $\boldsymbol{ED}$, If $\boldsymbol{E}$ follows $\boldsymbol{D}$. If $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ and $\boldsymbol{E} \in \mathcal{D}(\mathcal{C};\mathcal{B})$, then $\boldsymbol{ED} \in \mathcal{D}(\mathcal{C};\mathcal{A})$. The notation "output; input" is that it is compatible with the definition of series:

$$\underbrace{\boldsymbol{E}}_{\mathcal{D}(\mathcal{C};\mathcal{B})} \underbrace{\boldsymbol{D}}_{\mathcal{D}(\mathcal{B};\mathcal{A})} \in \mathcal{D}(\mathcal{C};\mathcal{A}).$$

The only problem with this choice is that usually diagrams are drawn with signals going left to right (Figure 4.1).

REMARK 4.11. ($\mathcal{D}(\mathcal{A};\mathcal{A})$ *is a monoid.*) In the set $\mathcal{D}(\mathcal{A};\mathcal{A})$ of systems with same input

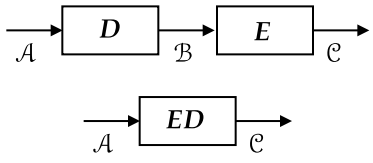


Figure 4.1. The series of two systems $\boldsymbol{D}$ and $\boldsymbol{E}$, if $\boldsymbol{E}$ follows $\boldsymbol{D}$, is written as $\boldsymbol{ED}$. This clashes with the convention of drawing diagrams with signals flowing left to right.

and output space $\mathcal{A}$, the series is well defined on all elements of the set. Moreover, there is a an identity $\mathsf{IdSys}_{\mathcal{A}}$, corresponding to the identity system for which the output is equal to the input. The identity satisfies $\boldsymbol{D}\,\mathsf{IdSys}_{\mathcal{A}} = \boldsymbol{D}\,\mathsf{IdSys}_{\mathcal{A}} = \boldsymbol{D}$. These properties make $\mathcal{D}(\mathcal{A};\mathcal{A})$ a monoid.

4.4. Loops

An important operation is closing the loop around systems. In discrete time, the definition is easy.*

DEFINITION 4.12 (Closing the loop). Consider a system $\boldsymbol{D} \in \mathcal{D}(\mathcal{A};\mathcal{A})$. The closed-loop system $\mathsf{Loop}(\boldsymbol{D})$ is obtained by adding a unit delay to the output and connect it to its own input (Figure 4.1a). This definition is well posed because all signals are well defined at all times.

There might be situations where the loop needs to be closed only around some of the ports. To be rigorous, the Loop function should be parametrized by a list of which output signals map to which input signals, but this is not done explicitly.

The type of $\mathsf{Loop}(\boldsymbol{D})$ depends on the type of $\boldsymbol{D}$. There are three interesting cases:

(1) If the system $\boldsymbol{D} \in \mathcal{D}(\mathcal{A};\mathcal{A})$ has input and output in the same space $\mathcal{A}$, then the result belongs to $\mathcal{D}(\mathcal{A};\emptyset)$, which means it is an autonomous system. One consequence of Definition 4.1 is that an autonomous system is the same thing as a

monoid: A semigroup with an identity element. See Definition C.2.

*In continuous time, no delay is added, which might result in algebraic constraints.

unit delay: A dynamical system whose output is the same as the input, delayed by one time step. See Definition 4.21.

autonomous dynamical system: A system with no input.

stochastic process:

$$\mathcal{D}(\mathcal{A};\emptyset) \cong \mathsf{StocProcesses}(\mathcal{A}).$$

Therefore, in this case the Loop function maps a black box system to a stochastic process:

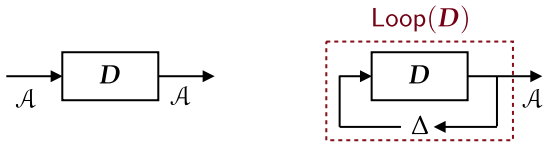
$$\mathsf{Loop} : \mathcal{D}(\mathcal{A};\mathcal{A}) \to \mathsf{StocProcesses}(\mathcal{A}).$$

(2) If the system $\boldsymbol{D} \in \mathcal{D}(\mathcal{A} \times \mathcal{B};\mathcal{A})$ has more output signals than input signals, then the result is a stochastic process on the output signals (Figure 4.1b):

$$\mathsf{Loop} : \mathcal{D}(\mathcal{A} \times \mathcal{B};\mathcal{A}) \to \mathsf{StocProcesses}(\mathcal{A} \times \mathcal{B}).$$

(3) If there is an additional input, the result of Loop is again a black box system, because its output still depends on the input (Figure 4.1c):

$$\mathsf{Loop} : \mathcal{D}(\mathcal{A};\mathcal{A} \times \mathcal{C}) \to \mathcal{D}(\mathcal{A};\mathcal{C}).$$



$$\mathsf{Loop} : \mathcal{D}(\mathcal{A};\mathcal{A}) \to \mathsf{StocProcesses}(\mathcal{A})$$

(a) Closing the loop around a system in $\mathcal{D}(\mathcal{A};\mathcal{A})$ gives a stochastic process on $\mathcal{A}$.

$$\mathsf{Loop} : \mathcal{D}(\mathcal{A} \times \mathcal{B};\mathcal{A}) \to \mathsf{StocProcesses}(\mathcal{A} \times \mathcal{B})$$

(b) Closing the loop around a system in $\mathcal{D}(\mathcal{A} \times \mathcal{B};\mathcal{A})$ gives a stochastic process on $\mathcal{A} \times \mathcal{B}$.

$$\mathsf{Loop} : \mathcal{D}(\mathcal{A};\mathcal{A} \times \mathcal{C}) \to \mathcal{D}(\mathcal{A};\mathcal{C})$$

(c) Closing the loop around a system in $\mathcal{D}(\mathcal{A};\mathcal{A} \times \mathcal{C})$ gives a system in $\mathcal{D}(\mathcal{A};\mathcal{C})$.

Figure 4.1. The Loop operation (Definition 4.12) closes the loop around a system. The result is either another system or just a stochastic process, depending on the type of input and output of the original system.

4.5. Statistics of a Black Box

The following are some technical constructions needed later.

The AllOutputs function gives all possible output sequences of a system.

DEFINITION 4.13 (AllOutputs). Define the map

$$\mathsf{AllOutputs} : \mathcal{D}(\mathcal{B};\mathcal{A}) \to \mathsf{powerset}(\mathsf{Sequences}(\mathcal{B})), \tag{4.1}$$

such that $\mathsf{AllOutputs}(\boldsymbol{D})$ is the set of all possible output sequences generated by the system $\boldsymbol{D}$:

$$\mathsf{AllOutputs}(\boldsymbol{D}) = \{\boldsymbol{b} \in \mathsf{Sequences}(\mathcal{B}) \mid \exists \boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A}) : \ (\boldsymbol{D}\boldsymbol{a})(\boldsymbol{b}) \neq 0\}.$$

REMARK 4.14. The shortcut notation "$\rightrightarrows \mathcal{X}$" is used to denote a function whose codomain is the powerset of $\mathcal{X}$. Using this notation, the definition 4.1 is rewritten as

$$\mathsf{AllOutputs} : \mathcal{D}(\mathcal{B};\mathcal{A}) \rightrightarrows \mathsf{Sequences}(\mathcal{B}).$$

To interpret the condition, note that $\boldsymbol{D}\boldsymbol{a} \in \mathsf{StocProcesses}(\mathcal{B})$, and that a stochastic process is a probability measure on sequences (Definition B.7). Hence the condition $(\boldsymbol{D}\boldsymbol{a})(\boldsymbol{b}) \neq 0$ means that the probability of observing the sequence $\boldsymbol{b}$ is nonzero.

LEMMA 4.15 (Properties of AllOutputs).

(1) *The identity system produces all sequences:*

$$\mathsf{AllOutputs}(\mathsf{IdSys}_{\mathcal{A}}) = \mathsf{Sequences}(\mathcal{A}).$$

(2) *Prefiltering a system* $\boldsymbol{F}$ *with another system* $\boldsymbol{E}$ *does not enlarge the set of sequences that can be produced:*

$$\mathsf{AllOutputs}(\boldsymbol{F}\,\boldsymbol{E}) \subseteq \mathsf{AllOutputs}(\boldsymbol{F}).$$

The AllOutcomes function gives all possible results, when the system is in closed loop with another system.

DEFINITION 4.16 (AllOutcomes). For a system $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$, define the set $\mathsf{AllOutcomes}(\boldsymbol{D}) \subset \mathsf{StocProcesses}(\mathcal{B}\times\mathcal{A})$ as the set of all possible observed statistics of the closed loop system, as one tries pairing $\boldsymbol{D}$ with all possible systems $\boldsymbol{F} \in \mathcal{D}(\mathcal{A};\mathcal{B})$:

$$\mathsf{AllOutcomes}(\boldsymbol{D}) = \bigcup_{\boldsymbol{F}\in\mathcal{D}(\mathcal{A};\mathcal{B})} \mathsf{Loop}(\boldsymbol{F}\,\boldsymbol{D}).$$

4.6. Special Classes of Systems

DEFINITION 4.17 (Deterministic system). A system $\boldsymbol{D}$ is *deterministic* if, for every length k, $\boldsymbol{b} = \boldsymbol{D}\boldsymbol{a}$ implies that $\boldsymbol{b}_{:k}$ is a deterministic function of $\boldsymbol{a}_{:k}$.

$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$ is the set of all deterministic systems.

A *memoryless* system has the same representation power as a conditional distribution (Definition B.3).

DEFINITION 4.18 (Memoryless (or instantaneous) system). In a memoryless system the probability distribution of the output $\boldsymbol{b}_k$ depends only on $\boldsymbol{a}_k$.

$\mathcal{D}_{\mathrm{inst}}(\mathcal{B};\mathcal{A})$ is the set of all memoryless systems.

DEFINITION 4.19 (Finite-memory system). In a system with finite memory Δ, the output $\boldsymbol{b}_k$ depends only on $\boldsymbol{a}_{k-\Delta:k}$ and $\boldsymbol{b}_{k-\Delta:k-1}$.

$\mathcal{D}_{\mathsf{fm}}(\mathcal{B};\mathcal{A})$ is the set of all finite-memory systems.

4.6.1. *Special systems*

Finally, for future reference, these are some special systems that have already been used in the definition.

DEFINITION 4.20 (Identity system $\mathsf{IdSys}_{\mathcal{A}}$). The identity on the monoid $\mathcal{D}(\mathcal{A};\mathcal{A})$ is called $\mathsf{IdSys}_{\mathcal{A}}$.

DEFINITION 4.21 (Delay system Δ). For any space $\mathcal{A}$, the delay system $\Delta \in \mathcal{D}(\mathcal{A};\mathcal{A})$ implements the relation $b_k = a_{k-1}$.

DEFINITION 4.22 (Accumulator system $\mathbb{III}$). For any space $\mathcal{A}$, the accumulator system $\mathbb{III} \in \mathcal{D}(\mathsf{Sequences}(\mathcal{A});\mathcal{A})$ implements the relation $b_k = \langle a_0, \ldots, a_k \rangle$.

4.7. Composition Rules

The series of two systems is another system: if $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ and $\boldsymbol{E} \in \mathcal{D}(\mathcal{C};\mathcal{B})$, then $\boldsymbol{E}\,\boldsymbol{D} \in \mathcal{D}(\mathcal{C};\mathcal{A})$. But we will be quite liberal in applying systems to other objects, such as sequences and stochastic processes. Table 4.2 gives the type of the expression $y\,x$, as a function of the type of x and y.

Table 4.2. Composition rules for different kinds of objects

type of y	*type of y*	*type of x*
$\mathcal{B}$	$\mathsf{Functions}(\mathcal{B};\mathcal{A})$	$\mathcal{A}$
$\mathsf{ProbMeasures}(\mathcal{B})$	$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$	
$\mathsf{ProbMeasures}(\mathcal{B})$	$\mathsf{Functions}(\mathcal{B};\mathcal{A})$	$\mathsf{ProbMeasures}(\mathcal{A})$
$\mathsf{ProbMeasures}(\mathcal{B})$	$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$	
$\mathsf{Sequences}(\mathcal{B})$	$\mathsf{Functions}(\mathcal{B};\mathcal{A})$	$\mathsf{Sequences}(\mathcal{A})$
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$	
$\mathsf{Sequences}(\mathcal{B})$	$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$	
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathcal{D}(\mathcal{B};\mathcal{A})$	
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathsf{Functions}(\mathcal{B};\mathcal{A})$	$\mathsf{StocProcesses}(\mathcal{A})$
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$	
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$	
$\mathsf{StocProcesses}(\mathcal{B})$	$\mathcal{D}(\mathcal{B};\mathcal{A})$	
$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$	$\mathsf{Functions}(\mathcal{C};\mathcal{B})$	$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathsf{Conditional}(\mathcal{C};\mathcal{B})$	
$\mathcal{D}_{\mathrm{det}}(\mathcal{B};\mathcal{A})$	$\mathcal{D}_{\mathrm{det}}(\mathcal{C};\mathcal{B})$	
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathcal{D}(\mathcal{C};\mathcal{B})$	
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathsf{Functions}(\mathcal{C};\mathcal{B})$	$\mathcal{D}(\mathcal{B};\mathcal{A})$
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathsf{Conditional}(\mathcal{C};\mathcal{B})$	
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathcal{D}_{\mathrm{det}}(\mathcal{C};\mathcal{B})$	
$\mathcal{D}(\mathcal{C};\mathcal{A})$	$\mathcal{D}(\mathcal{C};\mathcal{B})$	

4.8. Inverting Systems

Systems that are "invertible" are of particular interest for our goals.

DEFINITION 4.23 (Right- and left-invertible systems). A system $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ is *right-invertible* if there exists a system $\boldsymbol{D}^R \in \mathcal{D}(\mathcal{A};\mathcal{B})$ such that the series of the two systems is equivalent to the identity system: $\boldsymbol{D}\boldsymbol{D}^R = \mathsf{IdSys}_{\mathcal{A}}$.

Equivalently, $\boldsymbol{D} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ is *left-invertible* if there exists a system $\boldsymbol{D}^L \in \mathcal{D}(\mathcal{A};\mathcal{B})$ such that that $\boldsymbol{D}^L\boldsymbol{D} = \mathsf{IdSys}_{\mathcal{B}}$.

Right-invertible does not imply left-invertible and vice versa.

The left and right inverses are constrained to be causal because all elements of $\mathcal{D}(\mathcal{B};\mathcal{A})$ are causal systems. For example, the delay Δ is not causally invertible.

Left-invertible implies deterministic only on finite-dimensional spaces.

EXAMPLE 4.24 (Left-invertible does not imply deterministic for infinite-dimensional spaces). Here is a construction of a left-invertible system $\boldsymbol{D} \in \mathcal{D}(\mathcal{A};\mathcal{A})$ which is not deterministic. Let the set $\mathcal{A}$ be $\{\square,\boxdot\}^{\mathbb{N}}$, which represents the set of infinite binary strings (Definition A.1). The system $\boldsymbol{D}$ is instantaneous, and thus is a map from strings to probabilities over strings:

$$\boldsymbol{D} : \{\square,\boxdot\}^{\mathbb{N}} \to \mathsf{ProbMeasures}(\{\square,\boxdot\}^{\mathbb{N}}).$$

This map shifts each bit of the infinite sequence one step to the right , and adds to the first place one bit z, which is either $\square$ or $\boxdot$ with uniform probability:

$$\boldsymbol{D}(\langle x_1, x_2, x_3, \dots \rangle) = \langle z, x_1, x_2, x_3, \dots \rangle\,, \quad z \sim \mathsf{Uniform}(\{\square,\boxdot\}).$$

This system is left-invertible, having the left inverse $\boldsymbol{D}^L$, which shifts the infinite sequence one step to the left (thus dropping the first bit):

$$\begin{aligned} \boldsymbol{D}^L : \{\square,\boxdot\}^{\mathbb{N}} &\to \{\square,\boxdot\}^{\mathbb{N}}, \\ \langle x_1, x_2, x_3, \dots \rangle &\mapsto \langle x_2, x_3, \dots \rangle\,. \end{aligned}$$

The "trick" that makes this example possible is that in an infinite-dimensional space there is an infinite amount of space to fit an infinite amount of extra payload. This example corresponds exactly to Hilbert's infinite hotel [54]: there is always space for an extra guest (in this case, the guest is random bit z). See Example A.22 for the same principle written in terms of injective/surjective maps.

Also right-invertible systems might be non deterministic on infinite-dimensional spaces.

EXAMPLE 4.25 (Right-invertible does not imply deterministic for infinite-dimensional spaces). Here is a construction of a right-invertible system $\boldsymbol{D} \in \mathcal{D}(\mathcal{A};\mathcal{A})$ which is not

deterministic. The setup is the same as Example 4.24, using the space of infinite strings $\mathcal{A} = \{\square, \boxdot\}^{\mathbb{N}}$.

The instantaneous system $\boldsymbol{D}$ operates conditionally on the value of the first bit read. If the bit is $\square$, then the output is the sequence shifted to the left. Otherwise, the output is completely random.

$$\boldsymbol{D} : \{\square, \boxdot\}^{\mathbb{N}} \quad \to \quad \mathsf{ProbMeasures}(\{\square, \boxdot\}^{\mathbb{N}}),$$

$$\boldsymbol{D}(\langle x_1, x_2, x_3, \dots \rangle) \quad = \quad \begin{cases} \langle x_2, x_3, \dots \rangle & \text{if } x_1 = \square, \\ \langle z_1, z_2, z_3, \dots \rangle & \text{if } x_1 = \boxdot, \end{cases} \tag{4.1}$$

$$z_i \sim \mathsf{Uniform}(\{\square, \boxdot\}). \tag{4.2}$$

The right inverse $\boldsymbol{D}^R$ is

$$\boldsymbol{D}^R : \{\square, \boxdot\}^{\mathbb{N}} \quad \to \quad \{\square, \boxdot\}^{\mathbb{N}},$$

$$\langle x_1, x_2, x_3, \dots \rangle \quad \mapsto \quad \langle \square, x_1, x_2, x_3, \dots \rangle .$$

The "trick" is that with infinite payload to exploit, it is possible to encode in the signal a bit that switches the randomness on and off.

Having established that left-invertible and right-invertible alone do not imply deterministic, the next natural question is whether a system which is being left- and right-invertible is necessarily deterministic. The answer is yes (Lemma 4.28), but to arrive at such result a number of intermediate results are needed.

In general, the series $\boldsymbol{FE}$ being deterministic does not imply that $\boldsymbol{F}$ is deterministic, as shown in Example 4.25. However, $\boldsymbol{F}$ must be deterministic on at least part of the input

sequences, namely those produced by $\boldsymbol{E}$.

LEMMA 4.26. *Let $\boldsymbol{E} \in \mathcal{D}(\mathcal{B};\mathcal{A})$ and $\boldsymbol{F} \in \mathcal{D}(\mathcal{C};\mathcal{B})$. If $\boldsymbol{FE}$ is deterministic, then $\boldsymbol{F}$ is deterministic on the sequences contained in* $\mathsf{AllOutputs}(\boldsymbol{E})$.

PROOF. Choose any sequence $\boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A})$. Let $\boldsymbol{c}$ be the result of applying $\boldsymbol{FE}$ to $\boldsymbol{a}$. Because $\boldsymbol{FE}$ is deterministic, we know that $\boldsymbol{c}$ is a fixed sequence, and not a stochastic process:

$$\boldsymbol{c} = \boldsymbol{FEa} \in \mathsf{Sequences}(\mathcal{C}).$$

On the other hand, in general, $\boldsymbol{Ea} \in \mathsf{StocProcesses}(\mathcal{B})$. Let $\mathsf{Support}(\boldsymbol{Ea}) \subset \mathsf{Sequences}(\mathcal{B})$ be the set of all sequences that can be produced by the stochastic process $\boldsymbol{Ea}$. For each $\boldsymbol{b} \in \mathsf{Support}(\boldsymbol{Ea})$, $\boldsymbol{c} = \boldsymbol{Fb}$ is a sequence, and hence $\boldsymbol{F}$ is deterministic on $\boldsymbol{b}$. This holds for every $\boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A})$, therefore $\boldsymbol{F}$ is deterministic on the set

$$\bigcup_{\boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A})} \mathsf{Support}(\boldsymbol{Ea}),$$

which is another way to write $\mathsf{AllOutputs}(\boldsymbol{E})$. □

EXAMPLE 4.27. Notice how this condition is verified in (4.1). The series $\boldsymbol{DD}^R$ is deterministic, hence $\boldsymbol{D}$ is deterministic on the sequences produced by $\boldsymbol{D}^R$.

LEMMA 4.28. *If $\boldsymbol{D} \in \mathcal{D}(\mathcal{A};\mathcal{A})$ is left- and right-invertible, then it is deterministic.*

PROOF. If $\boldsymbol{D}^L$ is the left inverse of $\boldsymbol{D}$, then $\boldsymbol{D}^L\boldsymbol{D} = \mathsf{IdSys}_{\mathcal{A}}$. From the second point of Lemma 4.15 it follows that

$$\mathsf{AllOutputs}(\mathsf{IdSys}_{\mathcal{A}}) \subseteq \mathsf{AllOutputs}(\boldsymbol{D}^L),$$

and that $\mathsf{AllOutputs}(\mathsf{IdSys}_{\mathcal{A}}) = \mathsf{Sequences}(\mathcal{A})$. Hence $\mathsf{AllOutputs}(\boldsymbol{D}^L) = \mathsf{Sequences}(\mathcal{A})$. If

a system has both a left and right inverse, then the two coincide, because $\boldsymbol{D}(\mathcal{A};\mathcal{A})$ is a monoid (Lemma C.3). Hence $\boldsymbol{D}^R = \boldsymbol{D}^L$, and

$$\mathsf{AllOutputs}(\boldsymbol{D}^L) = \mathsf{AllOutputs}(\boldsymbol{D}^R) = \mathsf{Sequences}(\mathcal{A}). \tag{4.3}$$

If $\boldsymbol{D}^R$ is the right-inverse of $\boldsymbol{D}$, then $\boldsymbol{D}\boldsymbol{D}^R = \mathsf{IdSys}_{\mathcal{A}}$. From Lemma 4.26, this implies that $\boldsymbol{D}$ is deterministic on $\mathsf{AllOutputs}(\boldsymbol{D}^R)$, which is equal to $\mathsf{Sequences}(\mathcal{A})$ by (4.3). □

4.9. Group Structure on Invertible Systems

We have established that left- and right-invertible systems are deterministic. It is easy to see that there is a group structure on this set.

DEFINITION 4.29 (Group of invertible systems). Systems in $\mathcal{D}(\mathcal{A};\mathcal{A})$ that are left- and right-invertible are deterministic and form a group, denoted $\mathcal{D}^\star(\mathcal{A})$.

PROOF. It was already noted that $\mathcal{D}(\mathcal{A};\mathcal{A})$ is a monoid. Each element in $\mathcal{D}^\star(\mathcal{A})$ has an inverse; because the left-inverse is right-invertible (and vice versa), the inverse belongs in $\mathcal{D}^\star(\mathcal{A})$ as well. Therefore, $\mathcal{D}^\star(\mathcal{A})$ is a group. □

This group will be very useful in the rest of this dissertation. The following are specialized subsets of $\mathcal{D}^\star(\mathcal{A})$.

DEFINITION 4.30 (Instantaneous invertible systems). $\mathcal{D}^\star_{\mathsf{inst}}(\mathcal{A})$ is the subset of $\mathcal{D}^\star(\mathcal{A})$ of systems which are also instantaneous (Definition 4.18).

DEFINITION 4.31 (Finite-memory invertible systems). $\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{A})$ is the subset of $\mathcal{D}^\star(\mathcal{A})$ which also have finite memory (Definition 4.19).

These are both subgroups of $\mathcal{D}^\star(\mathcal{A})$, because the properties of being instantaneous or with finite memory are preserved by composition and inversion.

The class $\mathcal{D}^\star(\mathcal{A})$ includes also causally invertible systems that are not instantaneous.

EXAMPLE 4.32 (Difference encoder/decoder pair). As a simple example, consider a pair of linear systems $\boldsymbol{E}/\boldsymbol{D}$ that work as an encoder/decoder pair.

The encoder $\boldsymbol{E} \in \mathcal{D}^\star(\mathbb{R};\mathbb{R})$ has input $\boldsymbol{a}$, output $\boldsymbol{b}$, and hidden state $\boldsymbol{r}$:

$$\boldsymbol{b} = \boldsymbol{E}(\boldsymbol{a}),$$

$$\begin{cases} r_0 & = 0, \\ r_{k+1} & = a_k, \\ b_k & = a_k - r_k \end{cases}$$

The decoder $\boldsymbol{D} \in \mathcal{D}^\star(\mathbb{R};\mathbb{R})$ has input $\boldsymbol{c}$, output $\boldsymbol{d}$, and hidden state $\boldsymbol{q}$:

$$\boldsymbol{d} = \boldsymbol{D}(\boldsymbol{c}),$$

$$\begin{cases} q_0 & = 0, \\ q_{k+1} & = d_k, \\ d_k & = q_k + c_k. \end{cases}$$

One way to prove that these are inverse of each other is to write the dynamics in the canonical form for discrete-time systems:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k + Du_k. \end{aligned}$$

For the decoder, we obtain the matrices

$$\boldsymbol{D}: \qquad A = 1 \quad B = 1 \quad C = 1 \quad D = 1,$$

and for the decoder, we obtain

$$\boldsymbol{E}: \qquad A = 0 \quad B = 1 \quad C = -1 \quad D = 1.$$

Using the Matlab commands

```
1  E = ss(1,1,1,1,[]); zpk(E)
2  D = ss(0,1,−1,1,[]) zpk(D)
```

we can find the transfer functions are

$$D(z) = \frac{z}{z-1}, \qquad E(z) = \frac{z-1}{z},$$

from which we can see that the two systems are inverse of each other.

EXAMPLE 4.33 (Lossless video encoding). Let $\mathcal{A}$ be the set of bit strings of arbitrary length. We consider two systems on $\mathcal{D}(\mathcal{A};\mathcal{A})$ that perform compression/decompression of a video sequence.

The first system, called encoder, takes as input an image stream image_k, at each time computing packet_k, which contains the incremental information:

$$\text{packet}_k = \text{encoder}(\text{packet}_{:k-1}, \text{image}_k).$$

The packet at time k depends on the previous frames previously transmitted and instantaneously on the image at time k. In video codecs such as MPEG, the incremental information

encodes the deviation of the data image_k from a prior constructed from the previous images, taking into account some model of the video stream (e.g., motion is mostly coherent across large parts of the scene).

The second system, called decoder, does the opposite operation. Given the sequence of encoded packets, it recreates the image sequence. The image at time k depends on the previous images decoded and the incremental information contained in the current packet:

$$\text{image}_k = \text{decoder}(\text{image}_{:k-1}, \text{packet}_k).$$

Note that these two systems are inverses of each other.

4.10. Representation Nuisances

The group of invertible systems has a natural group action (Definition C.35) over the set of all systems. Let $\mathcal{D}(\mathcal{Y};\mathcal{U})$ be the set of all systems with observations in $\mathcal{Y}$ and commands in $\mathcal{U}$. The group of invertible systems $\mathcal{D}^\star(\mathcal{Y})$ acts on $\mathcal{D}(\mathcal{Y};\mathcal{U})$ via series composition. According to the order of the series, this is can be either a left or right group action.

DEFINITION 4.34 (Left group action). For $g \in \mathcal{D}^\star(\mathcal{Y})$ and $\boldsymbol{D} \in \mathcal{D}(\mathcal{Y};\mathcal{U})$, the left group action "$\cdot$" is

$$g \cdot \boldsymbol{D} = g\boldsymbol{D}. \tag{4.1}$$

The group action is the just the series composition of the two systems g and $\boldsymbol{D}$ (Figure 4.1a).

The interpretation is that the observations are transformed in an invertible way. Because $\mathcal{D}^\star(\mathcal{Y})$ is a group, one can find a system $g^{-1} \in \mathcal{D}^\star(\mathcal{Y})$ such that $g^{-1} \cdot (g \cdot \boldsymbol{D}) = \boldsymbol{D}$. Recall that all of these systems are causal. Therefore, one can reconstruct any statistics that could be computed from the observations of the system $\boldsymbol{D}$ from the system $g \cdot \boldsymbol{D}$, provided that we can estimate the element g that acted on the data. For this reason, we call $\mathcal{D}^\star(\mathcal{Y})$

"representation nuisances", in the sense that they change the *representation* of the observations, but not the available information (Figure 4.1b).

Likewise, we can define representation nuisances for the commands as a right action. Take the group $\mathcal{D}^\star(\mathcal{U})$ of invertible systems over the command space $\mathcal{U}$.

DEFINITION 4.35 (Right group action). For $h \in \mathcal{D}^\star(\mathcal{U})$ and $\boldsymbol{D} \in \mathcal{D}(\mathcal{Y};\mathcal{U})$, the left group action "$\cdot$" is

$$\boldsymbol{D} \cdot h = \boldsymbol{D}h. \tag{4.2}$$

Note that there is a difference with respect to (4.1), as it is a composition in the opposite order: first h, then $\boldsymbol{D}$ (Figure 4.1c).

Also in this case the interpretation is that the group action changes only the *representation* of the commands, because it is always possible to find another system $h^{-1} \in \mathcal{D}^\star(\mathcal{U})$ such that $(\boldsymbol{D} \cdot h) \cdot h^{-1} = \boldsymbol{D}$ (Figure 4.1d). This implies that any level of control we had for the original system is still preserved for the perturbed system.

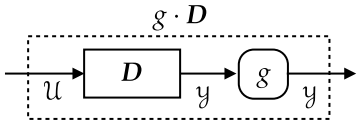
As a further step, we consider the direct product of $\mathcal{D}^\star(\mathcal{U})$ and $\mathcal{D}^\star(\mathcal{Y})$, which is a group that comprises nuisances both on the observations and the commands.

DEFINITION 4.36 (Representation nuisances). The set $\mathcal{D}^\star(\mathcal{Y};\mathcal{U}) = \mathcal{D}^\star(\mathcal{Y}) \times \mathcal{D}^\star(\mathcal{U})$ is a group with the operation

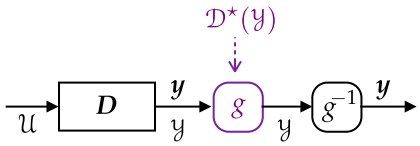
$$\langle g_2, h_2 \rangle \langle g_1, h_1 \rangle = \langle g_2 g_1, h_1 h_2 \rangle \tag{4.3}$$

and it is called the group of *representation nuisances*.

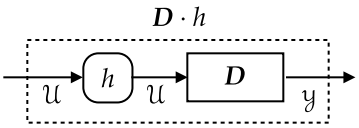
There are a couple of subtle points on how we have defined this group, due to the fact that the order of the operations for elements acting on commands and observations are exchanged. To show that the operation (4.3) makes a respectable group operation, one



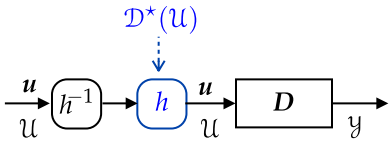
(a) A nuisance acting on the observations…



(b) …can be, in theory, compensated.



(c) A nuisance acting on the commands…



(d) …can be, in theory, *pre-compensated*.

Figure 4.1. The groups $\mathcal{D}^\star(\mathcal{U})$ and $\mathcal{D}^\star(\mathcal{Y})$ are called "representation nuisances" because they can be causally inverted. Therefore, they do not change the intrinsic properties of the system (such as observability and controllability), but just its representation.

needs to check that that it is associative:

$$\begin{aligned}
\langle g_3, h_3\rangle \left(\langle g_2, h_2\rangle \langle g_1, h_1\rangle\right) &= \langle g_3, h_3\rangle \langle g_2 g_1, h_1 h_2\rangle \\
&= \langle g_3 g_2 g_1, h_1 h_2 h_3\rangle \\
&= \langle (g_3 g_2) g_1, h_1 (h_2 h_3)\rangle \\
&= \langle g_3 g_2, h_2 h_3\rangle \langle g_1, h_1\rangle \\
&= \left(\langle g_3, h_3\rangle \langle g_2, h_2\rangle\right) \langle g_1, h_1\rangle .
\end{aligned}$$

Next, we define the action of the group on $\mathcal{D}(\mathcal{Y};\mathcal{U})$; this is just the combination of the two actions previously described separately for $\mathcal{D}^\star(\mathcal{Y})$ and $\mathcal{D}^\star(\mathcal{U})$.

DEFINITION 4.37 (Action of representation nuisances). The group of representation nuisances $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ acts on $\mathcal{D}(\mathcal{Y};\mathcal{U})$ as follows: if $x = \langle g, h\rangle \in \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ and $\boldsymbol{D} \in \mathcal{D}(\mathcal{Y};\mathcal{U})$ then $x \cdot \boldsymbol{D} = g\boldsymbol{D}h$.

To show that this is a proper group action, one need to show that, for any two representation nuisances $x_1, x_2 \in \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$, the action of one followed by the other is equivalent to the action of the product:

$$\begin{aligned} x_2 \cdot (x_1 \cdot \boldsymbol{D}) &= x_2 \cdot (g_1 \boldsymbol{D} h_1) \\ &= g_2 g_1 \boldsymbol{D} h_1 h_2 \\ &= (g_2 g_1)\boldsymbol{D}(h_1 h_2) \\ &= (x_2 x_1) \cdot \boldsymbol{D}. \end{aligned}$$

4.11. The Geometry of Bootstrapping

A group action defines an equivalence relation on the set on which it acts: two elements of the set are in the same equivalence class if there is a group element that transforms one into the other.

In this case, two systems $\boldsymbol{D}, \boldsymbol{E}$ in $\mathcal{D}(\mathcal{Y};\mathcal{U})$ are considered equivalent if there is a representation nuisance $g \in \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ such that $g \cdot \boldsymbol{D} = \boldsymbol{E}$. If this is the case, we can transform one into the other just by changing their representation. This set of equivalence classes is named $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$.

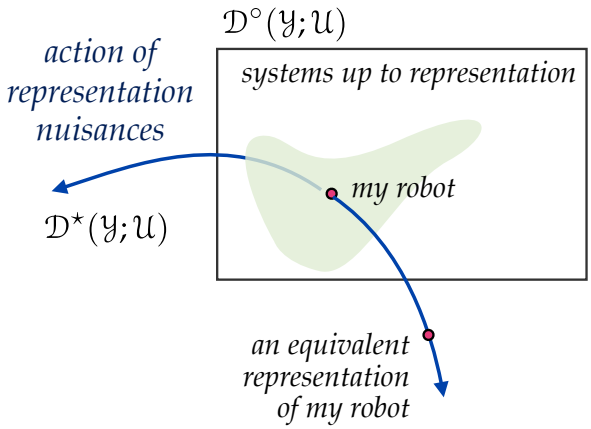


Figure 4.1. The geometry of bootstrapping: the set of all dynamical systems $\mathcal{D}(\mathcal{Y};\mathcal{U})$ can be factorized as $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U}) \times \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$, where the set of equivalence classes $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$ represents the physical properties of the system that are invariant to the representation.

DEFINITION 4.38 (Physical systems). We denote by $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$ the equivalence classes:

$$\mathcal{D}^\circ(\mathcal{Y};\mathcal{U}) = \mathcal{D}(\mathcal{Y};\mathcal{U}) \;/\; \mathcal{D}^\star(\mathcal{Y};\mathcal{U}) \,.$$

The interpretation is that the elements of $\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$ (which are sets of systems, equivalent up to representations) represent "physical" systems, as opposed to the elements of $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ which represent a particular choice of representation (Figure 4.1).

The next chapter formalizes the problem of bootstrapping, and argues that an ideal bootstrapping agent should be agnostic to the choice of representation, and that the degree of failure encodes the semantic assumptions about the data.

REMARK 4.39 (Other equivalence relations for dynamical systems). The "up to representation" equivalence relation is different from the many equivalence relations found in control theory. It subsumes *bisimulation [55]* equivalence.

The equivalence relation in the theory of realization (e.g., [56, 57]) is different as it considers systems having the same input-output behavior. The equivalence relation studied in the context of differential flatness [58–60] is different as well.

CHAPTER 5

Bootstrapping Agents

This chapter describes the scenario in which to place bootstrapping agents. It defines the two-stage protocol in which the agent interacts with the world: first an exploration/learning phase, then an acting/exploitation phase. The meaning of what is a goal *for a bootstrapping agent is also clarified. Once the structure is in place, it is possible to derive necessary invariant properties for a bootstrapping agent to be optimal.*

Table 5.1. Symbols used in this chapter

Definition of the problem		
$\boldsymbol{w}$	$\in \mathcal{D}(\mathcal{Y};\mathcal{U})$	The world with which the agent interacts.
$\mathcal{G}$	$\subset \mathsf{StocProcesses}(\mathcal{Y}\times\mathcal{U})$	Bootstrapping "goal" (desirable outcomes).
Definition of bootstrapping agent		
$\mathcal{A}$	$= \langle \mathcal{M}, \mathsf{expl}_{\mathcal{A}}, \mathsf{act}_{\mathcal{A}}\rangle$	A tuple defining a bootstrapping agent.
$\mathcal{M}$	(any set)	The agent's model space.
$\boldsymbol{m}$	$\in \mathcal{M}$	Learned model.
expl	$\in \mathcal{D}(\mathcal{U}\times\mathcal{M};\mathcal{Y})$	Exploration/learning strategy.
act	$: \mathcal{M}\to\mathcal{D}(\mathcal{U};\mathcal{Y})$	Agent's action/behaviors.
$\mathsf{Agents}(\mathcal{Y};\mathcal{U})$		The sets of all agents with formats $\mathcal{Y}/\mathcal{U}$.
Derived quantities		
$\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})$	$\in \mathsf{ProbMeasures}(\mathcal{D}(\mathcal{U};\mathcal{Y}))$	Behavior instantiated in the learning phase.
$\mathsf{WtoR}_{\mathcal{A}}(\boldsymbol{w})$	$\in \mathsf{StocProcesses}(\mathcal{Y}\times\mathcal{U})$	Final statistics of the agent's interaction with the world.
$\mathsf{success}^{\mathcal{G}}_{\mathcal{A}}$	$\subset \mathcal{D}(\mathcal{Y};\mathcal{U})$	Worlds for which the agent is successful.
Invariance analysis		
$\mathsf{G}_{\mathcal{A}}$	$\leq \mathcal{D}^{\star}(\mathcal{Y};\mathcal{U})$	Representations nuisances to which the agent is invariant.
$\mathsf{C}_{\mathcal{A}}$	$\subset \mathcal{D}(\mathcal{Y};\mathcal{U})$	... on this particular set.
$\mathsf{stab}(\mathcal{G})$		Symmetries of the goal set $\mathcal{G}$.

5.1. Observations and Commands

A bootstrapping agent interacts with the *"world"*. The world is everything that is unknown to the agent. For an agent embodied in a robotic body, the world is the series of the unknown sensors, the external environment, and the unknown actuators. The world is called "sensorimotor cascade", when the discussion is specific to the robotic setting.

The world and the agent communicate through the streams of observations and commands. The *observations* are what flows from the sensors to the agent; the *commands* are what flows from the agent to the actuators, and that the agent can choose freely.

At each time t, the observations $\boldsymbol{y}_t$ take values in a set $\mathcal{Y}$ called *observations space*. The commands $\boldsymbol{u}_t$ belong to the *command space* $\mathcal{U}$. The sets $\mathcal{Y}$ and $\mathcal{U}$ are the interface between the agent and the world, and they specify the *format* of the data which the agent can handle. Another signal that might come from the world to the agent is a reward signal. This can be included as part of the observations $\boldsymbol{y}_t$.

These are a few examples of observations spaces:

$\mathcal{Y} = \{\square, \boxdot\}$ The observations are an uninterpreted bit.

$\mathcal{Y} = \mathbb{R}$ The observations are a real number.

$\mathcal{Y} = \mathbb{R}^{n_y}$ The observations are a vector of n_y real numbers.

$\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$ The observations are a vector of n_y sensels, each taking values in the same set $\overline{\mathcal{Y}}$,

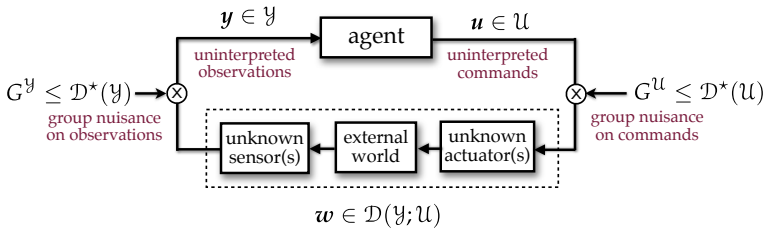


Figure 5.1. In the bootstrapping scenario, the embodied agent interacts with the "world", which is the series of the unknown actuators, the external environment, and the unknown sensors. The figure also shows the representation nuisances, which act as disturbances between the agent and the world.

not otherwise described.

$\mathcal{Y} = \mathsf{Continuous}(\mathbb{R}; [0,1]^2)$ The observations are a continuous real function on the square $[0,1]^2$.

$\mathcal{Y} = \mathsf{Continuous}(\mathbb{R}; \mathcal{S})$ The observations are a continuous field on some space $\mathcal{S}$.

$\mathcal{Y} = \mathsf{Functions}(\{0,\dots,254\}; \{1,\dots,640\} \times \{1,\dots,480\})$ The observations are a function defined on a 640×480 grid which takes one of 255 possible values.

REMARK 5.1. (*Formats vs semantic assumptions*) The last three examples show the distinction between the two concepts that were called "format" of the data, and the "semantic assumptions" about that data. The format of the data is just the choice of the sets $\mathcal{Y}$ and $\mathcal{U}$, which are the interface between the agent and the world. The semantic assumptions are what is implicitly implied about the data beyond the format. In the following, there are never comparisons between different formats, such as comparing an agent that uses real-valued observations ($\mathcal{Y} = \mathbb{R}$) and one that sees a discretized version of the same process (e.g., $\mathcal{Y} = \mathbb{N}$). The discussion is centered on comparing the semantic assumptions about the data: for example, there can be two agents which both see real-valued observations, but one is "more powerful" (in a sense which will be made precise) because it makes "less assumptions" about the meaning of those real values.

REMARK 5.2. (*Common abstractions*) While we do want to study the "radical" problem of bootstrapping from uninterpreted bits, often the discussion is at a higher level, considering more structured spaces, like many of the examples above. From an engineering perspective, it is clear that there is always a tradeoff between the efficiency/practicality of an agent, and the assumptions that must be made about the world. From this perspective, the goal is not necessarily to create the most generic agent, but rather to understand

sensels: The discrete elements of the agent's observations vector; from *sens*or *el*elemnts, in analogy to *pixels* (*pict*ure *el*ements).

what is this tradeoff, and more specifically, to understand how to formally characterize the "assumptions" of a given agent.

REMARK 5.3. (*Mathematical abstraction will be allowed*) Like all theories, at different points we will oscillated between realism and some level of mathematical abstraction, which is necessary to obtain any result. Consider the second assumption above. It might seem relatively innocuous to assume that the observations space $\mathcal{Y}$ corresponds to the real numbers $\mathbb{R}$. However, this sole assumption makes the model incompatible with physical reality: a real number corresponds to an infinite string, and it is not physically possible for an infinite amount of information to flow from the world to the agent in one instant.

5.2. Two-stage Interaction with the World

The interaction between agent and the world happens in two phases. During the learning/exploration phase, the agent is allowed to interact freely with the world. The result of the learning phase is a model learned by the agent. From this model, the agent instantiate some behavior that is performed during the action phase.

This formalization does not allow to investigate the tradeoff between *exploration* (using time/resources to estimate a model of the environment) and *exploitation* (using the collected information to perform the given task), which is an important issue in the study of intelligent agents. However, a proper formalization of this concept would imply committing to a specific way to quantify this tradeoff. Sooner than later, one is trapped in dealing with the consequences of one arbitrary formalization (e.g., hypothesizing some kind of reward scheme, and assuming exponential discounting of future rewards).

How much time or data does the agent need to learn? This is perhaps the central question of the more mathematical part of machine learning (à la Vapnik). From that perspective, if an agent has an infinite amount of data, then it can learn perfectly any distribution; rather, the interesting question is understanding what are the bounds/guarantees that can be derived on the learning result.

This effort goes in an orthogonal direction. Even if an agent is allowed to observe an infinite amount of sensorimotor data, the fidelity of the model learned, and consequently its performance on a task, is bounded by the complexity, flexibility, and the intrinsic bias of the parametric structure used by the agent. This analysis quantifies exactly these characteristics, and gives an interpretation in terms of semantic assumptions.

Nevertheless, while the considerations about the data needed are not the focus of the theoretical contribution delineated in this part, selecting a model that requires relatively few data is an important concern in the next part (Part 2).

5.3. Defining Bootstrapping Agents

All ingredients are ready to formalize exactly what is a bootstrapping agent. If $\mathcal{U}$ is the command space, $\mathcal{Y}$ is the observations space, then the "world" $\boldsymbol{w}$ is an element of the set of "black boxes" $\mathcal{D}(\mathcal{Y};\mathcal{U})$ (Definition 4.1).

A *bootstrapping agent* is defined as a tuple whose elements define the separate learning and acting phase.

DEFINITION 5.4 (Bootstrapping agent). A *bootstrapping agent* for a world belonging to $\mathcal{D}(\mathcal{Y};\mathcal{U})$ is a tuple $\mathcal{A} = \langle \mathcal{M}, \mathsf{expl}, \mathsf{act} \rangle$ such that:

$\mathcal{M}$ is a set (called *model space*);

$\mathsf{expl} \in \mathcal{D}(\mathcal{U} \times \mathcal{M}; \mathcal{Y})$ is the *exploration strategy*;

Vladimir N. Vapnik: Russian mathematician who formalized statistical learning theory.

$\mathsf{act} : \mathcal{M} \to \mathcal{D}(\mathcal{U}; \mathcal{Y})$ is the *action strategy*.

$\mathsf{Agents}(\mathcal{Y}; \mathcal{U})$ is the set of agents using the formats $\mathcal{Y}/\mathcal{U}$.

The *exploration strategy* $\mathsf{act}_{\mathcal{A}}$, defined as an element of $\mathcal{D}(\mathcal{U} \times \mathcal{M}; \mathcal{Y})$, is a dynamical system whose input are the observations (in $\mathcal{Y}$), and that has two output signals: the commands (in $\mathcal{U}$) that drive the exploration, and the *model* (in $\mathcal{M}$) which is being estimated. The world "model" is used mostly as a mnemonics because this quantity is an opaque value that ought to make sense only to the agent. Even though it is called here "model space", there are no constraints on the set $\mathcal{M}$. The names "model", like "learn" and "act", serve mostly as mnemonics.

During the exploration phase, the exploration strategy $\mathsf{act}_{\mathcal{A}}$ interacts in closed loop with the world $\boldsymbol{w}$. Using the Loop operator (Definition 4.12), the result is $\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\, \boldsymbol{w})$, and it is a stochastic process on $\mathcal{M} \times \mathcal{Y} \times \mathcal{U}$. Ignoring the commands and observations, consider it a stochastic process on the set $\mathcal{M}$ only:

$$\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\, \boldsymbol{w}) \in \mathsf{StocProcesses}(\mathcal{M}).$$

It is assumed that, after a long enough interaction, the system has converged to the stationary distribution, which can be extracted using Final operator (Definition B.8):

$$\mathsf{Final}(\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\, \boldsymbol{w})) \in \mathsf{ProbMeasures}(\mathcal{M}). \tag{5.1}$$

As a particular case, the model will have converged to one particular value in $\mathcal{M}$.

The *action strategy* $\mathsf{act}_{\mathcal{A}}$ is a map from $\mathcal{M}$ to $\mathcal{D}(\mathcal{Y}; \mathcal{U})$; this means that the learned model $\boldsymbol{m} \in \mathcal{M}$ is converted to a system $\mathsf{act}_{\mathcal{A}}(\boldsymbol{m}) \in \mathcal{D}(\mathcal{U}; \mathcal{Y})$ which does the actual interaction with the world. This system is called sometimes a "behavior" "instantiated" by the agent, to distinguish it from the whole agent.

All the systems mentioned can possibly have an internal state and randomized behavior (we now reap the fruits of the rather abstract definition of $\mathcal{D}(\mathcal{Y};\mathcal{U})$).

EXAMPLE 5.5. An agent embodied in a robot might include in $\boldsymbol{m}$ a description of the sensor geometry and the statistics of the environment; from that description $\boldsymbol{m}$, the agent then instantiates the behavior $\mathsf{act}_{\mathcal{A}}(\boldsymbol{m}) \in \mathcal{D}(\mathcal{Y};\mathcal{U})$, which might include the logic for estimation of an internal state.

In general, the model $\boldsymbol{m}$ is distributed according to the distribution (5.1), therefore, for a fixed world $\boldsymbol{w}$, one observes a distribution over possible behaviors. This function is given the name "world-to-behavior".

DEFINITION 5.6 (World-to-behavior $\mathsf{WtoB}_{\mathcal{A}}$). Define the "world-to-behavior" function as

$$\begin{aligned} \mathsf{WtoB}_{\mathcal{A}} : \mathcal{D}(\mathcal{Y};\mathcal{U}) &\to \mathsf{ProbMeasures}(\mathcal{D}(\mathcal{U};\mathcal{Y})) \\ \boldsymbol{w} &\mapsto \mathsf{act}_{\mathcal{A}}(\mathsf{Final}(\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\,\boldsymbol{w}))). \end{aligned}$$

Note that $\mathsf{expl}_{\mathcal{A}}\,\boldsymbol{w}$ is the series of the exploration strategy and the world, $\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\,\boldsymbol{w})$ are their closed loop statistics, $\mathsf{Final}(\mathsf{Loop}(\mathsf{expl}_{\mathcal{A}}\,\boldsymbol{w})) \in \mathsf{ProbMeasures}(\mathcal{M})$ is the distribution of the model learned, and the whole result is the distribution of the agent behavior.

When the agent instantiates the behavior $\mathsf{act}_{\mathcal{A}}(\boldsymbol{m})$, the loop is closed again with the world, and the result $\mathsf{Loop}(\mathsf{act}_{\mathcal{A}}(\boldsymbol{m})\,\boldsymbol{w})$ is a stochastic process on $\mathcal{Y}\times\mathcal{U}$:

$$\mathsf{Loop}(\mathsf{act}_{\mathcal{A}}(\boldsymbol{m})\,\boldsymbol{w}) \in \mathsf{StocProcesses}(\mathcal{Y}\times\mathcal{U}).$$

The "final result" of the interaction is then the average of this function by the distribution of $\boldsymbol{m}$. This function is given the name "world-to-result" $\mathsf{WtoR}_{\mathcal{A}}$.

DEFINITION 5.7 (World-to-result $\mathsf{WtoR}_{\mathcal{A}}$). For an agent $\mathcal{A}$, the "world-to-result" function $\mathsf{WtoR}_{\mathcal{A}}$ is defined as

$$\begin{aligned} \mathsf{WtoR}_{\mathcal{A}} : \mathcal{D}(\mathcal{Y};\mathcal{U}) &\to \mathsf{StocProcesses}(\mathcal{Y}\times\mathcal{U}) \\ \boldsymbol{w} &\mapsto \mathsf{Loop}(\mathsf{act}_{\mathcal{A}}(\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}))\,\boldsymbol{w}). \end{aligned}$$

Now that the interaction between world and agent is clearly defined, it is possible to reason about the agent's goals.

5.4. Defining the Agent's Goals

There are at least three qualitatively different ways to give a goal to the agent: external rewards, intrinsic motivation, and explicit tasks. They are not mutually exclusive. As discussed in the introduction , in an ideal agent, intrinsic motivation (e.g., "curiosity") guides the exploration, explicit tasks represent primitive skills, and pursuing an external reward might eventually guide the behavior. Their differences are briefly summarized here, especially for what regards their implications on the formalization of the problem. Considering all three approaches in the same framework leads to a rather abstract definition of goal.

5.4.1. *External rewards*

In reinforcement learning, one assumes that there is a reward signal from the world to the agent. In this construction, it can be added either as a separate signal from the observations $\boldsymbol{y}$ or simply assuming that it is part of the observations. The goal of the agent is to "maximize" this reward. It is necessary to be more precise and choose how the agent should feel like about 1) uncertainty; and 2) time. As for uncertainty, there is a large body of work (from Pascal on) that indicates that a rational agent should maximize the *expectation* of its rewards. As for time, one must decide how to discount future rewards with respect

to immediate results. It can be shown that geometric discounting (weighting the future rewards by γ^t, for some $\gamma \in (0, 1)$) is an appropriate choice for rational agents.

REMARK 5.8. (*Do we need rational agents?*) As a side note, it is not at all clear that the goal of AI is to design rational agents. The only example of intelligence we have—humans and other primates—do not appear to be anywhere close to perfect rational agents. In fact, much research in econometrics is dedicated to show how as shown how much humans deviate from the rational agent assumption. In general, humans are *risk averse* (a bird in the hand is worth two in the bush).

It is possible to create simple models of why this could be a reasonable heuristics for an agent, but in the end the true answer is that this is the result of human evolution, which is a complex social process. (For example, there is an asymmetry between genders, as women are generally more risk averse [61]).

5.4.2. *Intrinsic motivation*

A relatively recent line of research (e.g., [48–50]) is concerned with defining *intrinsic motivation* for intelligent agents. The basic idea is that part of the agent's interaction with the world can be guided by some intrinsic criteria which are finalized to acquire skills, not immediately useful for the execution of a task. This formalizes things such as *curiosity* or *play,* which are essential in cognitive development.

5.4.3. *Explicit tasks*

One alternative is defining explicit tasks in the spirit of control theory. For example, consider the problem we call *servoing*, which informally is stated as "Given the goal observation $\check{\boldsymbol{y}}$, choose the commands $\boldsymbol{u}$ such that the observations $\boldsymbol{y}$ eventually match $\check{\boldsymbol{y}}$".

While it is possible to frame this as a supervised learning problem, for example by manufacturing the reward function $R = \|\boldsymbol{y} - \check{\boldsymbol{y}}\|$, the conceptual difference is that the objective

function is not opaque, but rather it is known explicitly as a function of the observations.

These are the kind of tasks that will be studied most extensively here, for two reasons. One reason is that one might define a hierarchy of tasks that describe the essential skills of bootstrapping agents (Chapter 8). The other reason is that there are many aspects to a "learning" problem. Roughly speaking, these are:

(1) The *model identification* problem: what can the agent do?

(2) The *reward identification* problem: what should the agent aim to obtain?

(3) The *control* problem: *how* should the agent do it?

Using explicit tasks, we get rid of the second problem. Once there is a model and a goal to maximize, the control problem is *conceptually* easy (but possibly *computationally* very hard). The model identification problem is instead the one which is conceptually hard.

5.4.4. *A uniform interface for defining goals*

All three of these approaches can be considered under the same interface. Somehow they describe whether an agent is "correct" and "optimal", with some falsifiable assertion which can be judged by looking at the interaction of the agent with the world. Hence a goal will is defined as a subset of $\mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$ that indicates which are the "desirable" outcomes.

Intuitively, one might think that the goal depends on the particular world. This is reflected in the following temporary definition of goal.

DEFINITION 5.9 (Temporary definition of bootstrapping goal). A *bootstrapping goal* is a function

$$\mathcal{G} : \mathcal{D}(\mathcal{Y};\mathcal{U}) \rightrightarrows \mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$$

that associates to each world $\boldsymbol{w}$ the set $\mathcal{G}(\boldsymbol{w}) \subset \mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$, representing the desirable outcomes for the agent when interacting with the world $\boldsymbol{w}$.

This definition is admittedly quite abstract, but it does cover all situations of interest, and it can be reconciled with more traditional definitions.

EXAMPLE 5.10. One way to construct $\mathcal{G}(\boldsymbol{w})$ for a criterion that defines an optimal behavior $\boldsymbol{F}^\star(\boldsymbol{w}) \in \mathcal{D}(\mathcal{U};\mathcal{Y})$ is to set $\mathcal{G}(\boldsymbol{w})$ as simply the resulting interaction statistics for the optimal agent: $\mathcal{G}(\boldsymbol{w}) = \mathsf{Loop}(\boldsymbol{F}^\star(\boldsymbol{w})\boldsymbol{w})$.

The choice of the function $\mathcal{G}$ is not entirely arbitrary, and must respect two basic coherence properties:

(1) It does not make sense to define as desirable the outcomes that are impossible to obtain. For a fixed world $\boldsymbol{w}$, the set $\mathsf{AllOutcomes}(\boldsymbol{w})$ (Definition 4.16) describes all possible statistics that can be generated. Thus it is required that

$$\mathcal{G}(\boldsymbol{w}) \subseteq \mathsf{AllOutcomes}(\boldsymbol{w}). \tag{5.1}$$

(2) The goal must be observable for the agent. Consider two different worlds $\boldsymbol{w}_1, \boldsymbol{w}_2$. Suppose that a certain outcome $x \in \mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$ is considered feasible for the first world ($x \in \mathsf{AllOutcomes}(\boldsymbol{w}_1)$). Then, if the same outcome is desirable for the other world ($x \in \mathcal{G}(\boldsymbol{w}_2)$), then necessarily it must be desirable also for the first ($x \in \mathcal{G}(\boldsymbol{w}_1)$), simply because the agent cannot distinguish two worlds that appear the same externally. This can be written as

$$\mathcal{G}(\boldsymbol{w}_2) \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) \subseteq \mathcal{G}(\boldsymbol{w}_1). \tag{5.2}$$

The consequence of (5.1) and (5.2) is that it is not necessary to specify the function $\mathcal{G}$ for each world.

LEMMA 5.11. *Defining the set $\overline{\mathcal{G}}$ as the union of all desirable states:*

$$\overline{\mathcal{G}} = \bigcup_{\boldsymbol{w} \in \mathcal{D}(\mathcal{Y};\mathcal{U})} \mathcal{G}(\boldsymbol{w}), \tag{5.3}$$

the desirable states for a particular world can be found as the intersection of $\overline{\mathcal{G}}$ with the possible outcomes:

$$\mathcal{G}(\boldsymbol{w}) = \overline{\mathcal{G}} \cap \mathsf{AllOutcomes}(\boldsymbol{w}).$$

PROOF. From (5.2), for any world $\boldsymbol{w}_3$,

$$\mathcal{G}(\boldsymbol{w}_3) \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) \subseteq \mathcal{G}(\boldsymbol{w}_1).$$

Together with (5.2), this implies

$$(\mathcal{G}(\boldsymbol{w}_2) \cup \mathcal{G}(\boldsymbol{w}_3)) \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) \subseteq \mathcal{G}(\boldsymbol{w}_1).$$

By induction from two worlds $\boldsymbol{w}_2, \boldsymbol{w}_3$ to the whole set $\mathcal{D}(\mathcal{Y};\mathcal{U})$, one arrives at the set $\overline{\mathcal{G}}$:

$$\overline{\mathcal{G}} \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) \subseteq \mathcal{G}(\boldsymbol{w}_1).$$

One can show that this is not an inclusion but rather an equality between sets, by working on the left side. In the union (5.3), there is also $\boldsymbol{w}_1$, so $\overline{\mathcal{G}} = \overline{\mathcal{G}} \cup \mathcal{G}(\boldsymbol{w}_1)$:

$$\left[\mathcal{G}(\boldsymbol{w}_1) \cup \overline{\mathcal{G}}\right] \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) \subseteq \mathcal{G}(\boldsymbol{w}_1).$$

Because $\cap$ is distributive over $\cup$,

$$[\mathcal{G}(\boldsymbol{w}_1) \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1)] \cup \left[\overline{\mathcal{G}} \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1)\right] \subseteq \mathcal{G}(\boldsymbol{w}_1).$$

From condition (5.1), it follows that $\mathcal{G}(\boldsymbol{w}_1) \subseteq \mathsf{AllOutcomes}(\boldsymbol{w}_1)$. Hence $\mathcal{G}(\boldsymbol{w}_1) \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1) = \mathcal{G}(\boldsymbol{w}_1)$. Substituting this one obtains

$$\mathcal{G}(\boldsymbol{w}_1) \cup \left[\overline{\mathcal{G}} \cap \mathsf{AllOutcomes}(\boldsymbol{w}_1)\right] \subseteq \mathcal{G}(\boldsymbol{w}_1),$$

which makes the inclusion an equality:

$$\mathcal{G}(\boldsymbol{w}) = \overline{\mathcal{G}} \cap \mathsf{AllOutcomes}(\boldsymbol{w}).$$

□

The implication is that to specify the goal set, one just needs to specify the union of the goal set on all possible worlds. Therefore the temporary Definition 5.9 is amended with a simpler alternative.

DEFINITION 5.12 (Bootstrapping goal). A *bootstrapping goal* $\mathcal{G}$ is a subset of $\mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$ that represents the desirable outcomes for the agent when interacting with the world.

As was derived in Subsection 5.3, the interaction of an agent with the world is summarized by the function $\mathsf{WtoR}_{\mathcal{A}}$, which maps a world $\boldsymbol{w} \in \mathcal{D}(\mathcal{Y};\mathcal{U})$ to the resulting statistics $\mathsf{WtoR}_{\mathcal{A}}(\boldsymbol{w}) \in \mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$. At this point it is possible to forget the two-stage protocol and just use $\mathsf{WtoR}_{\mathcal{A}}$ as a proxy for studying the agent.

If the *goal* $\mathcal{G}$ can be considered a subset of $\mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$ representing the desirable outcomes, then, for a fixed goal $\mathcal{G}$, the preimage $\mathsf{WtoR}_{\mathcal{A}}^{-1}(\mathcal{G})$ is the set of worlds for which the agent is successful.

DEFINITION 5.13 (Success set). The success set $\mathsf{success}_{\mathcal{A}}^{\mathcal{G}} \subset \mathcal{D}(\mathcal{Y};\mathcal{U})$ is the set of worlds for which the agent succeeds in a particular goal $\mathcal{G}$:

$$\mathsf{success}_{\mathcal{A}}^{\mathcal{G}} = \mathsf{WtoR}_{\mathcal{A}}^{-1}(\mathcal{G}).$$

5.5. Necessary Invariance Properties of the Agent

Recall that a representation nuisance (Definition 4.36) is a group that acts on the world $\boldsymbol{w}$ by changing the representation of observations and commands. This is a "nuisance", in the sense that it can always be inverted to obtain the original world. This section derives necessary properties for the agent to be robust to these nuisances.

This is not a simple statement on the functions $\mathsf{act}_{\mathcal{A}}$ or $\mathsf{expl}_{\mathcal{A}}$, but rather one must consider the function $\mathsf{WtoB}_{\mathcal{A}}$ (Definition 5.6), which maps the world $\boldsymbol{w}$ to the agent's behavior instantiated in the acting phase. The necessary invariance properties for the agent are derived as properties of this function $\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})$.

The discussion is symmetric for observations and commands.

As for the observations, suppose that the world $\boldsymbol{w}$ is perturbed by a representation nuisance corresponding to a group $G \leq \mathcal{D}^{\star}(\mathcal{Y})$, so that the agent interacts with $g\boldsymbol{w}$. The agent is robust to this nuisance if it is able to compensate it. Let $\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})$ be the agent behavior for the original world $\boldsymbol{w}$. If a nuisance $g \in G \leq \mathcal{D}^{\star}(\mathcal{Y})$ acts on the world, the world is transformed as

$$\boldsymbol{w} \mapsto g\boldsymbol{w}, \tag{5.1}$$

to obtain exactly the same result, it is necessary that the function $\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})$ satisfies

$$\mathsf{WtoB}_{\mathcal{A}}(g\boldsymbol{w}) = \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})g^{-1}. \tag{5.2}$$

To see why this should hold, consider the world-agent series in the acting phase, which is written as

$$\text{series without nuisance:} \qquad \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})\boldsymbol{w}.$$

If the world changes according to (5.1) and the agent changes according to (5.2), then the

series is

$$\text{series with nuisance:} \qquad (\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w})\, g^{-1})(g\, \boldsymbol{w}).$$

Because the two elements cancel each other, the result is that the series is invariant to the nuisance. Intuitively, this has the interpretation that, the commands sent to the agent to the world are independent of the observations representation.

The discussion for the commands is symmetric. A nuisance $h \in H \leq \mathcal{D}^\star(\mathcal{U})$ acts on the commands and transforms the world as

$$\boldsymbol{w} \mapsto \boldsymbol{w}\, h.$$

In this case, the agent must satisfy

$$\mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}\, h) = h^{-1}\, \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}). \tag{5.3}$$

If this holds, the agent-world series is invariant to the nuisance:

$$\boldsymbol{w}\, h\, \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}\, h) = \boldsymbol{w}\, h\, h^{-1}\, \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}) = \boldsymbol{w}\, \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}).$$

Intuitively, this has the interpretation that the world gives the same observations (that is, the state is unchanged), notwithstanding a change of representations for the commands.

These properties can be written more compactly by considering that we already defined the set of representation nuisances (Definition 4.36) $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$, which captures both observations and commands nuisances. Considering a generic element $x = \langle g, h\rangle \in \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ and its dual $x_* = \langle h, g\rangle \in \mathcal{D}^\star(\mathcal{U};\mathcal{Y})$, the invariance condition is succinctly written

$$\mathsf{WtoB}_{\mathcal{A}}(x \cdot \boldsymbol{w}\,) = x_*^{-1} \cdot \mathsf{WtoB}_{\mathcal{A}}(\boldsymbol{w}). \tag{5.4}$$

The set of representation nuisances $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ is quite large, so it is unrealistic to expect that the agent is invariant to all nuisances. A practical agent will be invariant only to a subgroup $\mathrm{G}_\mathcal{A} \leq \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ of all nuisances. In practice, it is common that the invariance of the agent also depends on the set $C_\mathcal{A} \subset \mathcal{D}(\mathcal{Y};\mathcal{U})$ to which the world $\boldsymbol{w}$ belongs. The following defines the invariance properties of the agent with respect to a tuple $\langle C_\mathcal{A}, \mathrm{G}_\mathcal{A}\rangle$.

DEFINITION 5.14 (Invariance properties of a bootstrapping agent). Given a subset of the worlds $C_\mathcal{A} \subset \mathcal{D}(\mathcal{Y};\mathcal{U})$ and a subgroup of the representation nuisances $\mathrm{G}_\mathcal{A} = \mathrm{G}_\mathcal{A}^\mathcal{Y} \times \mathrm{G}_\mathcal{A}^\mathcal{U} \leq \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$, an agent $\mathcal{A}$ is *invariant on* $\langle C_\mathcal{A}, \mathrm{G}_\mathcal{A}\rangle$ if it holds that

$$\forall \boldsymbol{w} \in C_\mathcal{A}, \quad \forall x \in \mathrm{G}_\mathcal{A}, \quad \mathsf{WtoB}_\mathcal{A}(x \cdot \boldsymbol{w}) = x_*^{-1} \cdot \mathsf{WtoB}_\mathcal{A}(\boldsymbol{w}). \tag{5.5}$$

The condition can be written separately for observations and commands as follows:

$$\forall \boldsymbol{w} \in C_\mathcal{A}, \quad \forall g \in \mathrm{G}_\mathcal{A}^\mathcal{Y}, \qquad \mathsf{WtoB}_\mathcal{A}(g\boldsymbol{w}) = \mathsf{WtoB}_\mathcal{A}(\boldsymbol{w})g^{-1},$$

$$\forall \boldsymbol{w} \in C_\mathcal{A}, \quad \forall h \in \mathrm{G}_\mathcal{A}^\mathcal{U}, \qquad \mathsf{WtoB}_\mathcal{A}(\boldsymbol{w}h) = h^{-1}\mathsf{WtoB}_\mathcal{A}(\boldsymbol{w}).$$

Chapter 6 gives a catalog of semantic assumptions, and Chapter 7 describes the corresponding representation nuisances.

5.6. Invariance Properties of the Goal Set $\mathcal{G}$

In the usual perspective of control theory and machine learning alike, an objective (or an error function, reward, etc.) is supposed to be given as part of the problem statement, and outside of judgment. This chapter shows that, in the bootstrapping perspective, it is possible to judge whether an error function is better than another, because, depending on their symmetries, they imply different semantic assumptions which carry over to the

agent. Therefore, part of the problem of bootstrapping is also designing good error functions.

5.6.1. *One example*

The next example shows the consequences of choosing a particular error function over another. For simplicity, the formalization is given here in continuous time, but all conclusion apply also in discrete time.

The class of systems C is a subset of $\mathcal{D}(\mathcal{Y};\mathcal{U})$, with $\mathcal{U} = \{\boldsymbol{u} \in \mathbb{R}^2 \mid \|\boldsymbol{u}\| \leq 1\}$ and $\mathcal{Y} = \mathbb{R}^2$. Let the two-dimensional vector $\boldsymbol{q} \in \mathbb{R}^2$ represent the position in a plane of a solid body. The initial distribution for $\boldsymbol{q}$ is unknown (it is an unknown parameter). The commands $\boldsymbol{u}$ affect the pose like kinematic velocities, but in an unknown direction, represented by an orthogonal matrix $\mathbf{B} \in \mathsf{O}(2)$, which is another unknown parameter of the system. The observations $\boldsymbol{y} \in \mathbb{R}^2$ are simply the pose. The class C is thus defined

$$C = \begin{cases} \dot{\boldsymbol{q}} & = \mathbf{B}\boldsymbol{u}, \qquad \mathbf{B} \in \mathsf{O}(2) \\ \boldsymbol{q}_0 & \in \mathsf{ProbMeasures}(\mathbb{R}^2) \\ \boldsymbol{y} & = \boldsymbol{q}. \end{cases} \tag{5.1}$$

The goal for the agent is to stabilize the pose to $\boldsymbol{y} = \mathbf{0}$. The details of how this is encoded matter greatly. Several error functions were discussed before (Section 3.9), among which

$$E_1 = \int \|\boldsymbol{y}\|_1 \,\mathrm{d}t, \qquad E_2 = \int \|\boldsymbol{y}\|_2 \,\mathrm{d}t, \tag{5.2}$$

which are going to be studied again here. Both express the same idea that the agent must stabilize the observations to $\boldsymbol{y} = \mathbf{0}$.

After the dynamics and an error function (either E_1 or E_2) are defined, the problem is well posed from a control theory perspective. From this information, it is possible to

design an optimal agent which identifies the unknown parameter for the model, and then uses it to solve the task by minimizing the given error function.

5.6.2. *Computing the system's symmetries*

The first step of the analysis consists in finding the symmetries of the set $C \subset \mathcal{D}(\mathcal{Y};\mathcal{U})$. The stabilizer of C (Definition C.39) $\mathsf{stab}^{\mathcal{D}^\star(\mathcal{Y};\mathcal{U})}(C)$ is the largest subgroup $G \leq \mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ such that $G \cdot C = C$. For this class of systems, the stabilizer is

$$\mathsf{stab}^{\mathcal{D}^\star(\mathcal{Y};\mathcal{U})}(C) = \mathsf{E}(2) \times \mathsf{O}(2).$$

The orthogonal group $\mathsf{O}(2)$ (Definition D.2) acts on the commands as $\boldsymbol{u} \mapsto \mathbf{X}\boldsymbol{u}$. The Euclidean group $\mathsf{E}(2)$ (Definition D.3) acts on the observations, and the action is $\boldsymbol{y} \mapsto \mathbf{A}\boldsymbol{y} + \boldsymbol{v}$, with $\mathbf{A} \in \mathsf{O}(2)$ and $\boldsymbol{v} \in \mathbb{R}^2$. These transformations preserve the class of system C, because, calling $\boldsymbol{u}' = \mathbf{X}\boldsymbol{u}$ and $\boldsymbol{y}' = \mathbf{A}\boldsymbol{y} + \boldsymbol{v}$, the new dynamics for $(\boldsymbol{u}', \boldsymbol{y}')$ is

$$g \cdot C = \begin{cases} \dot{\boldsymbol{q}} & = (\mathbf{ABX})\boldsymbol{u}, \\ \boldsymbol{q}_0 & \in \mathbf{A}\,\mathsf{ProbMeasures}(\mathbb{R}^2) + \boldsymbol{v}, \\ \boldsymbol{y} & = \boldsymbol{q}. \end{cases} \tag{5.3}$$

Because $\mathbf{ABX} \in \mathsf{O}(2)$ and $\mathbf{A}\,\mathsf{ProbMeasures}(\mathbb{R}^2) + \boldsymbol{v} = \mathsf{ProbMeasures}(\mathbb{R}^2)$, the new system is still an element of the class C.

---

stabilizer: The stabilizer of a set is the subgroup of a given group whose action leaves the set invariant. See Definition C.39.

5.6.3. *Computing the symmetries of the error function*

The second step is considering the symmetries of the error functions in (5.2). From the previous discussion (Table 3.3), the symmetries of the functions are

$$\begin{aligned} \mathsf{Sym}(E_1) &= \mathsf{D}_{\pm}(n) \times \mathsf{Perm}(n). \\ \mathsf{Sym}(E_2) &= \mathsf{O}(n). \end{aligned}$$

For $n = 2$, these groups can be written more explicitly as

$$\begin{aligned} \mathsf{Sym}(E_1) &= \left\{\mathbf{I}, \left(\begin{smallmatrix} +1 & 0 \\ 0 & -1 \end{smallmatrix}\right), \left(\begin{smallmatrix} -1 & 0 \\ 0 & -1 \end{smallmatrix}\right), \left(\begin{smallmatrix} -1 & 0 \\ 0 & +1 \end{smallmatrix}\right)\right\} \times \left\{\mathbf{I}, \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)\right\}, \\ \mathsf{Sym}(E_2) &= \left\{\pm \left(\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right), \theta \in [0, 2\pi)\right\}. \end{aligned}$$

The first error function is "much less" invariant than the second because $\mathsf{Sym}(E_1)$ is a finite set strictly contained in the infinite set $\mathsf{Sym}(E_2)$.

Usually objectives break the symmetries of the system. Consider the error function E_2, whose symmetries are the orthogonal group $\mathsf{O}(n)$. The system class C is invariant to $\mathsf{E}(2)$, which is the semidirect product of $\mathsf{O}(n)$ and $\mathbb{R}^2$, acting as translations. When this error function is considered together with this class of system, the translation symmetry is lost—and for good reason, because the error function dictates that $\boldsymbol{y} = \mathbf{0}$ is "special", and while $\mathbf{0}$ is a fixed point for the action of $\mathsf{O}(n)$, it is not preserved by translations. If one wants the agent to stabilize to a specific observation, it is unavoidable that the translation symmetry is lost.

But consider the symmetries of the first function: $\mathsf{Sym}(E_1)$ contains exactly those symmetries that maintain invariant the set of base vectors. This set is much smaller than the potential set of symmetries of the system. This can be interpreted as a semantic assumption: choosing this particular error function means imposing that a particular choice of

base vectors is significant (in addition that the particular point $\boldsymbol{y} = \mathbf{0}$ is significant). Consequently, an optimal agent for this error function will carry over this semantic assumption, and would not be as invariant as it could possibly be.

This is one aspect that makes bootstrapping a challenging problem: other than the problem of designing agents, there is also the problem of designing proper error functions.

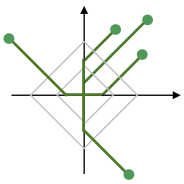
5.6.4. *Symmetries of the goal set $\mathcal{G}$*

To make this completely formal, and applicable even for goals that cannot be expressed by an error function, it is necessary to state this intuition with reference to the goal $\mathcal{G}$, which was defined in Section 5.4.4 as a subset of $\mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$. This subset has the interpretation of being the outcomes of the agent-world interaction that are deemed "desirable". If the goal can be cast as an optimization problem, then the set $\mathcal{G}$ can be constructed by taking the optimal agent and recording its interaction with the world.

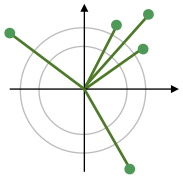
For this example, it is easy to describe the trajectories of an optimal agent (Figure 5.1). Not surprisingly, the symmetries of these trajectories are the same as the symmetries of the error function. The set of optimal trajectories, for all starting points, together with the relative commands, form the set $\mathcal{G} \subset \mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$. (In this case, those are just simple sequences, because the model has no stochasticity.) The symmetries of $\mathcal{G}$, or, more formally, the stabilizer

$$\mathsf{stab}(\mathcal{G}) \leq \mathcal{D}^{\star}(\mathcal{Y}; \mathcal{U}),$$

gives an upper bound for the invariance of the agent and expresses the semantic assumptions of the goal.



(a) $\int \|\boldsymbol{y}\|_1 \, \mathrm{d}t$



(b) $\int \|\boldsymbol{y}\|_2 \, \mathrm{d}t$,

Figure 5.1. Trajectories of an optimal agent for the system (5.1).

CHAPTER 6

A Catalog of Semantic Assumptions

This chapter describes a few semantic assumptions about black boxes. The challenge here is to describe the key assumptions, without fully instantiating the system. Each assumption is associated to the group of transformations that preserves it.

6.1. Legend

preconditions on format: These are preconditions about the *format* of the data (i.e., sets $\mathcal{Y}$ and $\mathcal{U}$) that need to be verified for this semantic assumption to make sense. For example, some of these semantic assumptions make sense only if $\mathcal{U}$ is a metric space.

preserved by: This is the largest representation nuisance that preserves this assumption. Some of these are explicit links to the representation nuisance's section in the nuisances catalog (Chapter 7).

implies: This field notes whether the assumption is stronger than another.

disrupted by: This is an explicit mention of a nuisance that would not preserve the semantic assumption.

6.2. Catalog

Assumption 1: Temporal changes are salient

---

representation nuisance: A causally invertible transformation of observations and commands. See Definition 4.36.

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

preconditions on format: $\overline{\mathcal{Y}}$ is a metric space

Roughly speaking, a "salient" stimulus is one to which it is worth dedicating the agent's attention and computational resources. Usually, in biology the definition does not get more precise than this.

We can give a simple temporary definition for our goals which uses the value of information. Suppose that some sensel z can acquire two states, $\square$ and $\boxdot$. We ask what is the value for the agent to not observe that variable, and just assume one of the two values. If one value is more salient than the other, then ignoring a salient value is more regrettable than ignoring a nonsalient value. In this example (Table 6.1), $z = \boxdot$ is the salient value.

Table 6.1. Costs incurred by an agent of not observing one sensel

cost incurred	$z = \square$	$z = \boxdot$
agent assumes $z = \square$	1	10
agent assumes $z = \boxdot$	1	1

Suppose that we have a notion of "saliency", then this semantic assumption states that the saliency of a sensel depends on its temporal derivative, and the larger the derivative, the more salient the sensel is. We need a metric on $\overline{\mathcal{Y}}$ to measure the change, so this semantic assumption applies only when $\overline{\mathcal{Y}}$ is a metric space. In discrete time, we might measure this change as $d^{\overline{\mathcal{Y}}}(y_i(k), y_i(k+1))$, where $d^{\overline{\mathcal{Y}}}$ is a metric on $\overline{\mathcal{Y}}$. The largest representation nuisance that preserves this property is the combination of any permutation with isometries of $\overline{\mathcal{Y}}$.

preserved by: $\mathsf{Perm}(n_{\boldsymbol{y}}) \times \mathsf{Isom}(\overline{\mathcal{Y}})^{n_y}$

disrupted by: Any nuisance that mixes the sensel values.

---

metric space: A metric space is a set endowed with a metric. See Definition E.1.

Assumption 2: Larger (or smaller) values are more salient

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

preconditions on format: $\overline{\mathcal{Y}}$ is totally ordered

Neurons communicate mainly through spikes [62]. While we still do not understand the neural code, we know that this code is *sparse*, in the sense that neurons are mostly silent. This makes sense evolutionarily because spiking consumes energy. It is also thought that a spike is more salient than silence (see Assumption 1 for a definition of *salient*).

Suppose that each sensels value belongs to a set $\overline{\mathcal{Y}}$ which is totally ordered; this simply means that we can say if a value is larger or smaller than another. In this case, an agent's semantic assumption might be that larger values are more salient (or vice versa). The largest nuisance preserving this property are the orientation-preserving homeomorphisms of $\overline{\mathcal{Y}}$, plus any permutation.

preserved by: Nuisance 9 ($\mathsf{Perm}(n_{\boldsymbol{y}}) \times \mathsf{Homeo}_+(\overline{\mathcal{Y}})^{n_y}$)

Assumption 3: Observations have "continuous" dynamics

preconditions on format: $\mathcal{Y}$ is a metric space

A common assumption for agents is that the observations are expected to change "slowly". One way to encode this assumption, in a way which is robust to noise, is to assume that $\mathcal{Y}$ is a metric space, and look at the statistics of the distance between two successive measurements

$$d(\boldsymbol{y}(t), \boldsymbol{y}(t+1)).$$

---

total order: An antisymmetric, transitive, and total binary relation. See Definition A.9.

A reasonable definition of "continuous" dynamics in discrete time is that the pdf

$$f(x) = \mathbb{P}(\ d(\boldsymbol{y}(t), \boldsymbol{y}(t+1)) = x\ )$$

is maximum for $x = 0$ and it is monotonically decreasing.

A generic homeomorphism of $\mathcal{Y}$ would not preserve this property, because it can warp distances in an unpredictable way. Clearly this property is preserved by the isometries $\mathsf{Isom}(\mathcal{Y})$.

preserved by: Nuisance 4 ($\mathsf{Isom}(\mathcal{Y})$)

disrupted by: Nuisance 3 ($\mathsf{Homeo}(\mathcal{Y})$)

Assumption 4: Sensels have similar statistics

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_{\boldsymbol{y}}}$

One semantic assumption that might simplify the development of an agent is that all sensels have similar statistics. For example, the agent might assume that the pdf of each sensel is the same. Then the agent can estimate this pdf faster by considering the samples from all sensels at the same time, rather than estimating one pdf for each sensel. Unfortunately, this assumption is only preserved by sensels permutations.

preserved by: Nuisance 1 ($\mathsf{Perm}(n_{\boldsymbol{y}})$)

Assumption 5: Sensels noise is independent

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_{\boldsymbol{y}}}$

Similarly, another assumption might be that the noise process acts independently on all sensels. This is a softer assumption, because it is preserved by any representation nuisance

pdf: Probability distribution function. See Definition B.6.

that transforms each sensel independently as well as permute them.

preserved by: $\mathsf{Perm}(n_{\boldsymbol{y}}) \times \mathsf{Aut}(\overline{\mathcal{Y}})^n$

Assumption 6: The observations correspond to a spatial field

preconditions on format: $\mathcal{Y}$ is a field on $\mathcal{S}$

preconditions on format: $\mathcal{S}$ is a metric space

Assume that the observations are a spatial field $\boldsymbol{y} : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S}$ is a manifold. For example, we might consider the observations from a camera as a spatial field on $[0,1]^2$ (the image space).

One tacit assumption might be that the space $\mathcal{S}$ corresponds to observations of a physical space. As a counterexample, consider using an image to encode some other unrelated information, for example by using luminance to encode bits (Figure 2.1a).

One way to formalize this assumption is to endow $\mathcal{S}$ with a metric, and expect that, for any two positions $s_1, s_2 \in \mathcal{S}$, the values $y(s_1)$ and $y(s_2)$ are, on average, more similar to each other if their distance $d(s_1, s_2)$ is small. Let R be a similarity measure. Then we could impose that the similarity is a function of the distance:

$$R(y(s_1), y(s_2)) = f(d(s_1, s_2)). \tag{6.1}$$

Such assumption would be invariant only to the isometries of $\mathcal{S}$. Moreover, imposing that the function f in (6.1) is the same at all sensels is quite restrictive.

A more robust formalization, slightly less elegant, is to say that the function $r_{s_1}(s) = R(y(s_1), y(s))$ (i.e., the similarity of the value at s_1 with respect to its neighbors) is *locally*

similarity measure: A function of two random variables that is 1 if they are identical.

geodesically concave. This property captures the same idea of local similarity, and it is invariant to all diffeomorphisms of $\mathcal{S}$.

This is more than the format of the data.

preserved by: Nuisance 6 ($\mathsf{Diff}(\mathcal{S})$)

Assumption 7: The spatial field is homogenous

preconditions on format: $\mathcal{Y}$ is a field on $\mathcal{S}$

Suppose that, as per the previous semantic assumption, the observations are a spatial field. For example, if the sensor is a camera, pixels close to each other are on average more similar than pixels far from each other. In practice, computer vision algorithms make much more assumption about the data than just locality. One typical assumption is that the signal statistics are "homogenous" across the image. For example, many algorithms use some sort of features (e.g., SIFT [63]) which are obtained by applying a filter bank at different scales at each point in the image (e.g., Gaussian filter with standard deviation equal to 8, 16, 32, 64 pixels). The scales are fixed across the image: this assumes that the image statistics are homogeneous across the image.

This can be formalized by requiring that statistics such as the covariance of the spatial gradient $\mathrm{cov}(\nabla_s y(s))$ are constant across the image. In this case, the assumption is invariant only to the isometries of $\mathcal{S}$.

implies: Assumption 6 (*The observations correspond to a spatial field*)

preserved by: Nuisance 5 ($\mathsf{Isom}(\mathcal{S})$)

---

geodesic convexity: Generalization of convexity for functions whose domain is a manifold. See Definition E.19.

Assumption 8: Observations are continuous in the states

preconditions on format: $\mathcal{Y}$ is a topological space

Suppose that there is some behaviorally relevant hidden state in the world, and that the observations are a continuous function of that hidden state. This is an assumption often done by algorithms that fit a policy from the instantaneous observations to the commands, and in doing so they assume (due to the internal representation used for such a policy) that the policy is a smooth function of the observations.

This property is preserved by any continuous transformations of the observations.

preserved by: Nuisance 3 ($\mathsf{Homeo}(\mathcal{Y})$)

Assumption 9: White noise

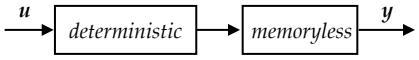
preconditions on format: None

A stochastic process is said to be *white* if the values at different instants are independent.* A system has "white noise" if the observations are corrupted by a white process. Usually "corrupted" means that the noise acts additively on the observations. That definition would imply that we also assume that $\mathcal{Y}$ is a vector space. A more general formalization of white noise is assuming that the world can be factorized in a deterministic system (Definition 4.17) followed by a a memoryless (Definition 4.18) stochastic system (Figure 6.1a). The dual concept for the commands would be if the world could be factorized in the opposite way (Figure 6.1b).

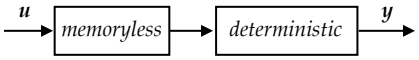
---

policy: A map from states (or observations) to actions (commands).

*If the process is Gaussian, then we can say equivalently that they are *uncorrelated*; but note that independent is equivalent to uncorrelated only for Gaussian variables.



(a) White noise on the observations



(b) White noise on the commands

Figure 6.1. Our definition of white noise on the observations is that we can factorize the system as a deterministic system followed by a stochastic memoryless system, or vice versa for the commands.

This assumption is preserved by all instantaneous representation nuisances.

preserved by: Nuisance 13 ($\mathsf{Aut}(\mathcal{Y})$)

disrupted by: Any non memoryless nuisance.

Assumption 10: The system is reversible

preconditions on format: None

A system is reversible if we can find a map $\rho : \mathcal{U} \to \mathcal{U}$, such that, for each command $\boldsymbol{u} \in \mathcal{U}$, giving the command $\boldsymbol{u}$ followed by $\rho(\boldsymbol{u})$ (or vice versa), takes the system back to its original state. In general, if a system is reversible, planning under uncertainty is "easy", because if some prediction is not verified, the agent can step back and return to a previous state.

This assumption is preserved by all instantaneous transformations of the commands.

preserved by: Nuisance 16 ($\mathsf{Aut}(\mathcal{U})$)

Assumption 11: Similar commands have similar effects

preconditions on format: $\mathcal{U}$ is a topological space

A common assumption is that the effect of two similar commands is close. For example, giving the command $\boldsymbol{u} = 1$ or $\boldsymbol{u} = 1.01$ takes the system to two similar states. This property is preserved by all homeomorphisms of $\mathcal{U}$.

The trouble for this is how to define the "effect" of a certain choice of commands from a bootstrapping perspective, that is, without referring to an unobservable "state". We give two alternatives.

(1) One possibility is define this using an input-to-output property. For example, we might require that the probability distribution of the future observations depends continuously on the commands. This does not require that $\mathcal{Y}$ has a well-defined topology, because it uses the topology of $\mathsf{ProbMeasures}(\mathcal{Y})$.

(2) Another possibility is using the concept of a task. A task induces the notion of an optimal command $\boldsymbol{u}^\star$. Different commands at time t will generally change the optimal command at time $t+1$. For a fixed time t, the optimal command at the next step $\boldsymbol{u}^\star_{t+1}$ is a function of the chosen command $\boldsymbol{u}_t$ and the unknown observations that the agent receives next: $\boldsymbol{u}^\star_{t+1} = f(\boldsymbol{u}_t, \boldsymbol{y}_t)$. This construction is for a fixed t, so the function f contains all the past experience up to time t. Considering the variable $\boldsymbol{y}_t$ as unknown, we can consider the partial function $F : \mathcal{U} \to (\mathcal{Y} \to \mathcal{U})$, such that the optimal command can be written as $\boldsymbol{u}^\star_{t+1} = F(\boldsymbol{u}_t)(\boldsymbol{y}_t)$. Then similar commands have similar effect if the map F is continuous. This implies using the topology of $\mathcal{Y}$, which needs to be assumed to be a topological space.

preserved by: Nuisance 14 ($\mathsf{Homeo}(\mathcal{U})$)

Assumption 12: One command does nothing

preconditions on format: None

One useful assumption for an agent is that there is one value $\boldsymbol{u}^{\mathsf{nop}} \in \mathcal{U}$ that corresponds to the actuators "resting", and the controllable part of the state space not changing. This property is preserved by any instantaneous transformation of the commands.

preserved by: Nuisance 16 ($\mathsf{Aut}(\mathcal{U})$)

Assumption 13: A known command does nothing

preconditions on format: None

One additional assumption is that the agent *knows* that special value $\boldsymbol{u}^{\mathsf{nop}}$ that corresponds to "resting". This property is preserved by any instantaneous transformation of the commands that keeps that special value fixed.

preserved by: Subgroup of $\mathsf{Aut}(\mathcal{U})$ fixing $\boldsymbol{u}^{\mathsf{nop}}$

implies: Assumption 12 (*One command does nothing*)

Assumption 14: Minus does the opposite

preconditions on format: $\mathcal{U} = \mathbb{R}^{n_{\boldsymbol{u}}}$

The command $-\boldsymbol{u}$ has the opposite effect of $+\boldsymbol{u}$. This implies that $\boldsymbol{u} = 0$ has the zero effect.

preserved by: Nuisance 17 ($\mathsf{Aut}(\mathbb{R}^{+}_{\circ})^{n_{\boldsymbol{u}}}$)

implies: Assumption 12 (*One command does nothing*)

implies: Assumption 13 (*A known command does nothing*)

implies: Assumption 10 (*The system is reversible*)

Assumption 15: More does more

preconditions on format: $\mathcal{U} = \overline{\mathcal{U}}^{n_u}$

preconditions on format: $\overline{\mathcal{U}}$ is totally ordered

Suppose that each command takes value in a set $\overline{\mathcal{U}}$, and that this set is totally ordered, so that we can distinguish a "small" command from a "large" command. One semantic assumption might be that "larger" commands have a larger "effect". See Subsection 6.2 for an intrinsic definitions of "effect".

This property is preserved by all orientation-preserving transformations of $\overline{\mathcal{U}}$.

preserved by: Nuisance 20 ($\mathsf{Perm}(n_{\boldsymbol{u}}) \times \mathsf{Homeo}_+(\overline{\mathcal{U}})^{n_u}$)

Assumption 16: Half does half

preconditions on format: $\mathcal{U} = \mathbb{R}^{n_u}$

This is another step toward assuming a full linear structure. Suppose there is a metric to measure the effect of a command. For example, we could take $\|\dot{\boldsymbol{y}}\|$ as a metric. Then this semantic assumption states that, for $\alpha > 0$, the effect of $\alpha \boldsymbol{u}$ is α times the effect of $\boldsymbol{u}$.

implies: Assumption 13 (*A known command does nothing*)

preserved by: The largest nuisance preserving this property are "star shaped" transformations of the kind $\boldsymbol{u}' = f\left(\boldsymbol{u}/\|\boldsymbol{u}\|\right)\boldsymbol{u}$, for any function $f : \mathbb{S}^{n_u - 1} \to \mathbb{R}^+_\circ$.

Assumption 17: The world has finite memory

preconditions on format: None

Recall that a system has finite memory if its observations can be predicted by looking at only a finite window of the previous observations and the commands accepted by the system (Definition 4.19). This is a particularly convenient assumption for the agent to do,

as it puts an upper bound on the computational resources that the agent must invest in creating a model for the system.

This property is preserved if the nuisances acting on the observations and commands have finite memory as well.

A different assumption would be to assume that the world has *bounded* memory. The largest representation nuisances preserving that property are the instantaneous transformations.

preserved by: Nuisance 10 ($\mathcal{D}^{\star}_{\mathsf{fm}}(\mathcal{Y})$)

preserved by: Nuisance 19 ($\mathcal{D}^{\star}_{\mathsf{fm}}(\mathcal{U})$)

Assumption 18: Commands are kinematic velocities

preconditions on format: $\mathcal{U} = \mathbb{R}^{n_u}$

One quite specific semantic assumption is that the commands represent kinematic velocities of the system. The following is one possible way to formalize this concept while keeping the assumptions on the rest of the system quite vague.

Assume that one state of the system evolves on a Lie group G, and that the commands $\boldsymbol{u}$ determine the velocity of g on G:

$$\dot{g}_t = g_t \, \mathbf{A} \, \boldsymbol{u}_t,$$

where $\mathbf{A}$ is a linear operator, representing unknown scaling or change of coordinates. We

---

Lie group: A topological group which is also a differentiable manifold with the same topology. See Definition C.44.

also allow the system to have another state ξ with its own dynamics, and that the observations $\boldsymbol{y}$ depend on both g and ξ:

$$\begin{aligned} \dot{\xi} &= f(\xi), \\ \boldsymbol{y} &= h(g, \xi). \end{aligned}$$

This system captures the model a mobile robot moving around, with h representing the model of a camera, and ξ the motion of people in the environment. Part 2 is concerned on how to model robots much more in detail.

This property is preserved by any linear transformation of the commands.

preserved by: Nuisance 18 ($\mathsf{GL}(n_{\boldsymbol{u}})$)

implies: Assumption 10 (*The system is reversible*)

implies: Assumption 13 (*A known command does nothing*)

implies: Assumption 16 (*Half does half*)

implies: Assumption 15 (*More does more*)

implies: Assumption 11 (*Similar commands have similar effects*)

Assumption 19: Commands determine *kinematic velocities*

preconditions on format: None

In contrast to the previous assumption, here we just assume that the commands determine the kinematic velocities, in the sense that there is a map $f : \mathcal{U} \to \mathbb{R}^k$ such that $f(\boldsymbol{u})$ can be interpreted as kinematic velocities (Figure 6.2).

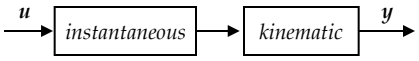


Figure 6.2. Positing that the commands *determine* the kinematic velocities is a much softer assumption that assuming that the commands are kinematic velocities themselves.

This is a much softer assumption, as it does not imply assumptions about the particular representation of the commands, such as Assumption 16 (*Half does half*), Assumption 15 (*More does more*) and Assumption 11 (*Similar commands have similar effects*). Consequently, this assumption is preserved by all instantaneous transformations of the commands.

implies: Assumption 10 (*The system is reversible*)

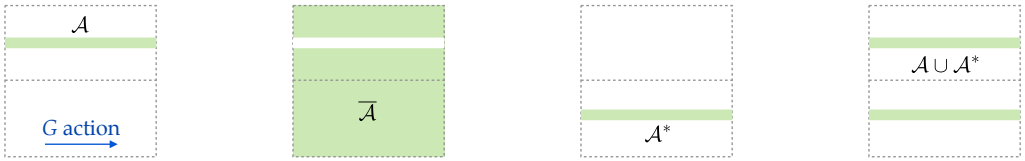
implies: Assumption 13 (*A known command does nothing*)

preserved by: Nuisance 16 ($\mathsf{Aut}(\mathcal{U})$)

6.3. Remarks

Most of these semantic assumptions are constraints to the class of systems to which the world can belong. That is, they describe a subset of $\mathcal{D}(\mathcal{Y};\mathcal{U})$. Classical properties of dynamical systems (such as linearity) also correspond to subsets of $\mathcal{D}(\mathcal{Y};\mathcal{U})$. However, some semantic assumptions are joint constraints among dynamics and task. For example, the concept of saliency (Assumption 1) requires that a goal is specified.

For the semantic assumptions that can describe a subset $\mathcal{A}$ of $\mathcal{D}(\mathcal{Y};\mathcal{U})$, the largest group that preserves them is what was called the stabilizer (Definition C.39) of the subset $\mathcal{A}$: $G = \mathsf{stab}^{\mathcal{D}^\star(\mathcal{Y};\mathcal{U})}(\mathcal{A})$, where $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ is the set of all representation nuisances. Clearly G does not identify $\mathcal{A}$ completely. In fact, G is also the stabilizer of the complement $\overline{A}$.



(a) $\mathcal{A}$ **(b)** $\overline{\mathcal{A}}$ **(c)** $\mathcal{A}^*$ **(d)** $\mathcal{A} \cup \mathcal{A}^*$

Figure 6.1. One representation nuisance does not pinpoint the semantic assumption exactly, as there are other assumptions that are preserved by the same group action.

As an example, consider the two assumptions:

$$\mathcal{A}: \quad \text{“larger values are more salient”,}$$

$$\overline{\mathcal{A}}: \quad \text{“larger values are } \textbf{not} \text{ more salient”.}$$

These are both conserved by monotonic transformations of the sensel values.

Moreover, there is often a “dual” of the semantic assumption which is preserved by the same group as well. In this case, it is

$$\mathcal{A}^*: \quad \text{“}\textbf{smaller} \text{ values are more salient”.}$$

Note that $\mathcal{A}^* \neq \overline{\mathcal{A}}$.

If we consider semantic assumptions as subsets of $\mathcal{D}(\mathcal{Y}; \mathcal{U})$, then we can create new assumptions by using the operations of set union and intersection (Figure 6.1). For example, we can define

$$\mathcal{A} \cup \mathcal{A}^*: \quad \text{“either larger or smaller values are more salient”,}$$

which is still a useful assumption, but leaves to the agent of figuring out one bit of information.

CHAPTER 7

A Catalog of Representation Nuisances

This chapter provides a catalog of representation nuisances. For each representation nuisance, the catalog reports whether there is an interpretation in terms of a semantic assumption (described in the previous chapter).

This has two uses: for analysis*, one should first compute what are the agent's symmetries, and then look up in this list what it means regarding the semantic assumptions; for* synthesis*, one should look up here what are the symmetries that the agent needs to show to prove that it is independent of certain semantic assumptions.*

7.1. Legend

preconditions on format: These are the assumptions about the *format* of the data (i.e., sets $\mathcal{Y}$ and $\mathcal{U}$) that need to be verified for this nuisance to make sense. For example, one can apply a permutation to the observations only if the observations are composed by discrete sensels, and not if they are a continuous field. Vice versa, one can apply a diffeomorphism only if the observations are a continuous field on some Riemannian manifold.

group: This is the group considered (or one isomorphic to it).

action: This is a description of the group action on commands/observations.

preserves: Some property or statistics that is *invariant* with respect to this nuisance.

equivariant: Some property or statistics that is *equivariant* with respect to this nuisance.

perturbs: Some property or statistics that is *perturbed* by this nuisance.

is largest preserving: Some semantic assumption, or system property for which this group is

the largest group that preserves it.

7.2. Nuisances Acting on the Observations

Nuisance 1: $\mathsf{Perm}(n_y)$ *– Permutations of the sensels*

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

group: $\mathsf{Perm}(n_y)$

action: $y_i \mapsto y_{\pi(i)}$, $\pi \in \mathsf{Perm}(n_y)$

This is the basic example of a representation nuisance: would your agent work if the sensels were scrambled?

If an agent has some assumptions on the identity of any signal (e.g., the first sensel encode the reward) then it will only be invariant with respect to a subgroup of this group.

Note that the observations space $\mathcal{Y}$ is supposed to be a set of discrete sensels, each taking value in the same set $\overline{\mathcal{Y}}$. This nuisance would not apply to the case where the observations are an infinite dimensional field.

equivariant: Pairwise statistics such as the correlation or covariance matrix are equivariant with respect to permutations.

is largest preserving: Assumption 4 (*Sensels have similar statistics*)

Nuisance 2: $\mathsf{GL}(n_y)$ *– Linear transformations of the observations*

preconditions on format: $\mathcal{Y} = \mathbb{R}^{n_y}$

A generic linear transformation of the observations can represent a variety of filtering operations.

preserves: Linearity and bilinearity of the dynamics.

equivariant: Covariance matrix (transforms as $\mathbf{P} \mapsto \mathbf{APA}^T$)

perturbs: Correlation matrix

Nuisance 3: $\mathsf{Homeo}(\mathcal{Y})$ – *Continuous transformations of the observations*

preconditions on format: $\mathcal{U}$ is a topological space

group: $\mathsf{Homeo}(\mathcal{Y})$

action: $\boldsymbol{y} \mapsto \varphi(\boldsymbol{y})$

is largest preserving: Assumption 8 (*Observations are continuous in the states*)

Nuisance 4: $\mathsf{Isom}(\mathcal{Y})$ – *Isometries of the observations*

preconditions on format: $\mathcal{Y}$ is a metric space

group: $\mathsf{Isom}(\mathcal{Y})$

action: $\boldsymbol{y} \mapsto \varphi(\boldsymbol{y})$

Invariance to isometries is a natural requirement in many problems. It can be often interpreted as an invariance to the data reference frame.

Note: do not confuse with Nuisance 5 ($\mathsf{Isom}(\mathcal{S})$*) .*

preserves: Assumption 3 (*Observations have "continuous" dynamics*)

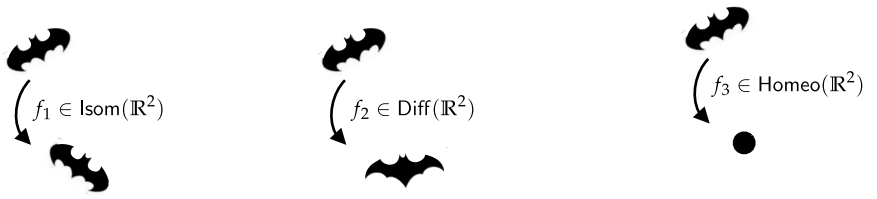
Nuisance 5: $\mathsf{Isom}(\mathcal{S})$ – *Isometries of the observations field*

preconditions on format: $\mathcal{Y}$ is a field on $\mathcal{S}$

group: $\mathsf{Isom}(\mathcal{S})$

action: $y(s) \mapsto y(\varphi(s))$

We assume that the observations $\boldsymbol{y}$ are a function over some manifold $\mathcal{S}$, and we consider the isometries of $\mathcal{S}$. An example of this nuisance would be to mount a robot's camera



(a) Isometries **(b)** Diffeomorphisms **(c)** Homeomorphisms

Figure 7.1. Difference between isometries, homeomorphisms, and diffeomorphisms, in the case $\mathcal{S} = \mathbb{R}^2$.

upside down. Isometries are a relatively small set of transformations, and typically finite-dimensional (Figure 7.1). For example, the isometries of the hyper sphere $\mathbb{S}^{n-1}$ are the orthogonal transformations $\mathsf{O}(n)$ (D.2); the isometries of $\mathbb{R}^n$ form the Euclidean group $\mathsf{E}(n)$ (Definition D.3).

Note: do not confuse with Nuisance 4 ($\mathsf{Isom}(\mathcal{Y})$*) .*

equivariant: Isometries commute with many operations that are expressed through the metric of $\mathcal{S}$, such as spherical smoothing of an image (Definition 19.1).

is largest preserving: Assumption 7 (*The spatial field is homogenous*)

Nuisance 6: $\mathsf{Diff}(\mathcal{S})$ *– Diffeomorphisms of the observations*

preconditions on format: $\mathcal{Y}$ is a field on $\mathcal{S}$

group: $\mathsf{Diff}(\mathcal{S})$

action: $y(s) \mapsto y(\varphi(s))$

We assume that the observations $\boldsymbol{y}$ are a function over some manifold $\mathcal{S}$, and we consider all diffeomorphisms of $\mathcal{S}$. This is a much larger set of transformations than only the isometries of $\mathcal{S}$; diffeomorphisms are an infinite-dimensional topological group (not a Lie

topological group: A group whose operations are continuous in a given topology. See

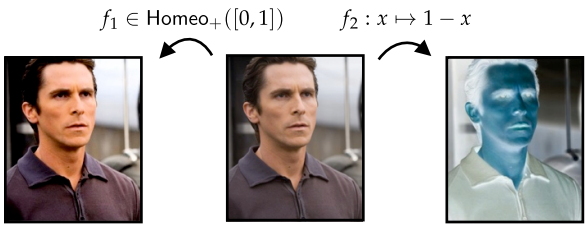


Figure 7.2. Contrast transformations. The human visual system is extremely robust to contrast transformations, represented by the action of the group $\mathsf{Homeo}_+(\overline{\mathcal{Y}})$.

group).

preserves: Because they are homeomorphisms, diffeomorphisms preserve the topology of $\mathcal{S}$. In addition to that, they preserve the differentiability of the signal; this allows considering statistics of the spatial gradient $\nabla_s y(s)$ (Figure 7.1).

perturbs: Any statistics that depends on the metric of $\mathcal{S}$ is perturbed by diffeomorphisms.

is largest preserving: Assumption 6 (*The observations correspond to a spatial field*)

Nuisance 7: $\mathsf{Homeo}_+(\overline{\mathcal{Y}})$ *– Acting jointly on all observations*

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

preconditions on format: $\overline{\mathcal{Y}}$ is totally ordered

preconditions on format: *(equivalently:* $\mathcal{Y}$ is a field on $\mathcal{S}$)

group: $\mathsf{Homeo}_+(\overline{\mathcal{Y}})$

action: $y_i \mapsto f(y_i)$, for $f \in \mathsf{Homeo}_+(\overline{\mathcal{Y}})$

action: *(equivalently:* $y(s) \mapsto f(y(s))$, for $f \in \mathsf{Homeo}_+(\mathbb{R})$)

This nuisance is often called a *contrast* transformation (Figure 7.2).

Nuisance 8: $\mathsf{Aff}(\mathbb{R})^{n_y}$ *– Affine transformation of the single sensel*

Definition C.43.

preconditions on format: $\mathcal{Y} = \mathbb{R}^{n_y}$

group: $\mathsf{Aff}(\mathbb{R})^{n_y}$

action: $y_i \mapsto a_i y + b_i$, for $a_i \neq 0$, $b_i \in \mathbb{R}$

preserves: This is the largest nuisance that preserves the correlation between sensel values.

Nuisance 9: $\mathsf{Perm}(n_{\boldsymbol{y}}) \times \mathsf{Homeo}_+(\overline{\mathcal{Y}})^{n_y}$ *– Positive saliency*

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

preconditions on format: $\overline{\mathcal{Y}}$ is totally ordered

group: $\mathsf{Perm}(n_{\boldsymbol{y}}) \times \mathsf{Homeo}_+(\overline{\mathcal{Y}})^{n_y}$

action: $y_i \mapsto f_i(y_{\pi(i)})$

is largest preserving: Assumption 2 (*Larger (or smaller) values are more salient*)

Nuisance 10: $\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{Y})$ *– Dynamical nuisances with finite memory acting on observations*

preconditions on format: None

group: $\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{Y})$

action: $\boldsymbol{y} \mapsto \boldsymbol{D}\boldsymbol{y}$ (system-signal product), for $\boldsymbol{D} \in \mathcal{D}^\star_{\mathsf{fm}}(\mathcal{Y})$

is largest preserving: Assumption 17 (*The world has finite memory*)

perturbs: Any statistics of $\boldsymbol{y}$ defined as an expectation over time (e.g., correlation) is disrupted by this and other dynamical nuisances.

Nuisance 11: $\textsc{dsmplti}(1)$ *– Stable, minimum phase systems*

preconditions on format: $\mathcal{Y} = \mathbb{R}^{n_y}$

action: $y_i \mapsto \boldsymbol{D}y_i$, for $\boldsymbol{D} \in \textsc{dsmplti}(1)$ a stable, minimum phase system

intuition: This is a dynamical nuisance, which acts on the observations by filtering them using a discrete-time stable, minimum-phase finite-dimensional linear time-invariant dynamical system. While this is a mouthful, this is the minimum set of attributes to describe a linear system such that it is always possible to find a causal inverse. See Example 4.32.

perturbs: Even if the same dynamical system filters all the sensels, most statistics such as the correlation are perturbed by this nuisance.

preserves: It is possible to work in the Laplace domain to find robust statistics. Let $\mathcal{L}$ indicate the Laplace transform of a signal. Then the transform of the filtered signal is the product of the transfer function of the filter and the transform of the signal:

$$\mathcal{L}(\boldsymbol{D}y_i) = \mathsf{TF}(\boldsymbol{D})\mathcal{L}(y_i).$$

This implies that one might take statistics such as the ratios $\mathcal{L}(y_i)/\mathcal{L}(y_j)$ to be invariant with respect to the action of this nuisance.

Nuisance 12: $\mathsf{Aut}(\overline{\mathcal{Y}})^{n_u}$ *– Known labeling of observations*

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

group: $\mathsf{Aut}(\overline{\mathcal{Y}})^{n_y}$

action: $y_i \mapsto f_i(y_i)$, for $f_i \in \mathsf{Aut}(\overline{\mathcal{Y}})$

The action of this nuisance changes the representation of each sensel in an independent way. The order of the sensels is not changed, and no sensel values are mixed together.

perturbs: Every simple statistics like the correlation is perturbed by this nuisances.

---

transfer function: An equivalent representation of linear systems from an input-output perspective.

preserves: Mutual information and related nonparametric statistics are preserved.

Nuisance 13: $\mathsf{Aut}(\mathcal{Y})$ *– Instantaneous transformations of the observations*

preconditions on format: None.

group: $\mathsf{Aut}(\mathcal{Y})$

action: $\boldsymbol{y} \mapsto g(\boldsymbol{y})$, for $g \in \mathsf{Aut}(\mathcal{Y})$

preserves: Any mutual information-like statistics between observations and commands.

is largest preserving: Assumption 9 (*White noise*)

7.3. Nuisances Acting on the Commands

Nuisance 14: $\mathsf{Homeo}(\mathcal{U})$ *– Continuous transformations of the commands*

preconditions on format: $\mathcal{U}$ is a topological space

group: $\mathsf{Homeo}(\mathcal{U})$

action: $\boldsymbol{u} \mapsto f(\boldsymbol{u})$, for $f \in \mathsf{Homeo}(\mathcal{U})$

This is the largest nuisance that still preserves the intuitive notion that similar command have similar effect.

perturbs: Properties such as linearity are perturbed by this nuisance.

is largest preserving: Assumption 11 (*Similar commands have similar effects*)

Nuisance 15: $\mathsf{Aut}(\overline{\mathcal{U}})^{n_u}$ *– Any transformation of the single command*

preconditions on format: $\mathcal{U} = \overline{\mathcal{U}}^{n_u}$

group: $\mathsf{Aut}(\overline{\mathcal{U}})^{n_u}$

action: $u_i \mapsto f_i(u_i)$, for $f_i \in \mathsf{Aut}(\overline{\mathcal{U}})$

intuition: This nuisance changes the representation of each command but does not mix them together. To be invariant with respect to this nuisance, it is likely that the agent must use a nonparametric representation of the commands.

Nuisance 16: $\mathsf{Aut}(\mathcal{U})$ *– Instantaneous transformations of the commands*

preconditions on format: None

group: $\mathsf{Aut}(\mathcal{U})$

action: $\boldsymbol{u} \mapsto f(\boldsymbol{u})$, for $f \in \mathsf{Aut}(\mathcal{U})$

This is the largest set of "static" representation nuisances.

is largest preserving: Assumption 10 (*The system is reversible*)

is largest preserving: Assumption 12 (*One command does nothing*)

is largest preserving: Assumption 19 (*Commands* determine *kinematic velocities*)

Nuisance 17: $\mathsf{Aut}(\mathbb{R}_{\circ}^{+})^{n_{\boldsymbol{u}}}$ *– Nonlinear transformations preserving symmetry*

preconditions on format: $\mathcal{U} = \mathbb{R}^{n_{\boldsymbol{u}}}$

group: $\mathsf{Aut}(\mathbb{R}_{\circ}^{+})^{n_{\boldsymbol{u}}}$

action: $u_i \mapsto \mathsf{sgn}(u_i) f_i(|u_i|)$, for $f_i \in \mathsf{Aut}(\mathbb{R}_{\circ}^{+})$

This is an example that shows that often a simple semantic assumption corresponds to a quite complicated representation nuisance. This is the largest nuisance that preserves the semantic assumption that $-\boldsymbol{u}$ has the "opposite" effect than $+\boldsymbol{u}$ (Subsection 6.2). In this case the notation is quite opaque. The group is $\mathsf{Aut}(\mathbb{R}_{\circ}^{+})^{n_{\boldsymbol{u}}}$ because it consists of $n_{\boldsymbol{u}}$ independent transformations acting on each command separately. Each command is transformed in a symmetric way. The constraint that the map φ acting on a single command

must preserve is that it must be an odd function: $\varphi(u) = -\varphi(-u)$. This implies that $\varphi(u)$ can be written as

$$\varphi(u) = \mathsf{sgn}(u_i) f(|u_i|)$$

for some arbitrary map $f : \mathbb{R}^+_\circ \to \mathbb{R}^+_\circ$. There are no constraints on f, therefore it can be any automorphism of the positive reals $\mathbb{R}^+_\circ$.

is largest preserving: Assumption 14 (*Minus does the opposite*)

Nuisance 18: $\mathsf{GL}(n_{\boldsymbol{u}})$ *– Linear transformations of the commands*

preconditions on format: $\mathcal{U} = \mathbb{R}^{n_{\boldsymbol{u}}}$

group: $\mathsf{GL}(n_{\boldsymbol{u}})$

action: $\boldsymbol{u} \mapsto \mathbf{A}\boldsymbol{u}$

A linear transformation of the commands preserves many semantic assumptions that often have a physical interpretation. For example, if the commands are velocities, a linear transformation is just a change of reference frame.

is largest preserving: Assumption 18 (*Commands are kinematic velocities*)

Nuisance 19: $\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{U})$ *– Dynamical nuisances with finite memory acting on the commands*

preconditions on format: None

group: $\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{U})$

action: $\boldsymbol{u} \mapsto \boldsymbol{D}\boldsymbol{u}$ (system-signal product), for $\boldsymbol{D} \in \mathcal{D}^\star_{\mathsf{fm}}(\mathcal{U})$

perturbs: "Reversibility", as defined in Subsection 6.2, while preserved by all instantaneous nuisances, is perturbed by this one.

Nuisance 20: $\mathsf{Perm}(n_{\boldsymbol{u}}) \times \mathsf{Homeo}_+(\overline{\mathcal{U}})^{n_u}$ *– Monotonic transformations of single command*

preconditions on format: $\mathcal{U} = \overline{\mathcal{U}}^{n_u}$

preconditions on format: $\overline{\mathcal{U}}$ is a topological space

group: $\mathsf{Perm}(n_{\boldsymbol{u}}) \times \mathsf{Homeo}_+(\overline{\mathcal{U}})^{n_u}$

action: $u_i \mapsto f_i(u_{\pi(i)})$, for $\pi \in \mathsf{Perm}(n_{\boldsymbol{u}})$ and $f_i \in \mathsf{Homeo}_+(\overline{\mathcal{U}})$.

is largest preserving: Assumption 15 (*More does more*)

CHAPTER 8

Tasks for Bootstrapping Agents

This chapter describes a few examples of tasks for bootstrapping agents. Defining bootstrapping tasks is not easy, as they must refer to uninterpreted observations and commands, and be (ideally) completely invariant to representation nuisances. This chapter discusses a series of tasks which are relevant to any bootstrapping agent, and how they form a hierarchy that probes the increasing required skills from the agent.

8.1. Challenges in Defining Bootstrapping Tasks

The first challenge in defining bootstrapping tasks is that they must be formulated with respect to uninterpreted streams of observations and commands. It is not possible to describe a task such as "fetch me a beer from the fridge", unless one also defines what is a "beer", a "fridge", and "fetch" in terms of the uninterpreted vectors $\boldsymbol{y}$ and $\boldsymbol{u}$.

A related challenge is that the task should also be *achievable* for multiple worlds. For example, a task such as "stabilize the observations to $\boldsymbol{y} = \mathbf{0}$" cannot be achievable if $\mathbf{0}$ is not part of the agent's world, or if there is some observations noise.

Another challenge, already discussed before (Section 5.6), is that the description of a task by error functions or performance measures often implies imposing some extra semantic assumptions about the system.

These problems are not solved at this time and affect most, if not all, the tasks described in this chapter.

8.2. Tasks for Disembodied Agents

8.2.1. *Prediction*

If an agent can predict the future observations (in as much as it is possible due to uncertainty), then it has learned a good model for the world. The prediction problem can be formulated in many variations, differing according to the prediction horizon, the quality of the prediction, and the way that performance is measured. These three aspects are independent of each other. Even for the simple problem of prediction one can find more than a dozen variants, each requiring different skills from the agent.

Regarding the prediction horizon, it is useful to distinguish at least between three qualitatively different cases: "instantaneous, "short" horizon, and "long" horizon:

"*instantaneous*" *prediction,* in the sense of predicting the derivative of the observations or similar instantaneous quantity, is the easiest and can usually be done using a reduced model of the dynamics, such as a linearized model;

"short" horizon prediction will refer to predicting the future observations based on the current observations only, without accessing previous memory. Formally, this is the case where $\boldsymbol{y}_t$ is a sufficient statistics for predicting $\boldsymbol{y}_{t+\Delta}$;

"long" horizon prediction requires accessing memories of previous observations.

It is also useful to define *relaxed* prediction problems, in the cases where an approximate prediction is sufficient. The following are three levels on a smooth qualitative–quantitative spectrum:

Qualitative The agent must predict which sensels will change in $\boldsymbol{y}_{t+\Delta}$ compared with $\boldsymbol{y}_t$.

Sign-prediction The agent must predict which sensels will change, and whether their values increase or decrease.

This assumes that it is possible to find an order for the sensel values (compare

Assumption 2 (Larger (or smaller) values are more salient)).

Quantitative The agent must predict the future observations $\boldsymbol{y}_{t+\Delta}$ exactly.

Finally, one must decide how to measure performance. Here the previous discussion on error functions is relevant (compare Section 5.6).

8.2.2. *Skills built on top of prediction*

If the agent is able to predict the next observations, then it is possible to implement more complicated task on top of this ability (Figure 8.1).

Define an *anomaly* a mismatch between predictions and observations. Anomaly detection is an important skill for bootstrapping agents, as it makes the agent aware that its model of the world is incomplete.

If the agent is embodied in a robotic body, most of the changes in the observations are due to self-motion, in a way which can be learned. Occasionally some of the changes are due to other agents moving in space, or other unmodeled effects in the dynamics.

By looking for coherent temporal traces of the anomaly signal, it is possible to detect another agent moving in the same environment.

If the same sensel appears to be consistently anomalous with respect to a learned model, it is likely that it is faulty. Fault detection can be realized by averaging the anomaly signal [64].

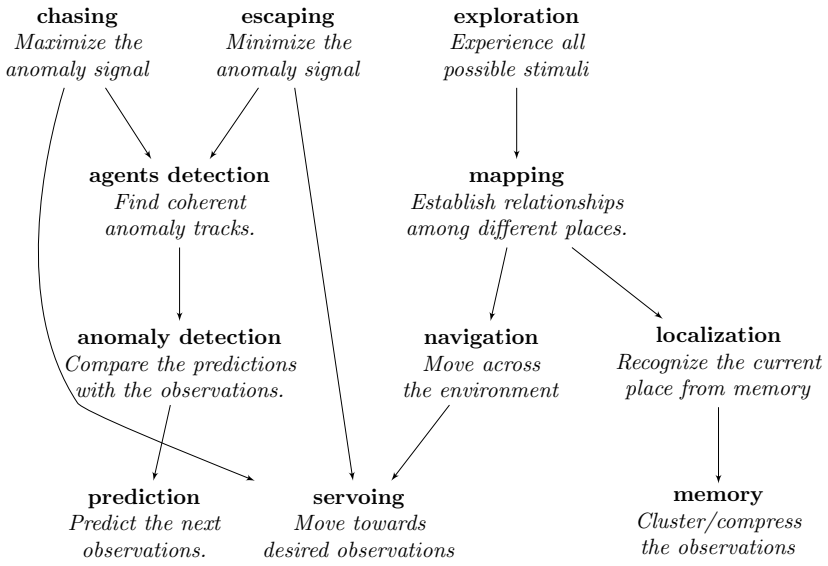


Figure 8.1. A hierarchy of tasks for bootstrapping agents.

8.3. Tasks for Embodied Agents

Several specific tasks can be defined for agents embodied in a robotic body. To define these tasks, it is assumed that the observations represent the output of a sensor attached to the robot.

The concept of "place" can be defined directly in the observations space, as a neighborhood of a given observation vector:

$$\mathsf{Neighbors}(\boldsymbol{y}_\circ) = \{\boldsymbol{y} \in \mathcal{Y} \mid d^{\mathcal{Y}}(\boldsymbol{y}, \boldsymbol{y}_\circ) < \alpha\}.$$

This is a generalization of the idea of working in "image space" in applications such as visual servoing. However, care must be given as to the way one defines the metric space, because committing to a certain distance $d^{\mathcal{Y}}$ might carry over certain semantic assumptions. An alternative definition of "place" is given by Kuipers and colleagues as part of the spatial semantic hierarchy [65, 66]: if one has already a policy, then a "place" can be defined as a subset of the states in which the policy has a predictable result.

8.3.1. *Servoing*

Going to a designated place, described by some observation vector $\check{\boldsymbol{y}}$, is a task that will be used often in Part 2. The difficulty of the task essentially depends on the distance between the current state and the goal state. Just like prediction, it will be useful to distinguish between "short" and "long" horizon instances of the problem.

8.3.2. *Chasing and escaping*

Several tasks can be defined on top of servoing. If the agent can detect moving objects, then one can define *chasing* or *escaping* as servoing of the detection signal: for escaping the target, the detection signal should be minimized (i.e., agent gets farther), while for chasing a target, the anomaly signal should be maximized (i.e., agent gets closer).

8.3.3. *Localization and mapping*

Other spatial abilities can be defined on top of the concept of place and basic tasks such as servoing. Localization (which place is this?) and metrical/topological mapping (which sequence of actions brings from one place to another?) based on minimal semantics have been demonstrated by Milford [67] in a bio-plausible setting.

All these tasks form a hierarchy that goes from basic skills to complex behaviors (Figure 8.1).

Part 2

Learning Models of Robotic Sensorimotor Cascades

CHAPTER 9

Robot Sensors and Actuators

This chapter introduces the basic notations for describing robot motions and robot sensors. Three "canonical" robot sensors are described and their dynamics are derived.

Table 9.1. Symbols used in this chapter

Motion		
$\mathcal{Q}$	$\leq \mathrm{SE}(3)$	Configuration space.
$\boldsymbol{q}$	$\in \mathcal{Q}$	Configuration.
$\mathbf{R}$	$\in \mathrm{SO}(3)$	Attitude (orientation).
$\boldsymbol{t}$	$\in \mathbb{R}^3$	Position.
$\boldsymbol{v}$	$\in \mathbb{R}^3$	Linear velocity in body frame.
$\boldsymbol{\omega}$	$\in \mathbb{R}^3$	Angular velocity in body frame.
$\hat{\boldsymbol{\omega}}$	$\in \mathrm{so}(3)$	Angular velocity as a matrix.
Sensors		
$\mathcal{S}$	(a manifold)	Sensel space.
$\mathcal{O}$	(a field)	Sensor output space (often $\mathbb{R}$).
$\mathsf{Images}(\mathcal{S})$		All functions from $\mathcal{S}$ to $\mathcal{O}$.
m		Inner product tensor on $\mathcal{S}$.
$\mathcal{F}$	$: \mathbb{R}^3 \to \mathbb{R}$	The field sampled by a field-sampler.
σ^s	> 0	Distance to obstacles in direction s.
μ^s	> 0	"Nearness" (inverse of distance).
Environment statistics		
Maps	$= \mathsf{Shapes} \times \mathrm{SE}(3)$	Parametrization of the environment.
$\boldsymbol{p}$	$\in \mathrm{SE}(3)$	Environment pose.
$\boldsymbol{s}$	$\in \mathsf{Shapes}$	Environment shape.
p_{T}	$\in \mathsf{ProbMeasures}(\mathsf{Maps})$	Training distribution.
$\mathsf{Sym}(p_{\mathrm{T}})$	$\leq \mathrm{SE}(3)$	Symmetries of the training distribution.

9.1. Robot Motion

This section introduces the minimal notation necessary for describing robot kinematics. General references for robot dynamics are Murray et al. [68], Siciliano et al. [69], and Springer's *Handbook of Robotics* [3]. The best reference for probabilistic sensor modeling is Thrun et al. [16].

We assume that the underlying dynamics is a rigid body controlled in velocity. Let $\mathcal{Q}$ be the *configuration space* in which the robot moves. Assume that $\mathcal{Q}$ is a subgroup of $\mathrm{SE}(3)$, such as $\mathrm{SE}(3)$ itself, $\mathrm{SE}(2)$ (planar motion), $\mathrm{SO}(3)$ (pure rotations), $\mathbb{R}^3$ (pure translations), or $\mathbb{R}$, for a robot constrained to live on a straight line.

The configuration $\boldsymbol{q} \in \mathcal{Q}$ can be written as a function of the position $\boldsymbol{t} \in \mathbb{R}^3$ and the attitude $\mathbf{R} \in \mathrm{SO}(3)$. The linear velocity $\boldsymbol{v} \in \mathbb{R}^3$ is a three-dimensional vector, and it is expressed in the body frame. The angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ is also a three-dimensional vector giving the instantaneous angular velocities around the three principal axes in the body frame. Using the "hat map" (Definition E.12), the vector is mapped to an antisymmetric matrix $\hat{\boldsymbol{\omega}} \in \mathsf{so}(3)$.

LEMMA 9.1 (Kinematics of rigid body). *Let the configuration space* $\mathrm{SE}(3)$ *be parametrized with the position* $\boldsymbol{t} \in \mathbb{R}^3$ *and the attitude* $\mathbf{R} \in \mathrm{SO}(3)$*. Then the dynamics are*

$$\begin{cases} \dot{\boldsymbol{t}} &= \mathbf{R}\,\boldsymbol{v}, \\ \dot{\mathbf{R}} &= \mathbf{R}\,\hat{\boldsymbol{\omega}}. \end{cases}$$

REMARK 9.2. Note that both $\boldsymbol{v} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ are expressed in the body frame. For

example, the constant velocities

$$\begin{cases} \boldsymbol{v} &= (1,0,0)^T, \\ \boldsymbol{\omega} &= (0,1,1)^T. \end{cases}$$

describe an upward counter clockwise spiral motion.

9.2. Exteroceptive Robot Sensors

We consider sensors composed of a set of sensory elements (*sensels*) that are physically related to one another. We write the observations as

$$\boldsymbol{y} = \{y^s\}_{s\in\mathcal{S}},$$

where s is the sensel position ranging over the *sensel space* $\mathcal{S}$. Sometimes, when there are already too many indices around, the notation

$$\boldsymbol{y} = \{y(s)\}_{s\in\mathcal{S}}$$

is used instead.

EXAMPLE. In the case of a camera, the sensels span the visual sphere $\mathbb{S}^2$; s corresponds to a pixel's direction, and y^s to the intensity measured by that pixel.

The sensel space $\mathcal{S}$ interacts with the configuration space $\mathcal{Q}$.

DEFINITION 9.3 (Sensel space). The *sensel space* $\mathcal{S}$ is a manifold on which there is defined an action of the configuration space. For every $\boldsymbol{q} \in \mathcal{Q}$ and $s \in \mathcal{S}$, we can define the element $\boldsymbol{q} \cdot s \in \mathcal{S}$, and $\boldsymbol{q}_1 \cdot (\boldsymbol{q}_2 \cdot s) = (\boldsymbol{q}_1\boldsymbol{q}_2) \cdot s$.

EXAMPLE. For example, for a pan-tilt-roll "robotic" camera, $\mathcal{S} = \mathbb{S}^2$, $\mathcal{Q} = \mathrm{SO}(3)$, and the action $\boldsymbol{q} \cdot s$ corresponds to applying the rotation $\boldsymbol{q}$ to $s \in \mathbb{S}^2$.

The sensel values returned by the sensors lie in a certain *output space* $\mathcal{O}$. For simplicity, we will just assume $\mathcal{O}$ to be $\mathbb{R}$; everything can be extended to more complicated output spaces.

EXAMPLE. For a color camera, $\mathcal{O}$ would be the RGB space; for a range-finder, $\mathcal{O}$ would be $\mathbb{R}^+$ (distances).

At each time, the sensor returns the observations as a function from $\mathcal{S}$ to $\mathbb{R}$. Independently of the sensor, this function is called "image", and the set of all functions from $\mathcal{S}$ to $\mathbb{R}$ is written as $\mathsf{Images}(\mathcal{S})$. Sometimes, it will be needed to assume that these are continuous or differentiable.

We assume that there is an inner product defined on $\mathsf{Images}(\mathcal{S})$ that allows to measure the dissimilarity of two observations by the norm induced by the inner product**.** We use the tensor notation to represent the inner product. Let y^s represent the value of $\boldsymbol{y}$ at the sensel $s \in \mathcal{S}$. We put the index up in "y^s" with analogy to covariant tensors. Given two observations $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$, their inner product $\langle\langle \boldsymbol{y}_1, \boldsymbol{y}_2 \rangle\rangle$ can be represented by contracting y_1^s, y_2^v with a $(0,2)$ tensor m_{sv}: $\langle\langle \boldsymbol{y}_1, \boldsymbol{y}_2 \rangle\rangle = m_{sv} y_1^s y_2^v$. Here, using the Einstein convention, summation (integration) is assumed over indices that appear twice (up and down). The inner product allows to define a norm $\|\boldsymbol{y}\|^2 = \langle\langle \boldsymbol{y}, \boldsymbol{y} \rangle\rangle$ as well as a conjugation operation $\boldsymbol{y} \mapsto \boldsymbol{y}^*$ by $y_s^* = m_{sv} y^v$.

9.2.1. *Maps and relative sensors*

Call "map" everything needed to compute the sensor output, apart from the robot pose.

EXAMPLE. For a range-finder, the map includes the 3D environment structure; for a camera, it includes the texture, reflectance, and illumination information as well.

We use a construction typical of stochastic geometry [70, 71]: we assume that the set of maps Maps can be factorized into a "shape" and "pose" component, in the sense that, for each map, there are many others that share the same shape (including color, texture, etc.), but they are rototranslated. Therefore, let the map space be factorized as $\mathsf{Maps} = \mathsf{Shapes} \times \mathrm{SE}(3)$, where Shapes is called *shape space.* An element of Maps is a tuple $\langle \boldsymbol{s}, \boldsymbol{p} \rangle$, with $\boldsymbol{s} \in \mathsf{Shapes}$ and $\boldsymbol{p} \in \mathrm{SE}(3)$.

A "relative" sensor is one that can be "carried" by the robot.

DEFINITION 9.4 (Relative sensor). Given a sensel space $\mathcal{S}$, the configuration space $\mathcal{Q}$, and a shape-pose space $\mathsf{Maps} = \mathsf{Shapes} \times \mathrm{SE}(3)$, the map $y : \mathsf{Maps} \times \mathcal{Q} \times \mathcal{S} \to \mathcal{O}$ corresponds to a *relative sensor* if the following two properties hold for all $x \in \mathcal{Q}$:

$$y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, s) = y(\langle \boldsymbol{s}, x\boldsymbol{p} \rangle, x\boldsymbol{q}, s), \qquad \text{[P1]} \tag{9.1}$$

$$y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, s) = y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}x^{-1}, x \cdot s). \qquad \text{[P2]} \tag{9.2}$$

Property P1 corresponds to the fact that there is an intrinsic ambiguity in choosing the frame of reference. The world and the robot have both a pose with respect to some fixed coordinate frame, but the output of the sensor depends only of the *relative* pose $\boldsymbol{q}^{-1} \cdot \boldsymbol{p}$ (let $x = \boldsymbol{q}^{-1}$ in (9.1) to see this).

Property P2 describes the fact that the robot is "carrying" the sensor: ultimately the output at sensel s depends only on $\boldsymbol{q} \cdot s$, therefore it is invariant if we apply x to s and multiply $\boldsymbol{q}$ by x on the right.

9.3. Three Canonical Robot Sensors

This section derives the models for three "canonical" robot sensors: field-samplers, range finders, and cameras (Table 9.2). The goal is understanding in what way these are similar or dissimilar.

9.3.1. *Simplifications in the analysis*

There are several simplifications done here and in the next chapter, which only complicate the exposition but would not invalidate the main conclusions.

noiseless The models do not include noise explicitly. All learning procedures are robust to additive white noise on the observations.

continuous time Real sensors give observations in discrete time; for simplicity, here they are described by continuous-time ODE s.

continuous space Real sensor have a discrete number of sensels. The models presented here return observations on some continuous manifold $\mathcal{S}$.

no field-of-view limitations Most sensors has a limited field of view. The dynamics of the sensels at the border are clearly different than the dynamics inside. For the next three chapter, this is ignored, while it is dealt with explicitly in Section 13.1.1.

static environment In principle, any model of a sensor should include all possible phenomena that might affect the observations, such as moving objects in the environment. The models described here only cover fixed environments.

9.3.2. *Field-samplers*

ODE: Ordinary Differential Equation

Table 9.2. Dynamics of canonical robot sensors

	$\mathcal{S}$	*meaning of* $y(s)$	*dynamics*
field sampler	$\mathbb{R}^3$	intensity of a field	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \nabla_i y^s \boldsymbol{v}^i$
camera	$\mathbb{S}^2$	luminance of a sensel	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \mu^s \nabla_i y^s \boldsymbol{v}^i$
range-finder	$\mathbb{S}^2$	distance readings	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + (\nabla_i \log y^s - s_i^*) \boldsymbol{v}^i$

These equations are valid for sensels far from occlusions and the border of the field of view. Note that the dynamics of the three sensors is formally the same for rotations.

DEFINITION 9.5 (Ideal field-sampler). Let the sensels space be $\mathcal{S} = \mathbb{R}^3$. The sensor $\boldsymbol{y}$ is a field-sampler if there exists a field $\mathcal{F} : \mathbb{R}^3 \to \mathbb{R}$ such that

$$y^s = \mathcal{F}(\boldsymbol{t} + \mathbf{R}\, s),$$

where $\boldsymbol{t} \in \mathbb{R}^3$ and $\mathbf{R} \in \mathrm{SO}(3)$ are the sensor position and attitude.

The field-sampler is general enough to represent olfactory and temperature sensors (see, e.g., [72, 73]).

PROPOSITION 9.6 (Dynamics of field-samplers). *The dynamics of a field-sampler are bilinear in* $\boldsymbol{y}$ *and the sensor velocities* $\boldsymbol{v}, \boldsymbol{\omega}$:

$$\dot{y}^s = (\nabla_i y^s) \boldsymbol{v}^i + (s \times \nabla y^s)_i \boldsymbol{\omega}^i. \tag{9.1}$$

PROOF. The derivative of the observations are given by

$$\dot{y}^s = \nabla \mathcal{F}|_{z = \boldsymbol{t} + \mathbf{R}s} \cdot (\mathbf{R}\boldsymbol{v} + \mathbf{R}\hat{\boldsymbol{\omega}} s), \tag{9.2}$$

where we used the fact that $\dot{\boldsymbol{t}} = \mathbf{R}\boldsymbol{v}$ and $\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\omega}}$ (Lemma 9.1). We want to write $\nabla \mathcal{F}$ as a function of y. Note that, inverting the sensor model, we obtain $\mathcal{F}(z) = y(\mathbf{R}^T(\boldsymbol{z} - \boldsymbol{t}))$. Taking the derivative of that relation, we obtain $\nabla \mathcal{F} \cdot x = \nabla y|_{s = \mathbf{R}^T(z - \boldsymbol{t})} \cdot \mathbf{R}^T x$. Substituting

in (9.2), we obtain

$$\begin{aligned} \dot{y}^s &= \nabla y|_{s=\mathbf{R}^T(z-\boldsymbol{t})}\mathbf{R}^T(\mathbf{R}\boldsymbol{v}+\mathbf{R}\hat{\boldsymbol{\omega}}s) \\ &\qquad (\text{Because } \mathbf{R}^T=\mathbf{R}^{-1}) \\ &= \nabla_i y^s(\boldsymbol{v}+\hat{\boldsymbol{\omega}}s)^i \qquad (9.3) \\ &\qquad (\text{Using Lemma E.14}) \\ &= (\nabla_i y^s)\boldsymbol{v}^i + (s\times\nabla y^s)_i\boldsymbol{\omega}^i. \qquad (9.4) \end{aligned}$$

□

9.3.3. *Range finders*

Each reading of an ideal range finder measures the distance from a an origin point (in $\mathbb{R}^3$) to the closest obstacle in a certain direction (in $\mathbb{S}^2$). Real range finders have very rich noise models [16].

PROPOSITION 9.7. *Let y^s be the range reading (distance to the obstacle in direction s). Then the dynamics of y^s are*

$$\dot{y}^s = (\nabla_i \log y^s - s_i^*)\boldsymbol{v}^i + (s\times\nabla y^s)_i\,\boldsymbol{\omega}^i. \qquad (9.5)$$

PROOF. (This proof is due to Shuo Han) The model for the rotation part is analogous to the field-sampler and camera. Hence we are only concerned in proving the result for translation. For clarity, we use the more widespread notation, and let the range readings be σ (rather than $\boldsymbol{y}$). Write $\sigma=\sigma(s,\boldsymbol{t})$ as a function of the direction s and the robot position $\boldsymbol{t}\in\mathbb{R}^3$. Then we have to prove that

$$\frac{\partial}{\partial \boldsymbol{t}}\sigma(s,\boldsymbol{t}) = \nabla\log\sigma(s,0) - s^T.$$

Without loss of generality, we can assume we are computing the derivative at $\boldsymbol{t} = 0$. In a neighborhood of 0, it holds that

$$\|\boldsymbol{t} + \sigma(s)s\| = \sigma\left(\frac{\boldsymbol{t} + \sigma(s)s}{\|\boldsymbol{t} + \sigma(s)s\|}, 0\right), \tag{9.6}$$

as can be seen by geometric inspection of Fig. 9.1. The proof is based on the implicit function theorem applied to the relation (9.6). Define the function $n(v) : \mathbb{R}^3 \to \mathbb{R}^3$ as the vector v normalized by its module: $n(v) \triangleq v/\|v\|$. Then the following holds:

$$F(\sigma, s, \boldsymbol{t}) = \|\boldsymbol{t} + \sigma(s)s\| - \sigma(n(\boldsymbol{t} + \sigma(s)s), 0) = 0.$$

We can compute the derivative $\frac{\partial}{\partial \boldsymbol{t}}\sigma(s, \boldsymbol{t})$ using the implicit function theorem applied to F:

$$\frac{\partial \sigma}{\partial \boldsymbol{t}} = \left(\frac{\partial F}{\partial \sigma}\right)^{-1} \frac{\partial F}{\partial \boldsymbol{t}}.$$

To this end, we first recall that $\frac{\partial}{\partial v}\|v\| = \frac{v^T}{\|v\|}$, and we compute the derivative of $n(v)$ as

$$\begin{aligned}
\frac{\partial}{\partial v} n(v) &= \frac{\partial}{\partial v}\frac{v}{\|v\|} = \frac{I}{\|v\|} + v\frac{\partial}{\partial v}\frac{1}{\|v\|} = \frac{I}{\|v\|} + v\frac{\partial}{\partial v}\frac{1}{\|v\|} = \frac{I}{\|v\|} - \frac{v}{\|v\|^2}\frac{\partial}{\partial v}\|v\| \\
&= \frac{I}{\|v\|} - \frac{v}{\|v\|^2}\frac{v^T}{\|v\|} = \frac{1}{\|v\|}\left(I - \frac{vv^T}{\|v\|^2}\right) = \frac{1}{\|v\|}\left(I - n(v)n(v)^T\right).
\end{aligned}$$

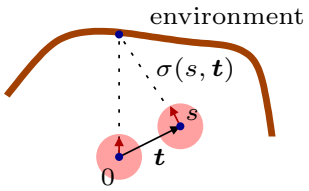


Figure 9.1. Geometry of range-finder sensing, used in the proof for Proposition 9.7.

We use the shortcut $x = \boldsymbol{t} + \sigma(s)s$, and $\sigma_0(s) = \sigma(s, 0)$.

$$\frac{\partial F}{\partial \boldsymbol{t}} = \frac{x^T}{\|x\|} - \nabla_u \sigma_0(u)(1 - uu^T)|_{u=n(x)} \frac{1}{\|x\|} \left(I - n(x)n(x)^T\right).$$

For the other, we simply obtain $\frac{\partial F}{\partial \sigma} = \frac{\partial F}{\partial p} s$. We compute $\frac{\partial \sigma}{\partial \boldsymbol{t}}$:

$$\begin{aligned}
\frac{\partial \sigma}{\partial \boldsymbol{t}} &= -\frac{\partial F/\partial p}{\partial F/\partial \sigma} = -\frac{\frac{x^T}{\|x\|} - \nabla_u \sigma_0(u)(1 - uu^T)|_{u=n(x)} \frac{1}{\|x\|} \left(I - n(x)n(x)^T\right)}{\left(\frac{x^T}{\|x\|} - \nabla_u \sigma_0(u)(1 - uu^T)|_{u=n(x)} \frac{1}{\|x\|} \left(I - n(x)n(x)^T\right)\right) s} \\
&\qquad \text{(Simplifying the } \|x\|\text{.)} \\
&= -\frac{x^T - \nabla_u \sigma_0(u)(1 - uu^T)|_{u=n(x)} \left(I - n(x)n(x)^T\right)}{x^T s - \nabla_u \sigma_0(u)(1 - uu^T)|_{u=n(x)} \left(I - n(x)n(x)^T\right) s}.
\end{aligned}$$

This expression is valid in a neighborhood of $\boldsymbol{t} = 0$. We now compute the limit as $\boldsymbol{t} \to 0$. We have

$$x \to \sigma(s)s, \qquad \|x\| \to \sigma(s), \qquad n(x) \to s.$$

Substituting all of these, we obtain

$$\frac{\partial \sigma}{\partial \boldsymbol{t}} = -\frac{\sigma(s)s^T - \nabla_s \sigma_0(s)(1 - ss^T)\left(I - ss^T\right)}{\sigma(s)s^T s - \nabla_s \sigma_0(s)(1 - ss^T)\left(I - ss^T\right) s}.$$

Using the fact that $(I - ss^T)s = 0$, and $\nabla_s \sigma_0(s)(1 - ss^T) = \nabla_s \sigma_0(s)$ (the gradient is tangent to s), we simplify it to

$$\frac{\partial \sigma}{\partial \boldsymbol{t}} = -\frac{\sigma(s)s^T - \nabla_s \sigma_0(s)}{\sigma(s)} = \frac{\nabla_s \sigma_0(s)}{\sigma(s)} - s^T = \nabla_s \log \sigma(s) - s^T.$$

□

The "$-s_i^*$" term means that if the velocity $\boldsymbol{v}$ is in the direction on s, then the range decreases (the remaining nonlinear term $\nabla_i \log \sigma^s$ is less intuitive).

9.3.4. *Vision sensors*

In general, the sensel space of a camera is $\mathcal{S} = \mathbb{R}^3 \times \mathbb{S}^2$: each pixel captures the light arriving to a particular focus point (in $\mathbb{R}^3$) from a particular direction on the unit sphere ($\mathbb{S}^2$). For simplicity, we consider a central camera with only one focus point, so that the sensel space is just $\mathcal{S} = \{\mathbf{0}\} \times \mathbb{S}^2$.

The physics of light scattering can be quite complicated. The following definition is applicable only if the surfaces have a Lambertian light model, far from occlusions, and considering a static world. See Soatto [74] for a formal definition that takes into account occlusions and shading.

PROPOSITION 9.8. *Let y^s, $s \in \mathbb{S}^2$, be the luminance signal captured by the camera. Let μ^s be the nearness, the inverse of the distance in direction s. Then the dynamics of $\boldsymbol{y}$ are*

$$\dot{y}^s = \mu^s \nabla_i y^s \boldsymbol{v}^i + (s \times \nabla y^s)_i \boldsymbol{\omega}^i. \tag{9.7}$$

Note that the nearness μ is a hidden state for the dynamics. This hidden state has a dynamics of its own, which is given by the range-finder dynamics (9.5).

9.4. Training and Environment Statistics

In identification it is often necessary to impose certain observability conditions on the data that make the system identifiable. In this context, these conditions correspond to various constraints on the geometry and the statistics of the environment, as well as the exploration behavior of the agent, which is summarized by the training distribution p_{T}.

DEFINITION 9.9 (Training distribution). The *training distribution*

$$p_{\mathrm{T}} \in \mathsf{ProbMeasures}(\mathsf{Maps} \times \mathcal{Q})$$

is the distribution over maps and poses experienced by the agent during the learning phase.

The symmetries of this training distribution summarize whether the agent has had a "uniform" exploration of the environment.

DEFINITION 9.10. Define the *symmetry group* of p_{T} as the subgroup $\mathsf{Sym}(p_{\mathrm{T}})$ of the configuration space $\mathcal{Q}$ such that, for all $g \in \mathsf{Sym}(p_{\mathrm{T}})$,

$$p_{\mathrm{T}}(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}) = p_{\mathrm{T}}(\langle \boldsymbol{s}, g \cdot \boldsymbol{p} \rangle, \boldsymbol{q}).$$

EXAMPLE 9.11. Consider a planar robot ($\mathcal{Q} = \mathrm{SE}(2)$). If we believe that the robot experience did not privilege one particular orientation over the others, then we would set $\mathsf{Sym}(p_{\mathrm{T}})$ to be the group of planar rotations $\mathsf{SO}(2)$.

Whether the training distribution was "symmetric enough" depends on what sensor the robot is using. For example, $\mathsf{Sym}(p_{\mathrm{T}}) = \mathsf{SO}(2)$ would be enough for making the statistics of a 2D range-finder uniform across the field of view, but not if the sensor was a 3D range-finder. This is formalized by the concept of a "mixing" distribution.

DEFINITION 9.12 (Mixing distribution). Consider two couples of sensels (s_1, v_1), (s_2, v_2), where $s_1, v_1, s_2, v_2 \in \mathcal{S}$. Let d be the metric on the manifold $\mathcal{S}$. We call the training distribution *mixing* for $\mathcal{S}$ if $d(s_1, v_1) = d(s_2, v_2)$ implies that there exists a $g \in \mathsf{Sym}(p_{\mathrm{T}})$ such that $(s_2, v_2) = (g \cdot s_1, g \cdot v_1)$.

---

subgroup: A subset of a group that forms a group under the same operation. See Definition C.5.

If the distribution is mixing, the statistics of the observations are easy to compute. These are mainly technical lemmas that are needed in the successive chapters.

PROPOSITION 9.13. *For a mixing training distribution, the expectation of any function of two sensels $s, v \in \mathcal{S}$ is only a function of their distance; for all functions $\phi : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$, we can write $\mathbb{E}\{\phi(y^s, y^v)\}$ as $\varphi(d(s,v))$ for some function $\varphi : \mathbb{R}^+_\bullet \to \mathbb{R}$.*

PROOF. The proof consists in showing that $\mathbb{E}\{\phi(y^s, y^v)\}$ has the same value for any two couples of sensels (s_1, v_1), (s_2, v_2) that have the same distance. Write the expectation

for (s_2, v_2) by showing the dependence of $\boldsymbol{y}$ on the pose $\boldsymbol{q}$ and the world $\langle \boldsymbol{s}, \boldsymbol{p} \rangle$:

$$
\begin{aligned}
\mathbb{E}\{\phi(y^{s_2}, y^{u_2})\} &= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, s_2), y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, v_2))\} \\
&\qquad \text{(By assumption, } s_2 = x s_1 \text{ and } u_2 = x v_1 \text{ for some } x \in \mathsf{Sym}(p_{\mathrm{T}})\text{.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, x s_1), y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, x v_1))\} \\
&\qquad \text{(Because this is a relative sensor, property P2 holds.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}x, s_1), y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}x, v_1))\} \\
&\qquad \text{(Now using property P1 applied to } (\boldsymbol{q}x)^{-1}\text{.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, (\boldsymbol{q}x)^{-1}\boldsymbol{p} \rangle, (\boldsymbol{q}x)^{-1}\boldsymbol{q}x, s_1), \\
&\qquad\quad y(\langle \boldsymbol{s}, (\boldsymbol{q}x)^{-1}\boldsymbol{p} \rangle, (\boldsymbol{q}x)^{-1}\boldsymbol{q}x, v_1))\} \\
&\qquad \text{(Simplifying, and using the fact that } (\boldsymbol{q}x)^{-1} = x^{-1}\boldsymbol{q}^{-1}\text{.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, x^{-1}\boldsymbol{q}^{-1}\boldsymbol{p} \rangle, e, s_1), y(\langle \boldsymbol{s}, x^{-1}\boldsymbol{q}^{-1}\boldsymbol{p} \rangle, e, v_1))\} \\
&\qquad \text{(If } x \text{ is in the group of symmetries } \mathsf{Sym}(p_{\mathrm{T}})\text{, } x^{-1} \text{ is as well.)} \\
&\qquad \text{(Using the mixing property (Definition 9.12), we can remove } x^{-1}\text{.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, \boldsymbol{q}^{-1}\boldsymbol{p} \rangle\rangle, e, s_1), y(\langle \boldsymbol{s}, \boldsymbol{q}^{-1}\boldsymbol{p} \rangle, e, v_1))\} \\
&\qquad \text{(Reusing property P1 in the other direction.)} \\
&= \mathbb{E}\{\phi(y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, s_1), y(\langle \boldsymbol{s}, \boldsymbol{p} \rangle, \boldsymbol{q}, v_1))\} \\
&= \mathbb{E}\{\phi(y^{s_1}, y^{v_1})\}.
\end{aligned}
$$

Because $\mathbb{E}\{\phi(y^s, y^v)\}$ has the same value for all couples of sensel with a fixed distance, it must be a function of only the distance. □

Several corollaries follow.

COROLLARY 9.14. For a relative sensor in the mixing case, the covariance of two sensels is a function of only their distance:

$$\operatorname{cov}(y^s, y^v) = f(d(v, s)).$$

PROOF. This follows directly from Proposition 9.13, applied to the function

$$\phi(y^s, y^v) = (y^s - \mathbb{E}\{y^s\})\,(y^v - \mathbb{E}\{y^v\})\,.$$

□

COROLLARY 9.15. In a mixing environment, the expected value of the sensels does not depend on s:

$$\mathbb{E}\{y^s\} = \overline{y}.$$

PROOF. Apply Proposition 9.13 with $s = u$ and the function $\phi(y^s, y^s) = y^s$. The expectation depends on s only through $d(s, s) = 0$, and therefore it is independent of s. □

COROLLARY 9.16. The gradient of y with respect to the sensor space has expected value 0:

$$\mathbb{E}\{\nabla y^s\} = 0.$$

PROOF. This is simple consequence of the linearity of the expectation:

$$\mathbb{E}\{\nabla y(s)\} = \nabla \mathbb{E}\{y(s)\} = \nabla \overline{y} = 0.$$

□

COROLLARY 9.17. More generally, the expectation of the gradient of any function f :

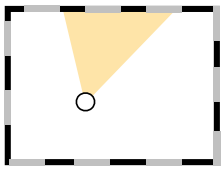


Figure 9.1. Example of nonmonotone environment.

$\mathcal{O} \to \mathcal{O}$ of the observations is 0:

$$\mathbb{E}\{\nabla f(y^s)\} = 0.$$

We define a property of the environment useful in the future.

DEFINITION 9.18 (Monotone environment). The environment is *monotone* if the covariance of the values of two sensels is a monotone function of the distance between the sensels.

Not all environments are monotone. It is true in general that the covariance reaches the maximum when the distance is 0, however, then it is not always monotonically decreasing. Usually that means that there is some structure in the environment. Figure 9.1 shows a counter-example: suppose that a robot with a camera is an environment that (on average) appears periodic on the retina with period $\Delta\theta$; then the correlation between pixels will be an oscillatory function of the distance with period approximately $\Delta\theta$.

9.5. Related Work for Learning Dynamics

The main approaches to learning models for dynamical systems are briefly summarized below. The main contribution with respect to previous work is to study models that are specifically tailored to robotic sensorimotor cascades. The methodological difference is the effort towards an analytical rather than empirical discussion. We have derived explicit models for these sensorimotor cascades, therefore we can try to characterize exactly what is the approximation obtained by the classes of models that will be introduced. Moreover,

we look explicitly at the performance for a closed loop behavior (servoing) which will be characterized analytically. The investigation of the agent's invariance to representation nuisance is orthogonal to the class of models chosen, so it could be equally well applied to all these other works.

9.5.1. *The systems/control identification approach*

System identification [24, 25] has a long history [75], and there is a complete theory for linear systems as well as for some limited classes of nonlinear systems; see Verdult [76] for a tutorial introduction. Identification for "generic" nonlinear systems can be approached as a generic regression problem using Volterra series [77], which is, very roughly speaking, the equivalent of a "Taylor expansion" for dynamical systems. Recently system identification has been approached with much more complicated models such as Gaussian processes [78] (e.g., [27]).

9.5.2. *Learning from sensorimotor data*

Many works in machine learning used models that work with raw sensory data, and, to a lesser extent, with sensori*motor* data. Deep belief networks have been used to represent motions and transformations at the pixel level [79–82]. The work by Roberts et al. [83] about learning optic flow-fields is particularly related to the BGDS models presented in Chapter 9.

9.5.3. *Learning generic MDP/POMDP*

Markov Decision Processes (MDP) and Partially Observable Markov Decisions Processes (POMDP) are dynamical systems with a discrete state space that evolves with arbitrary transitions. These dynamics can be learned with spectral methods [84, 85].

Predictive State Representations (PSR) [86–89] are an alternative representation for dynamical systems, in which, rather than modeling the state explicitly, the dynamics is represented by a series of *tests* (i.e., observable functions of the state). There is some analogy with the definition of black box given in Chapter 4.

CHAPTER 10

Learning Sensor Geometry

This chapter considers the problem of recovering the sensor geometry from the statistics of a set of scrambled sensels. The problem is studied in the context of camera calibration, and the results are competitive with the state of the art.

The materials in this chapter come from a paper jointly written with Davide Scaramuzza.

Table 10.1. Symbols used in this chapter

Problem statement		
Y_{ij}	$\in [-1,+1]$	Similarity matrix.
f	$: \mathbb{R}^{+}_{\bullet} \to [-1,+1]$	Distance-to-similarity function.
$\mathsf{infr}(f)$	> 0	Informative radius of f.
φ		Generic similarity statistics.
$\mathcal{M}$		Target manifold.
$\mathcal{S}$	$\subset \mathcal{M}^n$	Solution of the embedding problem.
Performance measures		
ρ_{sp}		Spearman performance measure.
ρ^{*}_{sp}		Normalized Spearman performance.
rad		Radius of a point distribution.
Algorithms		
SK		Shepard-Kruscall algorithm.
SKv+w		Proposed algorithm.
SKv		Proposed algorithm (without warping).
C_{ij}		Cosine matrix.
$\mathbf{S}$	$\in \mathbb{R}^{3\times n}$	Point coordinates stacked in a matrix.
$\mathsf{sorted}(\boldsymbol{x})$		Sorted vector.
$\mathsf{order}(\boldsymbol{x})$		Rank of a vector (Definition A.28).

10.1. Calibration by Correlation

In many applications, from classic photogrammetry tasks to autonomous robotics, camera calibration is a necessary preliminary step before using the camera data [90]. Calibration is necessary even for off-the-shelf cameras, as the properties of an optical system typically differ substantially from the stated manufacturer's specifications. *Extrinsic* camera calibration is concerned with recovering the pose (position and orientation) of the camera with respect to another camera, or another reference frame of interest. *Intrinsic* camera calibration is concerned with estimating the origin and direction of the line of sight of each pixel; this information allows us to put into correspondence the image of an object with the position of the object in the world. Some scientific applications require estimating other characteristics of the optical system, such as the point-spread function. In this chapter, we focus on intrinsic camera calibration for central cameras.

In a central camera, the lines of sight of every pixel intersect in a single point. Therefore, the intrinsic calibration information consists of the direction of each pixel on the visual sphere ($\mathbb{S}^2$). If a camera is noncentral, then one needs to know, for each pixel, also its spatial position (in $\mathbb{R}^3$) in addition to its direction (in $\mathbb{S}^2$). A noncentral camera can be approximated as a central camera only if the displacement of each pixel's origin is negligible with respect to the distance to the objects in the scene. This assumption is generally satisfied in applications such as robotics, but might not be satisfied for more uncommon applications and optical systems. A general description of how the properties of lenses, mirrors, and sensors contribute to the geometry of the optical system is outside of the scope of this chapter; a recent tutorial is given by Sturm et al. [91].

10.1.1. *Established techniques for intrinsic calibration*

The most widely used techniques for intrinsic camera calibration (from now on, simply "calibration") require the use of a known calibration pattern, and that the cameras optics can be well represented by a restricted family of models. Several calibration software tools are available online as open source. The Matlab Calibration Toolbox [92] works for pin-hole cameras and implements a mix of techniques appeared in the literature [93, 94]. The model used for pin-hole cameras is parametrized by the center of projection, the focal length, and radial and tangential distortion, which accounts for the possibility of the image sensor being not perpendicular to the optical axis. Other calibration toolboxes [95–102] can be used for calibrating omnidirectional catadioptric cameras, obtained by placing a mirror on top of a conventional camera, such that the optical axis coincides with the mirror's axis, or with fish-eye cameras (dioptric). The parameters of the model are the center of projection in image coordinates and the profile of the radial distortion. These methods are relatively simple to use. In most of them, the user prints out a calibration pattern consisting of a black and white checkerboard, and collects several pictures of the pattern from different points of view. A semi-interactive procedure is used to identify the corners of the calibration pattern. Given this information, the software automatically solves for the calibration parameters. The algorithms rely on the fact that the pattern is known to lie on a plane, which allows recovering the parameters of the homography describing the world-to-image transformation, and that the nonlinear parts of the model (*e.g.*, distortion) are simple enough that they can be recovered using generic nonlinear optimization.

Recently, there have been several works to improve on these techniques, to make them more flexible by enlarging the family of optics considered, or making the calibration procedure more convenient. Grossberg and Nayar [103] describe a method for calibrating an

arbitrary imaging system, in which the pixels are allowed to have an arbitrary configuration on the visual sphere, that is based on an active display. Espuny and Gil [104] describe a technique that does not require a known image pattern, but is based on known sensor motion.

10.1.2. *Calibration by correlation*

We describe an approach to intrinsic camera calibration based exclusively on low-level statistics of the raw pixel streams, such as the inter-pixel correlation. To the best of our knowledge, Grossmann et al. [105] were the first to propose this idea for the problem of camera calibration, albeit they were inspired by work done in developmental robotics and related fields [106–108].

The basic premise is that the statistics of the raw pixel stream contain information about the sensor geometry. Let $y_i(t)$ be the luminance perceived at the i-th pixel at time t. If we compare the sequences $\{y_i(t)\}_t$ and $\{y_j(t)\}_t$ for the i-th and j-th pixel, we expect to find that they are more similar the closer the two pixels are on the visual sphere. The geometry of the sensor can be recovered if one can find a statistics of the two sequences that is precisely a function of the pixels distances. More formally, let $s_i \in \mathbb{S}^2$ be the direction of the i-th pixel on the visual sphere, and let $d(s_i, s_j)$ be the geodesic distance on the sphere between the directions s_i and s_j. Let $\varphi : \mathbb{R}^T \times \mathbb{R}^T \to \mathbb{R}$ indicate a real-valued statistics of two sequences of length T. For example, the statistics φ can be the sample correlation, the mutual information, or any other information-theoretical divergence between two sequences, such as the "information distance" [105]. Define the *similarity* Y_{ij} between two pixels using φ:

$$\mathrm{Y}_{ij} = \varphi(\{y_i(t)\}_t, \{y_j(t)\}_t).$$

The assumption that must be verified for the method to work, which we will call the *monotonicity condition*, is that the similarity is a function f of the pixel distance:

$$Y_{ij} = f(d(s_i, s_j)), \tag{10.1}$$

and that this f is monotonic, therefore, invertible.

Grossmann et al. assume to know the function f, obtained with a separate calibration phase, by using a sensor with known intrinsic calibration experiencing the same scene as the camera being calibrated. Therefore, using the knowledge of f, one can recover the distances from the similarities: $d(s_i, s_j) = f^{-1}(Y_{ij})$. They describe two algorithms for recovering the pixel positions given the inter-pixel distances. The first algorithm is based on *multidimensional scaling* (which we will recall in the following sections) and solves for all pixel directions at the same time. The authors observe that this method is not robust enough for their data, and propose a robust nonlinear embedding method, inspired by Sammon [109] and Lee et al. [110]. This second algorithm is iterative and places one pixel per iteration on the sphere, trying to respect all constraints with previously placed points.

Compared with traditional calibration methods, the "calibration by correlation" approach is attractive because it does not require a parametric model of the camera geometry, control of the sensor motion, or particular properties of the scene. However, the results reported by Grossmann et al. do not compare favorably with traditional methods. The authors focus their quantitative analysis mainly on the accuracy of the estimation and inversion of the function f. They find that, for the information distance, f is reliably invertible only for $d(s_i, s_j) \leq 30°$.* For large field of view, the estimated distributions appear

*Compare Fig. 6 in [105], which shows the graph of f as a function of distance; and Fig 8ab, which shows the error for estimating f^{-1}.

significantly "shrunk" on the visual sphere.† Moreover, they find that, in practice, the function f is sensitive to the scene content; in their data, they find that applying the function f^{-1} estimated with a calibration rig to the data taken from a different camera leads to over-estimation of the angular distances.‡

We start from the same premise of Grossmann et al., namely that it is possible to find a statistics of the pixel stream that depends on the pixel distance. However, rather than assuming the function f known, we formulate a joint optimization problem, in which we solve for both the directions $\{s_i\}$ and the function f. In this way, there is no need for a preliminary calibration phase with a sensor of known geometry. However, the problem becomes more challenging, requiring different analytic and computational tools.

Section 10.2 gives a formal description of the joint optimization problem. Section 10.3 discusses the conditions under which one can expect a monotonic relation between pixel distance and pixel statistics. We show that, if the camera undergoes uniform random motion, then necessarily all pairwise statistics between pixel values must depend on the pixel distance only. This suggests that a good way to collect data for camera calibration is to wave it around as randomly as possible, a theory we verify in practice.

Section 10.4 gives an observability analysis of the problem. The observability depends both on the manifold's *local geometric properties* (curvature) as well as on *global topological properties* (connectedness). In $\mathbb{R}^m$, the scale is not observable, but, surprisingly, it is observable in $\mathbb{S}^2$ and other spaces of nonzero curvature, which makes the problem more constrained than in Euclidean space.

Section 10.5 discusses the performance measures that are adequate for the problem.

†See Section 4.4.1 and Fig. 13 in [105]. Note the shrinkage of the distribution (no quantitative measure is given in the paper).

‡See Section 4.4.2 in [105].

The Procrustes error (i.e., alignment up to rotations) is an intuitive choice, but it is not admissible because it is not invariant to all symmetries of the problem. We use the *Spearman score* as an admissible and observable performance measure.

Section 10.6 describes our algorithm, which is an extension of the classical Shepard-Kruskal (SK) algorithm [111–114]. The major extension is an extra step necessary to recover the correct scale when it is observable; this step is critical for accurate calibration.

Section 10.7 discusses the experimental results for the case of camera calibration. The algorithm is evaluated for three different cameras: a pin-hole (45° FOV), a fish-eye (150° FOV), and an omnidirectional catadioptric camera (360° × 100° FOV). The results obtained are comparable with those obtained using conventional methods.

10.2. Nonmetric Embedding

Let $\mathcal{M}$ be a Riemannian manifold, and let d be its geodesic distance. We formalize the problem of metric embedding from nonmetric measurements as follows.

PROBLEM 10.1. Given a symmetric matrix $\mathbf{Y} \in \mathbb{R}^{n \times n}$, estimate the set of points $\mathcal{S} = \{s_i\}_{i=1}^n$ in a given manifold $\mathcal{M}$, such that $Y_{ij} = f(d(s_i, s_j))$ for some (unknown) monotonic function $f : [0, \infty) \to \mathbb{R}$.

Without loss of generality, we assume the similarities to be normalized so that $-1 \leq Y_{ij} \leq 1$ and $Y_{ii} = 1$. This implies $f(0) = 1$, and that f is nonincreasing. For camera calibration, the manifold $\mathcal{M}$ will generally be the unit sphere $\mathbb{S}^2$; however, we formulate a slightly more generic problem. We will be especially interested in showing how the observability of the problem changes if $\mathcal{M}$ is chosen to be $\mathbb{S}^1$ (the unit circle) or $\mathbb{R}^m$ instead of $\mathbb{S}^2$.

If the function f was known, it would be equivalent to know directly the matrix of distances. The problem of finding the positions of a set of points given their distance

matrix is often called "metric embedding". In the Euclidean case ($\mathcal{M} = \mathbb{R}^m$), the problem is classically called *Multidimensional Scaling* (MDS), and was first studied in psychometry in the 1950s. Cox and Cox [115] describe the statistical origins of the problem and give an elementary treatment, while France and Carroll [114] give an overview of the algorithmic solutions.

The scenario described in Problem 10.1 is sometimes called nonmetric multidimensional scaling. The word "nonmetric" is used because the metric information, contained in the distances $d(s_i, s_j)$, is lost by the application of the unknown function f. In certain applications, it is not important for the reconstructed points to be recovered accurately. For example, in psychometry, one might use these techniques essentially for visualization of high-dimensionality datasets; in that case, one only wants a *topologically correct* solution. If that is the case, one can just choose an arbitrary $\tilde{f}$ different from the true f; as long as $f(0) = \tilde{f}(0)$, the results will be topologically correct. However, in the camera calibration setting, we are explicitly interested in obtaining a *metrically accurate* solution.

Problem 10.1 is a chicken-and-egg problem in the two unknowns f and $\{s_i\}_{i=1}^n$: knowing the function f, one can estimate the distances as $f^{-1}(\Upsilon_{ij})$, and use standard MDS to solve for $\{s_i\}_{i=1}^n$; conversely, knowing the distances, it is trivial to estimate f. But is it possible to estimate both at the same time? To the best of our knowledge, there has not been any claim about whether *accurate metric embedding from* nonmetric *measurements* is possible. In this paper, we will show that the answer depends on the properties of the manifold $\mathcal{M}$. Specifically, while for $\mathbb{R}^m$ the scale is not observable, we show that accurate metric embedding is possible for $\mathbb{S}^2$. Consequently, it is possible to calibrate a camera from any statistics that respects the monotonicity condition (10.1), even if the function f is a priori unknown.

We briefly mention several other problems that can be formalized in the same way, and

that will motivate us to solve the problem in a slightly more generic way than what strictly needed for camera calibration. In developmental robotics and similar fields [106–108], a common scenario is that an agent starts from zero knowledge of its sensors, and its first concern is to recover the geometry of the sensor (possibly a camera, but also a range-finder or other robotic sensor) by considering simple statistics of the sensor streams. In *sensor networks* (see, e.g., [116]**AC:** add other in grossman), one basic problem is localizing the nodes in space based on relative measurements of wi-fi strength. Assuming the signal is a function of the distance, we arrive to the same formalization, using $\mathbb{R}^n$ as the target manifold. More generally, this formalization covers many embedding problems in machine learning, where the data is assumed to be in a metric space, but the available similarities, perhaps obtained by comparing vectors of *features* of the data, cannot be interpreted directly as distances in the original metric space.

10.3. When is Similarity a Function of the Sensels Distance?

The basic assumption of our method is that it is possible to find a statistics of the pixel luminance that satisfies the monotonicity condition (10.1). We state a result that guarantees that *any* pairwise statistics is asymptotically a function of the distance between the pixels, if the camera undergoes uniformly random motion, in the sense that the camera's orientation (a rotation matrix $\mathbf{R}$) is uniformly distributed in $\mathsf{SO}(3)$ (the set of rotation matrices). This result is a particularization of Proposition 9.13.

PROPOSITION 10.2. *If the probability distribution of the camera orientation* $\mathbf{R}$ *is uniform in* $\mathsf{SO}(3)$*, the expectation of a function of the luminance of two pixels depends only on the pixel distance: for all functions* $g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$*, there exists a function* $f : \mathbb{R}^+ \to \mathbb{R}$*, such that*

$$\mathbb{E}\{\, g(y(s_i), y(s_j))\,\} = f(d(s_i, s_j)).$$

PROOF. The luminance at pixel $s \in \mathbb{S}^2$ at time t can be written as

$$y(s,t) = h(\boldsymbol{t}(t), \mathbf{R}(t)\, s),$$

where $\boldsymbol{t} \in \mathbb{R}^m$ is the sensor position, $\mathbf{R} \in \mathsf{SO}(3)$ is the sensor orientation, and $h : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathbb{R}$ is a function that describes the environment. In the following, we drop the dependence on time.

Consider two pairs of pixels (s_i, s_j) and (s_k, s_l) having the same distance:

$$d(s_i, s_j) = d(s_k, s_l).$$

We will show that this constraint is enough for the pairwise statistics to be equal:

$$\mathbb{E}\{g(y(s_i), y(s_j))\} = \mathbb{E}\{g(y(s_k), y(s_l))\}. \tag{10.1}$$

Because there is no other relation between the two pairs of pixels other than their distance, the statistics g depends only on the distance.

If the probability distribution of $\mathbf{R}$ is uniform on $\mathsf{SO}(3)$, that is, it is the Haar measure $\mathsf{SO}(3)$, then it is also invariant to a rotation (i.e., left/right actions): for all functions z and rotations $\mathbf{X}$, $\mathbb{E}\{z(\mathbf{R})\} = \mathbb{E}\{z(\mathbf{R}\mathbf{X})\}$.

In our case, we have that for any $\mathbf{X}$,

$$\begin{aligned} \mathbb{E}\{g(y(s_i), y(s_j))\} &= \mathbb{E}\{g(h(\boldsymbol{t}, \mathbf{R}\, s_i), h(\boldsymbol{t}, \mathbf{R}\, s_j))\} \\ &= \mathbb{E}\{g(h(\boldsymbol{t}, \mathbf{R}\, \mathbf{X} s_i), h(\boldsymbol{t}, \mathbf{R}\, \mathbf{X} s_j))\}. \end{aligned} \tag{10.2}$$

Because $d(s_i, s_j) = d(s_k, s_l)$, there exists an $\mathbf{X}$ such that

$$s_k = \mathbf{X} s_i, \qquad s_l = \mathbf{X} s_j.$$

By substituting this $\mathbf{X}$ in (10.2) we obtain (10.1). □

In particular, this is valid for the correlation between pixel values, as the correlation can be written as $\mathsf{corr}(y_i, y_j) = \mathbb{E}\{g(y(s_i), y(s_j))\}$ with $g(y(s_i), y(s_i)) = (y(s_i) - \overline{y})(y(s_i) - \overline{y})$. Most other similarity statistics can be written in the same fashion.

When is similarity monotonic?

Proposition 10.2 ensures that (10.1) holds for some function f, but it does not ensure that such function f is monotone. To find conditions that guarantee that f is monotone it is necessary to introduce some model of the environment. Essentially, f might not be monotone if there is some long-range "structure" in the environment. We describe an artificial counterexample in which f is not monotone.

EXAMPLE 10.3. Imagine a room, shaped like a parallelepiped with the base of size $L \times L$ and height $H \gg L$ (Fig. 10.1). Suppose an omnidirectional camera is suspended in the middle of the room, equidistant from the walls, ceiling, and floor. From that position, the images of ceiling and floor gets projected on the visual sphere in an area contained in a spherical cap of radius $\delta = 2\arccos(H/\sqrt{H^2 + L^2})$. For example, for $L = 5\,\mathrm{m}$ and $H = 10\,\mathrm{m}$, we obtain $\delta \simeq 28°$. This implies that, if two pixels observe the ceiling at the same time, they cannot be more than 28° apart. Assume that the floor and the ceiling are painted of a uniform white, and the walls have very intricate black-white patterns, well approximated by white noise. We let the camera undergo random rotational motion, and we compute the correlation of the pixel luminance. Consider now two pixels at distance $d(s_i, s_j) = 60°$. Note that any two pixels at this distance will never look both at the ceiling at the same time, because the apparent size of the ceiling is $\delta = 28°$. Hence, there are three possibilities: (1) they are both looking at the walls; (2) one is looking at the walls, another at the ceiling; (3) one is looking at the walls, another at the floor. In all cases, one is

looking at the white noise on the walls. Therefore, the correlation of two pixels streams at distance 60° is 0: $f(60°) = 0$. Consider now two pixels at distance 180° (one exactly opposite to the other on the visual sphere). For these pixels, there are two possibilities: (1) they are both looking at the walls; (2) one looks at the ceiling, the other at the floor. Because floor and ceiling are the same color, the luminance of these two pixels has a slight positive correlation: $f(180°) > 0$. Therefore, the function f is not monotonic, because $f(0°) = 1$, $f(60°) = 0$, and $f(180°) > 0$.

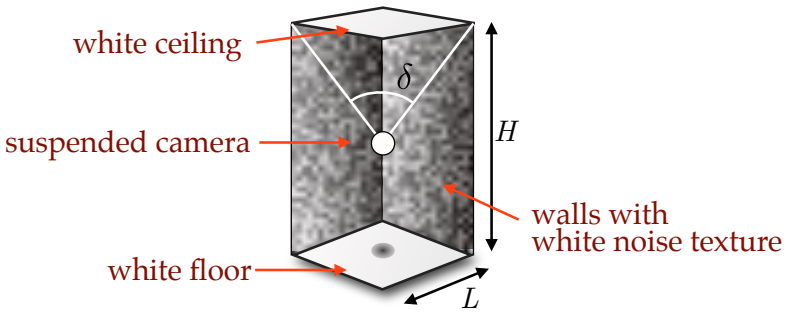


Figure 10.1. Environment used in Example 10.3.

10.4. Observability of Sensor Geometry Reconstruction

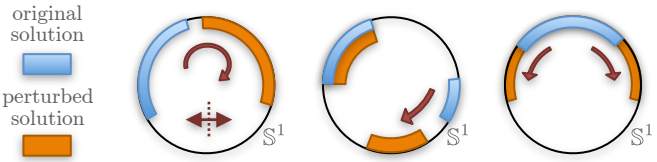
A *symmetry* of an estimation problem is any joint transformation of the unknowns (in this case, the directions $\mathcal{S} = \{s_i\}$ and the function f) that does not change the observations (in this case, the similarities Y_{ij}). Studying the observability of the problem means describing what symmetries are present. In this section, we first give a tour of the symmetries of this problem, before presenting the main result in Proposition 10.8.

10.4.1. *Isometries*

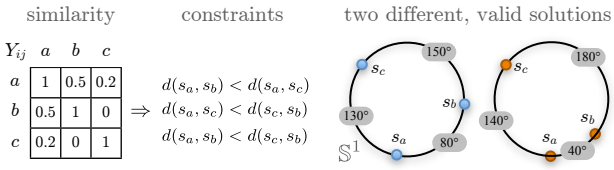
It is easy to see that the similarities Y_{ij} are preserved by the isometries of the domain $\mathcal{M}$.

10.4.2. *Sliding*

Define the "informative radius" of f as follows.



(a) Isometries (b) Sliding (c) Warping



(d) Wiggling: an extreme case, with only three points.

Figure 10.1. Symmetries of the estimation problem, illustrated in the case $\mathcal{M} = \mathbb{S}^1$. (a) Isometries (for $\mathbb{S}^1$, rotations and reflections) are unobservable because they preserve distances between points. (b) If f is non invertible on the whole domain, disconnected components of $\mathcal{S}$ can move isometrically independently from each other; we call this "sliding". (c) In manifolds with zero curvature (e.g., $\mathbb{R}^n$ and $\mathbb{S}^1$, but not $\mathbb{S}^2$) the scale is not observable; formally, a linear warping (Definition 10.5) does not violate the problem constraints. (d) If the set of points is finite, the constraints are not violated by small perturbations of the points, called "wigglings".

DEFINITION 10.4. For a function $f : \mathbb{R}^+_\circ \to \mathbb{R}$, let $\mathsf{infr}(f)$ be the maximum r such that f is invertible in $[0, r]$.

If the set $\mathcal{S}$ has two components distant more than $\mathsf{infr}(f)$ from each other, one component can be isometrically moved independently of the other, without changing the observations (Fig. 10.1b); we call this "sliding".

10.4.3. *Linear warping*

We define a *linear warping* as a map that scales the interpoint distances uniformly by a constant.

DEFINITION 10.5. A linear warping of $\mathcal{M}$ is a map $\varphi_\alpha : \mathcal{M} \to \mathcal{M}$ such that, for some $\alpha > 0$, for all $s_1, s_2 \in \mathcal{M}$,

$$d(\varphi_\alpha(s_1), \varphi_\alpha(s_2)) = \alpha\, d(s_1, s_2),$$

If a linear warping exists, then it is a symmetry of the problem. In fact, suppose that $(f, \{s_i\})$ is a solution of the problem. Construct another solution $(f', \{s'_i\})$, with $f' = \frac{1}{\alpha} f$ and $s'_i = \varphi_\alpha(s_i)$. We would not be able to distinguish between these two solutions, as they would give the same observations.

More generally, we could define "generic warpings" as follow.

DEFINITION 10.6. A *generic warping* is a map $\varphi_m : \mathcal{M} \to \mathcal{M}$ such that $d(\varphi(s_1), \varphi(s_2)) = m(d(s_1, s_2))$, for some monotonic function $m : \mathbb{R}^+_\circ \to \mathbb{R}^+_\circ$.

10.4.4. *Wiggling*

A peculiar aspect of Problem 10.1 is that the unknowns $\mathcal{S} = \{s_i\}_{i=1}^n$ live in a continuous space $\mathcal{M}$, but the observations $\mathrm{Y}_{ij} = f(d(s_i, s_j))$ are actually equivalent to a set of *discrete inequalities*, a fact which is very well explained by Agarwal et al. [117]. In fact, because the function f is completely unknown (the only constraint being its monotonicity), all that we can infer about the inter-point distances from the matrix $\mathbf{Y}$ is their ordering: for all $(i, j), (k, l)$, if $\mathrm{Y}_{ij} \gtreqless \mathrm{Y}_{kl}$, then we can infer $d(s_i, s_j) \lesseqgtr d(s_k, s_l)$, but nothing more. Therefore, the sufficient statistics in the matrix Y_{ij} is the *ordering* of the entries, not their specific *values*. This raises the question of whether precise metric reconstruction is possible, if the available observations are a set of discrete inequalities. In fact, given a point distribution $\{s_i\}$ of n points, the position of the generic point s_j is constrained by n^2 inequalities (many of which redundant). Inequalities cannot constrain a specific position for s_j in the manifold $\mathcal{M}$; rather, they specify a small finite area in which all constraints are satisfied.

Therefore, for each solution, an individual point has a small neighborhood in which it is free to "wiggle" without violating any constraint. In general, we call these perturbations "wiggling":

DEFINITION 10.7. A *wiggling* of a set $\{s_i\} \subset \mathcal{M}$ is a map $\varphi : \mathcal{M} \to \mathcal{M}$ that preserves the ordering of the distances: for all i, j, k, l:

$$d(s_i, s_j) < d(s_k, s_l) \Leftrightarrow d(\varphi(s_i), \varphi(s_j)) < d(\varphi(s_k), \varphi(s_l)).$$

The size of the allowed wiggling decreases with the density of points; for n points uniformly distributed in $\mathcal{M}$, one can show that the average wiggling space is in the order of $o(1/n)$ per single point (i.e., keeping the others fixed). In the limit as the points become dense, it is possible to show that wiggling degenerates to rigid linear warpings. For very sparse distributions, the effect of wiggling can be quite dramatic (Fig. 10.1d).

10.4.5. *Main result*

The following proposition establishes which of the previously described symmetries are present in the problem, as a function of the geometric properties of the space $\mathcal{M}$, the point distribution, and the function f.

PROPOSITION 10.8. *Assume the set $\mathcal{S} = \{s_i\}$ is an open subset of $\mathcal{M}$ whose closure has only one connected component. Let the available measurements be $Y_{ij} = f(d(s_i, s_j))$, where $f : \mathbb{R}^+_\circ \to \mathbb{R}$ is a monotone function with $\mathsf{infr}(f) > 0$. Then:*

- *If $\mathcal{M}$ has nonzero curvature (e.g., $\mathbb{S}^2$), then it is possible to recover f exactly, and $\mathcal{S}$ up to isometries.*
- *If $\mathcal{M}$ has zero curvature:*

Table 10.2. Observability classes

class	*space*	*curvature*	*extra assumptions*	*symmetries*
A	$\mathbb{S}^{\geq 2}$	> 0	-	wigglings, isometries
B	$\mathbb{S}^1$	0	$\mathsf{rad}(\mathcal{S}) + \mathsf{infr}(f) \geq 2\pi$	wigglings, isometries
C	$\mathbb{S}^1$	0	$\mathsf{rad}(\mathcal{S}) + \mathsf{infr}(f) < 2\pi$	wigglings, isometries, linear warpings
D	$\mathbb{R}^n$	0	-	wigglings, isometries, linear warpings
A	$\mathbb{H}^n\mathbb{H}^n$	< 0	-	wigglings, isometries

- *If $\mathcal{M}$ is simply connected (e.g., $\mathbb{R}^m$), then it is possible to recover f only up to scale, and $\mathcal{S}$ up to isometries plus a "linear warping" (Definition 10.5).*
- *If $\mathcal{M}$ is not simply connected (e.g., $\mathbb{S}^1$), the scale can be recovered if $\mathsf{infr}(f)$ is large enough.*

  For $\mathbb{S}^1$, this happens if

$$\mathsf{rad}(\mathcal{S}) + \mathsf{infr}(f) \geq \pi, \tag{10.1}$$

  where $\mathsf{rad}(\mathcal{S})$ is the radius of $\mathcal{S}$ (Definition E.2).

The observability breaks down as follows:

- *"Sliding" occurs if $\mathcal{S}$ has multiple components with Hausdorff distance greater than $\mathsf{infr}(f)$.*
- *If $\mathcal{S}$ has a finite number of points, there is a "wiggling" uncertainty, in the order of $o(1/n)$ for uniform distributions.*

The results are summarized in Table 10.2 and the various observability classes are labeled A–D for later reference. We note that the observability results depend both on the *local geometrical properties* of the space (curvature) as well as the *global topological properties* (connectedness).

10.4.6. *Proof overview*

The starting point is considering that the largest unobservable transformations are the set of wigglings (Definition 10.7), because they are exactly those that keep constant the order of the inter-points distances, which is the sufficient statistics for the estimation problem. All other symmetries—isometries (Definition E.3), linear warping (Definition 10.5), generic warping (Definition 10.6) are a specialized version of wigglings. Moreover, an isometry is a linear warping with $\alpha = 1$, and a linear warping is a specialization of a generic warping. In summary, just by the definition of the various transformations, we have the following chain of inclusions:

$$\text{isometries} \subset \begin{matrix}\text{linear}\\ \text{warpings}\end{matrix} \subset \begin{matrix}\text{generic}\\ \text{warpings}\end{matrix} \subset \text{wigglings}.$$

Isometries and warpings are very structured transformations, but wigglings are in general discontinuous. The next step in the analysis is understanding in what cases the set of wigglings is more structured. Proposition 10.9 shows that, as the number of points becomes large (in the limit, infinite), wigglings are constrained to be generic warpings. Thus, if $\mathcal{S}$ has an infinite number of points, we have the following:

$$\text{isometries} \subset \begin{matrix}\text{linear}\\ \text{warpings}\end{matrix} \subset \begin{matrix}\text{generic}\\ \text{warpings}\end{matrix} \overset{n\to\infty}{=} \text{wigglings}.$$

With this assumption, we now can study a much more well-behaved set of transformations. Proposition 10.10 gives the unexpected result that, in general, there exist no generic nonlinear warpings (Definition 10.6), a result that does not depend on the manifold yet (i.e., we did not consider topology or curvature). Intuitively, there is no way to deform the distances in a nonlinear way that maintains the consistency of all constraints. The proof is based on an elementary argument based on the fact that any generic warping must preserve geodesics (Lemma 10.11). Thus, only by assuming that the number of points is large, and with no assumptions on the manifold, we can conclude that

$$\text{isometries} \subset \underset{\text{warpings}}{\overset{\text{linear}}{}} = \underset{\text{warpings}}{\overset{\text{generic}}{}} = \text{wigglings}.$$

This means that the largest group of symmetries of the problem is composed by linear warping. At this point, we have to consider the property of the manifold. Proposition 10.12 shows that, if the manifold has nonpositive curvature, then all linear warpings are necessarily isometries (the scaling factor is 1):

$$\mathcal{M}$$

$$\text{isometries} \overset{\text{curved}}{=} \underset{\text{warpings}}{\overset{\text{linear}}{}} = \underset{\text{warpings}}{\overset{\text{generic}}{}} = \text{wigglings}.$$

This means that for the sphere $\mathbb{S}^2$ and the hyperbolic plane, isometries are the largest group of symmetries. This is surprising, because it means that *we can recover the scale*, even though the measurements available are completely non-metric. Instead, for Euclidean spaces, it is easy to see that a linear warping is always unobservable. Finally, Proposition 10.13 discusses the special case of the circle. Because the topology is not simply connected, it is possible to establish additional constraints: intuitively, an arbitrary warping is not allowed, because if the distribution $\mathcal{S}$ is inflated too much, the tails will "crash" into each other and violate the problem constraints.

10.4.7. *Proof details*

PROPOSITION 10.9. *If $\mathcal{S}$ is a connected open set, all wigglings are generic warpings.*

PROOF. The intuition is that a non-trivial wiggling is possible only if there are "gaps" between the points; as the points get denser, the gaps close and the wiggling degenerates to a warping.

Note that the definition of wiggling does not imply any particular property of the map φ such as continuity. It is a map defined only the subset $\mathcal{S}$ of $\mathcal{M}$. There is no information of

how φ behaves outside of $\mathcal{S}$. However, if $\mathcal{S}$ is an open subset of $\mathcal{M}$, then necessarily φ must have certain regularities.

First of all, it should necessarily be a continuous map. This can be seen directly from the relation $d(s_i, s_j) < d(s_k, s_l) \Leftrightarrow d(\varphi(s_i), \varphi(s_j)) < d(\varphi(s_k), \varphi(s_l))$ if we let $s_i = s_k$ and consider two sequences $s_j^{(m)} \overset{m\to\infty}{\to} s_i$ and $s_l^{(m)} \overset{m\to\infty}{\to} s_k$.

Consider two pairs of points s_i, s_j at distance $\delta = d(s_i, s_j)$. Consider two other pairs of points s_k, s_l with the same relative distance $\delta = d(s_k, s_l)$ — because the set is open, and the distance is continuous, s_k can be found in a neighborhood of s_i and s_j in a neighborhood of s_l. Because $d(s_i, s_j) = d(s_k, s_l)$, the wiggling direction constraint implies that $d(\varphi(s_i), \varphi(s_j)) = d(\varphi(s_k), \varphi(s_l))$. Because s_k, s_l have no other relation to s_i, s_j other than their distance, it follows that the distance of two points transformed by φ only depends on their initial distance: $d(\varphi(s_i), \varphi(s_j)) = m(d(s_i, s_j))$, for some possibly nonlinear function m. Because φ is continuous, this holds for all points in $\mathcal{S}$, therefore φ is a generic warping. □

PROPOSITION 10.10. *All generic warpings are linear warpings.*

PROOF. The proof relies on Lemma 10.11 below, which says that generic warpings preserve the geodesics. This means that, if the midpoint between A and B is C, then $\varphi(C)$ is the midpoint between $\varphi(A)$ and $\varphi(C)$. Let $d(A, C) = d(C, B) = \ell$. Then $d(\varphi(A), \varphi(C)) = d(\varphi(C), \varphi(B)) = m(\ell)$. We can find two different expressions for $d(\varphi(A), \varphi(B))$:

$$d(\varphi(A), \varphi(B)) = m(d(A, B)) = m(2\ell), \quad \text{and}$$

$$d(\varphi(A), \varphi(B)) = d(\varphi(A), \varphi(C)) + d(\varphi(C), \varphi(B)) = 2m(\ell).$$

It follows that $m(\ell) = \frac{1}{2}m(2\ell)$. Generalize this reasoning to an equal division of the geodesics in k parts, to derive $m(x) = \frac{1}{k}m(kx)$, for all $x > 0$ and integers $k \geq 1$. Take the derivative of both sides with respect to x to obtain $m'(x) = m'(kx)$. For any $y > 0$, let

$x = y/k > 0$, and let $k \to \infty$, to obtain $m'(y) = m'(0)$, which implies that m is a linear function. □

LEMMA 10.11. *A generic warping preserves geodesics. More formally, for $A, B \in M$ and $t \in [0,1]$, let $g(A,B,t)$ be the geodesic between A and B. If $\varphi : M \to M$ is a warping, then $g(\varphi(A), \varphi(B), t) = \varphi(g(A,B,t))$.*

PROOF. As a base case, we prove the statement for the midpoint. Suppose that there exists a geodesic between A and B. Let C be the midpoint between A and B, with $d(A,C) = d(C,B) = L$. Let $a = \varphi(A)$ and $b = \varphi(B)$ be the transformed points. Let $c = g(a, b, \frac{1}{2})$ be the midpoint between a and b, with $d(a,c) = d(c,b) = \ell$. Using some elementary properties of geodesics, we shall derive that $\varphi(C) = c$.

Because c is the midpoint, the shortest path between a and b goes through c:

$$d(a,c) + d(c,b) \leq d(a,x) + d(x,b), \qquad \text{for all } x.$$

Write this for $x = \varphi(C)$:

$$d(a,c) + d(c,b) \leq d(a,\varphi(C)) + d(\varphi(C),b)$$

On the right-hand side, substitute $d(a,\varphi(C)) = d(\varphi(A),\varphi(C)) = m(d(A,C))$, using the definition of warping. Likewise $d(\varphi(C),b) = d(\varphi(C),\varphi(B)) = m(d(C,B))$, giving

$$d(a,c) + d(c,b) \leq m(d(A,C)) + m(d(C,B)).$$

The point c is the midpoint, so let $\ell = d(a,c) = d(c,b)$, and $L = d(A,C) = d(C,B)$. We obtain that $\ell \leq m(L)$.

We can do the same computation with A and B. Because C is the midpoint between A

and B, we have that $d(A,C)+d(C,B) \leq d(A,x)+d(x,B)$, for all x. Write it for $x=\varphi^{-1}(c)$ and substitute $A=\varphi^{-1}(a)$ and $B=\varphi^{-1}(b)$ to obtain $2L \leq d(A,\varphi^{-1}(c))+d(\varphi^{-1}(c),B) = d(\varphi^{-1}(a),\varphi^{-1}(c))+d(\varphi^{-1}(c),\varphi^{-1}(b)) = m^{-1}(d(a,c))+m^{-1}(d(c,b)) = 2m^{-1}(\ell)$, which gives us $\ell \geq m(L)$. Together with $\ell \leq m(L)$, we conclude that $\ell = m(L)$. This means that $d(a,\varphi(C)) = d(\varphi(C),b) = \ell$, and hence $\varphi(C)$ is the midpoint between a and b. Because the midpoint is unique, it follows that $c=\varphi(C)$.

We have proved that $g(\varphi(A),\varphi(B),\frac{1}{2}) = \varphi(g(A,B,\frac{1}{2}))$. By dividing the original geodesics, and applying the reasoning above recursively, one can show that $g(\varphi(A),\varphi(B),\frac{a}{2^b}) = \varphi(g(A,B,\frac{a}{2^b}))$ for all integers $b \geq 0$ and $a \leq 2^b$. The set of dyadic rationals $a/2^b$ is dense in $[0,1]$, and the functions $t \mapsto g(\varphi(A),\varphi(B),t)$ and $t \mapsto \varphi(g(A,B,t))$ are continuous, because they are compositions of continuous functions. If two continuous functions on the same domain X agree on a dense subset of X, the agree on the whole domain. Therefore, it holds that $g(\varphi(A),\varphi(B),t) = \varphi(g(A,B,t))$ for all $t \in [0,1]$. □

PROPOSITION 10.12. *For $\mathbb{S}^m, m \geq 2$ and the hyperbolic plane, all linear warpings are isometries.*

PROOF. This is true for all manifolds with nonzero curvature, but the m-sphere and the hyperbolic plane admit an elementary proof based on spherical/hyperbolic geometry. Firstly, note that a linear warping is a conformal map (Definition E.21) as the Jacobian is uniformly α times an orthogonal matrix. Conformal maps preserve angles between geodesics.

Now it is time to recall high school facts about spherical geometry: the sides of a spherical triangle are uniquely determined by its angles. The same is true for the hyperbolic plane [118].

Consider now three points in $\mathcal{S}$ and the induced spherical/hyperbolic triangle. Under a

linear warping, its internal angles are preserved because a linear warping is conformal. Because the angles are preserved, the sides of the triangle are preserved as well, and therefore the distance between points is unchanged. Hence any linear warping is an isometry. □

PROPOSITION 10.13. *If* $\mathcal{M} = \mathbb{S}^1$ *and* $\mathsf{rad}(\mathcal{S}) + \mathsf{infr}(f) < 2\pi$, *a linear warping with* $\alpha \leq (2\pi - \mathsf{rad}(\mathcal{S}))/\mathsf{infr}(f)$ *is unobservable.*

PROOF. (sketch) This can be verified directly; the upper bound on α ensures that the tails of $\mathcal{S}$ do not overlap in the informative range of f. This result does not hold for $\mathbb{S}^2$, where the geometry of the problem constrains linear warpings to be isometries ($\alpha = 1$). □

10.5. Measuring Performance

The performance of an algorithm must be measured in a way compatible with the observability of the problem. We expect an error score to be *invariant,* meaning that it is conserved by the symmetries of the problem. If a score is not invariant, it is measuring something that is not possible to estimate. We expect an error score to be *complete*, meaning that it is minimized only by the solutions of the problem. If an error score is not complete, it cannot be used to distinguish solutions from non solutions. Finally, we wish the error score to be *observable*, in that it can be computed from the data, without the ground truth.

10.5.1. Distances-based performance measures

In our case, several classical error measures, widely used in other contexts, do not satisfy all these properties. The Procrustes error is defined as the mean distance between the solution $\{s_i\}_{i=1}^n$ and the ground truth $\{\bar{s}_i\}_{i=1}^n$, after choosing the best isometry that makes the two sets overlap [119].

DEFINITION 10.14. The *Procrustes error* e_{pr} is defined as

$$e_{\text{pr}}(\{\bar{s}_i\}, \{s_i\}) \triangleq \min_{\varphi \in \mathsf{Isom}(\mathcal{M})} \frac{1}{n} \sum_{i=1}^{n} d(\bar{s}_i, \varphi(s_i)). \tag{10.1}$$

This error score is unsuitable in our case, because, while it is invariant to isometries, it is not invariant to the other symmetries, namely linear warpings and wigglings. This means that, if we are considering an instance of the problem where the scale is not observable, using the Procrustes error can produce misleading results (we will show this explicitly in Section **??**). Moreover, there is the problem that not all points contribute equally to this performance measure. When aligning the two points sets, the points near the center of the distribution will be always more aligned, and the errors will accumulate for the points at the borders of the distribution. To eliminate this problem, we can consider the error on the interpoint distances rather than the absolute position of the points.

DEFINITION 10.15. The *mean relative error* e_r is the mean error between the inter-point distances:

$$e_{\text{r}}(\{\bar{s}_i\}, \{s_i\}) \triangleq \frac{1}{n^2} \sum_{i,j=1}^{n} |d(\bar{s}_i, \bar{s}_j) - d(s_i, s_j)|. \tag{10.2}$$

This error function is still invariant to isometries, does not need an optimization problem to be solved, and all pairs of points contribute equally. Moreover, it can be easily modified to be invariant to linear warpings: because linear warpings scale the distances uniformly, we achieve invariance by optimizing over an unknown scale.

DEFINITION 10.16. The *mean scaled relative error* e_{sr} is the relative error after the optimal warping:

$$e_{\mathsf{sr}}(\{\bar{s}_i\}, \{s_i\}) \triangleq \min_{\alpha > 0} \frac{1}{n^2} \sum_{i,j=1}^{n} |d(\bar{s}_i, \bar{s}_j) - \alpha\, d(s_i, s_j)|. \tag{10.3}$$

This is invariant to warpings. However, it is still not invariant to wigglings. To achieve invariance to wiggling we have to change approach.

10.5.2. Spearman-correlation-based performance measures

We introduce the *Spearman score*: an invariant, complete, and observable score for all observability classes. It is based on the idea of *Spearman correlation*, which measures a possibly *nonlinear* dependence between two variables, in contrast with the usual correlation, which can only assess linear dependence. The Spearman correlation is a common tool in applied statistics, but it is not widely used in engineering. The idea is that, to assess nonlinear relations, we should consider not the value of each datum, but rather their *order* (or *rank*) in the sequence.

DEFINITION 10.17. Let $\mathsf{order} : \mathbb{R}^n \to \mathrm{Perm}(n)$ be the function that computes the order (or rank) of the elements of a vector. For example,

$$\mathsf{order}([2012, 1, 15]) = [2, 0, 1].$$

DEFINITION 10.18. The *Spearman correlation* between two sequences x, y is the Pearson correlation of their order vectors:

$$\mathsf{spear}(x, y) \triangleq \mathsf{corr}(\mathsf{order}(x), \mathsf{order}(y)).$$

LEMMA 10.19. *The Spearman correlation detects any nonlinear monotonic relation:* $\mathsf{spear}(x, y) = \pm 1$ *if and only if* $y = g(x)$ *for some monotonic function* g.

We use this fact to check whether there exists a monotonic function f such that $\mathrm{Y}_{ij} = f(d(s_i, s_j))$. Given a solution $\{s_i\}_{i=1}^n$, we compute the corresponding distance matrix, and then compute the Spearman correlation of the distance matrix to the similarity matrix. To that end, we need to first unroll the matrices into a vector using the operator $\mathsf{vec} : \mathbb{R}^{n \times n} \to \mathbb{R}^{n^2}$.

Table 10.3. Properties of performance measures

	invariant?				*complete?*				*observable?*
obs. class →	A	B	C	D	A	B	C	D	
Procrustes error (10.1)	✗†	✗†	✗§	✗§	✓	✓	✗	✗	✗
Relative error (10.2)	✗†	✗†	✗§	✗§	✓	✓	✗	✗	✗
Scaled relative error (10.3)	✗†	✗†	✗§	✗§	✓	✓	✓	✓	✗
Spearman score (10.4)	✓	✓	✓	✓	✓	✓	✓	✓	✓

†: Not invariant to wigglings.
§: Not invariant to linear warping and wigglings.

DEFINITION 10.20. The *Spearman score* of a solution $\{s_i\}_{i=1}^n$ is the Spearman correlation between the (flattened) similarity matrix and the (flattened) distance matrix $\mathbf{D} = [\mathrm{D}_{ij}] = [d(s_i, s_j)]$:

$$\rho_{\mathrm{sp}}(\{s_i\}) \triangleq |\mathsf{spear}(\mathsf{vec}(\mathbf{Y}), \mathsf{vec}(\mathbf{D}))|. \tag{10.4}$$

The Spearman score is invariant to all symmetries of the problem, including wigglings, which by definition preserve the ordering of the distances. It is also complete because if $\rho_{\mathrm{sp}}(\{s_i\}, \mathrm{Y}_{ij}) = 1$, then there exists an f such that $\mathrm{Y}_{ij} = f(d(s_i, s_j))$.

If the data is corrupted by noise, $\rho_{\mathrm{sp}} = 1$ might not be attainable. In that case, it makes sense to normalize the score by the score of the ground truth.

DEFINITION 10.21. The *Normalized Spearman score* is

$$\rho^*_{\mathrm{sp}}(\{\bar{s}_i\}, \{s_i\}) \triangleq \frac{\rho_{\mathrm{sp}}(\{s_i\})}{\rho_{\mathrm{sp}}(\{\bar{s}_i\})}. \tag{10.5}$$

Table 10.3 summarizes the properties of the performance measures discussed.

10.6. Algorithm

We describe an extension of the classic Shepard-Kruskal algorithm (SK) [111–114] that we call SKv+w (SK variant + warping). The basic idea of SK is to use standard MDS[§]

[§]Given an $n \times n$ distance matrix $\mathbf{D}$, the best embedding in $\mathbb{R}^m$ can be found by solving for the top m eigenvectors of an $n \times n$ semidefinite positive matrix corresponding to a "normalized" version of $\mathbf{D}$ [114, 119].

Algorithm 1 The SKv+w embedding algorithm for a generic manifold $\mathcal{M}$.

Input: similarities $\mathbf{Y} \in \mathbb{R}^{n\times n}$; manifold-specific functions: $\mathsf{MDS}_{\mathcal{M}}$, $\mathsf{distances}_{\mathcal{M}}$, $\mathsf{init}_{\mathcal{M}}$. **Output:** $\mathbf{S} \in \mathcal{M}^n$.

1 **for** $\mathbf{D}^0$ **in** $\mathsf{init}_{\mathcal{M}}(\mathsf{order}(\mathbf{Y}))$: # Some manifolds need multiple starting points.
2 $\quad \mathbf{S}^0 = \mathsf{MDS}_{\mathcal{M}}(\mathbf{D}^0)$ # Compute first guess by MDS.
3 $\quad$ **for** $k = 1, 2, \ldots$ **until** s^k converged:
4 $\quad\quad \mathbf{D}^k = \mathsf{distances}_{\mathcal{M}}(\mathbf{S}^{k-1})$ # Compute current distances.
5 $\quad\quad \mathbf{D}^k_\star = \mathsf{vec}^{-1}(\mathsf{sorted}(\mathsf{vec}(\mathbf{D}))[\mathsf{order}(\mathsf{vec}(\mathbf{Y}))])$ # Nonparametric fitting and inversion of f.
6 $\quad\quad \mathbf{S}^k = \mathsf{MDS}_{\mathcal{M}}(\mathbf{D}^k_\star)$ # Embed according to the modified distances.
7 $\quad\quad s^k = \mathsf{spearman\_score}(\mathbf{S}^k, \mathbf{Y})$ # Use the Spearman score for checking convergence
8 $\mathbf{S}^\star = \mathbf{S}^{k^\star}$, where $k^\star = \arg\max_k s^k$ # Find best iteration according to the score.
9 **if** $\mathcal{M}$ is $\mathbb{S}^m, m \geq 2$: # Find optimal warping factor to embed in the sphere.
10 $\quad \mathbf{D}^\star = \mathsf{distances}_{\mathcal{M}}(\mathbf{S}^\star)$
11 $\quad \alpha^\star = \arg\min_\alpha \sigma^\alpha_{m+1}/\sigma^\alpha_{m+2}$, where $\{\sigma^\alpha_i\} = \mathsf{singular\_values}(\cos(\alpha \mathbf{D}^\star))$
12 $\quad$ **return** $\mathsf{MDS}_{\mathcal{M}}(\alpha^\star \mathbf{D}^\star)$ # Embed the warped distances.
13 **return** $\mathbf{S}^\star$

$\mathcal{M}$-specific initializations: $\quad \mathsf{init}_{\mathbb{R}^m}(\mathbf{oY}) \triangleq \mathbf{oY}$; $\quad \mathsf{init}_{\mathbb{S}^m}(\mathbf{oY}) \triangleq \{\pi\mathbf{oY}/n^2, 2\pi\mathbf{oY}/n^2\}$.

on Y_{ij} to obtain a first guess for $\{s_i\}$. Given this guess, one can obtain a rough estimate $\tilde{f}$ of f; given $\tilde{f}$, one can apply $\tilde{f}^{-1}$ to Y_{ij} to obtain an estimate of the distances D_{ij}; then one solves again for $\{s_i\}$ using MDS. The SK algorithm does not give accurate metric reconstruction. Our goal was to obtain a general algorithm that could work in all corner cases of the observability analysis. The algorithm described here will be shown to be robust across a diverse set of benchmarks on different manifolds, with a vast variation of shapes of f and noise levels. To this end, we extended the SK algorithm in several ways. In the following, some parts are specific to the manifold: we indicate by $\mathsf{MDS}_{\mathcal{M}}$ a generic implementation of MDS on the manifold $\mathcal{M}$, so that $\mathsf{MDS}_{\mathbb{R}^n}$ is the classical Euclidean MDS, and $\mathsf{MDS}_{\mathbb{S}^n}$ is the spherical MDS employed by Grossmann et al. [105].

10.6.1. *EM-like iterations (lines 3–7 of Algorithm 1)*

A straightforward extension is to iterate the alternate estimation of $\{s_i\}$ and f in an EM-like fashion. This modification has also been introduced in other SK variants [114]. This iteration improves the solution, but still does not give metrically accurate solutions.

10.6.2. *Choice of first guess for the distance matrix (line 1)*

Assuming that the similarities have already been normalized ($-1 \leq \mathrm{Y}_{ij} \leq 1$), the standard way to obtain an initial guess D^0_{ij} for the distance matrix is to linearly scale the similarities, setting $\mathrm{D}^0_{ij} \propto 1 - \mathrm{Y}_{ij}$. This implies that, given the perturbed similarities $\mathrm{Y}^\star_{ij} = g(\mathrm{Y}_{ij})$ for some monotone function g, the algorithm starts from a different guess and has a different trajectory. However, because the sufficient statistics $\mathsf{order}(\mathrm{Y}^\star_{ij}) = \mathsf{order}(\mathrm{Y}_{ij})$ is conserved, we expect the same solution. The fix is to set $\mathrm{D}^0_{ij} \propto \mathsf{order}(\mathrm{Y}_{ij})$ (making sure the diagonal is zero), so that the algorithm is automatically invariant to the shape of f.

10.6.3. *Multiple initializations (line 1)*

We observed empirically that multiple initializations are necessary for the case of $\mathbb{S}^m$. In particular, if one scales D^0_{ij} such that $0 \leq \mathrm{D}^0_{ij} \leq \pi$, all solutions generated have diameter $\leq \pi$; if one scales D^0_{ij} such that $0 \leq \mathrm{D}^0_{ij} \leq 2\pi$, all solutions have diameter $\geq \pi$. Extensive tests show that one of the two starting points always allows convergence to the true solution (the other being stuck in a local minimum). In Algorithm 1 this is represented by a manifold-specific function $\mathsf{init}_{\mathcal{M}}$ returning the list of initial guesses for $\mathbf{D}$.

10.6.4. *Non-parametric inversion of f (line 5)*

We have to find some representation for f, of which we do not know the shape, and use this representation to compute f^{-1}. In this kind of scenarios, a common solution is to use a flexible parametric representation for f, such as splines or polynomials. However, parametric fitting is typically not robust to very noisy data. A good solution is to use completely *non-parametric* fitting of f. Suppose we have two sequences $\{x_i\}, \{y_i\}$ which implicitly model a noisy relation $y_i = f(x_i) + \text{noise}$ for some monotone f. Our goal is to estimate the sequence $\{f^{-1}(y_i)\}$. Let $\mathsf{sorted}(\{x_i\})$ be the sorted sequence $\{x_i\}$. Then non-parametric inversion can be obtained by using the order of $\{y_i\}$ to index into the

sorted $\{x_i\}$ array[¶]:

$$\{f^{-1}(y_i)\} \simeq \mathsf{sorted}(\{x_i\})[\mathsf{order}(\{y_i\})].$$

This is seen in line 5 applied to the (unrolled) distance and similarity matrices.

10.6.5. *Spearman Score as convergence criterion (line 3)*

The iterations are stopped when the Spearman score converges. In practice, we observed that after 4–7 iterations the score has negligible improvement for all benchmarks. This score is also used to choose the best solution among multiple initializations (line 8).

10.6.6. *Warping recovery phase (lines 9–12)*

The most important change we introduce is a "warping recovery" phase that changes the qualitative behavior of the algorithm in the case of $\mathbb{S}^m, m \geq 2$. As explained in the observability analysis, in curved spaces the scale of the points distribution is observable. However, the SK algorithm (i.e., lines 3–7 of Algorithm 1) cannot compensate what we call a linear warping (Definition 10.5); in fact, it is easy to see that if $\mathbf{D}_0$ is a fixed point of the loop, also $\alpha\mathbf{D}_0$, for $\alpha > 0$, is a fixed point. In other words, the "null space" of the Shepard-Kruskal algorithm appears to be the group of linear warpings. Therefore, we implemented a simple algorithm to find the scale that best embeds the data onto the sphere, based on the fact that if $\mathbf{D}$ is a distance matrix for a set of points on $\mathbb{S}^m$, then the cosine matrix $\cos(\mathbf{D})$ must have rank $m+1$. Therefore, to find the optimal scale, we look for the optimal $\alpha > 0$ such that $\cos(\alpha\mathbf{D})$ is closest to a matrix of rank 3. This is implemented in lines 9–12, where the ratio of the $(m+1)$-th and the $(m+2)$-th singular value is chosen as a robust measure of the rank.

[¶]The square brackets here indicate indexing into the array, as in most programming languages (e.g., Python).

While it would be interesting to observe the improvements obtained by each variation to the original algorithm, for reasons of space we focus only on the impact of the warping recovery phase. We call SKv the SKv+w algorithm *without* the warping recovery phase (i.e., without the lines 9–12).

10.6.7. *Algorithm complexity*

The dominant cost of SKv+w lies in the truncated SVD decomposition needed for $\mathsf{MDS}_{\mathcal{M}}$ in the inner loop; the exact decomposition takes $O(n^3)$, which is, in practice, in the order of 5 ms for $n = 100$ and 500 ms for $n = 1000$ on current hardware[||]. There exist faster approximations to speed up the MDS step; see, e.g., the various Nystrom approximations [120].

10.7. Camera Calibration Results

10.7.1. *Hardware*

We use three different cameras, covering all practical cases for imaging systems: a perspective camera ("FLIP" in the following), a fish-eye camera ("GOPRO"), and an omnidirectional catadioptric camera ("OMNI"). FLIP: The Flip Mino HD [121] is a $100 consumer-level video recorder (Fig 10.1a). It has a 45° FOV; it has a 3X optical zoom, not used for these logs. GOPRO: The GOPRO camera [122] is a $300 rugged professional-level fish-eye camera for outdoor use (Fig 10.2a). The field of view varies slightly between 127° and 170° according to the resolution chosen; for our tests, we chose a resolution corresponding to a 150° field of view. OMNI: We used a custom-made omnidirectional catadioptric camera (Fig. 10.3a). This is a small, compact system very popular for micro aerial platforms, such as quadrotors [123, 124]. The camera is created by connecting a perspective camera to

[||] Tests executed using Numpy 1.5, BLAS compiled with Intel MKL, on a 2.67Ghz Intel Xeon core.

a hyperbolic mirror. The resulting field of view is 360° (horizontally) by 100° (vertically). The images have much lower quality than the FLIP and GOPRO (Fig. 10.2b). Table 10.4 summarizes the statistics of the three datasets.

Table 10.4. Dataset statistics

camera	FOV	*fps*	*resolution*	*subsampling*	n	*length*
FLIP	45°	30	1280×720	24 × 24 grid	1620	57416
GOPRO	150°	30	1280×720	24 × 24 grid	1620	29646
OMNI	360°	20	640×480	8 × 8 grid	1470	13131

10.7.2. *Manual calibration*

We calibrated the cameras using conventional techniques, to have a reference to which to compare our method. We calibrated the FLIP using the Matlab Calibration Toolbox [92], which uses a pin-hole model plus second-order distortion models. We calibrated the GOPRO and the OMNI using the OCamCalib calibration toolbox [102], using a fourth-order polynomial for describing the radial distortion profile [95, 96, 125]. Both methods involve printing out a calibration pattern (a checkerboard), taking several pictures of the board, then identifying the corners of the board using a semi-interactive procedure. Some examples of the calibration images used are shown in Fig. 10.1c and 10.2c.

10.7.3. *Data collection*

The environment in which the log is taken influences the spatial statistics of the images. The data logs were taken in a diverse set of environments. For the FLIP, the data was taken outdoors in the Caltech campus, which has a large abundance of natural elements. For the GOPRO, the data was taken in the streets of Philadelphia, a typical urban environment. For the OMNI, the data was taken indoors in a private apartment and an office location. Examples of the images collected are shown in Fig. 10.1b, 10.2b, 10.3b. In all cases, the cameras were held in one hand and waved around "randomly", trying to exercise at least

three degrees of freedom (shoulder, elbow, wrist), so that the attitude of the camera was approximately uniformly distributed in $\mathrm{SO}(3)$. We did not establish a more rigorous protocol, as these informal instructions produced good data. Data taken by exercising only one degree of freedom of the arm (e.g., forearm, with the wrist being fixed) did not satisfy the monotonicity assumption. Another example of data that we tried that did not satisfy the assumption was data from an omnidirectional camera mounted on a car[**].

10.7.4. *Data processing*

For all cameras, the original RGB stream of each pixel was converted to a one-dimensional signal by computing the luminance. We also subsampled the original images with a regular grid so that we could work with a reduced number of points. For the OMNI data, we used masking to only consider the annulus around the center (Fig. 10.3c), therefore excluding the reflection of the camera in the mirror and the interior of the box which lodged the camera. We used the correlation between the pixel luminance values as the similarity statistics: $\mathrm{Y}_{ij} = \mathsf{corr}(\, y_i(t),\ y_j(t)\,)$, where $y_i(t)$ indicates the luminance of the i-th pixel at time t. This simple statistics was the most useful across cameras (Section 10.7.6 discusses other possible choices of the similarity statistics). We found that the monotonicity condition is well verified for all three cameras. To plot these statistics, we assume the calibration results obtained with conventional techniques as the ground truth. The joint distribution of the similarity Y_{ij} and the distance $d(s_i, s_j)$ is shown in Fig. 10.4b, 10.5b, 10.6b. For these logs, the spatial statistics were quite uniform: at a distance of 45°, the inter-pixel correlation was in the range 0.2–0.3 for all three cameras. For the GOPRO and OMNI data, the correlation is 0 at around 90°. The correlation is negative for larger distances. The different average luminance between sky and ground (or floor and ceiling) is a possible explanation

[**]Because the car motion is mostly planar, a portion of the pixels always observes the sky (a featureless scene), while others observe the road (a scene richer in features).

for this negative correlation. The OMNI data is very noisy for distances in the range 90°–180°, as the sample correlation converges more slowly for larger distances. To check that the monotonicity condition is satisfied, regardless of the shape of f, it is useful to look at the Spearman diagrams in Fig. 10.4c, 10.5c, 10.6c, for the FLIP, GOPRO, and OMNI, respectively. These diagrams show, instead of similarity (Y_{ij}) versus distance ($d(s_i, s_j)$), the order of the similarities ($\mathsf{order}(Y_{ij})$) versus the order of the distances ($\mathsf{order}(d(s_i, s_j))$). The correlation of those gives the Spearman score (Definition 10.20). If there was a perfectly monotonic relation between similarity and distance, the Spearman diagram would be a straight line, regardless of the shape of f, and the Spearman score would be 1 (Lemma 10.19).

10.7.5. *Calibration results*

The results of manual calibration and calibration using our method are graphically shown in Fig. 10.4d, 10.5d, 10.6d. The plots show the data using spherical coordinates (azimuth/elevation). There is a number of intuitive remarks that can be made on the results by direct observations of the resulting point distributions (or, better, its 3D equivalent). For the FLIP data (Fig. 10.4d) the reconstructed directions lie approximately on a grid, as expected. For this data, and the GOPRO as well, the estimated points are more regular at the center of the field of view than on the borders. This is probably due to the fact that the pixels at the border have less constraints. The estimated FOV is very similar to the result given by the manual calibration (43° instead of 45°). For the GOPRO data (Fig. 10.5d) the shape of the sensor is well reconstructed, except for the two upper corners of the camera. The estimated FOV matches the manual calibration (153° instead of 150°). For the OMNI data (Fig. 10.6d) the shape of the sensor is overall well reconstructed, but it is more noisy than the FLIP or GOPRO. This is to be expected as the monotonicity relation is not as well respected (Fig. 10.6e).

Table 10.5. Calibration results (normalized Spearman score)

dataset			*norm. Spearman score* ρ^*_{sp}			
$\mathcal{S}$	FOV	f	*g. truth*	SKv	SKv+w	MDS
FLIP	45°	corr(y)	1	0.9998	**1.0006**	0.9709
GOPRO	150°	corr(y)	1	1.0027	**1.0029**	0.9702
OMNI	360°	corr(y)	1	**1.0288**	**1.0288**	0.9831

It can be concluded that our method gives results reasonably close to manual calibration, even for cases like the OMNI where the monotonicity condition holds only approximately. As predicted by the observability analysis, the scale can be reconstructed even without knowing anything about the function f.

We now look at quantitative performance measures. As explained before, the only admissible performance measure is the Spearman score, shown in Table 10.5. When judged by this performance measure, the SKv+w algorithm is slightly better than the manual calibration (the normalized Spearman score is larger than 1). In other words, the estimated distribution is actually a better fit of the similarity data than the manual calibration results. This implies that the imprecision in the estimate is a limitation of the input data rather than of the optimization algorithm; to obtain better results, we should improve on the data rather than improving the algorithm.

The Procrustes error (Equation 10.1) is the most intuitive performance measure (but not invariant to wiggling). The results are shown in Table 10.6. The error with respect to manual calibration is an average of 0.7° for the FLIP data, 3.5° for the GOPRO data, and 9.5° for the OMNI data. The table shows both the results with and without the warping phase (SKv+w and SKv, respectively). This makes it clear that the warping phase is necessary to obtain a good estimate of the directions, especially for the FLIP data. The difference is lower for the GOPRO data and negligible for the OMNI data. Intuitively, the warping phase takes advantages of what can be called "second-order" constraints, in the sense that

Table 10.6. Calibration results (Procrustes error)

dataset			*Procrustes error*		
$\mathcal{S}$	FOV	f	SKv	SKv+w	MDS
FLIP	45°	corr(y)	24.05°	**0.74°**	15.16°
GOPRO	150°	corr(y)	4.72°	**3.53°**	6.20°
OMNI	360°	corr(y)	**9.48°**	**9.48°**	32.43°

they allow us to establish the scale at small FOV, but they disappear as the FOV tends to zero, because a small enough section of $\mathbb{S}^2$ looks flat (like $\mathbb{R}^2$). Finally, it is clear that the accuracy of MDS is much lower than SKv or SKv+w. In general, MDS obtains topologically correct solutions, but the scale is never correctly recovered, or the data appears otherwise deformed.

These results seem to outperform the results shown in Grossmann et al.: compare, for example, Figure 13 in [105]. Note that their method assume that the function f is known, obtained through a separate calibration phase. In principle, with much more information, their results should be better. Without having access to their data, we can only speculate on the reason. Perhaps the simplest explanation is that they do not "wave around" the camera for collecting the data; and therefore the monotonicity condition might not be as well satisfied. Moreover, they use a similarity statistics which has very low informative radius (30°), which might cause problems, even though the robust nonlinear embedding algorithm they use should be robust to this fact.

10.7.6. *Results for different similarity statistics*

Proposition 10.2 ensures that *any* statistics is a function of the pixel distance, but this result is limited in three ways: (1) it is only an asymptotic result, valid as time tends to infinity; (2) it assumes a perfectly uniform attitude distribution; and (3) it does not ensure that the function f is invertible (monotonic). Therefore, it is still an engineering matter to

find a statistics which is (1) robust to finite data size; (2) robust to a non-perfectly uniform trajectory; and (3) has a large invertible radius. An exhaustive treatment of this problem is delegated to future work. Here, we briefly show the results for three other statistics in addition to the luminance correlation. All statistics are defined as the correlation of an instantaneous function of the luminance and can be efficiently computed using streaming methods. The first variant consists in applying an instantaneous contrast transformation $c : y \mapsto y^2$ to the luminance before computing the correlation:

$$\mathrm{Y}_{ij} = \mathsf{corr}(\, c(y_i(t)),\, c(y_j(t))\,). \tag{10.1}$$

The second statistic is the correlation of the temporal derivative $\dot{y} = \frac{d}{dt}y$ of the luminance:

$$\mathrm{Y}_{ij} = \mathsf{corr}(\, \dot{y}_i(t),\, \dot{y}_j(t)\,). \tag{10.2}$$

This was inspired by recent developments in neuromorphic hardware [126]. Finally, we consider the correlation of the *sign* of the luminance change, as it is invariant to contrast transformations:

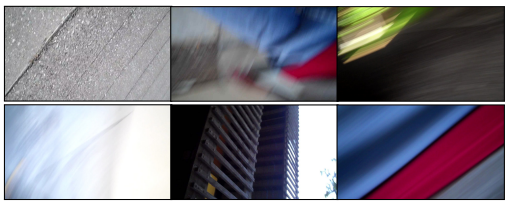
$$\mathrm{Y}_{ij} = \mathsf{corr}(\, \mathsf{sgn}(\dot{y}_i(t)),\, \mathsf{sgn}(\dot{y}_j(t))\,). \tag{10.3}$$

Table 10.7 shows the Spearman score obtained by using these on the OMNI data (the most challenging dataset). We find, in this case, that the contrast-scaled luminance (10.1) is slightly better than the simple correlation; the solution found is qualitatively similar. The two other similarity statistics (10.2) and (10.3) have much lower scores; for them, the monotonicity assumption is not well verified: their distributions are not informative for large distances. It is clear that there is a huge design space for similarity statistics. In the end, we did not find any statistic which was better than the simple correlation *uniformly* for all our three data sets. Therefore, we consider this an open research question.

Table 10.7. Results with different similarity statistics

dataset		Spearman score	
$\mathcal{S}$	f	g. truth	SKv+w
OMNI	$\mathrm{corr}(y)$	0.9173	**0.9438**
OMNI	$\mathrm{corr}(c(y))$	0.9212	**0.9465**
OMNI	$\mathrm{corr}(\dot{y})$	0.8550	**0.9211**
OMNI	$\mathrm{corr}(\mathrm{sgn}(\dot{y}))$	0.8739	**0.9077**

(a) The Flip Mino

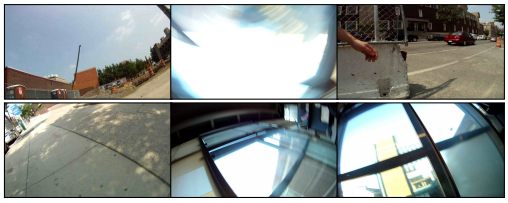


(b) Some frames from the calibration sequence.

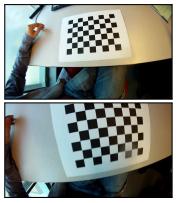
(c) Calibration patterns

Figure 10.1. The FLIP camera is a consumer-level portable video recorder. The data for calibration is taken while walking in the Caltech campus, with the camera in hand, and "randomly" waving the arm, elbow, and wrist.

(a) The GOPRO camera

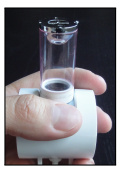


(b) Some frames from the calibration sequence

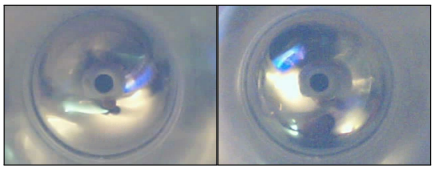


(c) Calibration pattern

Figure 10.2. The GOPRO camera is a rugged consumer camera for outdoors use. It uses a fish-eye lens with 170° field of view.



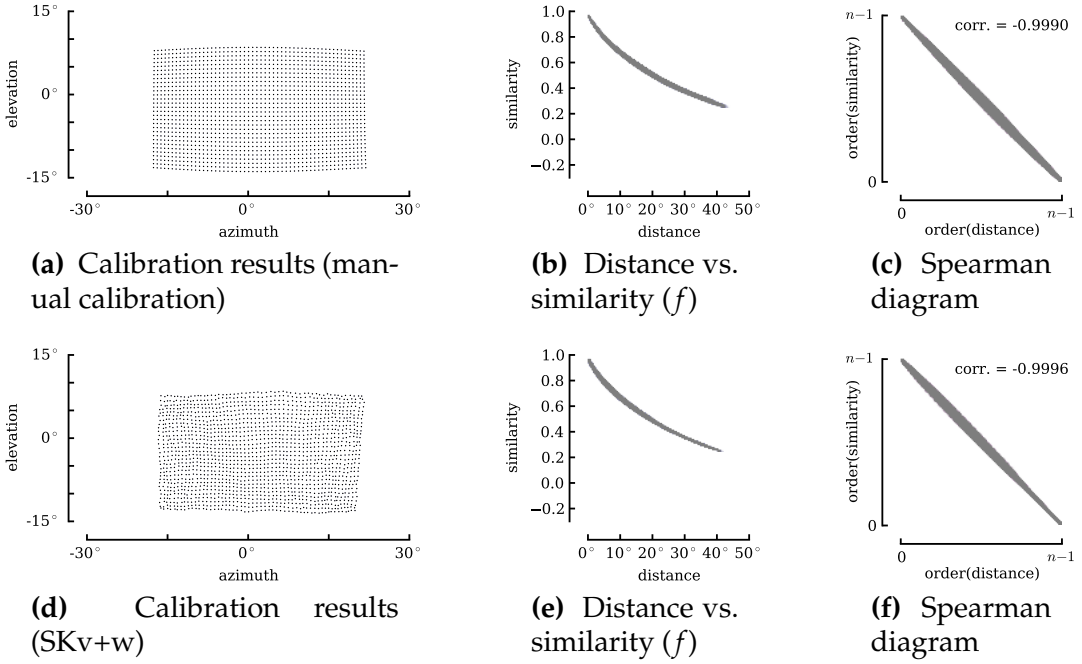
(a) Catadioptric camera



(b) Some frames from the calibration sequence

(c) Mask used

Figure 10.3. Note the small dimensions of this omnidirectional catadioptric camera, very well suited for aerial robotics applications. The data quality is much lower than for the FLIP and GOPRO data.



(a) Calibration results (manual calibration)

(b) Distance vs. similarity (f)

(c) Spearman diagram

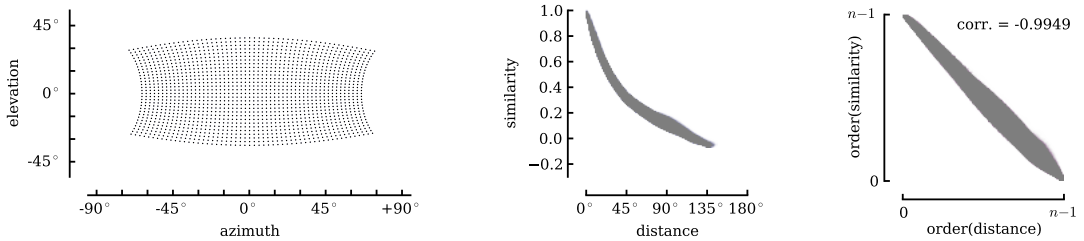
(d) Calibration results (SKv+w)

(e) Distance vs. similarity (f)

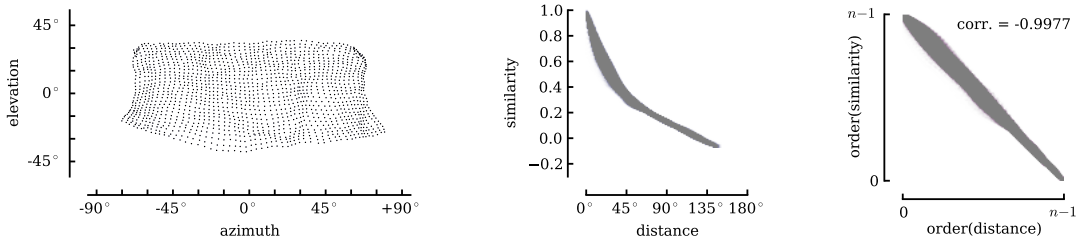
(f) Spearman diagram

Figure 10.4. Calibration results for the FLIP data using $\mathrm{corr}(y)$ as the similarity statistics.

Legend for Figure 10.4,10.5,10.6: The first row (fig. a,b,c) shows the results of calibration using conventional methods, while the second row (d,e,f) shows the results of our algorithm. The first column (a, d) shows the points distribution on the sphere, displayed using azimuth/elevation coordinates. The second column (b, e) shows the joint distribution of pixel distance ($d(s_i, s_j)$) and pixels similarities (Y_{ij}), which, in this case, is the correlation. This is the function f that we should fit. Finally, the third column (c, f) shows $\mathrm{order}(d(s_i, s_j))$ vs. $\mathrm{order}d(s_i, s_j)$ and their correlation, from which we derive the Spearman score.

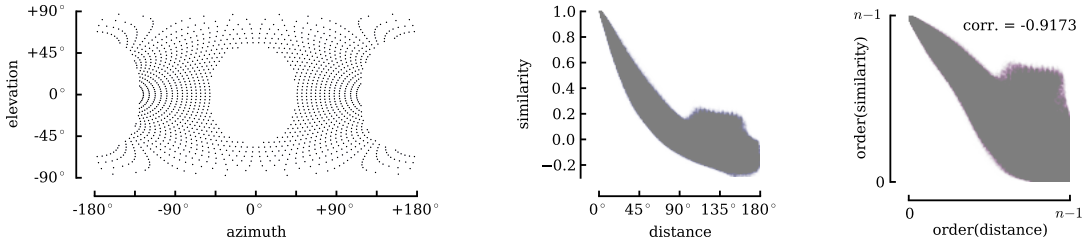


(a) Calibration results (manual calibration) **(b)** Distance vs. similarity (f) **(c)** Spearman diagram

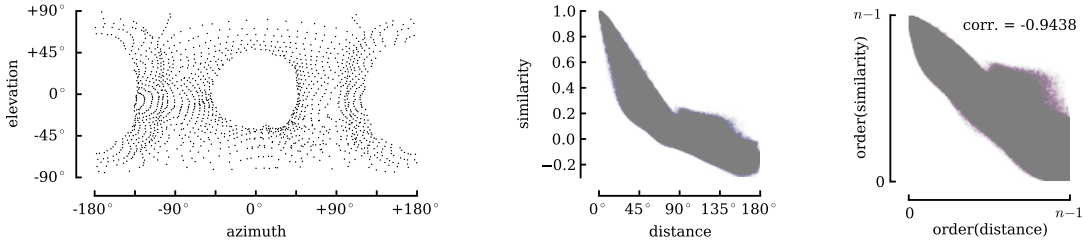


(d) Calibration results (SKv+w) **(e)** Distance vs. similarity (f) **(f)** Spearman diagram

Figure 10.5. Calibration results for the GOPRO data using $\mathrm{corr}(y)$ as the similarity statistics.



(a) Calibration results (manual calibration) **(b)** Distance vs. similarity (f) **(c)** Spearman diagram



(d) Calibration results (SKv+w) **(e)** Distance vs. similarity (f) **(f)** Spearman diagram

Figure 10.6. Calibration results for the OMNI data using $\mathrm{corr}(y)$ as the similarity statistics.

CHAPTER 11

Learning Bilinear Dynamics

This chapter describes the BDS *class of models, in which the dynamics of the observations is a bilinear function of the observations themselves, and shows how it approximates the canonical robot sensors.*

Table 11.1. Symbols used in this chapter

BDS models		
M^{s}_{vi}	$n_{\boldsymbol{u}} \times n_{\boldsymbol{y}} \times n_{\boldsymbol{y}}$ tensor	Tensor parametrizing the model.
P^{sv}	$n_{\boldsymbol{y}} \times n_{\boldsymbol{y}}$ tensor	Second moment matrix of $\boldsymbol{y}$.
$\mathbf{Q}$	$n_{\boldsymbol{u}} \times n_{\boldsymbol{u}}$ tensor	Second moment matrix of $\boldsymbol{u}$.
T^{svi}	$n_{\boldsymbol{u}} \times n_{\boldsymbol{y}} \times n_{\boldsymbol{y}}$ tensor	Statistics computed during learning.
$\mathsf{BDS}(n;k)$	$\subset \mathcal{D}(\mathbb{R}^n;\mathbb{R}^k)$	The class of BDS systems.

11.1. Why Bilinear Systems

In the most general case, a continuous-time dynamical system can be written as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$; $\boldsymbol{y} = h(\boldsymbol{x})$, where $\boldsymbol{x}$ represents the hidden state. However, one seldom sees an explicit model of this kind for sensors such as cameras or range finders, because the function h should encode all information regarding the environment, and a closed form is impossible to write except in the simplest of environments. One alternative representation is focusing on the observations dynamics $\dot{\boldsymbol{y}} = g(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{x})$. In most cases, the function g depends on the underlying unobservable state $\boldsymbol{x}$. An agent that does not have access to the state $\boldsymbol{x}$ (and its dynamics) cannot learn such a model. This motivates us to look at approximating the observations dynamics by disregarding the dependence on the state, thus looking for models of the form $\dot{\boldsymbol{y}} = g(\boldsymbol{y}, \boldsymbol{u})$. Because the agent has access to $\boldsymbol{y}$, $\dot{\boldsymbol{y}}$, and $\boldsymbol{u}$, learning the map g from the data is a well-defined problem.

Rather than trying to learn a generic nonlinear g, which appears to be a daunting task, especially for cases where $\boldsymbol{y}$ consists of thousands of elements (pixels of a camera), our approach has been to keep simplifying the model until one obtains something tractable. A second-order linearization of g leads to the expression

$$\dot{\boldsymbol{y}} = \boldsymbol{a} + A\boldsymbol{y} + B\boldsymbol{u} + C(\boldsymbol{y}, \boldsymbol{y}) + D(\boldsymbol{y}, \boldsymbol{u}) + E(\boldsymbol{u}, \boldsymbol{u}). \tag{11.1}$$

Here A and B are linear operators, but C, D, E are tensors (later we make the tensor notation more precise). If $\boldsymbol{y}$ and $\boldsymbol{u}$ have dimensions n_y and n_u, then C, D, E have dimensions, respectively, $n_y \times n_y \times n_y$, $n_y \times n_y \times n_u$, and $n_y \times n_u \times n_u$.

We can ignore some terms in (11.1) by using some semantic assumptions regarding our specific context.

- Assumption 13 (*A known command does nothing*)

If $\boldsymbol{u}$ represents a "movement" or "velocity" command, in the sense that if $\boldsymbol{u}$ is 0, then the pose does not change, and $\boldsymbol{y}$ does not change as well ($\boldsymbol{u} = 0 \Rightarrow \dot{\boldsymbol{y}} = 0$), we can omit the terms $\boldsymbol{a}$, $A\boldsymbol{y}$ and $C(\boldsymbol{y}, \boldsymbol{y})$, and we are left with $\dot{\boldsymbol{y}} = B\boldsymbol{u} + D(\boldsymbol{y}, \boldsymbol{u}) + E(\boldsymbol{u}, \boldsymbol{u})$.

- Assumption 14 (*Minus does the opposite*)

  If we assume that $\boldsymbol{u}$ is a symmetric velocity commands, in the sense that applying $+\boldsymbol{u}$ gives the opposite effect of applying $-\boldsymbol{u}$, then we can get rid of the $E(\boldsymbol{u}, \boldsymbol{u})$ term as well.

We are left with the model $\dot{\boldsymbol{y}} = B\boldsymbol{u} + D(\boldsymbol{y}, \boldsymbol{u})$, where D is a bilinear operator. We can incorporate $B\boldsymbol{u}$ into the second term by assuming there is a trivial observation whose value is always 1. In conclusion, our *ansatz* for a generic robotic sensor is a bilinear model of the kind $\dot{\boldsymbol{y}} = D(\boldsymbol{y}, \boldsymbol{u})$.

11.2. BDS

DEFINITION 11.1 (BDS). A system is a *bilinear dynamics sensor* (BDS), if observations and commands are vectors of real numbers, and the derivative of $\boldsymbol{y}$ depends linearly on $\boldsymbol{u}$ and $\boldsymbol{y}$. In formulas, there exists a (1,2) tensor $\mathbf{M}$ such that

$$\dot{y}^s = \sum_{i,v} \mathsf{M}^s_{vi} y^v u^i. \tag{11.1}$$

$\mathsf{BDS}(n;k) \subset \mathcal{D}(\mathbb{R}^n; \mathbb{R}^k)$ is the set of all such systems.

In this chapter, we use the Einstein convention according to which repeated up and down indices are automatically summed over, so that the explicit "$\sum$" symbol can be omitted. We can rewrite (11.1) more compactly as

$$\dot{y}^s = \mathsf{M}^s_{vi} y^v u^i.$$

11.2.1. *Symmetries of the* BDS *class*

PROPOSITION 11.2. *The* $\mathsf{BDS}(n;k)$ *family is closed with respect to the action of* $\mathsf{GL}(n)$ *on the observations and* $\mathsf{GL}(k)$ *on the commands:*

$$\mathsf{GL}(n) \cdot \mathsf{BDS}(n;k) \cdot \mathsf{GL}(k) = \mathsf{BDS}(n;k).$$

PROOF. Let $\boldsymbol{y}' = \mathbf{A}\boldsymbol{y}$ and $\boldsymbol{u} = \mathbf{B}\boldsymbol{u}'$, for some matrices $\mathbf{A} \in \mathsf{GL}(n)$, $\mathbf{B} \in \mathsf{GL}(k)$. Then the relation between $\dot{\boldsymbol{y}}'$, $\boldsymbol{y}'$ and $\boldsymbol{u}'$ is bilinear. □

11.3. Space and Time Discretization

While we have defined the model for continuous time and space. So, a natural question is whether the properties of the system (bilinearity) are preserved for discretized systems, both in time and space.

11.3.1. *Space discretization preserves bilinearity*

As for space discretization, we assume that the continuous function $y(s,t)$, $s \in \mathcal{S}$, is band limited, and it is sampled at a dense enough set of points to achieve reconstruction. If the sampling is uniform, we know they need to be at the Nyquist frequency. However, this is not a necessary condition. Margolis [127, Chapter 3] gives an elementary exposition on the topic of nonuniform sampling and reconstruction. See the classic paper by Yen [128] for examples of perturbations of uniform sequences that still allow reconstruction. Call a *sufficient sampling sequence* a set of n points on S^1, possibly irregular, such that it is possible to reconstruct the signal exactly.

PROPOSITION 11.3 (Spatial discretization preserves bilinearity). *Assuming the signal* $y(s,t)$ *is spatially band limited and it is sampled at a (possible irregular) set of points sufficient for reconstruction, spatial discretization preserves bilinearity, in the sense that if* $\dot{y}(s,t) = \nabla_s y(s,t)u(t)$

and $A = \{s_A^1, \ldots, s_A^n\} \in \left(\mathbb{S}^1\right)^n$ *is a sufficient sampling sequence of n elements, then the dynamics of* $y(A,t) = \{y(s_A^1,t), \ldots, y(s_A^n,t)\} \in \mathbb{R}^n$ *is* $\dot{y}(A,t) = \overline{\mathbf{V}} y(A,t)u(t)$ *where* $\overline{\mathbf{V}}$ *is constant* $n \times n$ *matrix.*

PROOF. By using the definition of gradient, we can write $\dot{y}$ as a limit:

$$\begin{aligned} \dot{y}(s,t) &= \nabla_s y(s,t)u(t) \\ &\qquad \text{(Definition of gradient)} \\ &= \lim_{\delta \to 0} \frac{y(s+\delta,t) - y(s,t)}{\delta} u(t). \end{aligned}$$

Writing this for each point in A:

$$\dot{y}(A,t) = \lim_{\delta \to 0} \frac{y(A+\delta,t) - y(A,t)}{\delta} u(t). \tag{11.1}$$

From Lemma 11.5 we know $A + \delta$ is a sufficient sampling sequence, and from Lemma 11.4 we can write $y(A+\delta,t)$ as a function of $y(A,t)$:

$$y(A+\delta,t) = \mathbf{V}_A^{A+\delta} y(A,t).$$

Substituting this back in (11.1):

$$\begin{aligned} \dot{y}(A,t) &= \lim_{\delta \to 0} \frac{\mathbf{V}_A^{A+\delta} y(A,t) - y(A,t)}{\delta} u(t) \\ &= \left(\lim_{\delta \to 0} \frac{\mathbf{V}_A^{A+\delta} - I}{\delta} \right) y(A,t)u(t) \\ &\doteq \overline{\mathbf{V}} y(A,t)u(t). \end{aligned}$$

This implies that the spatially discretized dynamics is still bilinear. □

LEMMA 11.4. *Suppose that* $A, B \subset (\mathbb{S}^1)^n$ *are two sufficient sampling sequences for a signal*

$y : \mathbb{S}^1 \to \mathbb{R}$. *Then* $y(B) = \mathbf{V}_A^B y(A)$ *for some matrix* $\mathbf{V}_A^B \in \mathbb{R}^{n\times n}$.

PROOF. An elementary proof is the following. Let $U = \{s_U^1, \dots, s_U^n\} \subset (\mathbb{S}^1)^n$ be a uniform sampling of $\mathbb{S}^1$. We know from classical results that, in this case, we can reconstruct the value $y(s)$ for any $s \in \mathbb{S}^1$ by linear interpolation using a basis of equal time-shifted functions:

$$y(s) = \sum_{i=1}^{n} \varphi(s - s^i) y(s^i). \tag{11.2}$$

(The basis functions are *sinc* but it is not relevant here.) In particular, given any sampling sequence $A = \{s_A^1, \dots, s_A^n\} \subset (\mathbb{S}^1)^n$, we can reconstruct its values $y(A)$ from the values $y(U)$ sampled at the uniform sequence by a linear matrix:

$$y(A) = \mathbf{V}_U^A y(U),$$

where $y(A) = \{y(s_A^1), \dots, y(s_A^n)\} \in \mathbb{R}^n$ is the irregularly sampled signal, $y(U) = \{y(s_U^1), \dots, y(s_U^n)\} \in \mathbb{R}^n$ is the signal sampled uniformly, and $\mathbf{V}_U^S$ is an $n \times n$ matrix whose entries are defined by the coefficients of (11.2):

$$(\mathbf{V}_U^A)_u^a = \sum_{i=1}^{n} \varphi(s_A^a - s_U^i) y(s_U^u).$$

If A is a sufficient sampling sequence, then it possible to express $y(U)$ as a function of $y(A)$. The inverse of a linear map is linear. Thus it is possible to write:

$$y(U) = \mathbf{V}_A^U y(A), \tag{11.3}$$

and necessarily $\mathbf{V}_A^U = \left(\mathbf{V}_U^A\right)^{-1}$. The same thing can be said for another sufficient sampling sequence B, thereby $y(B) = \mathbf{V}_U^B y(U) = \mathbf{V}_U^B \mathbf{V}_A^U y(A) \doteq \mathbf{V}_A^B y(A)$. □

LEMMA 11.5. *If A is a sufficient sampling sequence, then $A + \delta = \{s_A^1 + \delta, \dots, s_A^n + \delta\}$ is a*

sufficient sampling sequence.

REMARK 11.6. This discussion only involves existence formulas. Do not confuse this with a way to reconstruct the signal at the unknown samples. For inference, nowadays we would use Gaussian Processes regression [78], however it assumes certain shapes of the covariance (even if jointly determined) which are stronger assumptions about the signal than just being of finite bandwidth (which can be a constraint given by the physical structure).

11.3.2. *Time discretization preserves bilinearity only to first order*

Suppose the dynamics of $\boldsymbol{y}$ is bilinear ($\dot{\boldsymbol{y}} = \mathbf{M}\boldsymbol{u}\boldsymbol{y}$) and that the dynamics is discretize at intervals of length T. The value of $\boldsymbol{y}$ at the $(k+1)$-th instant is given by

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \int_{kT}^{kT+T} \mathbf{M}\boldsymbol{u}_t\boldsymbol{y}_t \,\mathrm{d}t.$$

Assuming that the commands take the constant value $\boldsymbol{u}_k$ in the interval $[kT, kT+T]$, the solution can be written using the matrix exponential (Lemma E.9) as

$$\boldsymbol{y}_{k+1} = \mathsf{mexp}(T(\mathbf{M}\boldsymbol{u}_k))\boldsymbol{y}_k.$$

The right-hand side is still linear in $\boldsymbol{y}$, but it is not linear in the commands. Using the expansion of the matrix exponential (E.1), the first few terms are

$$\mathsf{mexp}(T(\mathbf{M}\boldsymbol{u}_k) = \mathbf{I} + T\mathbf{M}\boldsymbol{u}_k + o(T^2).$$

The difference $\Delta\boldsymbol{y}_k = \frac{1}{T}(\boldsymbol{y}_{k+1} - \boldsymbol{y}_k)$ is then linear with respect to $\boldsymbol{u}$ only at the first order:

$$\Delta\boldsymbol{y}_k = \mathbf{M}\boldsymbol{u}_k\boldsymbol{y}_k + o(T^2).$$

See Elliot [129] for more insights on the dynamics of bilinear systems and the geometric properties of their flow maps.

11.4. Learning

In the learning phase, the agent randomly samples control commands $\boldsymbol{u}$ from any zero-mean distribution with positive definite covariance. Meanwhile, it estimates three quantities:

(1) The average observation at each sensel $\overline{\boldsymbol{y}}$:

$$\overline{y}^s = \mathbb{E}\{y^s\}, \tag{11.1}$$

(2) The $(2,0)$ covariance tensor $\mathbf{P}$:

$$\mathsf{P}^{sv} = \mathrm{cov}(y^s, y^v), \tag{11.2}$$

(3) The $(3,0)$ tensor $\mathbf{T}$ defined by

$$\mathsf{T}^{svi} = \mathbb{E}\{(y^s - \overline{y}^s)\, \dot{y}^v u^i\}. \tag{11.3}$$

These expectations can be computed online, using recursive definitions such as

$$\overline{y}^s(k+1) = \frac{k}{k+1}\overline{y}^s(k) + \frac{1}{k+1}y^s(k).$$

We note that these computations can be implemented on a neural architecture; (11.3) is similar to three-way Hebbian learning between $\boldsymbol{y}$, $\dot{\boldsymbol{y}}$, and $\boldsymbol{u}$. In fact, the expectation of the product of the three terms $(y^s - \overline{y}^s)$, $\dot{y}^v$, u^i can be thought as an approximation of the frequency that the three signals are active together.

The following proposition establishes that the tensor $\mathbf{T}$, tends to approximate the tensor $\mathbf{M}$ in (11.1).

LEMMA 11.7. *Let* $\mathbf{P}$, $\mathbf{Q}$ *be the covariance of* $\boldsymbol{y}$ *and* $\boldsymbol{u}$. *Then the tensor* $\mathbf{T}$ *tends asymptotically to*

$$\mathsf{T}^{svi} = \mathsf{M}^{s}_{qj}\mathsf{P}^{qv}\mathsf{Q}^{ij}. \tag{11.4}$$

PROOF. We can prove that this tends to:

$$\begin{aligned}
\mathsf{T}^{svi} &= \mathbb{E}\{\dot{y}^s\,(y^v - \overline{y}^v)\,u^i\} \\
&\qquad \text{(Substituting the model 11.1 for } \dot{y}\text{,and changing indices.)} \\
&= \mathbb{E}\{(\mathsf{M}^{s}_{qj}y^q u^j)(y^v - \overline{y}^v)u^i\} \qquad (11.5) \\
&\qquad \text{(Independence of } \boldsymbol{u}, \boldsymbol{y}\text{, and the fact that } \mathbf{M} \text{ is a constant.)} \\
&= \mathsf{M}^{s}_{qj}\mathbb{E}\left\{y^q\,(y^v - \overline{y}^v)\right\}\mathbb{E}\{u^i u^j\} \qquad (11.6) \\
&\qquad \left(\mathbb{E}\left\{y^q\,(y^v - \overline{y}^v)\right\} = \operatorname{cov}(y^q, y^v)\right) \\
&= \mathsf{M}^{s}_{qj}\mathsf{P}^{qv}\mathsf{Q}^{ij}.
\end{aligned}$$

□

In (11.4), we can observe a general pattern. Every quantity that the agent learns ultimately depends on three factors:

(1) *The agent's sensorimotor cascade.* In this case, $\mathbf{M}$.

(2) *The environment statistics*. In this case, the covariance $\mathbf{P}$ represents the effect of the specific environment in which learning takes place. For example, in the case of a camera, the covariance $\mathbf{P}$ depends on the statistics of the environment texture, and it would change in different environments.

(3) *The experience the agent had in such environment*. In this case, the tensor $\mathbf{Q}$ captures the kind of "training" the agent had in the environment.

11.5. Servoing

Models are only as good as the decisions they allow to make. We use servoing as an example task for a bootstrapping agent. Let $\check{\boldsymbol{y}}$ be some given "goal" observations. Then, based on the learned model, one can derive a control strategy that minimizes $\|\boldsymbol{y} - \check{\boldsymbol{y}}\|$.

PROPOSITION 11.8. *Assume that the agent is equipped with a* BDS *(Definition 11.1), that it has learned* $\mathbf{P}$ *and* $\mathbf{T}$ *using* (11.2)–(11.3), *and* $\mathbf{Q}$ *is positive definite. Then the control law*

$$u^i = -\sum_v \sum_q (y^v - \check{y}^v)^* \mathsf{T}^{svi} \mathsf{P}^{-1}_{vq} y^q \tag{11.1}$$

corresponds to a descent direction of the error metric $\|\boldsymbol{y} - \check{\boldsymbol{y}}\|$. *If the operators* $\{\mathsf{M}^s_{vi} y^v\}_{i=1}^k$ *commute,* $\check{\boldsymbol{y}}$ *is asymptotically stable.*

PROOF. The first part of the proof shows that the control is a descent direction for $V = \|\boldsymbol{y} - \check{\boldsymbol{y}}\|^2$. Defining the error signal $e^s = y^s - \check{y}^s$, we can write $V = \frac{1}{2} e^s m_{rs} e^r$. The

derivative of V can be computed as follows (sum symbols are omitted for clarity).

$$\begin{aligned}
\dot{V} &= \dot{e}^s m_{rs} e^r = \dot{y}^s m_{rs} e^r \\
&= [\mathsf{M}^s_{vi} u^i y^v] m_{rs} e^r \\
&\qquad \text{(Expanding } \boldsymbol{u}\text{.)} \\
&= -\mathsf{M}^s_{vi} [e^z m_{zw} \mathsf{T}^{wxi} \mathsf{P}^{-1}_{xp} y^p] y^v m_{rs} e^r \\
&\qquad \text{(Expanding } \mathbf{T}\text{.)} \\
&= -\mathsf{M}^s_{vi} e^z m_{zw} [\mathsf{M}^w_{qj} \mathsf{P}^{qx} \mathsf{Q}^{ij}] \mathsf{P}^{-1}_{xp} y^p y^v m_{rs} e^r \\
&\qquad \text{(Reordering everything.)} \\
&= -y^v \mathsf{M}^s_{vi} e^z m_{zw} \mathsf{M}^w_{qj} \mathsf{P}^{qx} \mathsf{P}^{-1}_{xp} y^p \mathsf{Q}^{ij} m_{rs} e^r \\
&\qquad \text{(Tensors } \mathbf{P} \text{ and } \mathbf{P}^{-1} \text{ cancel.)} \\
&= -(y^v \mathsf{M}^s_{vi} m_{rs} e^r) \mathsf{Q}^{ij} (y^p \mathsf{M}^w_{pj} m_{zw} e^z) \\
&\qquad \text{(Let } g_i = y^v \mathsf{M}^s_{vi} m_{rs} e^r\text{.)} \\
&= -g_i \mathsf{Q}^{ij} g_j \leq 0.
\end{aligned}$$

This proves that V never increases. If we prove that g is never 0 in a neighborhood of $\check{\boldsymbol{y}}$, then $\dot{V} < 0$, and V is then a Lyapunov function, making $\check{\boldsymbol{y}}$ asymptotically stable. To prove this second part we have to use some tools from nonlinear system theory [130, Chapter 11, page 540]. In general, because the Lie algebra is nilpotent of order 0, any solution of (11.1) can be written in the form

$$\boldsymbol{y} = \exp(\mathsf{M}^{\cdot}_{\cdot 1} b_1) \exp(\mathsf{M}^{\cdot}_{\cdot 2} b_2) \cdots \exp(\mathsf{M}^{\cdot}_{\cdot k} b_k) \check{\boldsymbol{y}}, \tag{11.2}$$

where the b_j are known as the *Philip Hall coordinates*, and exp represents the exponential of

a linear operator. We are interested only on the behavior near $\check{\boldsymbol{y}}$, therefore we can linearize each of the term as

$$\exp(\mathsf{M}^{\cdot}_{\cdot j} b_j) = I + \mathsf{M}^{\cdot}_{\cdot j} b_j + o(\|\boldsymbol{b}\|^2).$$

The linearized version of (11.2) is then

$$y^s = \check{y}^s + \mathsf{M}^s_{vj} b^j \check{y}^v + o(\|\boldsymbol{b}\|^2).$$

From this we get $e^s = \mathsf{M}^s_{vj} \check{y}^v b^j + o(\|\boldsymbol{b}\|^2)$, which we substitute in the definition of g_i to obtain

$$\begin{aligned} g_i &= \check{y}^v \mathsf{M}^s_{vi} m_{rs} e^r + o(\|\boldsymbol{b}\|^2) \\ &= \check{y}^v \mathsf{M}^s_{vi} m_{rs} (\mathsf{M}^r_{qj} \check{y}^q b^j) + o(\|\boldsymbol{b}\|^2). \end{aligned}$$

Recall that we are in a neighborhood of $\check{\boldsymbol{y}}$ (but not precisely at $\check{\boldsymbol{y}}$). This implies that $\boldsymbol{b}$ is not zero; otherwise, from (11.2) we get $\boldsymbol{y} = \check{\boldsymbol{y}}$. Assuming that the $\mathsf{M}^r_{qj} \check{y}^q$ commute, we also have that $\mathsf{M}^r_{qj} \check{y}^q b^j \neq 0^r$. In fact, if they commute, we can write (11.2) as $\boldsymbol{y} = \exp(\mathsf{M}^r_{qj} \check{y}^q b^j) \check{\boldsymbol{y}}$ and the argument of the exponential must be different from 0. These two together imply that $g_i = \check{y}^v \mathsf{M}^s_{vi} m_{rs} (\mathsf{M}^r_{qj} \check{y}^q b^j) \neq 0$ near the origin. Here the abundance of indices is masquerading the simplicity of the assertion. Without indices: suppose there is a vector $\boldsymbol{v} \neq 0$, and a linear operator A such that $A\boldsymbol{v} \neq 0$; then $A^* A \boldsymbol{v} \neq 0$. In this case, $\boldsymbol{v} = \boldsymbol{b}$ and $A^v_i = \check{y}^v \mathsf{M}^s_{vi}$. □

REMARK 11.9. It is a classic result [131] that, if a system such as (11.1) is nonholonomic, there exists no smooth controller that stabilizes $\check{\boldsymbol{y}}$ asymptotically. In particular (11.1) is smooth in $\boldsymbol{y}$, therefore it cannot work in the nonholonomic case. Instead, the requirement that the operators $\{\mathsf{M}^s_{vi} y^v\}_{i=1}^k$ commute is a technical necessity for having a compact proof

and can probably be relaxed.

11.6. Invariance to Linear Transformations

Proposition 11.2 has shown that the BDS class is closed with respect to the action of linear transformations, both on the commands and the observations. This does not automatically imply that the agent behavior has those symmetries as well. Section 5.6 has shown that the symmetries of the agent behavior depend on the symmetries of the task. In the previous section, the servoing task was described as minimizing the norm of the error metric

$$\|\boldsymbol{y} - \check{\boldsymbol{y}}\|. \tag{11.1}$$

This error metric is symmetric only with respect to orthogonal transformations of the observations:

$$\begin{aligned} \boldsymbol{y} &\mapsto \mathbf{A}\boldsymbol{y}, \qquad \mathbf{A} \in \mathsf{O}(n_{\boldsymbol{y}}), \\ \check{\boldsymbol{y}} &\mapsto \mathbf{A}\check{\boldsymbol{y}}. \end{aligned} \tag{11.2}$$

Therefore, while the BDS class of models is invariant to $\mathsf{GL}(n_{\boldsymbol{y}})$, an agent that would implement this strategy would only be invariant to $\mathsf{O}(n_{\boldsymbol{y}})$, which is a much smaller group. As for the commands, the agent is invariant to $\mathsf{GL}(n_{\boldsymbol{u}})$, because the error function (11.1) does not depend on $\boldsymbol{u}$.

There are two approaches to make the agent more invariant: making the task invariant, or making the observations canonical.

11.6.1. *Invariant task design*

The first approach consists in designing a different error function that takes into account the desired symmetries. One possibility is to choose

$$\|\mathsf{P}^{-\frac{1}{2}}(\boldsymbol{y}-\check{\boldsymbol{y}})\|. \tag{11.3}$$

Because the covariance matrix transforms as $\mathbf{P} \mapsto \mathbf{A}\mathbf{P}\mathbf{A}^T$, the new error metric is invariant to all linear transformations of the observations.

The control strategy (11.1) is easily amended, by noticing that, for any positive definite tensor $\mathbf{X}$, the control

$$u^j = -\sum_{r,s,v} \mathsf{Q}^{ij} e^r \mathsf{X}_{rs} \mathsf{M}^s_{vi} y^v \tag{11.4}$$

is a descent direction for the error metric $\frac{1}{2} e^r \mathsf{X}_{sr} e^s$.

11.6.2. *Invariance by canonization*

The second approach consists in applying a *canonization operator*. The invariant metric (11.3) coincides with the old metric (11.1) if the covariance matrix is equal to the identity. If the agents observations are filtered by a whitening operator that makes the observations canonical by enforcing $\mathbf{P} = \mathbf{I}$, then the agent has gained a larger invariance group. This general canonization approach is the object of Part 3.

Using either of these strategies leads to this formal result about the agent.

PROPOSITION 11.10. *The servoing behavior of a* BDS *agent is invariant to the representation nuisance* $\mathsf{GL}(n) \times \mathsf{GL}(k)$ *acting on* $\mathsf{BDS}(n,k)$.

Table 11.2. BDS approximations

	dynamics	*learned tensors*
field sampler	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \nabla_i y^s \boldsymbol{v}^i$	
		$\mathsf{T}^{svi} = \nabla^{\mathbb{R}^3}_j \mathsf{P}^{sv} \mathsf{Q}^{ij}, \quad i \in [1,3]$
		$\mathsf{T}^{svi} = (s \times \nabla^{\mathbb{R}^3} \mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6]$
camera	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + \mu^s \nabla_i y^s \boldsymbol{v}^i$	
		$\mathsf{T}^{svi} = \overline{\mu}^s \nabla^{\mathbb{S}^2}_j \mathsf{P}^{sv} \mathsf{Q}^{ij}, \quad i \in [1,3]$
		$\mathsf{T}^{svi} = (s \times \nabla^{\mathbb{S}^2} \mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6]$
range-finder	$\dot{y}^s = (s \times \nabla y^s)_i \, \boldsymbol{\omega}^i + (\nabla_i \log y^s - s^*_i) \boldsymbol{v}^i$	
		$\mathsf{T}^{svi} = \nabla^{\mathbb{S}^2}_j \beta(\mathsf{P}^{sv}) \mathsf{Q}^{ij}, \quad i \in [1,3]$
		$\mathsf{T}^{svi} = (s \times \nabla^{\mathbb{S}^2} \mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6]$

For the camera and range-finder, which are not exactly BDS, these tensors could be considered a "projection" of the nonlinear dynamics to the BDS space. The tensor $\mathbf{P}$ is the covariance of $\boldsymbol{y}$; the tensor $\mathbf{Q}$ is the covariance of $\boldsymbol{u}$; $\overline{\mu}^s$ is the average nearness (inverse of distance) in direction s.

11.7. BDS Approximation of Canonical Sensors

11.7.1. *Field-samplers*

The field-sampler dynamics (Lemma 9.6) are exactly bilinear, so it can be represented as a BDS. We can compute the exact form for the learned tensor $\mathbf{T}$. Assuming the general case of a fully actuated rigid body in $\mathrm{SE}(3)$, the tensor $\mathbf{T}$ has 6 components for the last index. The first three ($1 \leq i \leq 3$) correspond to linear velocity, and the last three ($4 \leq i \leq 6$) to the angular velocity. We assume that in the training distribution linear and angular velocity are uncorrelated, so that we can show the results separately.

PROPOSITION 11.11. *The learned tensor for a field-sampler* is

$$\mathsf{T}^{svi} = \nabla^{\mathbb{R}^3}_j \mathsf{P}^{sv} \mathsf{Q}^{ij}, \quad i \in [1,3], \tag{11.1}$$

$$\mathsf{T}^{svi} = (s \times \nabla^{\mathbb{R}^3} \mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6]. \tag{11.2}$$

PROOF. We show the computation for the linear velocity components:

$$\begin{aligned}\mathsf{T}^{svi} &\triangleq \mathbb{E}\{(y^s - \overline{y}^s)\,\dot{y}^v \boldsymbol{v}^i\} \\ &= \mathbb{E}\{(y^s - \overline{y}^s)\,(\nabla_j y^v)\boldsymbol{v}^j \boldsymbol{v}^i\} \\ &= \nabla_j \mathbb{E}\{(y^s - \overline{y}^s)\,y^v\}\mathbb{E}\{\boldsymbol{v}^j \boldsymbol{v}^i\} = \nabla_j \mathsf{P}^{sv}\mathsf{Q}^{ij}.\end{aligned}$$

The formula for the others is obtained similarly. □

11.7.2. *Vision sensors*

The dynamics of a camera (Proposition 9.8) are not exactly bilinear, because there is a hidden state, the nearness μ.

PROPOSITION 11.12. *The learned tensors for a camera are*

$$\begin{aligned}\mathsf{T}^{svi} &= \overline{\mu}^s \nabla_j^{\mathbb{S}^2} \mathsf{P}^{sv}\mathsf{Q}^{ij}, \quad i \in [1,3], \\ \mathsf{T}^{svi} &= (s \times \nabla^{\mathbb{S}^2}\mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6].\end{aligned} \tag{11.3}$$

PROOF. The proof is similar as Proposition 11.11: write the definition of T^{svi}, substitute (9.7) and carry on the computation. □

11.7.3. *Range finders*

PROPOSITION 11.13. *If the training distribution is mixing (Definition 9.12) and the environment is monotone (Definition 9.18) the learned tensors for a range-finder are*

$$\begin{aligned}\mathsf{T}^{svi} &= \nabla_j^{\mathbb{S}^2} \beta(\mathsf{P}^{sv})\mathsf{Q}^{ij}, \quad i \in [1,3] \\ \mathsf{T}^{svi} &= (s \times \nabla^{\mathbb{S}^2}\mathsf{P}^{sv})_j \mathsf{Q}^{ij}, \quad i \in [4,6]\end{aligned} \tag{11.4}$$

where $\beta(\mathsf{P}^{sv})$ is an element-wise scalar function of P^{sv}.

PROOF. The proof for the rotational part is the same as the camera. As for translation, notice that a compact way to write the dynamics is $\dot{\sigma^s} = (\nabla_j \log \sigma^s - s_j^*)\boldsymbol{v}^j$. Straight computation gives the following.

$$
\begin{aligned}
\mathsf{T}^{sxi} &= \mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\,\dot{\sigma}^x \boldsymbol{v}^i\} \\
&\qquad \text{(Using the observation dynamics.)} \\
&= \mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\,(\nabla_j \log \sigma^x - x_j^*)\boldsymbol{v}^j \boldsymbol{v}^i\} \\
&\qquad \text{(Separating the two terms)} \\
&= \mathsf{Q}^{ij}\left[\mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\,\nabla_j \log \sigma^x\} - \mathbb{E}\{x_j^*\,(\sigma^s - \overline{\sigma}^s)\}\right] \\
&\qquad \text{(The second term disappears given that } x_j^* \text{is a constant.)} \\
&= \mathsf{Q}^{ij}\left[\mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\,\nabla_j \log \sigma^x\}\right] \\
&\qquad \text{(We can pull out } \nabla \text{ due to linearity.)} \\
&= \mathsf{Q}^{ij}\nabla_j \mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\log \sigma^x\}.
\end{aligned}
$$

At this point, note that if we had σ^x instead of $\log \sigma^x$ inside the expectation, the result would be P^{sx}, the covariance of σ. Define $\mathsf{R}^{sx} = \mathbb{E}\{(\sigma^s - \overline{\sigma}^s)\log \sigma^x\}$. Then we can invoke Proposition 9.13 to say that R^{sx} is a function of $d(s,x)$. Because the environment is monotone, we know there is a 1-to-1 correspondence between P^{sx} and $d(s,x)$, therefore we can write $\mathbf{R}^{sx}$ as a function of P^{sx}. □

11.8. Simulations and experiments

We simulate a planar omnidirectional robot controlled in velocity. The commands are $\boldsymbol{u} = (u_1, u_2, u_3) = (\boldsymbol{v}_x, \boldsymbol{v}_y, \boldsymbol{\omega})$. We simulate a 180 range finder, an omnidirectional camera,

and a field sampler, with the sensels placed on a ring.

The learning procedure consists in placing the robot in a randomly generated map at random places (simulated as a uniform variable on the subset of $\mathrm{SE}(2)$ that does not intersect any obstacle), and simulate the sensor output $(\boldsymbol{y}, \dot{\boldsymbol{y}})$ when the robot chooses a random command $\boldsymbol{u}$ (simulated as a Gaussian random variable with spherical covariance).

We found out that, if one uses environment shapes which are too simple, the learned model will pick up the characteristic of the environment. For example, if a robot with a 360 range-finder is always placed in a room of the same size, say 10 m, it will learn that, if the readings in front measure 1 m, it is likely that the readings in the back measure 9 m — this knowledge is represented implicitly in the estimated covariance matrix, which will report strong negative correlation between readings in front and in the back. We do not want to see these effects, which are not representative of the real world, and to prevent them we simulate random environments composed of randomly sampled polygonal walls. See Fig. 11.1 for examples of the random environments used.

As long as there is some variability, the results are largely independent of the details of how the randomness is introduced. These observations motivated the definition of the environment's *symmetry group* (Definition 9.10) and how it affects the observation covariance.

For simulating a camera sensor, one should choose a random texture for the surfaces: for example, sample a luminance value independently for each 20 cm section of the surface, and smooth the result. Choosing structured inputs such as sinusoids introduces unwanted correlation. This motivated the definition of *monotone environment* (Definition 9.18).

For the field sampler, we simulated a random distribution of point sources with quadratic decay.

Figures 11.2 onward show the learned tensors. In all figures, blue means negative

and red positive; each figure is normalized independently from the others. Subfigures *b–c* show the covariance ($\mathbf{P}$) and information ($\mathbf{P}^{-1}$) tensors of the simulated sensors. These tensors are given as a function of the sensels position s, v. In these three simulated agents, where the sensels are placed on a semicircle, we let s, v be the angle with respect to the robot front. The covariance encodes information on the sensel topology. For the camera and range-finder, the covariance is sparse and local: only sensels that are very close to each other are correlated, and the correlation is a function of the sensels distance. The covariance/information matrix act very similarly to convolution/deconvolution operators.

Subfigures *d–f* show the learned tensor $\mathbf{T}$. If there are n sensels and 3 commands, then $\mathbf{T}$ is a $n \times n \times 3$ tensor. We show the 3 bidimensional slices $\mathsf{T}^{sv(1)}$, $\mathsf{T}^{sv(2)}$, $\mathsf{T}^{sv(3)}$. For example, the slice $\mathsf{T}^{sv(1)}$ describes how the linear velocity v_x is related to $y(s)$ and $\dot{y}(v)$. Subfigures *g-i* show the normalized tensor $\mathbf{T}\mathbf{P}^{-1}$ used in the control law (11.1). While $\mathbf{T}$ depends on the environment (because of the internal dependence on the covariance $\mathbf{P}$, which depends on the environment statistics), $\mathbf{T}\mathbf{P}^{-1}$ cancels out the environmental contribution. Perhaps the results for range-finder and camera are easier to interpret. If one imagines the slices T^{svi} as linear operators that, applied to $\boldsymbol{y}$, give $\dot{\boldsymbol{y}}$, it is evident that the agent learns local spatial gradients; we shall see this theoretically.

The pictures also show the (inferior) results with the control law

$$u^i = -\sum_v \sum_q (y^v - \check{y}^v)^* \mathsf{T}^{svi} y^*_s, \tag{11.1}$$

which is simpler than (11.1) because it omits the computation of the inverse of the covariance.

Figures 11.5 onward show the convergence properties of the control law (11.1). We are interested in evaluating the radius of convergence. We sample numerous environments and goal configurations; then we compute the control law in a volume around the goal

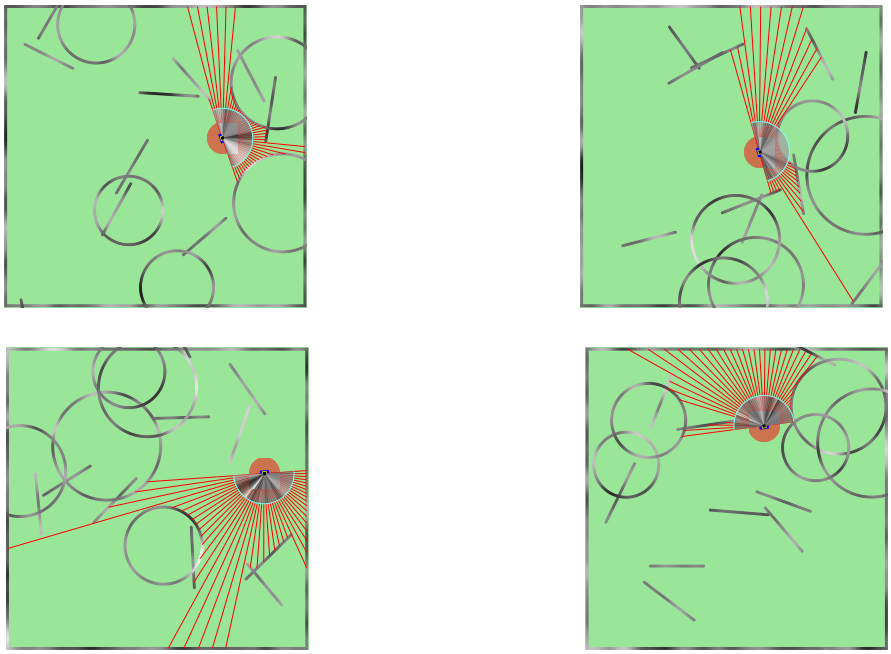


Figure 11.1. Examples of random worlds simulated for learning. The red rays represent the range-finder readings. The black and white arc around the robot represent the simulated output of camera. Note the random geometry and the random texture. Figure best viewed on screen zooming in on the details.

configuration. We show the results as "success maps": we slice the $\boldsymbol{q} = (x, y, \theta)$ volume at the three planes (x, y), (x, θ), (θ, y), and we count the percentage of times the control law pointed the robot in the right direction, meaning that it would have decreased the distance to the goal. In formulas: let the goal be $\boldsymbol{q} = (x, y, \theta) = (0, 0, 0)$. Then a "successful" command is one for which $d\|\boldsymbol{q}\|/dt \propto xu_1 + yu_2 + \theta u_3 < 0$. Light green means $> 99\%$; see the caption for the other values.

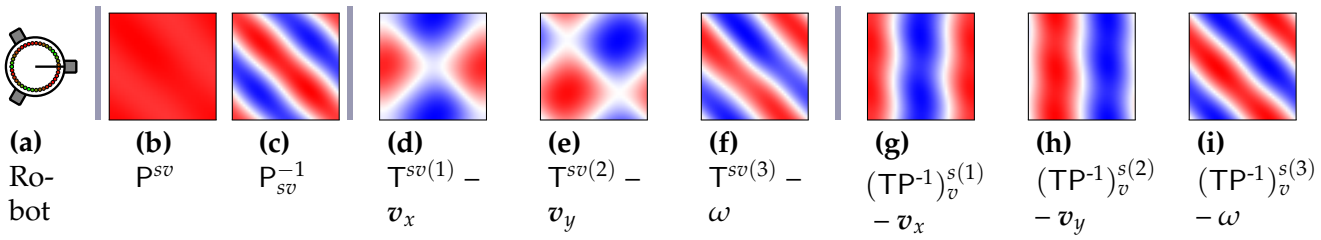


Figure 11.2. Tensors learned for robot with a field sampler, with sensels placed on a 360deg ring. Each axis corresponds to an angle on the ring. Red means positive; blue negative; white zero. Subfigures *e–f* show correlation and information matrix, as a function of the sensel positions $s, v \in \mathcal{S}$, which can be thought of the angle on the sensor ring. The correlation is almost identically 1; this depends on the statistics of the field we simulated. Figure *b–d* show the three slices of the learned tensor $\mathbf{T}$, for the three commands $(\boldsymbol{v}_x, \boldsymbol{v}_y, \boldsymbol{\omega})$. The tensor element T^{isv} represents the interaction between the i-th command, the observation y^s and the derivative $\dot{y}^v$. Figures *g–i* show the 3 slices of the normalized tensor $\mathbf{TP}^{-1}$. (Images best seen in color.)

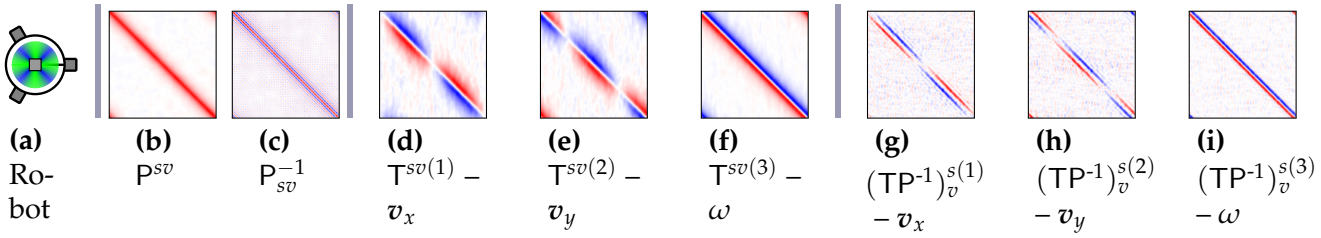


Figure 11.3. Tensors learned for robot with omnidirectional camera. See Fig. 11.2 for a general description of the figures. All pictures are a functions of two sensels $s, v \in \mathcal{S}$, which corresponds to the pixel orientation. In *e–f*, we can see that the covariance matrix of a camera is sparse and local: only nearby sensels interact. Figures *b–d* show the learned tensor $\mathbf{T}$; if one interprets each slice as a linear operator that, applied to $\boldsymbol{y}$, gives $\dot{\boldsymbol{y}}$, it is clear that all three represents functions of the gradient of $\boldsymbol{y}$. Depending on the particular environment, the gradient is more or less smoothed by the covariance. Figures *g–i* represent the slices of the normalized tensor $\mathbf{TP}^{-1}$: here the effect of the environment statistics are factored away and an even more local operator is obtained.

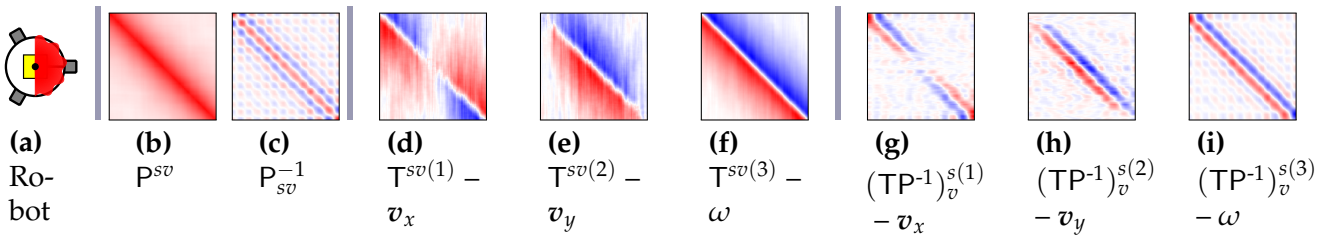


Figure 11.4. Tensors learned for robot with range-finder (180deg FOV). See Fig. 11.2 for a general description of the figures. It is interesting to compare these results with the camera results in Fig. 11.3. The covariance is less local: this means that, in the environment we simulated, the range readings are more correlated than the covariance; that is, they change less abruptly. As a consequence, the tensor $\mathbf{T}$ is less local than the corresponding tensor for the camera. Figures *g–i* show that the normalized tensor $\mathbf{TP}^{-1}$, where the effect of the environment statistics is removed, has a more local character.

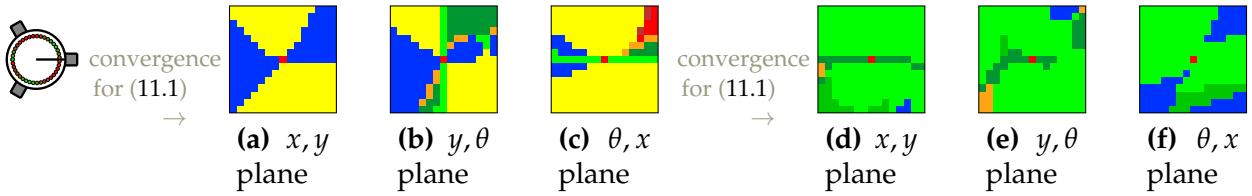


Figure 11.5. Statistics of the convergence of the two control laws for robot with a field sampler, with sensels placed on a 360deg ring. Figures (a)-(c) show the results for the simplified control law (11.1), the figures (d)-(f) show the results for the control law (11.1). We put the goal at the origin, and considered starting positions sampled in a 1m×1m×45deg parallelepiped around the goal. We show the convergence results along three slices in the planes x, y, y, θ, θ, x. The figures show the percentage of times (over 200 trials with random environments) that the control law indicated a direction decreasing the error metric. The color scale is: 0% <25% >25% >50% >75% >95% 100%. These figures are best seen on a computer screen.

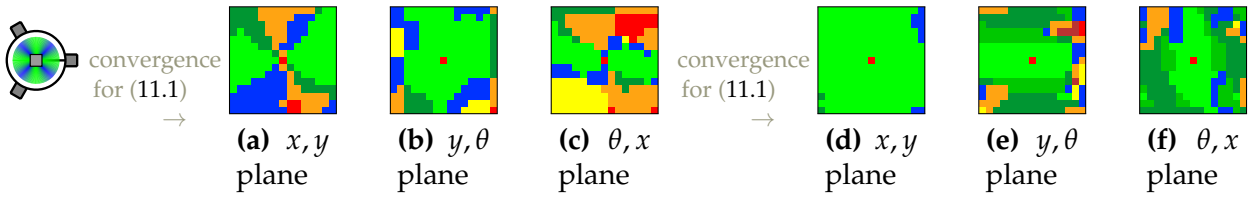
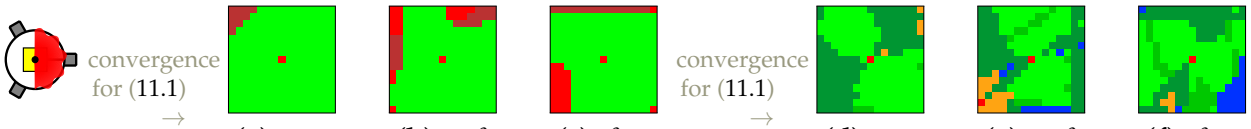


Figure 11.6. Convergence results for robot with omnidirectional camera. See the caption of Fig. 11.5 for an explanation of the color scales.



(a) x, y plane **(b)** y, θ plane **(c)** θ, x plane **(d)** x, y plane **(e)** y, θ plane **(f)** θ, x plane

Figure 11.7. Convergence results for robot with range-finder (180deg FOV). See the caption of Fig. 11.5 for an explanation of the color scales. For the range-finder, both control laws have large convergence radius.

CHAPTER 12

Learning Bilinear Flows

The class of bilinear gradient dynamics sensor *(*BGDS*) models systems with spatially coherent observations defined as a function on a manifold, and for which the dynamics $\dot{\boldsymbol{y}}$ is assumed to depend on $\boldsymbol{y}$ itself only through the spatial gradient $\nabla \boldsymbol{y}$.*

Table 12.1. Symbols used in this chapter

BGDS systems		
$\mathsf{Images}(\mathcal{S})$		Differentiable functions from $\mathcal{S}$ to $\mathbb{R}$.
$\nabla_d y(s)$	$(0, \dim(\mathcal{S}))$ tensor field on $\mathcal{S}$	Gradient of y.
$\mathsf{G}_i^d(s)$	$(\dim(\mathcal{S}), n_{\boldsymbol{u}})$ tensor field on $\mathcal{S}$	Model parameter.
$\mathsf{B}_i(s)$	$(0, n_{\boldsymbol{u}})$ tensor field on $\mathcal{S}$	Model parameter.
$\mathsf{BGDS}(\mathcal{S}; k)$	$\subset \mathcal{D}(\mathsf{Images}(\mathcal{S}); \mathbb{R}^k)$	Class of BGDS systems.
pop_ψ		Population code.
$\overline{\mu}^s$		Average nearness.
Learning (in physical space)		
$\mathsf{C}^i(s)$	$(n_{\boldsymbol{u}}, 0)$ tensor field on $\mathcal{S}$	Learned statistic.
$\mathsf{H}_d^i(s)$	$(n_{\boldsymbol{u}}, \dim(\mathcal{S}))$ tensor field on $\mathcal{S}$	Learned statistic.
$\mathsf{R}_{ef}(s)$	$(\dim(\mathcal{S}), \dim(\mathcal{S}))$ tensor field on $\mathcal{S}$	Covariance of $\nabla \boldsymbol{y}$.
Q^{ij}	$n_{\boldsymbol{u}} \times n_{\boldsymbol{u}}$	Covariance of $\boldsymbol{u}$.
Learning with alternative parametrization (logical space)		
$\mathcal{Z}$	manifold	Logical space.
$\boldsymbol{z}$	$\in \mathsf{Images}(\mathcal{Z})$	Transformed observations.
x	$\in \mathcal{Z}$	Logical coordinate.
φ	$\in \mathsf{Diff}(\mathcal{Z}; \mathcal{S})$	Coordinate change.
$\mathbf{J}$	$= \partial\varphi / \partial x$	Jacobian of φ.
$\mathsf{F}^i(x)$	(equivalent of $\mathsf{C}^i(s)$)	Tensors learned in alternative parametrization.
$\mathsf{M}_\delta^i(x)$	(equivalent of $\mathsf{H}_d^i(s)$)	
$\mathsf{S}_{mn}(x)$	(equivalent of $\mathsf{R}_{ef}(s)$)	

12.1. BGDS

DEFINITION 12.1. A system is a *bilinear gradient dynamics sensor* (BGDS), if the observations are a function on a manifold $\mathcal{S}$ ($\mathcal{Y} = \mathsf{Images}(\mathcal{S})$), the commands are real numbers ($\mathcal{U} = \mathbb{R}^{n_u}$) and the dynamics of the observations is bilinear with respect to the gradient of the observations and affine in the commands. Formally, there exist two tensor fields $\mathbf{G}$ and $\mathbf{B}$ on $\mathcal{S}$ such that

$$\dot{y}(s,t) = \sum_i (\mathsf{G}_i^d(s)\nabla_d y(s,t) + \mathsf{B}_i(s))u^i(t). \tag{12.1}$$

$\mathsf{BGDS}(\mathcal{S};k) \subset \mathcal{D}(\mathsf{Images}(\mathcal{S});\mathbb{R}^k)$ is the family of all such systems with k commands and sensel space $\mathcal{S}$.

In (12.1), the symbol s represents a spatial index (the position of the sensel on the space $\mathcal{S}$) and $\nabla_d y(s)$ denotes the d-th component of the gradient with respect to s.

The tensor field $\mathbf{G}$ represents the bilinear part of the dynamics, while $\mathbf{B}$ represents the purely affine part that does not depend on $\boldsymbol{y}$. Let $\dim(\mathcal{S})$ be the dimension of the space $\mathcal{S}$, and k the dimension of $\boldsymbol{u}$. Then the tensor field $\mathbf{G}$ can be represented by an $\dim(\mathcal{S}) \times n_{\boldsymbol{u}}$ matrix at each point of $\mathcal{S}$. If we discretize $\mathcal{S}$ to $n_{\boldsymbol{y}}$ sensels, then $\mathbf{G}$ has a total of $n_{\boldsymbol{y}} \times \dim(\mathcal{S}) \times n_{\boldsymbol{u}}$ elements. Instead, $\mathbf{B}$ is represented by only $n_{\boldsymbol{y}} \times n_{\boldsymbol{u}}$ elements.

12.1.1. *Symmetries of the* BGDS *class*

We now give the equivalent invariance properties of Proposition 11.2. For BDS models, we considered the effect of the linear group $\mathsf{GL}(n)$ on the observation; for BGDS, we consider the effect of diffeomorphisms of the manifold $\mathcal{S}$.

PROPOSITION 12.2. *The* $\mathsf{BGDS}(\mathcal{S};k)$ *family is closed with respect to diffeomorphisms* $\varphi \in \mathsf{Diff}(\mathcal{S})$ *that act on the observations as* $z(x,t) = y(\varphi(x),t)$*, and the action of* GL *on the commands:*

$$\mathsf{Diff}(\mathcal{S}) \cdot \mathsf{BGDS}(\mathcal{S};k) \cdot \mathsf{GL}(k) = \mathsf{BGDS}(\mathcal{S};k).$$

PROOF. For clarity, we prove the slightly more general result where the diffeomorphism is between two different spaces $\mathcal{S}$ and $\mathcal{Z}$:

$$\mathsf{Diff}(\mathcal{Z};\mathcal{S}) \cdot \mathsf{BGDS}(\mathcal{S};k) \cdot \mathsf{GL}(k) \subset \mathsf{BGDS}(\mathcal{Z};k).$$

Let $s \in \mathcal{S}$, $z \in \mathcal{Z}$, $s = \varphi(x)$, and $z(x,t) = y(\varphi(x),t)$. The gradients of the fields are related by

$$\nabla_d y(\varphi(x),t) = J_e^d(x)\nabla_e z(x,t), \tag{12.2}$$

where $\mathbf{J}$ is the jacobian of the diffeomorphism φ^{-1}. For the derivatives, we obtain

$$\begin{aligned}
\dot{z}(x,t) &= \dot{y}(\varphi(x),t) \\
&\qquad \text{(Substitution of dynamics)} \\
&= (\mathsf{G}_i^d(\varphi(x))\nabla_d y(\varphi(x),t) + \mathsf{B}_i(\varphi(x)))u^i(t) \\
&\qquad \text{(Substitution of gradient definition)} \\
&= (\mathsf{G}_i^d(\varphi(x))J_e^d(x)\nabla_e z(x,t) + \mathsf{B}_i(\varphi(x)))u^i(t) \\
&\triangleq (\tilde{\mathsf{G}}_i^d(x)\nabla_e z(x,t) + \tilde{\mathsf{B}}_i(x))u^i(t).
\end{aligned}$$

Therefore, after the diffeomorphism φ, the system dynamics is still bilinear, and their characteristic tensors are transformed by

$$\begin{aligned}
\tilde{\mathsf{G}}_i^d(x) &= J_e^d(x)\mathsf{G}_i^d(\varphi(x)), \\
\tilde{\mathsf{B}}_i(x) &= \mathsf{B}_i(\varphi(x)).
\end{aligned}$$

□

We can also look at the effect of diffeomorphisms of the *values* rather than the domain.

Suppose that the signal undergoes a local nonlinear transformation $f \in \mathsf{Diff}(\mathbb{R})$:

$$z(s) = f(y(s)).$$

What we find is that the BGDS class is closed with respect to all diffeomorphisms only if the tensor $\mathbf{B}$ is zero. If it is not zero, the function f must be an affine transformation.

PROPOSITION 12.3. *Assume $\boldsymbol{y}$ has* BGDS *dynamics. The dynamics of $z(s) = f(y(s))$ is* BGDS *only if either $\mathbf{B} = 0$ or $f \in \mathsf{Aff}(\mathbb{R})$.*

PROOF. Write $z(s) = f(x)$, with $x = y(s)$. The gradient of z is a function of the partial derivative of f:

$$\nabla_d z(s) = \frac{\partial f}{\partial x}|_{x=y(s)} \nabla_d y(s).$$

From (12.1) it follows:

$$\begin{aligned}
\dot{z}(s) &= \frac{\partial f}{\partial x}|_{x=y(s)} \dot{y}(s) \\
&= \frac{\partial f}{\partial x}|_{x=y(s)} ((\mathsf{G}_i^d(s) \nabla_d y(s,t) + \mathsf{B}_i(s)) u^i(t).) u^i \\
&= (\mathsf{G}_i^d(s) \frac{\partial f}{\partial x} \nabla_d y(s,t) + \frac{\partial f}{\partial x}|_{x=y(s)} \mathsf{B}_i(s)) u^i \\
&= (\mathsf{G}_i^d(s) \nabla_d z(s) + \frac{\partial f}{\partial x}|_{x=y(s)} \mathsf{B}_i(s)) u^i.
\end{aligned}$$

Therefore, the tensor $\mathbf{G}$ is invariant, while B is multiplied by the partial derivative:

$$\begin{aligned}
\tilde{\mathsf{G}}_i^d &= \mathsf{G}_i^d, \\
\tilde{\mathsf{B}}_i &= \frac{\partial f}{\partial x}|_{x=y(s)} \mathsf{B}_i.
\end{aligned}$$

For the dynamics to be BGDS, the tensor $\mathbf{B}$ should not depend on y, and the partial derivative of f does not depend on y only if f is an affine transformation. □

12.2. Learning

Learning a BGDS model from sensorimotor data can be done by a streaming algorithm that computes simple statistics of the data.

PROPOSITION 12.4 (Learning of BGDS models). *Consider a sensor with bilinear gradient dynamics. Assume that we can compute the gradient* $\nabla \boldsymbol{y}$. *Let* $\boldsymbol{u}$ *have second moment matrix* $\mathbf{Q}$. *Learn the three tensor fields* $\mathbf{C}^i$, $\mathbf{H}$, *and* $\mathbf{R}$ *according to*

$$\begin{aligned} \mathsf{C}^i(s) &\leftarrow \mathbb{E}\{\dot{y}(s)u^i\}, \\ \mathsf{H}^i_d(s) &\leftarrow \mathbb{E}\{\dot{y}(s)\nabla_d y(s)u^i\}, \\ \mathsf{R}_{ef}(s) &\leftarrow \mathbb{E}\{\nabla_e y(s)\nabla_f y(s)\}. \end{aligned}$$

If the training distribution is mixing (9.12), these statistics converge to

$$\begin{aligned} \mathsf{C}^i(s) &= \sum_j \mathsf{B}_j(s)\mathsf{Q}^{ij}, \\ \mathsf{H}^i_d(s) &= \sum_{D,j} \mathsf{G}^D_j(s)\mathsf{R}_{Dd}(s)\mathsf{Q}^{ij}. \end{aligned}$$

PROOF. The statistics $\mathbf{H}$ converges to

$$\begin{aligned}
\mathsf{H}_d^i(s) &= \mathbb{E}\{\dot{y}(s)\nabla_d y(s) u^i\} \\
&\quad \text{(Definition of gradient dynamics)} \\
&= \mathbb{E}\{[(\mathsf{G}_j^D(s)\nabla_D y(s) + \mathsf{B}_j(s))u^j]\, \nabla_d y(s)\, u^i\} \\
&\quad \text{(Independence of } \boldsymbol{y} \text{ and } \boldsymbol{u}\text{)} \\
&= \mathsf{G}_j^D(s)\left[\mathbb{E}\{\nabla_D y(s)\nabla_d y(s)\} + \mathsf{B}_j(s)\mathbb{E}\{\nabla_d y(s)\}\right]\mathbb{E}\{u^j u^i\} \\
&\quad \text{(The expectation of the gradient is 0 if mixing: } \mathbb{E}\{\nabla_D y(s)\} = 0.\text{)} \\
&= \mathsf{G}_j^D(s)\mathbb{E}\{\nabla_D y(s)\nabla_d y(s)\}\mathbb{E}\{u^j u^i\} \\
&= \mathsf{G}_j^D(s)\mathsf{R}_{Dd}(s)\mathsf{Q}^{ij}.
\end{aligned}$$

Note that the value of $\mathbf{H}$ is a combination of three factors: the dynamics ($\mathbf{G}$), the world properties ($\mathbf{R}$), and the commands used ($\mathbf{Q}$). The statistics $\mathbf{C}$ converges to:

$$\begin{aligned}
\mathsf{C}^i(s) &= \mathbb{E}\{\dot{y}(s)u^i\} \\
&\quad \text{(Definition of gradient dynamics.)} \\
&= \mathbb{E}\left\{\left[\left(\mathsf{G}_j^D(s)\nabla_D y(s) + \mathsf{B}_j(s)\right)u^j\right]u^i\right\} \\
&= (\mathsf{G}_j^D(s)\mathbb{E}\{\nabla_D y(s)\} + \mathsf{B}_j(s))\mathbb{E}\{u^j u^i\} \\
&= \mathsf{B}_j(s)\mathsf{Q}^{ij}.
\end{aligned}$$

□

12.2.1. *Prediction/anomaly detection*

PROPOSITION 12.5. *(Prediction/anomaly detection) We can define a signal $\hat{\dot{y}}(s)$ which predicts $\dot{y}(s)$ as:*

$$\hat{\dot{y}}(s) = \left[(\mathsf{H}^i_d(s)\mathsf{R}^{dD}(s)\nabla_D y(s) + \mathsf{C}^i(s)) \right] \mathsf{Q}_{ij}\boldsymbol{u}^j, \tag{12.1}$$

and an anomaly detection signal can be defined as

$$d(s) = \max\{-\hat{\dot{y}}(s)\dot{y}(s), 0\}. \tag{12.2}$$

(12.1) is simply the prediction given the learned model: equation (12.1) written using the learned tensors. The detector (12.2) returns a positive response when the predicted and observed derivative disagree on their sign; if the actual sensors is precisely a BGDS, this signal detects extraneous objects on the field of view.

12.3. Servoing

In analogy with Proposition 12.6, the following is a servoing strategy for a BGDS model.

PROPOSITION 12.6 (Servoing with BGDS). *The control command*

$$u^i = -\int_{\mathcal{S}} \left[\mathsf{H}^i_d(s)(\mathsf{R}(s)^{-1})^{dD}\nabla_D y(s) + \mathsf{C}^i(s) \right] (y(s) - \check{y}(s))\, ds \tag{12.1}$$

is a descent direction for $\|\boldsymbol{y} - \check{\boldsymbol{y}}\|$.

PROOF. The control command is

$$\begin{aligned} u^i &= -\int \left[\mathsf{H}^i_d(x)(\mathsf{R}(s)^{-1})^{de}\nabla_e y(s) + \mathsf{C}^i(s)\right](y(s)-\check{y}(s))\,ds \\ &= -\mathsf{Q}^{ij}\int [\mathsf{G}^D_j(s)\mathsf{R}_{Dd}(\mathsf{R}(s)^{-1})^{de} + \mathsf{C}^i(s)]\nabla_e y(s)(y(s)-\check{y}(s))\,ds \\ &\qquad \text{(The tensors } \mathbf{R} \text{ and } \mathbf{R}^{-1} \text{ cancel.)} \\ &= -\mathsf{Q}^{ij}\int \left[\mathsf{G}^d_j(s)\nabla_d y(s) + \mathsf{C}^i(s)\right](y(s)-\check{y}(s))\,ds \end{aligned} \tag{12.2}$$

Consider the error function $V = \frac{1}{2}\int (y(s)-\check{y}(s))^2 ds$. Its derivative is

$$\begin{aligned} \dot{V} &= \int (y(s)-\check{y}(s))\dot{y}(s)ds \\ &= \int (y(s)-\check{y}(s))\left[\left(\mathsf{G}^D_j(s)\nabla_D y(s) + \mathsf{B}_j\right)u^j\right]ds \\ &\qquad \text{(Letting } v_j = \textstyle\int (y(s)-\check{y}(s))(\mathsf{G}^D_j(s)\nabla_D y(s) + \mathsf{B}_j)ds\text{).} \\ &= -v_j\mathsf{Q}^{ij}v_i \le 0. \end{aligned}$$

□

12.4. Invariance to Reparametrization of the Sensel Space

In the previous section, we used the assumption that we can compute the gradient $\nabla y(s)$. That is equivalent to knowing the metric structure of the sensel space $\mathcal{S}$ complete sensor calibration, that is, knowing the positions of all sensels in the sensel space. For example, in the case of a camera, one should know the direction of each pixel on the visual sphere. By contrast, the BDS models did not have any assumption about the position of sensels in sensel space. We can make these two notions precise:

(1) A sensor is *fully calibrated* if we know the position of each sensel in the sensel space $\mathcal{S}$ (Figure 12.1a). This is the assumption needed for BGDS models (so far).

(2) A sensor is *uncalibrated* if we do not know anything about the position of sensels in $\mathcal{S}$ (Figure 12.1c). This is the assumption of BDS models.

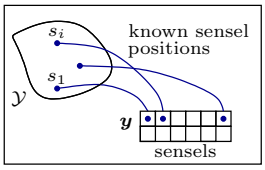
These two levels of prior information seem extreme. One intermediate level of knowledge is knowing the sensels topology, but not necessarily the precise metric arrangement on $\mathcal{S}$. Think of the output of an uncalibrated omnidirectional camera: while we know that if two pixels that are close in image space, then they are close in the sensel space $\mathcal{S} = \mathbb{S}^2$, we do not know their absolute position.

We can formalize this idea as follows. We introduce the concept of the sensor "logical space", as opposed to the "physical space". Before, we assumed that the observations were defined as a function $y : \mathcal{S} \to \mathbb{R}$. Now we assume that the sensor observation are returned as function $z : \mathcal{Z} \to \mathbb{R}$, defined on the "logical" space $\mathcal{Z}$ instead of the "physical" space $\mathcal{S}$.

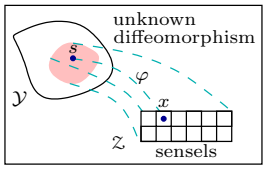
DEFINITION 12.7. A sensor is *logically calibrated* if its output is naturally defined on a certain *physical* sensel space $\mathcal{S}$ as $y : \mathcal{S} \to \mathbb{R}$, but the agent has access to another *logical* sensel space $\mathcal{Z}$ and the observations as a function $z : \mathcal{Z} \to \mathbb{R}$, and it holds that

$$z(x) = y(\varphi(x)), \tag{12.1}$$

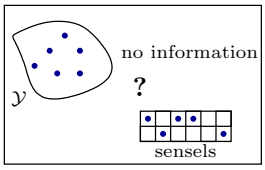
for some *unknown* differentiable and bijective map φ between $\mathcal{Z}$ and $\mathcal{S}$.



(a) Fully calibrated sensor



(b) Logically calibrated sensor



(c) Uncalibrated sensor

Figure 12.1. Three levels of prior information about the sensors: uncalibrated, logically calibrated, fully calibrated. The bootstrapping strategy in the first section works with uncalibrated sensor. In this section we show that to take advantage of the special structure possessed by a BGDS we only need to have a logically calibrated sensor.

In the case of a camera sensor, the physical space is the unit sphere $\mathbb{S}^2$, and the logical space is the pixel space $[1, W] \times [1, H]$, where $W \times H$ is the camera resolution. The logical variable is $x = (x_1, x_2)$ corresponding to the pixel coordinates. The map φ associates to a pixel its corresponding direction on the visual sphere. Knowledge of φ corresponds to knowing the camera intrinsic calibration. The function φ itself can be written explicitly only in particularly simple cases.

EXAMPLE 12.8. For a pinhole camera, given the focal length F and the sensor centers c_1, c_2, the map is

$$\begin{aligned} \varphi : [1, W] \times [1, H] &\rightarrow \mathbb{S}^2 \\ \varphi(x_1, x_2) &= \frac{(1, \frac{1}{F}(x_1 - c_1), \frac{1}{F}(x_2 - c_2))}{\|(1, \frac{1}{F}(x_1 - c_1), \frac{1}{F}(x_2 - c_2))\|}. \end{aligned}$$

If the agent learns a BGDS model in the logical space, then the corresponding servoing control minimizes the same error function, up to a scale factor of the metric being minimized that depends on the Jacobian determinant of φ.

PROPOSITION 12.9. *Consider a logically calibrated* BGDS. *Learn three tensor fields* $\mathbf{F}$, $\mathbf{M}$, $\mathbf{S}$ *according to*

$$\begin{aligned} \mathsf{F}^i(x) &\leftarrow \mathbb{E}\{\dot{z}(x) u^i\}, \\ \mathsf{M}^i_\delta(x) &\leftarrow \mathbb{E}\{\dot{z}(x) \nabla_\delta z(x) u^i\}, \\ \mathsf{S}_{mn}(x) &\leftarrow \mathbb{E}\{\nabla_m z(x) \nabla_n z(x)\}. \end{aligned}$$

Then the control strategy

$$u^i = -\int \left[\mathsf{M}^i_\delta(x)(\mathsf{S}^{-1})^{\delta D} \nabla_D z(x) + \mathsf{F}^i(x)\right] (z(x) - z_\star(x))\, dx \tag{12.2}$$

is minimizing the error function

$$\frac{1}{2}\int |\det \mathbf{J}|^{-1}(y(s)-\check{y}(s))^2\,\mathrm{d}s, \tag{12.3}$$

where $\mathbf{J} = \partial\varphi/\partial x$.

PROOF. The proof consists in expressing everything in the physical space. First we compute the results of bootstrapping. The learned tensor $\mathbf{F}$ is

$$\begin{aligned}
\mathsf{F}^i(x) &= \mathbb{E}\{\dot{z}(x)u^i\} \\
&= \mathbb{E}\{\dot{y}(\varphi(x))u^i\} \\
&= \mathbb{E}\{\left[\mathsf{G}_j^D(s)\nabla_D y(s) + \mathsf{B}_j(\varphi(x))\right]u^j u^i\} \\
&= \mathsf{B}_j(\varphi(x))\mathsf{Q}^{ij}.
\end{aligned}$$

The learned tensor $\mathbf{M}$ is

$$\begin{aligned}
\mathsf{M}_\delta^i(x) &= \mathbb{E}\{\dot{z}(x)\nabla_\delta z(x)u^i\} \\
&= \mathbb{E}\{\left[\mathsf{G}_j^D(\varphi(x))\nabla_D y(\varphi(x))u^j + \mathsf{B}_j(\varphi(x))u^j\right]\left[\nabla_d y(\varphi(x))\mathsf{J}_\delta^d\right]u^i\} \\
&= \mathsf{G}_j^D(s)\mathsf{J}_\delta^d\mathbb{E}\{\nabla_D y^s\nabla_d y^s\}\mathbb{E}\{u^j u^i\} \\
&= \mathsf{G}_j^D(s)\mathsf{J}_\delta^d\mathsf{R}_{Dd}\mathsf{Q}^{ij}.
\end{aligned} \tag{12.4}$$

For the covariance, we obtain

$$\begin{aligned} \mathsf{S}_{mn}(x) &= \mathbb{E}\{\nabla_m z(x) \nabla_n z(x)\} \\ &= \mathbb{E}\{\left(\mathsf{J}_m^d \nabla_d y(s)\right) (\mathsf{J}_n^e \nabla_e y(s))\} \\ &= \mathsf{J}_m^d \mathbb{E}\{\nabla_d y(s) \nabla_e y(s)\} \mathsf{J}_n^e \\ &= \mathsf{J}_m^d \mathsf{R}_{ed} \mathsf{J}_n^e. \end{aligned}$$

We can compute the inverse of the S_{mn} tensor by following the linear algebra rules:

$$(\mathsf{S}^{-1})^{mn} = (\mathsf{J}^{-1})_d^m (\mathsf{R}^{-1})^{ed} (\mathsf{J}^{-1})_e^n. \tag{12.5}$$

Now we can compute an equivalent expression for the control command (12.2):

$$u^i = -\int \left[\mathsf{M}_\delta^i(x)(\mathsf{S}^{-1})^{\delta D} \nabla_D z(x) + \mathsf{F}^i(x)\right] (z(x) - z_\star(x)) \, \mathrm{d}x.$$

For the first term, one finds that

$$\begin{aligned} \mathsf{M}_\delta^i(x)(\mathsf{S}^{-1})^{\delta D} \nabla_D z(x) &= \left[\mathsf{G}_j^g(f(x)) \mathsf{J}_\delta^h \mathsf{R}_{gh} \mathsf{Q}^{ij}\right] \times \\ &\quad \left[(\mathsf{J}^{-1})_d^\delta (\mathsf{R}^{-1})^{ed} (\mathsf{J}^{-1})_e^D\right] \left[\nabla_f y(\varphi(x)) \mathsf{J}_D^f\right]. \end{aligned}$$

The next step involves a couple of simplifications of $\mathbf{J}$ and $\mathbf{J}^{-1}$. We notice that $\mathsf{J}_\delta^h \left(\mathsf{J}^{-1}\right)_d^\delta = \mathrm{Id}_d^h$, and likewise $\mathsf{J}_D^f \left(\mathsf{J}^{-1}\right)_e^D = \mathrm{Id}_e^f$, where Id is the identity. Therefore we obtain

$$\begin{aligned} \mathsf{M}_\delta^i(x)(\mathsf{S}^{-1})^{\delta D} \nabla_D z(x) &= \mathsf{Q}^{ij} \int \mathsf{G}_j^g(\varphi(x)) \mathsf{R}_{gh} \mathrm{Id}_d^h (\mathsf{R}^{-1})^{ed} \mathrm{Id}_e^f \nabla_f y(\varphi(x)) \\ &= \mathsf{G}_j^f(\varphi(x)) \nabla_f y(\varphi(x)). \end{aligned}$$

The final form for the control law is

$$u^i = -\mathsf{Q}^{ij} \int \left[\mathsf{G}_j^f(\varphi(x)) \nabla_f y(\varphi(x)) + \mathsf{B}_j(\varphi(x)) \right] (z(x) - z_\star(x)) \, \mathrm{d}x.$$

At this point, notice that the control law is very similar to (12.2), but the integration is over the space x instead of s. We can do a change of variable; recalling that $s = \varphi(x)$, the volume form changes as $\mathrm{d}s = |\det \mathbf{J}| \, \mathrm{d}x$. The final expression for the commands is

$$u^i = -\mathsf{Q}^{ij} \int \frac{1}{|\det \mathbf{J}|} \left[\mathsf{G}_j^f(s) \nabla_f y(s) + \mathsf{B}_j(s) \right] (y(s) - \check{y}(s)) \, \mathrm{d}s.$$

This corresponds to minimizing the error function (12.3). □

This result implies that we do not have to know the full calibration of the sensor to take advantage of the fact that the sensor is a BGDS; the knowledge of the sensels positions up to an unknown diffeomorphism is sufficient.

12.4.1. *Invariance of the agent behavior*

However, the behavior is not completely invariant to diffeomorphisms, because the error function being minimized is slightly different. In complete analogy with Section 11.6, there are two possibilities: either amend the error function to be invariant, or introduce a canonization operator.

In Section 11.6 we used the covariance of the observations as the covariant statistics. Here, the analogous quantity is the *determinant* of the covariance of the *gradient* of the observations. Define the two scalar quantities in physical and logical space:

$$\begin{aligned} d_p(s) &= \det \mathbb{E}\{\nabla_d y^s \nabla_e y^s\}, \\ d_l(x) &= \det \mathbb{E}\{\nabla_f z^x \nabla_g z^x\}. \end{aligned} \tag{12.6}$$

These are observable quantities that allow to define an invariant error function.

LEMMA 12.10. *The error function is invariant to diffeomorphisms:*

$$\frac{1}{2}\int \frac{1}{\sqrt{d_p(s)}}(y(s)-\check{y}(s))^2\,\mathrm{d}s = \frac{1}{2}\int \frac{1}{\sqrt{d_l(x)}}(z(x)-\check{z}(x))^2\,\mathrm{d}x. \tag{12.7}$$

PROOF. From (12.6), it follows that

$$d_p(s) = |\det \mathbf{J}|^2 d_l(x).$$

Using a change of variable: and keeping in mind that $\mathrm{d}s = |\det \mathbf{J}|\,\mathrm{d}x$, one obtains

$$\begin{aligned}
& \frac{1}{2}\int \frac{1}{\sqrt{d_p(s)}}(y(s)-\check{y}(s))^2\,\mathrm{d}s \\
= \; & \frac{1}{2}\int \frac{1}{\sqrt{d_p(s)}}(y(s)-\check{y}(s))^2|\det \mathbf{J}|\,\mathrm{d}x \\
= \; & \frac{1}{2}\int \frac{1}{\sqrt{d_p(s)}}(z(x)-\check{z}(x))^2|\det \mathbf{J}|\,\mathrm{d}x \\
= \; & \frac{1}{2}\int \frac{1}{|\det \mathbf{J}|^2}\frac{1}{\sqrt{d_l(x)}}(z(x)-\check{z}(x))^2|\det \mathbf{J}|\,\mathrm{d}x \\
= \; & \frac{1}{2}\int \frac{1}{\sqrt{d_l(x)}}(z(x)-\check{z}(x))^2\,\mathrm{d}x.
\end{aligned}$$

□

The other approach consists in introducing a canonization operator. This is seen in detail in Subsection 18.5.

Table 12.2. BGDS approximation of canonical robotic sensors.

	dynamics	*learned tensors*
field sampler	$\dot{y}^s = (s \times \nabla y^s)_i \boldsymbol{\omega}^i + \nabla_i y^s \boldsymbol{v}^i$	
		$\mathsf{H}^{vi}(s) = \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}, \quad i \in [1,3].$
		$\mathsf{H}^{vi}(s) = \hat{s}^f_j \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}, \quad i \in [4,6],$
		$\mathsf{C}^i(s) = 0.$
camera	$\dot{y}^s = (s \times \nabla y^s)_i \boldsymbol{\omega}^i + \mu^s \nabla_i y^s \boldsymbol{v}^i$	
		$\mathsf{H}^{vi}(s) = \overline{\mu}^s \mathsf{R}_{jd}(s), \quad i \in [1,3],$
		$\mathsf{H}^{vi}(s) = \hat{s}^f_j \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}, \quad i \in [4,6],$
		$\mathsf{C}^i(s) = 0.$
range-finder	$\dot{y}^s = (s \times \nabla y^s)_i \boldsymbol{\omega}^i + (\nabla_i \log y^s - s^*_i)\boldsymbol{v}^i$	
		$\mathsf{H}^{vi}(s) = \overline{\mu}^s \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}, \quad i \in [1,3].$
		$\mathsf{H}^{vi}(s) = \hat{s}^f_j \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}, \quad i \in [4,6],$
		$\mathsf{C}^i(s) = -s^*_j \mathsf{Q}^{ij}, \quad i \in [1,3].$
		$\mathsf{C}^i(s) = 0, \quad i \in [4,6].$

12.5. Approximation to Canonical Robotic Sensors

In the previous section, we considered a generic BGDS sensor. Here, we consider the dynamics of our three model sensors, and discuss the quality of their BGDS approximation. The main results are given in Table 12.2.

12.5.1. *Field-samplers*

PROPOSITION 12.11 (BGDS approximation to field-sampler). The learned tensors for a field sampler are

$$\begin{aligned} \mathsf{H}^{vi}(s) &= \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}, \quad i \in [1,3], \\ \mathsf{H}^{vi}(s) &= \hat{s}^f_j \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}, \quad i \in [4,6], \\ \mathsf{C}^i(s) &= 0. \end{aligned}$$

PROOF. From (9.1), the dynamics of a field sampler are $\dot{y}(s) = (\nabla_j y(s))\boldsymbol{v}^j + (s \times$

$\nabla y(s))_j \boldsymbol{\omega}^j$. The computation for $\mathbf{C}$ goes as follows:

$$\begin{aligned}
\mathsf{C}^i(s) &= \mathbb{E}\{\dot{y}(s)u^i\} \\
&= \mathbb{E}\{[(\nabla_j y(s))\boldsymbol{v}^j + (s \times \nabla y(s))_j \boldsymbol{\omega}^j]u^i\} \\
&\qquad (\ \nabla \text{y is independent from the commands, and its expectation is } 0.) \\
&= 0.
\end{aligned}$$

We split the computation for $\mathbf{H}$ for the first and last three components. For the first three components ($i = [1,3]$), corresponding to the linear velocity:

$$\begin{aligned}
\mathsf{H}^i_d(s) &= \mathbb{E}\{\dot{y}(s)\nabla_d y(s)\boldsymbol{v}^i\} \\
&= \mathbb{E}\{[(s \times \nabla y^s)_i \boldsymbol{\omega}^i + \nabla_i y^s \boldsymbol{v}^i]\nabla_d y(s)\boldsymbol{v}^i\} \\
&\qquad (\text{Uncorrelated } \boldsymbol{v} \text{ and } \boldsymbol{\omega}.\ ) \\
&= \mathbb{E}\{[(\nabla_j y(s))\boldsymbol{v}^j]\nabla_d y(s)\boldsymbol{v}^i\} \\
&= \mathbb{E}\{\nabla_j y(s)\nabla_d y(s)\}\mathbb{E}\{\boldsymbol{v}^i\boldsymbol{v}^j\} \\
&= \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}
\end{aligned}$$

For the last three components ($i \in [4,6]$), corresponding to the angular velocity:

$$\begin{aligned}
\mathsf{H}^i_d(s) &= \mathbb{E}\{\dot{y}(s)\nabla_d y(s)\boldsymbol{\omega}^i\} \\
&= \mathbb{E}\{(s \times \nabla y(s))_j \boldsymbol{\omega}^j \nabla_d y(s)\boldsymbol{\omega}^i\} \\
&= \hat{s}^f_j \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}.
\end{aligned}$$

□

12.5.2. *Vision sensors*

PROPOSITION 12.12 (BGDS approximation to vision sensors). The learned tensors for a vision sensor are

$$\begin{aligned} \mathsf{H}^{vi}(s) &= \overline{\mu}^s \mathsf{R}_{jd}(s), \quad i \in [1,3], \\ \mathsf{H}^{vi}(s) &= \hat{s}^f_j \mathsf{R}_{fd}(s) \mathsf{Q}^{ij}, \quad i \in [4,6], \\ \mathsf{C}^i(s) &= 0. \end{aligned}$$

PROOF. For rotation, the model is the same as the model for field-samplers. As for the linear velocity, the computation is mostly the same, with the addition of the nearness μ^s:

$$\begin{aligned} \mathsf{H}^i_d(s) &= \mathbb{E}\{\dot{y}^s \nabla_d y^s \boldsymbol{v}^i\} \\ &= \mathbb{E}\{[(s \times \nabla y^s)_i \boldsymbol{\omega}^i + \mu^s \nabla_i y^s \boldsymbol{v}^i] \nabla_d y^s \boldsymbol{v}^i\} \\ &= \mathbb{E}\{[\mu^s (\nabla_j y^s) \boldsymbol{v}^j] \nabla_d y^s \boldsymbol{v}^i\} \\ &= \overline{\mu}^s \mathsf{R}_{jd}(s) \mathsf{Q}^{ij}. \end{aligned}$$

□

12.5.3. *Range finders*

PROPOSITION 12.13 (BGDS approximation to range finders). The learned tensors for a range finder are

$$\begin{aligned}
\mathsf{H}^{vi}(s) &= \overline{\mu}^s \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}, \quad i \in [1,3], \\
\mathsf{H}^{vi}(s) &= \hat{s}_j^f \mathsf{R}_{fd}(s)\mathsf{Q}^{ij}, \quad i \in [4,6], \\
\mathsf{C}^i(s) &= -s_j^* \mathsf{Q}^{ij}, \quad i \in [1,3]. \\
\mathsf{C}^i(s) &= 0, \quad i \in [4,6].
\end{aligned}$$

PROOF. For rotation, the model is the same as the model for field-samplers. For the other components:

$$\begin{aligned}
\mathsf{H}_d^i(s) &= \mathbb{E}\{\dot{y}^s \nabla_d y^s \boldsymbol{v}^i\} \\
&= \mathbb{E}\{[(\nabla_j \log y^s - s_j^*)\boldsymbol{v}^j]\nabla_d y^s \boldsymbol{v}^i\} \\
&\qquad (\mathbb{E}\{\nabla y\} = 0) \\
&= \mathbb{E}\{[(\nabla_j \log y^s)\boldsymbol{v}^j]\nabla_d y(s) \boldsymbol{v}^i\} \\
&\qquad (\text{derivative of logarithm}) \\
&= \mathbb{E}\left\{\frac{1}{y^s}\, \nabla_j y^s \nabla_d y^s\ \boldsymbol{v}^i \boldsymbol{v}^j\right\} \\
&\qquad (\text{independence of quantity and its gradient}) \\
&= \mathbb{E}\left\{\frac{1}{y^s}\right\} \mathsf{R}_{jd}(s)\mathsf{Q}^{ij} \\
&\qquad (\text{The inverse of the readings is the nearness.}) \\
&= \overline{\mu}^s \mathsf{R}_{jd}(s)\mathsf{Q}^{ij}.
\end{aligned}$$

For the tensor **C**:

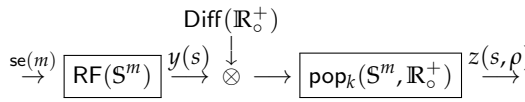
$$\begin{aligned} C^i(s) &= \mathbb{E}\{\dot{y}(s)\boldsymbol{v}^i\} \\ &= \mathbb{E}\{[(\nabla_j \log y^s - s_j^*)\boldsymbol{v}^j]\boldsymbol{v}^i\} \\ &= -s_j^* \mathbb{E}\{\boldsymbol{v}^j \boldsymbol{v}^i\} \\ &= -s_j^* \mathbf{Q}^{ij}. \end{aligned}$$

□

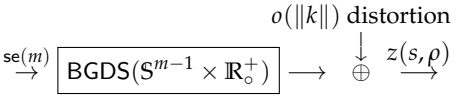
12.5.4. *Range finders (population code representation)*

Range-finders are not exactly BDS/BGDS because of the nonlinearity in the dynamics (Table 9.2). Here we show that a range finder model can be represented with small distortion as a BGDS, if the data is first preprocessed by a transformation similar to a population code.

DEFINITION 12.14 (Population code). Consider a signal y defined on a domain $\mathcal{S}$ ($y : \mathcal{S} \to \mathcal{O}$), and a symmetric kernel $\psi : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$. Then we call "population code" the signal $z : \mathcal{S} \times \mathcal{O} \to \mathbb{R}$ defined by $z(s, x) = \psi(x, y(s))$. The population code function pop_ψ



(a) Series of range-finder, unknown distortion, and population code .



(b) Equivalent model: BGDS with unstructured (i.e., non-group) nuisance.

Figure 12.1. A range-finder + population code can be approximated by a BGDS with low-distortion (Proposition 12.15).

has the signature

$$\mathsf{pop}_\psi(\mathcal{S};\mathcal{O}) : \mathsf{Images}(\mathcal{O}) \to \mathsf{Images}(\mathcal{S}\times\mathcal{O}).$$

Note that it augments the dimension of the observations from a field defined on $\mathcal{S}$ to a field defined on $\mathcal{S}\times\mathcal{O}$.

After this transformation, the dynamics appear simplified. Let $\mathsf{RF}(\mathbb{S}^m,k)$ be the family of all range-finders models for robots in $\mathsf{SE}(m)$ with k linear/angular velocity commands in $\mathsf{se}(m)$. The output of a rangefinder is a function defined on the sphere $\mathbb{S}^m$ to $\mathbb{R}^+_\circ$, therefore the family $\mathsf{RF}(\mathbb{S}^m;k)$ is a subset of $\mathcal{D}(\mathsf{Images}(\mathbb{S}^m);k)$.

PROPOSITION 12.15. *The dynamics of a range-finder whose output is filtered by a population code can be represented with small distortion by a* BGDS*, except near occlusions, with a small distortion bounded by $\|\psi\|$:*

$$\mathsf{pop}_\psi(\mathbb{S}^m;\mathbb{R}^+_\circ)\cdot\mathsf{RF}(\mathbb{S}^m,k)\subset\mathsf{BGDS}(\mathbb{S}^m\times\mathbb{R}^+_\circ,k)\oplus o(\|\psi\|).$$

For simplicity, we can picture the situation in the plane ($m=2$), but everything is valid in 3D as well. Let $O\subset\mathbb{R}^2$ be the subset of obstacles in the plane that are opaque to the range-finder. Define an indicator function $\delta_O:\mathbb{R}^2\to\{0,1\}$ such that $\delta_O(x)$ is 1 if the world is opaque at $x\in\mathbb{R}^2$. We define a field sampler sampling the field δ_O as follows:

$$\begin{aligned} f:\mathbb{R}^2 &\to \mathbb{R} \\ \boldsymbol{q} &\mapsto \delta_O(\mathbf{R}\boldsymbol{q}+\boldsymbol{t}), \end{aligned} \tag{12.1}$$

where $\boldsymbol{q}\in\mathbb{R}^2$ is the sensel position in robot frame, and $(\mathbf{R},\boldsymbol{t})\in\mathsf{SE}(2)$ is the robot pose. As previously discussed, every field sampler is a BGDS.

PROOF. Let $y:\mathbb{S}^1\to\mathbb{R}^+_\circ$ be the output of a range-finder. Define its population code

representation with a delta kernel ($k(x,y) = \delta_x(y)$) as $z(s,\rho) = \psi(\rho, y(s)) = \delta_{y(s)}(\rho)$. Note that in the population code for a range finder $s \in \mathbb{S}^1$ ranges over direction and $x \in \mathbb{R}^+_\circ$ ranges over distances; therefore, $z(s,\rho)$ is 1 if the obstacle in direction s is at distance x, so it represents a local polar map of the environment. Define the polar to Cartesian change of coordinates

$$\varphi : (s,\rho) \mapsto (\rho\ \cos(s),\ \rho\ \sin(s)),$$

which is a diffeomorphism except at the origin. The function z can be expressed as a function of the field sampler 12.1 as $z(s,\rho) = f(\varphi(s,\rho))$. This means that its output is a diffeomorphism of the output of a BGDS model.

Note, however, that so far we used a delta kernel $\psi = \delta$ and an indicator function δ_O for the obstacle set; this makes the output of the sensor sparse and discontinuous, which makes it impossible to represent the dynamics with partial differential equations. The solution is to use a kernel ψ with some smoothing (a trick used for different reasons in SLAM, leading to "relaxed" likelihood models), to which it (approximately) corresponds a smoothing of the field δ_O. The math only works exactly in the limit as $\psi \to \delta$, therefore there is a bounded small distortion. □

As a consequence, we know that the BGDS agent can learn, with small approximation, the dynamics of a range finder. We can take this further: suppose that we have a sensor whose output is an unknown function of the range; that is, instead of the ranges y_i, we have measurements $y'_i = f(y_i)$ for an unknown $f \in \mathrm{Diff}(\mathbb{R}^+_\circ)$. We can show that this transformation would be tolerated by the agent, because the resulting system is still a BGDS. In summary, it is possible to get the data from a range finder, distort it by a nonlinear function, shuffle the measurements, and then apply one of the calibration techniques that recover the topology, and the BGDS agent behavior will be invariant to all those group nuisances.

PROPOSITION 12.16. *Suppose that a nonlinear scaling $f \in \mathsf{Diff}_+(\mathbb{R}^+_\circ)$ acts on the output of a range-finder reading by reading, mapping $y(s) \mapsto f(y(s))$. Then the chain of sensor, scaling, and population code is still a* BGDS.

PROOF. We just need to note that, in general, nonlinear scaling commutes with the population code. Given: a dynamical system $\boldsymbol{D} \in \mathcal{D}(\mathsf{Images}(\mathcal{S}), \mathcal{U})$; a nonlinear scaling $f \in \mathsf{Diff}_+(\mathcal{O})$; a population code block $\mathsf{pop}_\psi(\mathcal{S}, \mathcal{O})$; we have that $\mathsf{pop}_\psi \cdot f \cdot \boldsymbol{D} = f \cdot \mathsf{pop}_\psi \cdot \boldsymbol{D}$. By the previous results $\mathsf{pop}_\psi \cdot \boldsymbol{D}$ is a BGDS. By Proposition 12.2, BGDS are closed to diffeomorphisms. □

12.6. Experiments

This section shows that BGDS bootstrapping agents can deal with the sensor suite of real-world robotic platforms with heterogeneous sensors. After minimal preprocessing, heterogeneous data from range-finders and cameras can be handled by exactly the same algorithm, with no other prior information. In the previous chapter, we evaluated BDS models on servoing. Here, we consider instead a passive task that can be tested on logged data. The task is anomaly detection: the agent must discover which changes in its stimuli can be explained by its own motion, and which are due to independent causes (e.g. objects moving in the field of view).

Datasets. We use data available from the Rawseeds project [132]. The robot is a differential drive platform with a full sensor suite onboard: 2 Sick range-finders (180 deg field of view, 181 readings, 70 Hz, max range of 80 m), 2 Hokuyo range-finders (270 field of view, 626 readings each, 10 Hz, max range of 6 m), one omnidirectional camera (RGB 640x640, 15fps), one frontal camera with fisheye lens (RGB 640×480, 30 fps), and a Point Gray Triclops (grayscale, $3 \times 320 \times 240$, 15 fps), GPS and IMU. The available logs correspond to a few hours of operation both indoors and outdoors, for a total of about 500 GB

of uncompressed data.

The commands $\boldsymbol{u} \in \mathbb{R}^2$ are the linear and angular velocities obtained from differentiating the odometry data. We set $u^0 = v$ (linear) and $u^1 = \omega$ (angular)—this is mainly for having clearer pictures, because we proved that the final result is invariant to any linear transformation.

Tests with range-finder data. For range-finder data, we demonstrate the bootstrapping pipeline as explained in Section 12.5.4, formalized in Fig. 12.1, and cartoonishly represented in Fig. 12.12a.

We take the data from the two Sick range finders, mounted with a heading of approximately 0 and 180 with respect to the robot. The raw readings are deformed by a nonlinear function $x \mapsto 1/x$ (nearness instead of range) and shuffled according to a random permutation. Thus the initial input to the pipeline is a set of 362 shuffled, warped sensels values $\{y_i\}_{i=1}^n$. An embedding algorithm [105] recovers the position of the sensels on the unit circle using a metric obtained by computing the information distances of the sensels. The reconstruction is accurate up to a diffeomorphism nuisance: that is, if $\theta_i \in \mathbb{S}^1$ is the real angle, we can estimate $\tilde{\theta}_i = \varphi(\theta_i)$ for $\varphi \in \mathsf{Diff}(\mathbb{S}^1)$. In practice, this means that we can reconstruct only the order of the sensels on the circle. We normalize each sensel value in the range $[0, 100]$ by computing the percentile of $y_i(t)$ to the whole history $y_i[-\infty, t]$; this normalizes the nonlinear scaling. Then we apply a population code filter with a small kernel ($\psi = 0.5$ against the $[0, 100]$ range). In the end, from shuffled distorted values $\{y_i\}_{i=1}^n$, we have obtained a 2D field $y(\tilde{\theta}, \tilde{\rho})$, where $\tilde{\theta} \in [0, 2\pi]$ is a (warped) angle and $\tilde{\rho} \in [0, 100]$ is a nonlinear function of the range. This "image" is diffeomorphic to a polar map of the environment; but notice that this is obtained not from prior geometric knowledge but only as the consequence of a data-agnostic pipeline.

In this case, the sensel space $\mathcal{S}$ ($\mathcal{S} = [0, 2\pi] \times [0, 100]$) has dimension $d = 2$ and there

are $k = 2$ commands (linear and angular velocity). Therefore the tensor $\mathbf{H}$ has 4 components, each of which is a field across the sensel space. The tensor $\mathbf{C}$ has 2 components (for linear and angular velocity), but we do not show it for the camera data, because it is in theory 0, and in practice insignificant noise. The 4 components of the tensor field $\mathbf{H}$ are shown in Fig. 12.13 as 4 false color images (white: zero, red: positive, blue: negative). Refer to the caption for some interpretation, not always intuitive. An example result of anomaly detection is shown in Fig. 12.15e. This is done by using the learned tensors to compute the prediction signal (12.1), and then computing the detection signal (12.2). The figure shows the total anomaly reported per sensel over time. Anomalies are reported constantly for the sensor horizons (front and back) where objects pop into view; the occasional traces correspond to people walking around the robot.

Tests with camera data. We stitched together the images from the omnidirectional camera, the frontal camera, and one of the stereo triplets to obtain a unique 640x480 frame from which we compute a grayscale signal (Fig 12.12b). In this case, we start from the knowledge of the correct sensel topology given from the raw images. However, there is still a diffeomorphism nuisance, because we assume no prior information on the intrinsic calibration of the cameras (the unknown diffeomorphism is the one—actually, there are three—that maps each pixel in the composite frame to the corresponding direction on the visual sphere).

The source image has three components (R, G, B). One must choose how to convert the raw RGB signal into a scalar quantity. In this paper, we are more interested in the general issues which are common across sensory modalities rather than specific issues of a particular sensor. We considered two filters: (1) a standard RGB to grayscale (luminance) conversion, and (2) grayscale, followed by the computation of the image contrast: $y(s) \mapsto \|\nabla y(s)\|$. We found that the results seem to be robust to the preprocessing step and should

be largely invariant for any other local filter applied to the images.

Before looking at the learning results, it is instructive to look at the first-order data statistics, such as the mean and variances of the signals (Fig. **??**). Already these simple statistics show that parts of the image are non informative: the borders of the omnidirectional camera and the camera reflection in the conic mirror have almost zero variance. One could use this information to ignore those parts; however we shall see that the tensor learning will ignore non informative parts automatically.

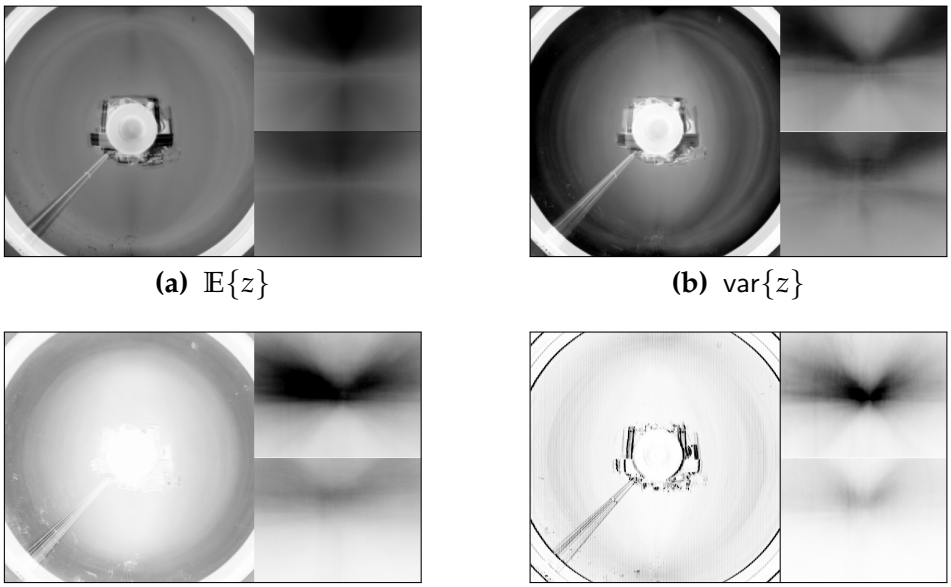
Fig. 12.14 shows the learning results, using the grayscale signal. Also in this case the sensel space $\mathcal{S} = [1, 640] \times [1, 480]$ has dimension $d = 2$ and there are $k = 2$ commands. Therefore, the tensor $\mathbf{H}$ has 4 components, each of which is a field across the sensel space, that can be displayed as 4 false colors images. Interpreting those images is not immediate. Remember that, ultimately, the tensor $\mathbf{H}_d^i$ shows how the derivative $\dot{y}$ is affected by the intensity of the d-th component of $\nabla_d y$ (∇_0 being the horizontal gradient and ∇_1 being the vertical gradient). Note that all fixed parts of the robot reflected in the mirror appear as white (zero values). In the field $\mathbf{H}_1^0$ (corresponding to linear velocity v and vertical gradient), it is easy to understand why the ceiling appears red (negative) and the floor blue (positive): if the robot moves with $v > 0$, and there is a positive gradient $\nabla_1 y > 0$, then one expects $\dot{y} > 0$ in the ceiling and $\dot{y} < 0$ for the floor. The part of the omnidirectional camera is all red (instead of blue) because the image is mirrored. The tensor $\mathbf{H}$ contains both intrinsic information about the sensor (direction of pixels) and extrinsic information (interaction sensor-command), as well as statistics of the environment (things on average further away correspond to lower response to translational motion).

One could make similar interpretations using the concept of *optic flow* (apparent retinal motion), however, notice we never compute optic flow and we do not assume the agent has the semantics of "motion" at all; we only use the quantities $\dot{y}$ and ∇y that can be

computed directly from the data without problems of regularization. Roberts *et al*. [83] present an analogous approach to learning "optic flow subspaces" using feature tracking and optimization.

Fig. **??** shows the analogous results using the contrast signal instead of grayscale. The learned tensors are very similar; the results should be invariant to all local image operations.

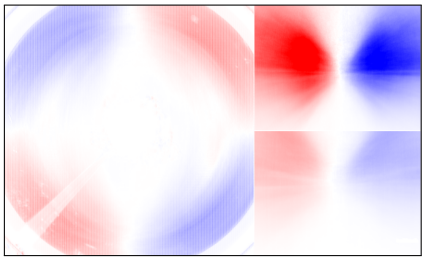
Fig. 12.15a–12.15d show an example of anomaly detection. As expected, objects (such as people) that move not coherently with the motion are readily detected. But the model also breaks down at occlusions and for very fast rotational motions; that is, where the signal evolution cannot be represented accurately using a smooth model such as (12.1).



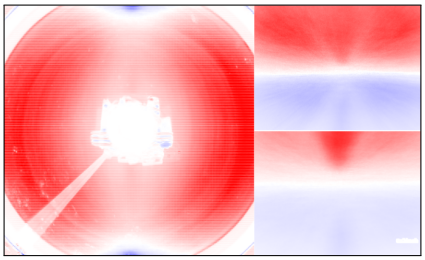
(a) $\mathbb{E}\{z\}$ **(b)** $\mathsf{var}\{z\}$

(c) $\mathsf{var}\{\dot{z}\}$ **(d)** $\mathsf{var}\{\nabla_1 z\}$

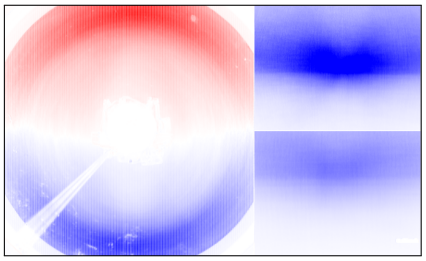
Figure 12.1. Statistics for the camera data (grayscale signal). In these figures we adopt the convention that white=zero and darker=positive. Subfigure (a) shows the observations mean. It is possible to see that the camera reflection remains fixed in the field of view, while the rest appears as a blur. Subfigure (b) shows the data variance. The part comprising the camera reflection appears as white, meaning that the variance is very small. Subfigure (c) shows the variance of $\dot{z}(x)$. This shows what parts of the image have the faster dynamics: apparently, more things happen in the distance rather than in the vicinity of the robot. Subfigure (d) shows the variance of $\nabla_1 z(x)$. It shows that some gradients are usually very strong, for example at the border between the subimages. Assuming that the world has uniform texture energy spectrum over the field of view, the energy of $\nabla z(x)$ is increasing with the distance, because when $z(x)$ is shrunk, the power spectrum of $\nabla z(x)$ increases.



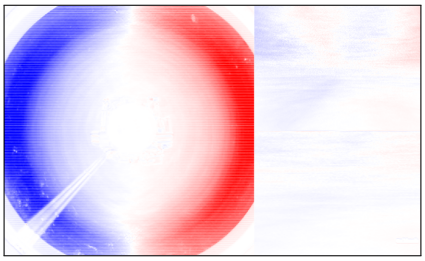
(a) $\mathbf{M}_1^1(x)$ (horiz. gradient and linear vel.)



(b) $\mathbf{M}_1^2(x)$ (vert. gradient and linear vel.)

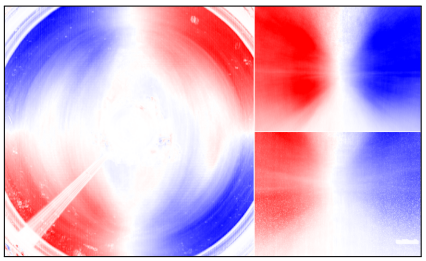


(c) $\mathbf{M}_2^1(x)$ (horiz. gradient and angular vel.)

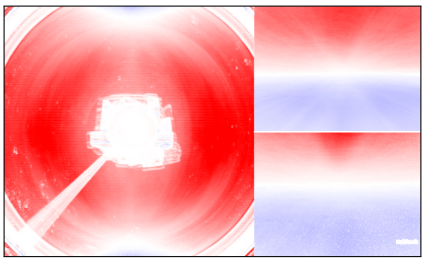


(d) $\mathbf{M}_2^2(x)$ (vert. gradient and angular vel.)

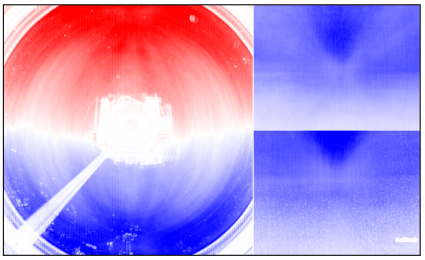
Figure 12.2. Learned tensors using **grayscale signal**. Blue means negative, red means positive, and white corresponds to zero. Collectively, these figures encode the sensors intrinsic and extrinsic calibration, along with some ancillary statistics about the environment, although the interpretation is not immediate. Note first that the parts that are not influenced by the robot motion (reflected camera, borders) appear as white, meaning that the tensor there is zero and negligible. The intensity depends on the average distance to the obstacle and to the pixel density on the visual sphere. The color depends on the pixel orientation on the visual sphere: see the text for more discussion.



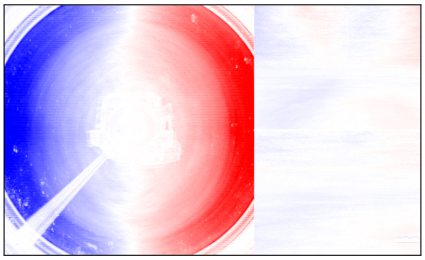
(a) Normalized $\mathbf{M}_1^1(x)$ (horiz. gradient and linear vel.)



(b) Normalized $\mathbf{M}_1^2(x)$ (vert. gradient and linear vel.)

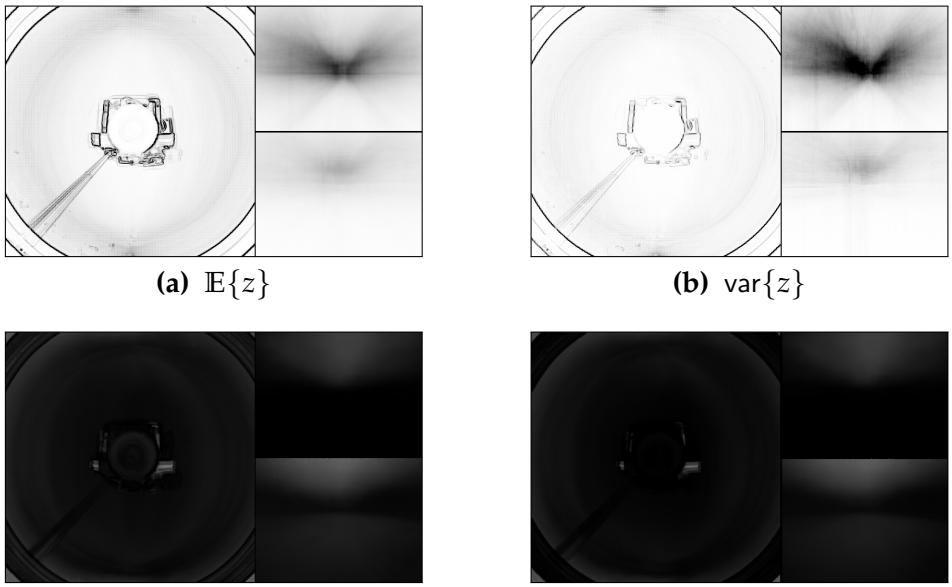


(c) Normalized $\mathbf{M}_2^1(x)$ (horiz. gradient and angular vel.)



(d) Normalized $\mathbf{M}_2^2(x)$ (vert. gradient and angular vel.)

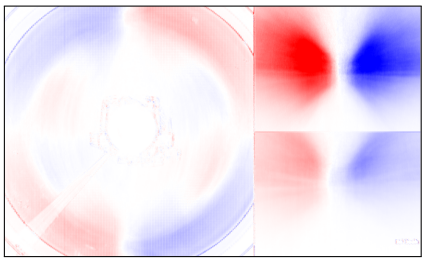
Figure 12.3. Learned tensors using **grayscale signal**, **normalized**. Blue means negative, red means positive, and white corresponds to zero. These figures are the equivalent of Fig. 12.2, but using the normalized signal (the gradients intensity are shown in Fig. 12.1). Although very similar, closer inspection shows some parts in which the normalization effect is evident. For example in (c) the immediate area around the robot base is given proper weight, which does not show in Fig. 12.2 (c).



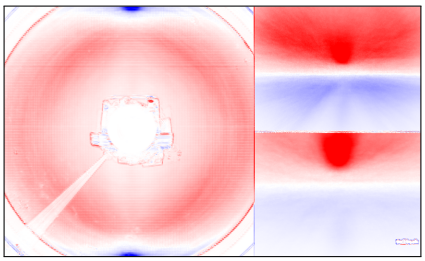
(a) $\mathbb{E}\{z\}$ **(b)** $\mathsf{var}\{z\}$

(c) $\mathsf{var}\{\dot{z}\}$ **(d)** $\mathsf{var}\{\nabla_1 z\}$

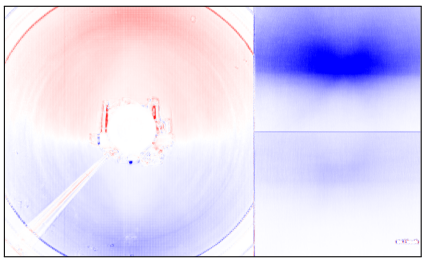
Figure 12.4. Statistics for the camera data (**contrast signal**). This figure shows the equivalent data of Fig. 12.1 for the contrast signal rather than for the grayscale signal. Note in (a) the very well defined contrast between the different cameras images, on the camera reflection, and mirror border. The subfigures (c) and (d), compared with the grayscale data in Fig. 12.1, show that the contrast signal is much more statistical uniform than the grayscale signal.



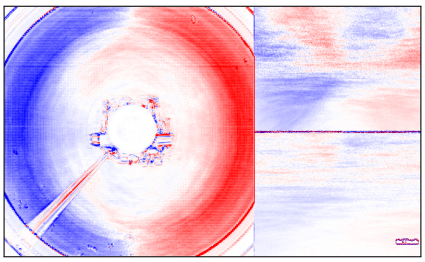
(a) $\mathbf{M}_1^1(x)$ (horiz. gradient and linear vel.)



(b) $\mathbf{M}_1^2(x)$ (vert. gradient and linear vel.)

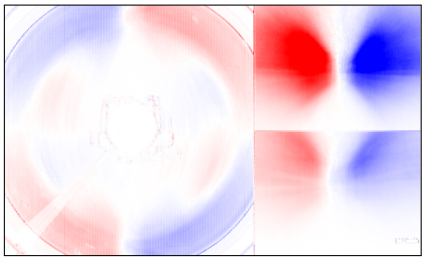


(c) $\mathbf{M}_2^1(x)$ (horiz. gradient and angular vel.)

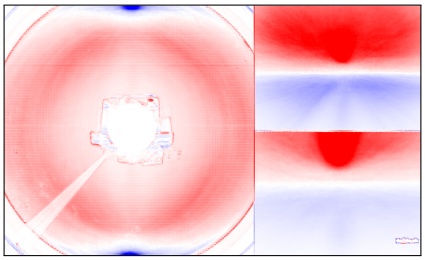


(d) $\mathbf{M}_2^2(x)$ (vert. gradient and angular vel.)

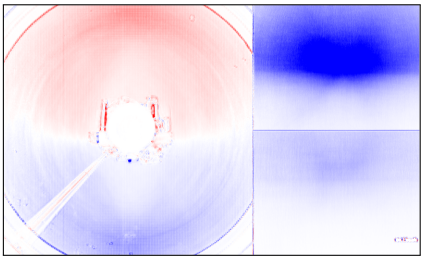
Figure 12.5. Learned tensors using **contrast signal**. Blue means negative, red means positive, and white corresponds to zero. This figure hows the equivalent data of Fig. 12.2 for the contrast signal rather than for the grayscale signal.



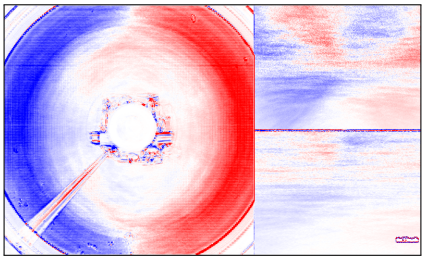
(a) Normalized $\mathbf{M}_1^1(x)$ (horiz. gradient and linear vel.)



(b) Normalized $\mathbf{M}_1^2(x)$ (vert. gradient and linear vel.)

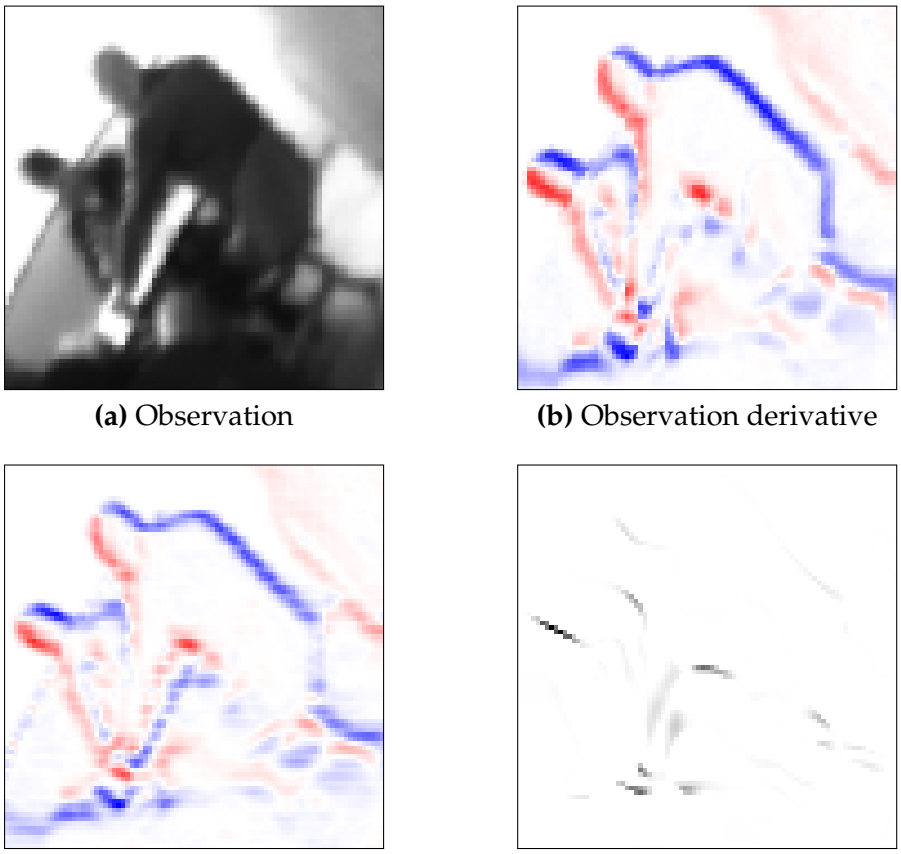


(c) Normalized $\mathbf{M}_2^1(x)$ (horiz. gradient and angular vel.)



(d) Normalized $\mathbf{M}_2^2(x)$ (vert. gradient and angular vel.)

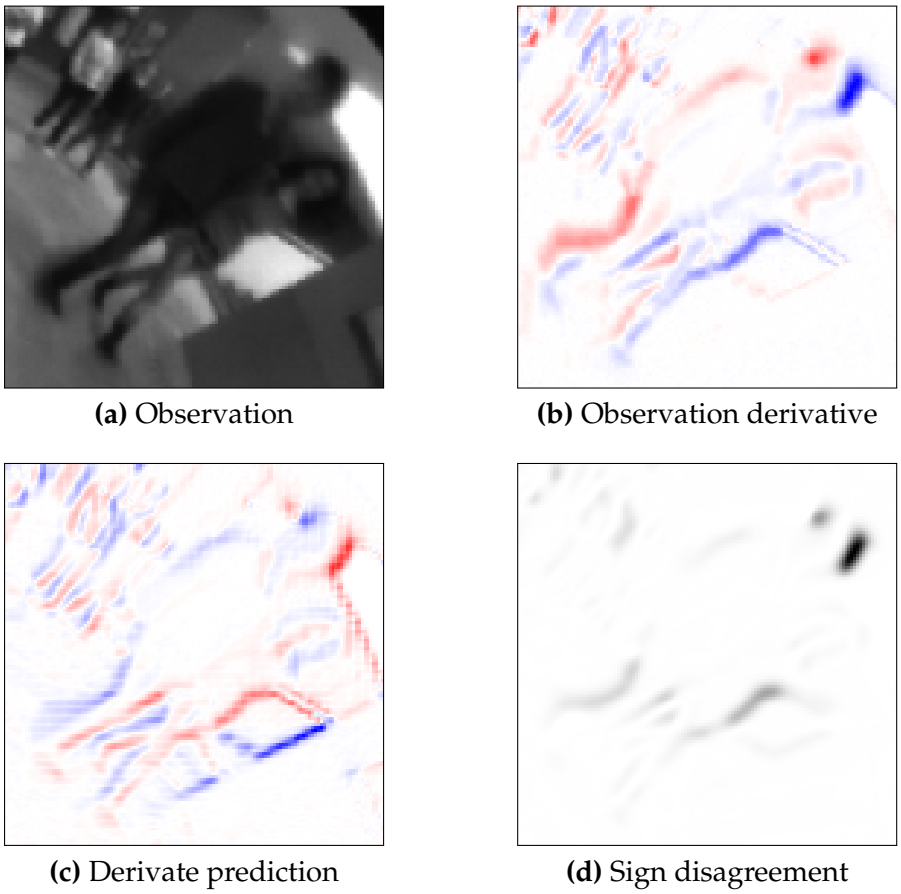
Figure 12.6. Learned tensors using **contrast signal**, **normalized**. Blue means negative, red means positive, and white corresponds to zero.



(a) Observation **(b)** Observation derivative

(c) Derivate prediction **(d)** Sign disagreement

Figure 12.7. Example of stationary image. The people in the image are mostly stationary in this image. Note that in (b) and (c) derivative observation and prediction roughly agree, albeit second order effects such as motion blur are not completely recovered.



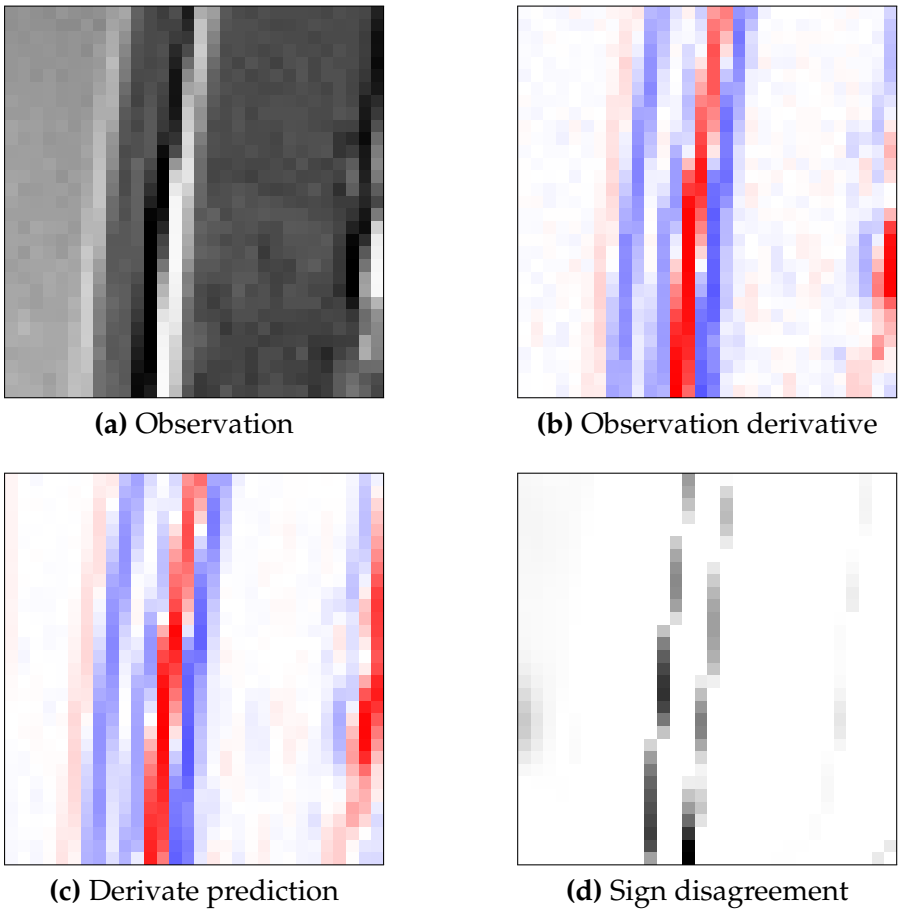
(a) Observation

(b) Observation derivative

(c) Derivate prediction

(d) Sign disagreement

Figure 12.8. Example of extraneous object detection. The people's independent motion gives an image derivative which has the opposite sign than the prediction of the BGDS model. The detection signal is active at the object's border.



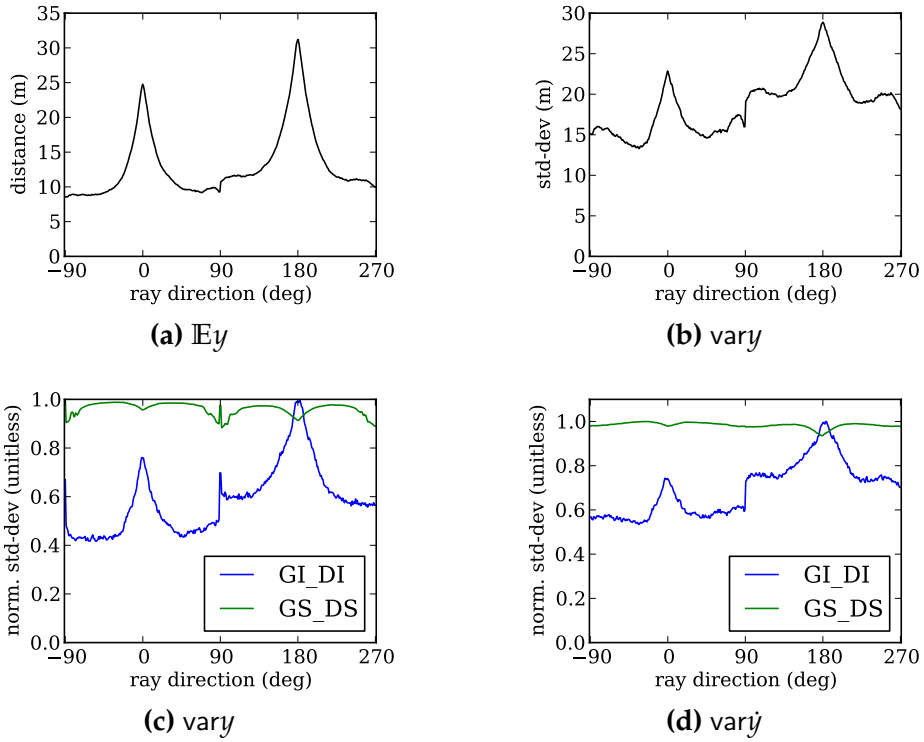
(a) Observation

(b) Observation derivative

(c) Derivate prediction

(d) Sign disagreement

Figure 12.9. Example of failure for very fine details. We noticed that the detection would give false positives for very fine textures. This figure shows a blowup of a 16px×16px detail of a door frame. The observed derivative and the prediction are apparently a very close match, but a closer examination reveals that the prediction is shifted 1px to the left. Subfigure (d) shows the detection signal: because of that shift, two bands are detected as inconsistent. We do not have a full explanation of this phenomenon, but probably it is linked to the fact that we approximate the derivative at instant k using the the difference $y_k - y_{k-1}$.



(a) $\mathbb{E}y$ **(b)** $\mathrm{var}y$ **(c)** $\mathrm{var}y$ **(d)** $\mathrm{var}\dot{y}$

Figure 12.10. Statistics for the range-finder data. Subfigure (a) and (b) show the observations mean and standard deviation. One can see that the hypothesis of the training distribution being mixed (Definition 9.12) is clearly not respected: the robot was driven along particular trajectories in very structured environments. In particular, note the peaks of distance at 0 and 180deg: the average shape is clearly that o a narrow corridor. Subfigures (c) and (d) show the variance of $\dot{z}$ and ∇z. Without filtering (configuration GI_DI), most of the signal energy is concentrated in front; while we see that filtering gradient and derivative with a sign() nonlinearity (configuration GS_DS) makes the energy constant across the field of view: that is, it mitigates the effect of large gradients and sudden changes in the signal.

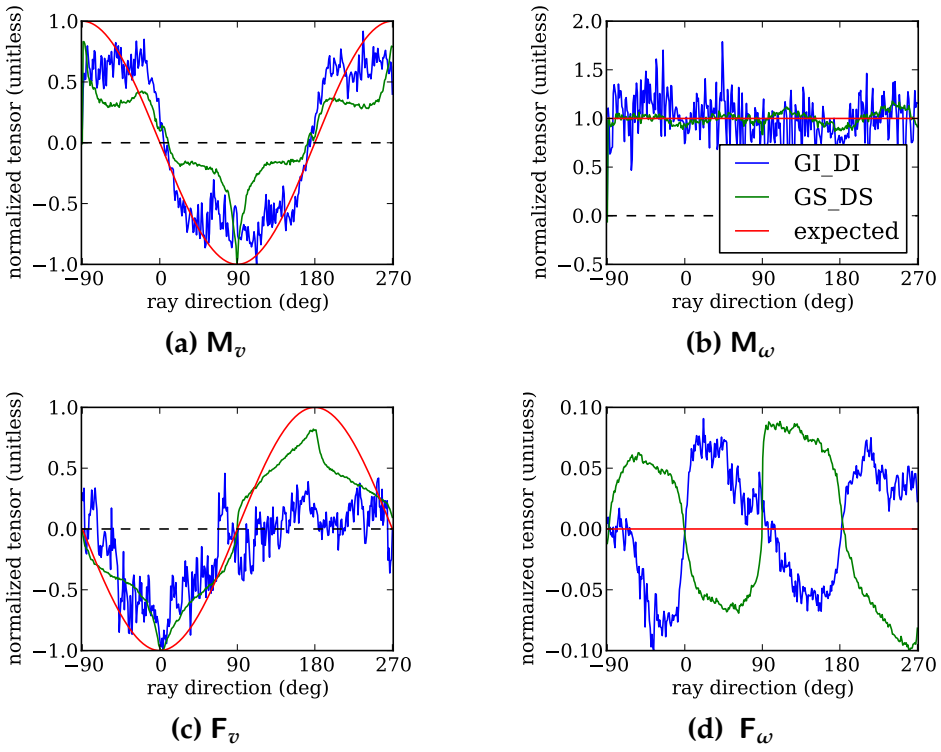
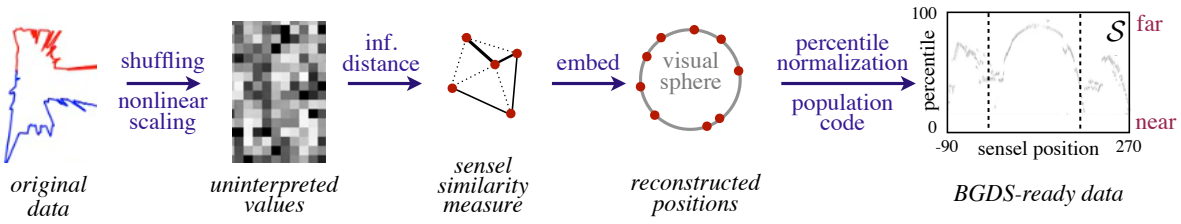
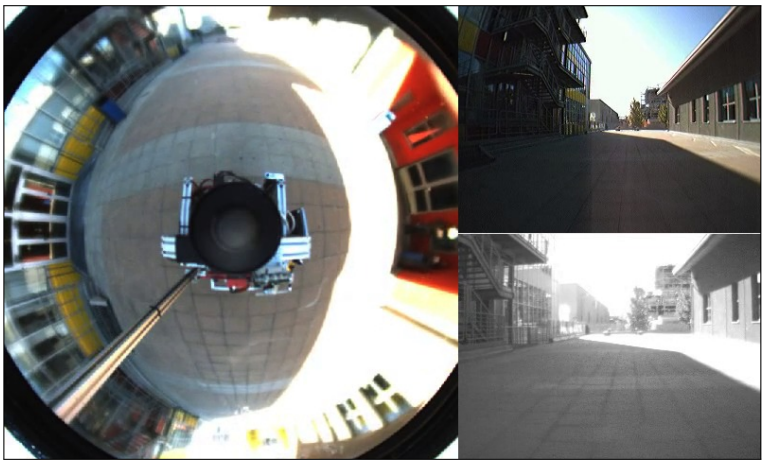


Figure 12.11. Tensors learned from range-finder data. We know the sensor configuration (two Sick range finders mounted in opposite directions aligned to the robot x-axis, therefore we can compute the expected result analytically (Table 11.2). We compare the results of using the raw derivative and gradient (GI_DI) and of filtering them with a sign() operation (GS_DS); the point being that such filtering help in obtaining smoother results, because the raw range data is very discontinuous. Subfigures (a),(b) show the two components $\mathbf{M}_v$ and $\mathbf{M}_\omega$ which refer to the bilinear part of the dynamics. In (a), the analytical result is $\phi \mapsto \sin(\phi)$. The learned tensors agree with the theory up to a nonlinear deformation, especially with the derivative and gradient filters. In (b), the theory predicts the response should be constant, and that is well verified. Subfigures (c),(d) show the tensors $\mathbf{F}_v$ and $\mathbf{F}_\omega$ that correspond to the affine part of the dynamics (that is, how $\boldsymbol{u}$ affects $\dot{\boldsymbol{z}}$ regardless of $\boldsymbol{z}$). The theory prediction for $\mathbf{F}_v$ is $\phi \mapsto \cos(\phi)$. The data agrees well for the frontal range finder, corresponding to the first half in the plot (-90deg to 90deg), while it does not for the rear range-finder. It is not clear why, but we can advance the hypothesis that this was somehow caused by the people following the robot, while the road ahead was typically clear. Note that the filtered version (GS_DS) successfully recovers the dynamics there. Finally, subfigure (d) depicts the tensor $\mathbf{F}_\omega$, which should be 0 according to the theory; in fact, the plot just shows magnified noise.)

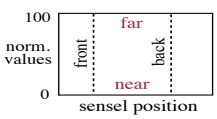


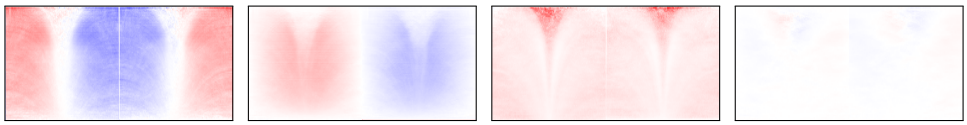
(a) Bootstrapping pipeline for range-finder data



(b) Composite camera frame

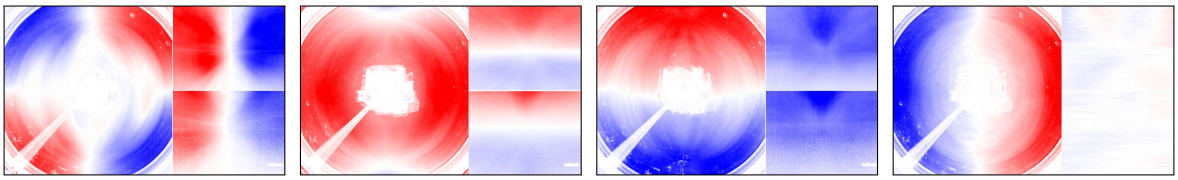
Figure 12.12. *Data sources used in the experiments.* (a) For range-finder data, we apply the complete bootstrapping pipeline as described in Section 12.5.4: we start from shuffled and distorted measurements; we compute sensels similarities with the information distance between the sensels values, from which an embedding on the visual sphere (circle) can be obtained [105]; finally, population coding (Definition 12.14) is used to obtain a two-dimensional "image" which is diffeomorphic to a polar map of the immediate surroundings. (b) For camera data, we use a composite frame obtained by stitching together the frames from an omnidirectional camera, a wide-angle frontal camera, and a gray-scale camera part of a stereo triplet. No previous knowledge of the optics is used.





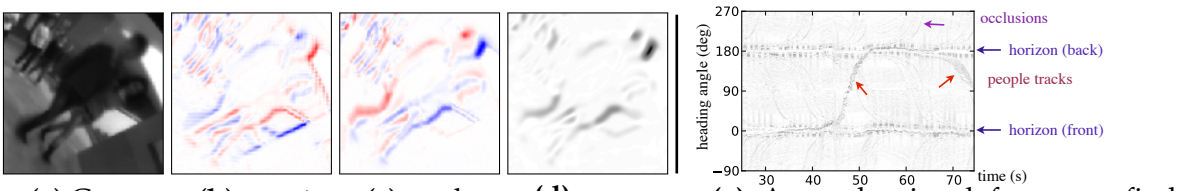
(a) $\mathbf{H}_0^0$ (hor. gradient and v) **(b)** $\mathbf{H}_1^0$ (ver. gradient and v) **(c)** $\mathbf{H}_0^1$ (hor. gradient and ω) **(d)** $\mathbf{H}_1^1$ (ver. gradient and ω)

Figure 12.13. *Tensors learned from range-finder data, with population code processing.* To understand these figures, it helps to think that the population code representation of range-finder data is diffeomorphic to a polar map of the environment. On the x-axis we have the angle (up to a diffeomorphism); on the y-axis we have the normalized values of the readings in percentiles (i.e., distance up to a diffeomorphism). In (c)-(d) the tensors that represent the interaction between angular velocity and horizontal and vertical gradient are respectively a constant and zero because the effect of rotation is just to translate horizontally these diagrams. (a)-(b) are not of easy interpretation, but the (anti) symmetry between the frontal and back range-finders is evident.



(a) $\mathbf{H}_0^0$ (hor. gradient and v) **(b)** $\mathbf{H}_1^0$ (ver. gradient and v) **(c)** $\mathbf{H}_0^1$ (hor. gradient and ω) **(d)** $\mathbf{H}_1^1$ (ver. gradient and ω)

Figure 12.14. *Tensors learned from camera data* (white: zero, red: positive, blue: negative). The tensor $\mathbf{H}$ encodes the sensors intrinsic and extrinsic calibration, along with some ancillary statistics about the environment. The interpretation is not immediate. Note first that the parts that are not influenced by the robot motion (reflected camera, borders) appear as white, meaning that the tensor there is zero and negligible. The intensity depends on the average distance to the obstacle and to the pixel density on the visual sphere. The color depends on the pixel orientation on the visual sphere: see the text for more discussion.



(a) Gray-scale data **(b)** $\dot{y}$ prediction **(c)** observed $\dot{y}$ **(d)** anomaly **(e)** Anomaly signal for range-finder data

Figure 12.15. *Anomaly detection using learned models.* We test the models on the task of anomaly detection, a passive task that can be done on logged data. The learned tensors are used to predict $\dot{y}$ using (12.1), and compute the anomaly detection signal (12.2). (a)–(d) show the results on camera data on a single frame. (e) shows the results for range finder data; the anomaly detection signal is shown over time (x axis) for each sensel (y axis). The constant traces represent the sensors horizon, where the model cannot predict the observations. The other traces represent people walking past the robot or in opposite direction. Occlusions (e.g., walking past open doors) also are detected as anomalies as the continuous model (12.1) cannot represent the discontinuous dynamics.

CHAPTER 13

Learning Diffeomorphisms

This chapter describes a more sophisticated model for robotic sensorimotor cascades. It is assumed that each command induces a diffeomorphism of the sensel space. In contrast with the previous models BDS *and* BGDS*, this model is not instantaneous and allows reasoning on potentially large robot displacements.*

Table 13.1. Symbols used in this chapter

Preliminaries		
$\mathcal{S}$	a manifold	Sensel space.
$\lvert\mathcal{S}\rvert$	> 0	Area of $\mathcal{S}$.
$d^{\mathcal{S}}$	$: \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+_\bullet$	Metric on $\mathcal{S}$.
$\mathsf{Id}_{\mathcal{S}}$	$: \mathcal{S} \to \mathcal{S}$	Identity of $\mathcal{S}$.
$\lvert\mathcal{U}\rvert$		Number of commands.
Models		
$\mathsf{DDS}(\mathcal{S};\mathcal{U})$	$\subset \mathcal{D}(\mathsf{Images}(\mathcal{S});\mathcal{U})$	Set of all DDS.
$\mathcal{V}$	$\subset \mathcal{S}$	Visible subset of $\mathcal{S}$.
$\mathsf{DDSL}(\mathcal{S},\mathcal{V};\mathcal{U})$	$\subset \mathcal{D}(\mathsf{Images}(\mathcal{S});\mathcal{U})$	Set of DDS with censored observations.
x	$\in \mathsf{Images}(\mathcal{S})$	Hidden state for a DDSL.
Learning and inference		
Γ		Learned uncertainty in a DDS.
$d_{\max}$	> 0	Bounds on the diffeomorphism.
ρ	> 0	Resolution of discretization.
d^{Diff}	$: \mathsf{Diff}(\mathcal{S}) \times \mathsf{Diff}(\mathcal{S}) \to \mathbb{R}^+_\bullet$	Distance between diffeomorphisms.
$\mathcal{D}_{\text{cmd}}$	$: \mathcal{U} \times \mathcal{U} \to \mathbb{R}^+_\bullet$	Distance between two commands.
$\mathcal{A}_{\text{cmd}}$	$: \mathcal{U} \times \mathcal{U} \to \mathbb{R}^+_\bullet$	"Anti-distance" between two commands.

13.1. DDS

DEFINITION 13.1 (DDS). A diffeomorphism dynamical system (DDS) is a discrete-time dynamical system on $\mathsf{Images}(\mathcal{S})$ with a finite commands alphabet

$$\mathcal{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{U}|}\}.$$

Each command $\boldsymbol{u}_j$ is associated to a diffeomorphism $\varphi_j \in \mathsf{Diff}(\mathcal{S})$. If $\boldsymbol{u}_j$ is active at time k, then the dynamics is

$$y_{k+1}(s) = y_k(\varphi_j(s)).$$

$\mathsf{DDS}(\mathcal{S}; \mathcal{U})$ is the set of all such systems on the manifold $\mathcal{S}$.

13.1.1. *Limited field of view*

Because a DDS can represent potentially large motions, it is important to include in the formalization also the idea of a limited field of view.

DEFINITION 13.2 (DDS with limited field of view). A DDS *with limited field of view* is a discrete-time dynamical system with hidden state $\boldsymbol{x}_k \in \mathsf{Images}(\mathcal{S})$, and a finite commands alphabet $\mathcal{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{U}|}\}$. Each command $\boldsymbol{u}_j$ is associated to a diffeomorphism $\varphi_j \in \mathsf{Diff}(\mathcal{S})$. The transition function from the state $\boldsymbol{x}_k$ at time k to the state $\boldsymbol{x}_{k+1}$ is given by

$$x_{k+1}(s) = x_k(\varphi(s)), \tag{13.1}$$

where φ is the diffeomorphism associated to the command given at time k. The observations $\boldsymbol{y} = \{y^s\}_{s \in \mathcal{S}}$ are a *censored* version of $\boldsymbol{x}$, in the sense that we only can see the state in

a subset $\mathcal{V} \subset \mathcal{S}$:

$$y_k(s) = \begin{cases} \boldsymbol{x}_k(s) & \text{if } s \in \mathcal{V}, \\ 0 & \text{if } s \notin \mathcal{V}. \end{cases} \tag{13.2}$$

$\mathsf{DDSL}(\mathcal{S}, \mathcal{V}; \mathcal{U})$ is the set of all such systems on the manifold $\mathcal{S}$.

13.1.2. *Symmetries for* DDS

PROPOSITION 13.3. *The family* $\mathsf{DDS}(\mathcal{S}; \mathcal{U})$ *is closed with respect to*

$\mathsf{Diff}(\mathcal{S})$ *Any diffeomorphism of the domain:*

$$y(s) \mapsto y(\varphi(s)), \quad \varphi \in \mathsf{Diff}(\mathcal{S}).$$

$\mathsf{Aut}(\mathbb{R})$ *Any automorphisms of the values:*

$$y(s) \mapsto f(y(s)), \quad f \in \mathsf{Aut}(\mathbb{R}).$$

$\mathsf{Aut}(\mathcal{U})$ *Any relabeling of the commands.*

13.2. Representing and Learning DDS

This section presents one approach to learning a DDS. Suppose that we are given training examples, consisting of tuples $\langle \boldsymbol{y}_k, \boldsymbol{u}_{j_k}, \boldsymbol{y}_{k+1} \rangle$, meaning that at time k we observed $\boldsymbol{y}_k$, then, after applying the command $\boldsymbol{u}_{j_k}$, we observed $\boldsymbol{y}_{k+1}$. Our objective is estimating the diffeomorphisms φ_j.

We assume that the $\mathcal{S}$ domain has been discretized into a finite number $n_{\boldsymbol{y}}$ of cells $\{s^i\}_{1 \le i \le n_{\boldsymbol{y}}} \subset \mathcal{S}$, such that the i-th cell has center $s^i \in \mathcal{S}$. We represent a diffeomorphism φ by its discretized version $\tilde{\varphi} : [1, n_{\boldsymbol{y}}] \to [1, n_{\boldsymbol{y}}]$ that associates to each cell s^i another cell $s^{i'}$, such that $i' = \tilde{\varphi}(i)$.

Learning can be done independently cell by cell. In fact, when the j-th command is applied, we expect that $y_{k+1}(s) = y_k(\varphi_j(s))$, as given by (13.1)–(13.2). This can be discretized in the following way, for the i-th cell: $y_{k+1}^{s^i} = y_k^{s^{\tilde{\varphi}_j(i)}}$. Therefore, the value $\tilde{\varphi}_j(i)$ can be found by minimizing the expected error:

$$\hat{\tilde{\varphi}}_j(i) = \arg\min_{i'} \mathbb{E}_{\boldsymbol{u}=\boldsymbol{u}_j}\{\|y_{k+1}(s^i) - y_k(s^{i'})\|\}. \tag{13.1}$$

This provides a point-by-point approximation of the diffeomorphism (continuity is preserved only asymptotically). Assuming we have a bound $d_{\max}$ on the maximum displacement over all diffeomorphisms:

$$\max_{j,s} d^{\mathcal{S}}(s, \varphi_j(s)) \leq d_{\max},$$

then the search for i' will not be needed to be extended to all cells in the domain, but only on the neighbors of i such that $d^{\mathcal{S}}(s^i, s^{i'}) \leq d_{\max}$. This is illustrated in Fig. 13.1c–13.1d for 2D domains. In practice, for each command and for each cell, we consider a square neighborhood of cells in which to search for the matching cell. This simple algorithm has complexity $\mathcal{O}(\rho^{2\dim(\mathcal{S})}|\mathcal{S}|d_{\max})$, where $|\mathcal{S}|$ is the area of the sensor, ρ the resolution, because for each of the $n_{\boldsymbol{y}} = \rho^{\dim(\mathcal{S})}|\mathcal{S}|$ cells, we have to consider a number of neighbors proportional to $\rho^{\dim(\mathcal{S})}d_{\max}$. Still, it is embarrassingly parallel, as the expectations in (13.1) can be computed separately, therefore it has decent performance if one uses vectorized operations.

13.2.1. *Estimating and propagating uncertainty*

In principle, one could treat each value $\tilde{\varphi}_j(i)$ as a random variable, and estimate its full distribution. In practice, this complexity is not needed in our application, and we limit ourselves to keeping track of a single scalar measure of uncertainty Γ_j^i, which we interpret

as being proportional to $\mathrm{Tr}(\mathrm{cov}(\varphi_j(s^i)))$. This uncertainty is computed from the value of the cost function (13.1):

$$\Gamma_j^i \simeq \mathbb{E}_{\boldsymbol{u}=\boldsymbol{u}_j}\{\|y_{k+1}(s^i) - y_k(s^{\tilde{\varphi}_j(i)})\|\}.$$

This uncertainty implicitly represents the limited field of view, because Γ_j^i is very large for cells s^i such that $\varphi_j(s^i) \notin \mathcal{V}$; that is, for cells whose values cannot be predicted because they depend on observations outside the field of view $\mathcal{V}$.

The DDS representation allows to compress a series of commands into one supercommand whose diffeomorphism is the composition of the individual diffeomorphisms. The composition of commands can keep track of the uncertainty as well. Suppose that we have two commands $\boldsymbol{u}_a$ and $\boldsymbol{u}_b$ and that we learned the two corresponding uncertain diffeomorphisms $\langle \tilde{\varphi}_a, \Gamma_a \rangle$ and $\langle \tilde{\varphi}_b, \Gamma_b \rangle$. Then the composite command $\boldsymbol{u}_c = \boldsymbol{u}_b \circ \boldsymbol{u}_a$ is represented by the pair $\langle \tilde{\varphi}_c, \Gamma_c \rangle$, where $\tilde{\varphi}_c$ is just the composition of $\tilde{\varphi}_a$ and $\tilde{\varphi}_b$:

$$\tilde{\varphi}_c(i) = \tilde{\varphi}_b(\tilde{\varphi}_a(i)),$$

but Γ_c takes into account both a *transport* and a *diffusion* component:

$$\Gamma_c(s^i) = \Gamma_a^{\tilde{\varphi}_b} + \Gamma_b^i.$$

13.3. Application to Camera Data

This section describes the application of the theory to camera data for a mobile robot. It is shown that the learned diffeomorphisms capture the motions, as well as the uncertainties due to the limited field of view of the camera. Based on the model, one can obtain long-term predictions of the observations given a sequence of commands, and these predictions correctly take into account the uncertainty due to the limited field of view.

Platform. We use an Evolution Robotics ER1 robot (Fig. 13.1a), with an on-board webcam producing 320×240 frames at ~7.5 Hz. The robot is driven through a variety of indoor and outdoor environments (Fig. 13.1b) for a total of around 50 minutes. The robot linear and angular velocities were chosen among the combinations of $\omega \in \{-0.2, 0, +0.2\}$ rad/s and $v \in \{-0.3, 0, +0.3\}$ m/s.

Results. Fig. 13.2 shows the resulting of the diffeomorphisms learning applied to this data. Not all combinations of commands are displayed; in particular, those corresponding to the robot backing up were not chosen frequently enough to obtain reliable estimates of the corresponding diffeomorphisms. Because we also learn the uncertainty of the diffeomorphisms, we can correctly predict effects due to the limited field of view for the camera. For example, for the first command, corresponding to a pure rotation to the right, we can predict that for 8 time steps we can predict the left half of the image, but we will not know anything about the right half (Fig. 13.2, bottom sequence).

13.4. Inferring the "Linear structure" of the Commands Space

Assuming a class of models, such as BDS/BGDS, where the commands have a linear effect on the dynamics (for example, if they correspond to kinematic velocities) automatically gives rich structural properties to the commands space. For example, the effect of applying $\boldsymbol{u}' = 2\boldsymbol{u}$ is twice larger than the effect of $\boldsymbol{u}$; the effect of $\boldsymbol{u} = 0$ corresponds to a null action, and the effect of $\boldsymbol{u}' = -\boldsymbol{u}$ is the opposite of $\boldsymbol{u}$. If the commands are kinematic, this structure can be lost if they are represented in a nonlinear way, for example if one has available the commands $\boldsymbol{u}' = f(\boldsymbol{u})$ instead of $\boldsymbol{u}$.

The DDS family does not assume that the command space $\mathcal{U}$ has any particular structure. By comparison, the BDS and BGDS models assume that $\mathcal{U}$ is a vector space and that the dynamics of $\boldsymbol{y}$ is linear in the commands. This implies a series of semantic assumptions, which are collectively referred to as "linear structure":

- Assumption 10 (*The system is reversible*)
- Assumption 11 (*Similar commands have similar effects*)
- Assumption 12 (*One command does nothing*)
- Assumption 13 (*A known command does nothing*)
- Assumption 14 (*Minus does the opposite*)
- Assumption 16 (*Half does half*)

These assumptions are not enforced a priori by a DDS. However, if these assumptions hold, then it is useful for the agent to be able to recognize this structure.

13.4.1. *Identifying redundant and null commands*

We call a pair of commands $\langle \boldsymbol{u}_j, \boldsymbol{u}_k \rangle$ *redundant* if they have the same effect on the observations. If two commands give the same effect, then one of them can be removed from the commands alphabet. Redundant commands can be recognized simply by looking at the distance between the corresponding diffeomorphisms.

DEFINITION 13.4 (Naive distance between diffeomorphisms). We will use the following distance between diffeomorphisms:

$$d^{\mathsf{Diff}}(\varphi_1, \varphi_2) = \int d^{\mathcal{S}}(\varphi_1(s), \varphi_2(s))\, d\mathcal{S}. \tag{13.1}$$

DEFINITION 13.5 (Distance between commands). The distance $\mathcal{D}_{\mathsf{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k)$ between two commands is defined as the difference between the corresponding diffeomorphisms, normalized by the average distance between all diffeomorphisms:

$$\mathcal{D}_{\mathsf{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k) = \frac{d^{\mathsf{Diff}}(\varphi_j, \varphi_k)}{\frac{1}{|\mathcal{U}|^2} \sum_l \sum_m d^{\mathsf{Diff}}(\varphi_l, \varphi_m)}.$$

The normalization makes this a unitless quantity that does not depend on the size of $\mathcal{S}$. As a special case, commands that have no effect can be easily identified by considering the distance of their diffeomorphism from the identity diffeomorphism $\mathsf{Id}_{\mathcal{S}}$.

13.4.2. *Identifying reversible commands*

A pair of commands $\langle \boldsymbol{u}_j, \boldsymbol{u}_k \rangle$ are *reversible* if applying $\boldsymbol{u}_j$ followed by $\boldsymbol{u}_k$ brings the system in the initial state, and vice versa; see Assumption 10 (*The system is reversible*). Two commands would be perfectly reversible if $\varphi_j = \varphi_k^{-1}$ or, equivalently, $\varphi_j^{-1} = \varphi_k$, or $\varphi_k \circ \varphi_j = \varphi_j \circ \varphi_k = \mathsf{Id}_{\mathcal{S}}$.

All of these conditions are equivalent in the continuum case and without noise, but might give slightly different results in practice due to numerical approximations and estimation noise. Somehow arbitrarily, we choose to define the *anti-distance* of a commands pair $\langle \boldsymbol{u}_j, \boldsymbol{u}_k \rangle$ as

$$\mathcal{A}_{\text{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k) = \frac{\frac{1}{2}(d^{\mathsf{Diff}}(\varphi_j, \varphi_k^{-1}) + d^{\mathsf{Diff}}(\varphi_j^{-1}, \varphi_k))}{\frac{1}{|\mathcal{U}|^2}\sum_l \sum_m d^{\mathsf{Diff}}(\varphi_l, \varphi_m)}.$$

If the anti-distance is zero, then the command pair is a perfectly reversible pair. The reason for averaging $d^{\mathsf{Diff}}(\varphi_j, \varphi_k^{-1})$ and $d^{\mathsf{Diff}}(\varphi_j^{-1}, \varphi_k)$ is that it enforces the symmetry condition $\mathcal{A}_{\text{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k) = \mathcal{A}_{\text{cmd}}(\boldsymbol{u}_k, \boldsymbol{u}_j)$.

13.4.3. *Invariance analysis*

Unfortunately, the distance (13.1) that we used so far is not left-invariant, in the sense that $d^{\mathsf{Diff}}(\alpha \circ \varphi_1, \alpha \circ \varphi_2) \neq d^{\mathsf{Diff}}(\varphi_1, \varphi_2)$ for any α that is not an isometry. This means that the thresholds to commands distances and anti-distances to decide if commands pairs are redundant or reversible pairs would have to be retuned if the parametrization of $\mathcal{S}$ change. A left-invariant distance is of the form

$$d^{\mathsf{Diff}}(\varphi_1, \varphi_2) = f(\varphi_1^{-1} \circ \varphi_2), \tag{13.2}$$

as it implies that

$$\begin{aligned} d^{\mathsf{Diff}}(\alpha \circ \varphi_1, \alpha \circ \varphi_2) &= d^{\mathsf{Diff}}((\alpha \circ \varphi_1)^{-1} \circ (\alpha \circ \varphi_2)) \\ &= d^{\mathsf{Diff}}(\varphi_1^{-1} \circ \alpha^{-1} \circ \alpha \circ \varphi_2) \\ &= d^{\mathsf{Diff}}(\alpha \circ \varphi_1, \alpha \circ \varphi_2). \end{aligned}$$

This distance has not been used because (13.2) implies explicitly constructing the inverse of diffeomorphism, which is not robust to noise if using the simple discretization introduced in this chapter.

This is a perfect example in which the format of the data allows the agent to be invariant to a large class of nuisances ($\mathsf{Diff}(\mathcal{S})$) but the limitation of the algorithm make it invariant to a smaller class ($\mathsf{Isom}(\mathcal{S}) \leq \mathsf{Diff}(\mathcal{S})$).

This naive discretization of diffeomorphisms is intuitive but does not conserve certain important properties. *Discrete geometry* is a discipline concerned with the discretization of objects in differential geometry to a discrete domain, and finds applications in area such as fluid mechanics and computer graphics. Gawlik *et. al.* [133] describe the *discrete diffeomorphism group*: if a certain manifold is approximated with a simplicial complex of n cells, discrete diffeomorphisms are represented as a certain subfamily of $n \times n$ stochastic matrices, in a way such that properties of continuous diffeomorphisms are preserved in the discretized version. In practice, using stochastic matrices allows each cell to correspond to multiple cells, even without considering uncertainty. Using such representation would probably improve the accuracy of the representation, but the possible accuracy gains are to be weighted with the increased computational complexity (from $\mathcal{O}(n)$ of the current method to $\mathcal{O}(n^2)$ of the discrete diffeomorphism group, even without considering uncertainty).

13.5. Application to Range Data

In this section, we apply the theory to sensorimotor cascades with range finder data. We preprocess the 1D range data to obtain a 2D population code representation, which allows to treat range data using exactly the same code as 2D images. We try the method on three dynamics: a differential-drive robot (where the commands are left/right track velocity), and, in simulation, with a unicycle dynamics (commanded in linear/angular velocity) and a car-like dynamics (commanded with steering angle and driving velocity). We show that the concept of commands distance/anti-distance allows to discover redundant commands and reversible commands pairs independently of the commands representation.

13.5.1. *Processing pipeline*

The pipeline that we use for processing the range finder data is shown in Fig. 13.3. Starting from a planar scan (Fig. 13.3*a*), we consider the polar representation (Fig. 13.3*b*). Then, we transform the 1D signal into a 2D signal by using a population code representation (Fig. 13.3*c*); each reading y^i is assigned a row of cells, and each cell is assigned a center $c^{i,k}$. The activation level of each cell is a function of the distance between y^i and $c^{i,k}$. Denoting the 2D signal $Y^{i,k}$, we set $Y^{i,k} = f(|y^i - c^{i,k}|)$ where f is a small Gaussian kernel ($\sigma = 1\%$ of the range of y^i). Once we have the 2D signal, we forget its origin as a range finder scan, and we treat it like any other image.

Fig. 13.3 shows also an example of prediction. Starting from the 2D signal Y_0 in Fig. 13.3*c*, and a learned diffeomorphism φ (represented here by Lena), we obtain the predicted signal Y_4 in Fig. 13.3*d* by applying 4 times the diffeomorphism φ (or, by first computing $\varphi' = \varphi \circ \varphi \circ \varphi \circ \varphi$, and then applying φ' to Y_0, which is the same, up to numerical errors). For visualizing the result, we can convert back to range readings (Fig. 13.3*e*) and range scan (Fig. 13.3f).

13.5.2. *Dynamics considered*

We consider three common mobile robot dynamics for wheeled mobile robots: unicycle (Fig. 13.4*a*), car-like (Fig. 13.5*b*), and differential-drive (Fig. 13.6*c*). All three dynamics have two commands: by appropriate normalization we can assume that $\boldsymbol{u} \in [-1, +1] \times [-1, +1]$ for all of them.

The unicycle and differential-drive have the same dynamics, but with different representations of the commands: the linear and angular velocity of the unicycle are linearly related to left/right wheel velocity for a differential drive. For the car-like dynamics, we assume that one command is the driving velocity, and the other is the instantaneous steering angle. The car-like dynamics is more restricted than the other two, as the vehicle cannot turn in place.

13.5.3. *Learning data*

As an example of a differential-drive robot, we use an iRobot *Landroid*, with a *Hokuyo* [134] range finder on board. The Hokuyo has a maximum range of 8m and an update frequency of 10 Hz. The field of view of a Hokuyo is 270°, but the sensor is partially obstructed by the WiFi antennas. The learning data is taken in a cluttered lab environment, for a total of about 45 minutes. We use simulated data for the unicycle and the car-like, simulating a 360° range finder with the same range as the Hokuyo. The simulated world is generated randomly from a collection of randomly placed polygons; the simulation is tuned to have approximately the same spatial statistics of the lab environment.

The commands alphabet is composed of the 9 canonical commands of the form (a, b), for $a, b \in \{-1, 0, 1\}$. The effect on the robot pose of choosing each canonical command is sketched in the grids in Fig. 13.4*b*, 13.5*b*, 13.6*b*.

13.5.4. *Learned diffeomorphisms*

The learned diffeomorphisms are shown in the grids in Fig. 13.4*c*, 13.5*c*, 13.6*c*. Here, the diffeomorphisms are visualized by their effect on the Lena template.

It turns out that learning diffeomorphisms of the population-code representation of range data is more challenging than learning diffeomorphisms of RGB images, because the data is much sparser (see, e.g., Fig. 13.3*cd*). It was surprising to see that, of all the diffeomorphisms learned, the most noisy result is for the commands that do not move the robot (Fig. 13.5*c*, middle row), the reason being that the motion we are trying to recover is small (actually, zero) with respect to the sensor noise. This uncertainty is appropriately captured by the estimate of Γ (not shown).

13.5.5. *Learned command structure*

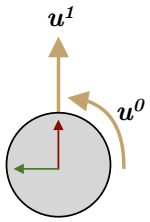
Tables 13.2, 13.3, 13.4 show the computed distance $\mathcal{D}_{\text{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k)$ and anti-distance $\mathcal{A}_{\text{cmd}}(\boldsymbol{u}_j, \boldsymbol{u}_k)$ for all commands pairs for the three dynamics considered.

For example, for the unicycle the reversible pairs are $\langle(-1,-1),(1,1)\rangle$, $\langle(1,0),(-1,0)\rangle$, $\langle(0,+1),(0,-1)\rangle$, $\langle(-1,1),(1,-1)\rangle$, and these are given the smaller values of anti-distance in Table 13.2b. The unicycle commands have a native linear structure, so reversible pairs are of the form $\langle(a,b),(-a,-b)\rangle$.

The car-like dynamics has the three null and redundant commands: $(0,0)$, $(0,1)$, and $(0,-1)$; these correspond to the driving velocity set to zero, and their corresponding diffeomorphism is the identity. The detected reversible pairs are $\langle(1,0),(-1,0)\rangle$, $\langle(1,1),(-1,1)\rangle$, $\langle(1,-1),(-1,$ these are of the form $\langle(a,b),(-a,b)\rangle$, which shows that the car-like dynamics is not linear in the original representation. Yet, we are able infer the linear structure from the analysis of the learned diffeomorphisms.

For the differential-drive dynamics, learned with real data, the pairs with the two

Table 13.2. Distance and anti-distance matrices for unicycle dynamics.



u^0: angular velocity
u^1: linear velocity

(a) Distance between commands

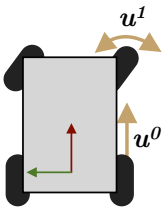
	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	0.68	0.70	1.19	1.21	1.23	0.86	0.92	0.91
(0,1)		-	0.93	1.04	0.92	1.25	0.97	0.86	1.17
(0,-1)			-	1.04	1.23	0.91	0.98	1.18	0.87
(1,0)				-	0.50	0.53	1.74	1.73	1.75
(1,1)					-	0.92	1.73	1.82	1.66
(1,-1)						-	1.73	1.82	1.66
(-1,0)							-	0.46	0.44
(-1,1)								-	0.84
(-1,-1)									-

(b) Anti-distance between commands

	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	0.85	0.83	0.97	1.03	1.02	1.34	1.38	1.38
(0,1)		-	**0.36**	1.07	1.26	0.95	1.17	1.35	1.08
(0,-1)			-	1.07	0.95	1.26	1.15	1.06	1.33
(1,0)				-	1.81	1.81	**0.21**	0.58	0.57
(1,1)					-	1.71	0.58	0.97	**0.28**
(1,-1)						-	0.57	**0.30**	0.95
(-1,0)							-	1.90	1.91
(-1,1)								-	1.84
(-1,-1)									-

lowest anti-distance are $\langle(1,0),(-1,0)\rangle$ and $\langle(1,1),(-1,-1)\rangle$. Then, there are a few false matches which have lower anti-distance than the other two pairs of reversible commands ($\langle(0,1),(0,-1)\rangle$ and $\langle(-1,1),(1,-1)\rangle$. This is probably due to the fact that computing the inverse of a diffeomorphism is very sensitive to noise, and currently we do not take into account the estimated diffeomorphism uncertainty, which is very large in this case, due to the limited field of view, and the antennas occlusions.

Table 13.3. Distance and anti-distance matrices for car-like dynamics.



u^0: driving velocity
u^1: steering angle

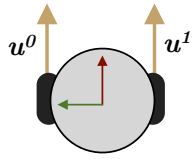
(a) Distance between commands

	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	**0.65**	**0.66**	0.85	1.51	0.94	0.85	0.96	1.49
(0,1)		-	**0.56**	0.75	1.40	0.97	0.76	0.97	1.39
(0,-1)			-	0.78	1.43	0.96	0.79	0.97	1.41
(1,0)				-	1.02	0.96	0.91	1.22	1.30
(1,1)					-	1.85	1.29	2.01	0.89
(1,-1)						-	1.23	0.80	2.00
(-1,0)							-	0.98	0.99
(-1,1)								-	1.85
(-1,-1)									-

(b) Anti-distance between commands

	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	1.20	1.22	1.05	1.04	1.66	1.01	1.67	1.03
(0,1)		-	1.10	0.95	1.07	1.56	0.91	1.57	1.07
(0,-1)			-	0.98	1.07	1.58	0.93	1.59	1.03
(1,0)				-	1.29	1.40	**0.41**	1.19	1.03
(1,1)					-	0.90	1.04	**0.25**	1.91
(1,-1)						-	1.14	2.02	**0.23**
(-1,0)							-	1.38	1.30
(-1,1)								-	0.91
(-1,-1)									-

Table 13.4. Distance and anti-distance matrix for Landroid (differential drive).



u^0: left wheel velocity
u^1: right wheel velocity

(a) Distance between commands

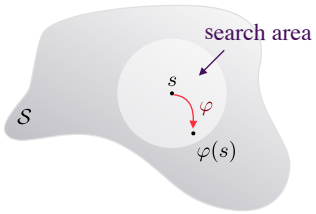
	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	1.34	1.25	1.41	0.89	1.49	0.52	0.89	0.86
(0,1)		-	1.12	0.55	0.88	0.81	1.34	1.74	0.87
(0,-1)			-	1.29	0.90	1.37	1.24	1.62	0.96
(1,0)				-	0.95	0.67	1.45	1.79	0.96
(1,1)					-	1.06	0.92	1.30	0.54
(1,-1)						-	1.51	1.86	1.03
(-1,0)							-	0.92	0.87
(-1,1)								-	1.24
(-1,-1)									-

(b) Anti-distance between commands

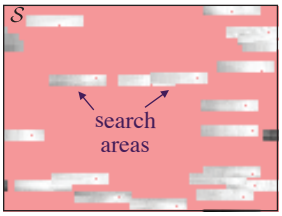
	(0,0)	(0,1)	(0,-1)	(1,0)	(1,1)	(1,-1)	(-1,0)	(-1,1)	(-1,-1)
(0,0)	-	1.08	1.46	1.07	1.32	1.33	1.71	1.98	1.26
(0,1)		-	1.76	2.08	1.49	2.21	1.14	1.39	1.46
(0,-1)			-	1.80	1.36	1.92	1.51	1.73	1.33
(1,0)				-	1.56	2.29	**1.06**	1.33	1.51
(1,1)					-	1.70	1.35	1.63	**0.95**
(1,-1)						-	1.36	1.26	1.65
(-1,0)							-	2.01	1.33
(-1,1)								-	1.60
(-1,-1)									-

(a) Robot platform

(b) Environments samples



(c) Geometry of the problem



(d) Algorithmic approximation

Figure 13.1. (a) For the first set of experiments we use an Evolution Robotics ER1 with an on-board camera. (b) The robot is driven through a variety of indoor and outdoor environments. (c) To learn the diffeomorphisms, we assume to have a bound on the maximum displacement $d(s, \varphi(s))$ on the manifold $\mathcal{S}$. (d) In the implementation, we are limited to square domains. The search area around each point is constrained to be a square, with given width and height, which are tunable parameters that affect the efficiency of the algorithm.

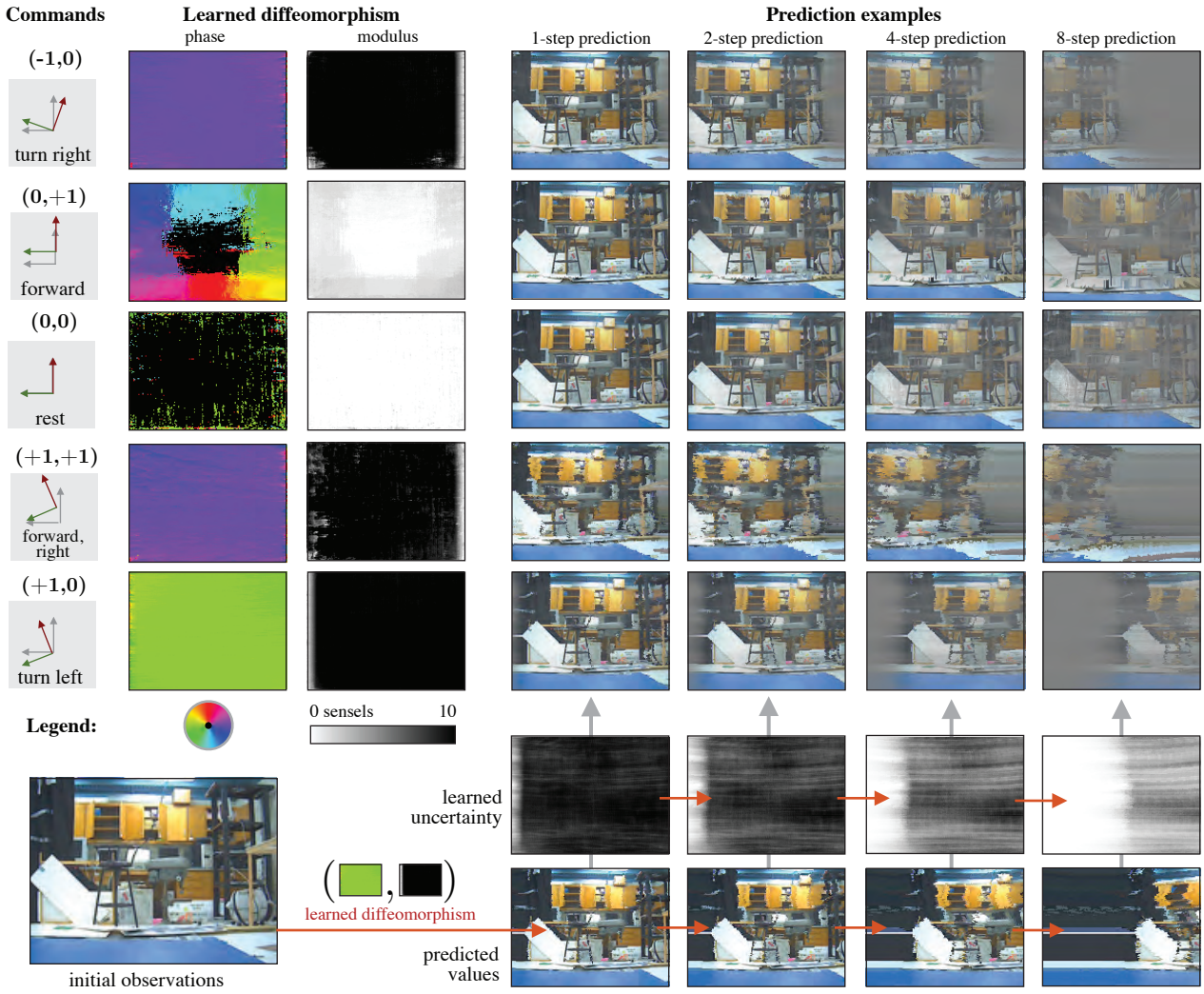


Figure 13.2. This figure shows a few of the learned diffeomorphisms learned from the camera data. Each row corresponds to a particular command given to the robot. The first column shows the effect of the command on the robot pose. The second and third row show the corresponding diffeomorphism, displayed using phase and modulus. The last four columns show how the learned diffeomorphisms can be used for prediction. The columns show the predicted image following the application of the command for 1, 2, 4, and 8 time steps. The uncertain parts of the predictions are shown in gray. The visualization of this uncertainty is done by propagating both the diffeomorphism uncertainty and the image values, as shown in the last two rows, and then blending the values with a solid gray rectangle according to the predicted uncertainty.

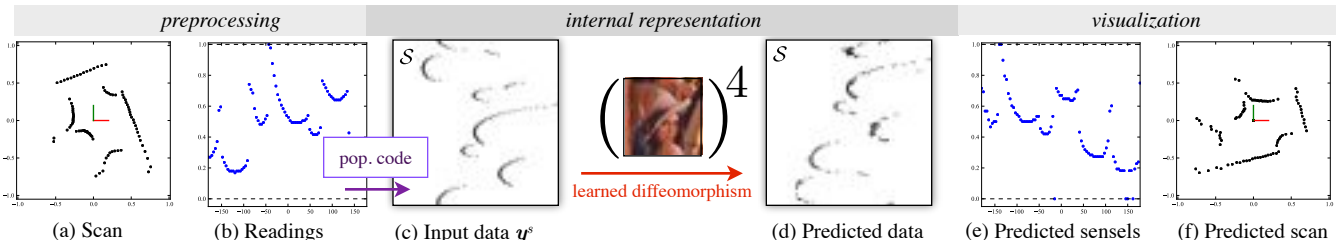


Figure 13.3. This figure shows the pipeline that we use for range-finder data. Starting from the scan (subfigure *a*) we consider the raw range-readings (i.e., the polar representation of the scan). Then we use a population code to obtain a 2D image from the 1D data. Once we have the 2D data, we forget about its origin as a range-finder scan, and we use exactly the same code we used for images. Here we show an example of prediction: one learned diffeomorphism, here represented by Lena, is applied 4 times to the image in *c* to obtain the predicted image in *d*. For visualization purposes, from the 2D image we can go back to the range readings (subfigure *e*) and obtain the predicted scan (subfigure *f*), which shows that the learned diffeomorphism corresponded to a pure rotation.

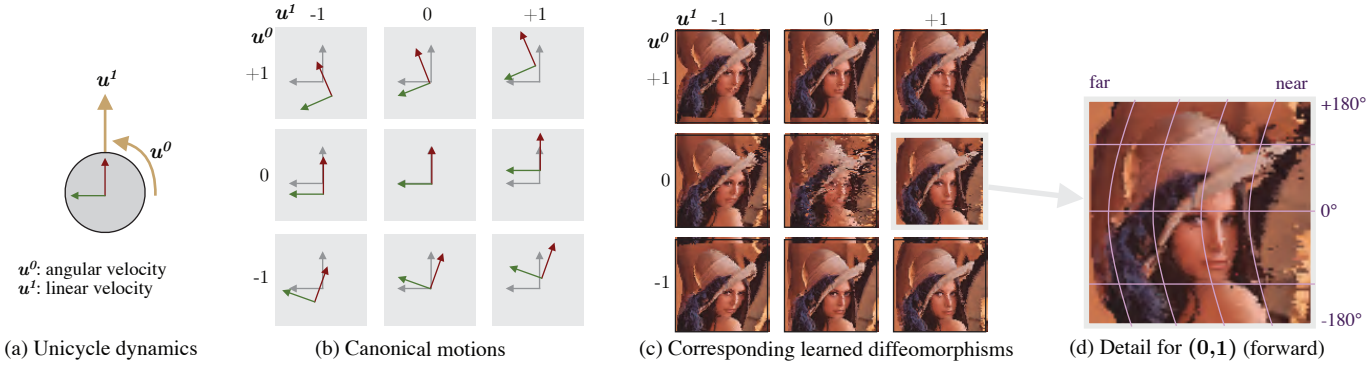
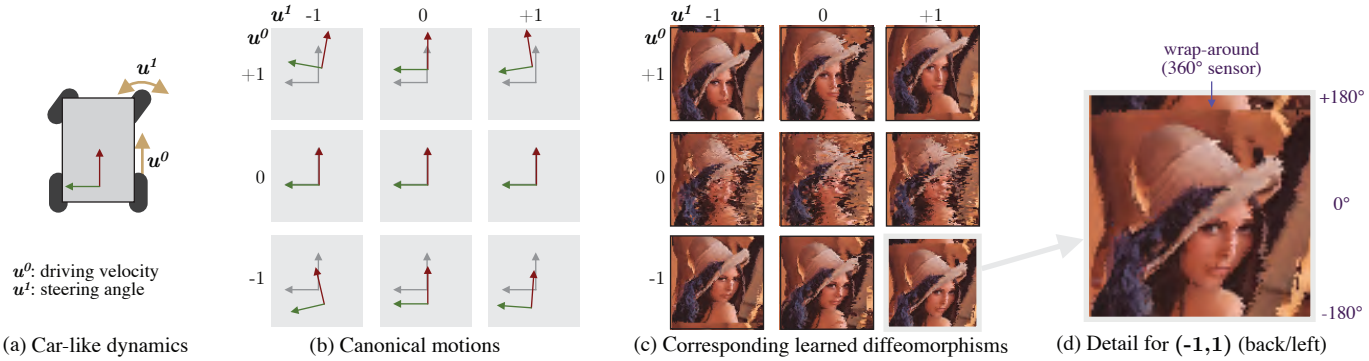
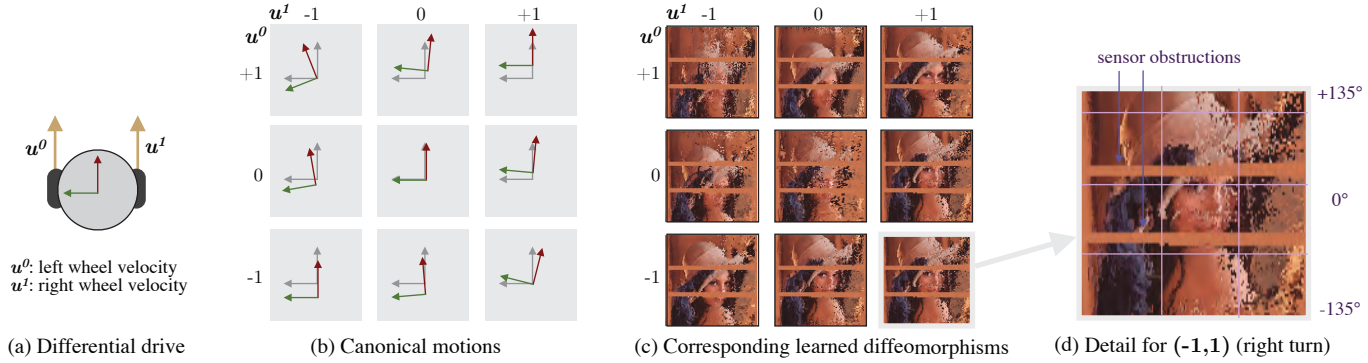


Figure 13.4. In this series of figures, subfigure *a* illustrates the robot dynamics; in this case, a unicycle dynamics. Subfigure *b* shows the effect on the robot pose of the 9 canonical commands. The gray arrows denote the initial pose, and the red/green arrows (for the x and y direction, respectively) show the robot final pose after applying the motion. Subfigure *c* shows the effect of the commands on the population code representation for a range-finder scan mounted on the robot (see Fig. 13.3 for an explanation of the preprocessing to obtain a 2D image from a 1D scan).



(a) Car-like dynamics (b) Canonical motions (c) Corresponding learned diffeomorphisms (d) Detail for (-1,1) (back/left)

Figure 13.5. A car-like dynamics, commanded in driving velocity and instantaneous steering angle, is more restricted than a unicycle as the robot cannot turn in place. Note that the three commands $(0,-1)$, $(0,0)$, $(0,+1)$ are equivalent and correspond to the robot staying in place.



(a) Differential drive (b) Canonical motions (c) Corresponding learned diffeomorphisms (d) Detail for (-1,1) (right turn)

Figure 13.6. A differential-drive dynamics is the same as a unicycle dynamics following a change of representation for the commands; compare subfigure *b* with Fig. 13.4b. The data we use for the differential drive comes from a real robot, mounting a Hokuyo range-finder with a 270° field of view. There are two antennas in front obstructing the range-finder, therefore the learned diffeomorphisms have two missing stripes.

Part 3

Invariance-based Analysis and Design

CHAPTER 14

Canonization

As we have seen, most of the complexity of the bootstrapping problem comes from the requirement that the behavior of an optimal agent should be invariant to a change in the representation. This was formalized in the previous sections with representation nuisances, transformations that change the data representation in a causally reversible way. One way for an agent to deal with nuisances is to be able to find an invariant representation of the world from the observed data. The scenario is shown geometrically in Figure 14.1: the group nuisance turns one point into any point of the orbit. Finding an invariant representation means being able to choose a "canonical" representative for each orbit.

The problem of finding invariant representations to group nuisances appears in many other contexts. For example, in reference to artificial and natural vision problems, Poggio et al. [51] argue that the stimulus variability is mostly due to the nuisances, rather than the object variability, and therefore normalizing the nuisances is the dominant problem of

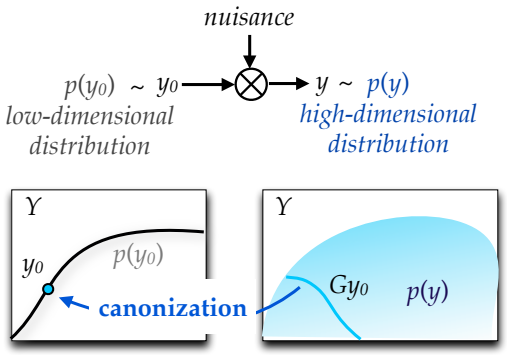


Figure 14.1. A group nuisance transforms nondeterministically the original data x_0 to an element of the orbit $G \cdot x_0$. The problem of finding an invariant representation consists in finding a procedure that is able to associate a "canonical" representative of the orbit starting from any element of the orbit.

computer vision.

While invariance as a theme is found in several other fields, the bootstrapping problem is more complicated in several respects that motivate our treatment. Firstly, the agent behavior must be simultaneously *invariant* with respect to the representation nuisances acting on the observations and *contra-variant* with respect to the nuisances acting on the commands. Instead, usually in passive problems such as object detection there is only one kind of invariance. Secondly, in the literature the groups are usually clearly defined and relatively small. For example, in classical mathematical statistics, most work went into deriving estimators equivariant to location-scale groups (e.g., [135]). In pattern recognition, one is concerned with invariance to Euclidean motions. In bootstrapping we are interested in much larger groups. Instead of an ad hoc approach for particular groups, we are led to considering systematic approaches to analysis and synthesis that allow modular solutions.

Outline: This part of the thesis introduces some analysis and synthesis tools that are of general interest beyond their application to bootstrapping.

Chapter 15 describes "group-spectral dossiers", which summarize the invariance properties of a mapping with respect to group actions, such as invariance and equivariance properties. These dossiers make it easy to analyze the properties of mapping compositions and inversions.

Chapter 16 describes "pontifical features", one possible way to obtain a canonical representation. Roughly speaking, strong pontifical features are functions transversal to the orbits of a group, which allow to choose one representative from each orbit. These were introduced by Soatto [52]. Finding one pontifical feature for a large group is challenging or computationally prohibitive. The idea is to study several classes of "weaker" pontifical features, each able to canonize a smaller subgroup. A taxonomy of features arises naturally according to the way that the features interact with each other.

Chapter 17 describes the algebra of pontifical features and their induced canonization operators.

CHAPTER 15

Group-spectral Dossiers

A "group-spectral dossier" summarizes how a map $f : \mathcal{X} \rightrightarrows \mathcal{Y}$ interacts with groups acting on $\mathcal{X}$ and $\mathcal{Y}$. The information in the dossier helps in reasoning about mapping composition and inversion.

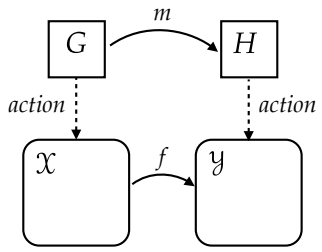


Figure 15.1. A group-spectral dossier summarizes how a function f between two sets $\mathcal{X}$ and $\mathcal{Y}$ interacts with groups acting on those sets (G acting on $\mathcal{X}$, and H acting on $\mathcal{Y}$). In some cases, it is possible to prove that f respects the structure of the group actions, in the sense that there exists a homomorphism $m : G \to H$ such that $f(g \cdot x) = m(g) \cdot f(x), \forall g \in G,\ x \in \mathcal{X}$.

15.1. Group-spectral Dossiers

DEFINITION 15.1 (Group-spectral dossier). The *group-spectral dossier* of a function $f : \mathcal{X} \rightrightarrows \mathcal{Y}$ is a "dossier" containing the following information:

(1) A description of Domain(f) and Codomain(f).

(2) A set of tuples HomMaps(f), which describe the way that f interacts with groups acting on $\mathcal{X}$ and $\mathcal{Y}$ (Definition 15.2).

For the case where domain and codomain coincide, we annotate in the dossier two other pieces of information:

(1) $\mathrm{EqSet}(f)$ is the set of fixed points of f:

$$\mathrm{EqSet}(f) = \{x \mid f(x) = x\}.$$

(2) $\mathrm{GrAct}(f)$ is a set of groups acting on $\mathcal{X}$. This describes whether the function can be written as a group action, in the sense that $G \in \mathrm{GrAct}(f)$ if there exists a group G acting on $\mathcal{X}$, and a map $\alpha : \mathcal{X} \to G$ such that $f(x) = \alpha(x) \cdot x$.

It is useful to summarize the dossier in a table, as shown in Figure 15.1.

DEFINITION 15.2 ($\mathrm{HomMaps}(f)$). Given a function $f : \mathcal{X} \rightrightarrows \mathcal{Y}$, the set $\mathrm{HomMaps}(f)$ is a set of three-terms tuples which describe the invariance properties of f. Each tuple is of the form $\langle G, m, H\rangle$, where the group G acts on $\mathcal{X}$, the group H acts on $\mathcal{Y}$, and m is a function from G to H.

$\boxed{\langle G, m, H\rangle} \in \mathrm{HomMaps}(f)$ The tuple $\langle G, m, H\rangle$, where G is a group acting on $\mathcal{X}$, H is a group acting on $\mathcal{Y}$, and $m : G \to H$, belongs to $\mathrm{HomMaps}(f)$ if

$$f(g \cdot x) = m(g) \cdot f(x), \qquad \forall g \in G,\ x \in X. \tag{15.1}$$

and m is a homomorphism between G and H. There are many interesting special cases, as described below.

$\boxed{\langle G, \sim, H\rangle} \in \mathrm{HomMaps}(f)$ The tuple $\langle G, \sim, H\rangle$ belongs to $\mathrm{HomMaps}(f)$ if G is a group acting on X, H is a group acting on Y, and it holds that (15.1) for some $\gamma : G \to H$ that is not a group homomorphism.

$\boxed{\langle G, \sim, \sim\rangle} \in \mathrm{HomMaps}(f)$ The tuple $\langle G, \sim, \sim\rangle$ belongs to $\mathrm{HomMaps}(f)$ if the resulting transformation is not a group.

$\mathrm{Domain}(f)$	$\overset{f}{\to}$	$\mathrm{Codomain}(f)$		
$\mathrm{EqSet}(f)$	$\equiv$	$\mathrm{EqSet}(f)$	$\mathrm{HomMaps}(f)$	
G	$\overset{m}{\to}$	H	$\langle G, m, H\rangle$	The action of G on x results in the action of H on $f(x)$.
G	$\overset{0}{\to}$	Id	$\langle G, 0, \mathsf{Id}\rangle$	(a special case: invariance)
G	$\overset{\mathsf{Id}}{\to}$	G	$\langle G, \mathsf{Id}, G\rangle$	(a special case: equivariance)
G	$\overset{\cdot^{-1}}{\to}$	G	$\langle G, \cdot^{-1}, G\rangle$	(a special case: contravariance)
Id	$\overset{*}{\to}$	H	$\langle \mathsf{Id}, *, H\rangle$	(a special case: a nuisance is introduced)
G	$\overset{\sim}{\to}$	H	$\langle G, \sim, H\rangle$	The result is a group, but not through a homomorphism.
G	$\overset{\sim}{\to}$	$\sim$	$\langle G, \sim, \sim\rangle$	The result is not a group.
	$\parallel G$		$G \in \mathrm{GrAct}(f)$	The function can be written as the action of G.

Figure 15.1. Tabular representation of the group-spectral dossier of a function f. Note that the notation is deficient: G and H are supposed to be acting on $\mathrm{Domain}(f)$ and $\mathrm{Codomain}(f)$, respectively, but the table does not show what is the group action.

As for the cases $\boxed{\langle G, m, H\rangle} \in \mathrm{HomMaps}(f)$, there are a few special cases worth mentioning:

$\boxed{\langle G, 0, \mathsf{Id}\rangle} \in \mathrm{HomMaps}(f)$ denotes invariance of f to the action of G.

$\boxed{\langle G, \mathsf{Id}, G\rangle} \in \mathrm{HomMaps}(f)$ expresses the equivariance of f to the action of G.

$\boxed{\langle \mathsf{Id}, *, H\rangle} \in \mathrm{HomMaps}(f)$ denotes that f introduces a group nuisance H.

15.2. Examples

EXAMPLE 15.3 (Invariant functions). Figure 15.1 shows the group-spectral dossier of a generic function $f : \mathcal{X} \to \mathcal{Y}$ which is invariant to the action of a group G. Note the definition of the right inverse f^{-1}, which maps $\mathcal{Y}$ to the equivalence classes of f in $\mathcal{X}$.

$$\begin{array}{ccc} \mathcal{X} & \xrightarrow{f} & \mathcal{Y} \\ \hline G & \xrightarrow{0} & \mathsf{Id} \end{array}$$

(a)

$$\begin{array}{ccc} & \overset{f^{-1}}{\rightrightarrows} & \\ \mathcal{Y} & & \mathcal{X} \\ \hline \mathsf{Id} & \xrightarrow{\star} & G \end{array}$$

(b)

Figure 15.1. Group-spectral dossier of an invariant function.

EXAMPLE 15.4 (Determinant). This example shows that there could be different group-spectral dossiers for the same function, with some being more useful than others. Consider the matrix determinant, which associates a real number to each square matrix:

$$\det : \mathbb{R}^{n\times n} \to \mathbb{R}.$$

One property of the determinant is that the determinant of the product is the product of the determinants:

$$\det(AB) = \det(A)\det(B). \tag{15.1}$$

Consider now the orthogonal group $\mathsf{O}(n)$ (Definition D.2), the set of square matrices whose rows and columns are orthonormal:

$$\mathsf{O}(n) = \{X \in \mathbb{R}^{n\times n} \mid XX^T = X^TX = I\}.$$

It is easy to see that the determinant of an orthogonal matrix is either $+1$ or -1.

Using the property (15.1), we obtain that

$$\det(XA) = \det(X)\det(A), \qquad \forall X \in \mathsf{O}(n), A \in \mathbb{R}^{n\times n}.$$

This relation should be interpreted as (15.1):

$$f(g\cdot x) \;\; = \;\; m(g)\cdot f(x), \qquad \forall g \in G, x \in \mathcal{X}, \tag{15.2}$$

where $\mathcal{X} = \mathbb{R}^{n\times n}$, $G = \mathsf{O}(n)$, $H = (\pm 1, \times)$, $f = \det$, $m = \det$. Moreover, m is a group homomorphism (Lemma D.15). The action on the left side is the action of $\mathsf{O}(n)$ on $\mathbb{R}^{n\times n}$; the action on the right side is the action of $(\mathbb{R}_\circ, \times)$ on $\mathbb{R}$. From all of this we can conclude that $\langle \mathsf{O}(n), \det, (\pm 1, \times)\rangle \in \text{HomMaps}(\det)$. This is expressed by line (1) in Figure 15.2b. (Note that it is incidental that, in this particular simple case, $f = m$.)

We can take the example one step further. The orthogonal matrices with determinant $+1$ form the special orthogonal group $\mathsf{SO}(n)$:

$$\mathsf{SO}(n) = \{X \in \mathsf{O}(n) \mid \det(X) = +1\}.$$

The determinant is invariant to the action of $\mathsf{SO}(n)$. This is shown explicitly in Figure 15.2b. What happens for the rest of $\mathsf{O}(n)$? Call $\mathsf{SO}^-(n)$ the rest of the elements in $\mathsf{O}(n)$:

$$\mathsf{SO}^-(n) = \{X \in \mathsf{O}(n) \mid \det(X) = -1\}.$$

The set $\mathsf{SO}^-(n)$ is not a group under the operation of matrix multiplication. But the two sets $\{\mathsf{SO}^-(n), \mathsf{SO}(n)\}$ form a group under the operation of subgroup product (Definition C.7), according to the multiplication table shown in Table 15.1a. Not incidentally, this is the same multiplication table of $(\pm 1, \times)$, shown in Table 15.1b. The two groups are isomorphic. Moreover, their actions (on $\mathbb{R}^{n\times n}$ and $\mathbb{R}$, respectively) are isomorphic as well, according to (15.2). This justifies line (2) in Figure 15.2b.

$\mathbb{R}^{n\times n}$	$\overset{\det}{\to}$	$\mathbb{R}$
$\mathsf{O}(n)$	$\overset{\det}{\to}$	$(\pm 1, \times)$

(a)

$\mathbb{R}^{n\times n}$	$\overset{\det}{\to}$	$\mathbb{R}$	
$\mathsf{SO}(n)$	$\overset{0}{\to}$	Id	(1)
$\{\mathsf{SO}^-(n), \mathsf{SO}(n)\}$	$\overset{\mathsf{Id}}{\to}$	$(\pm 1, \times)$	(2)

(b)

Figure 15.2. Two group-spectral dossier for the function det, looking at different properties of the function.

Table 15.1. Multiplication table for $\{\mathsf{SO}^-(n), \mathsf{SO}(n)\}$ and the operation of subgroup product, and its isomorphic equivalent $(\pm 1, \times)$.

(a)

(subgroup product)	$\mathsf{SO}(n)$	$\mathsf{SO}^-(n)$
$\mathsf{SO}(n)$	$\mathsf{SO}(n)$	$\mathsf{SO}^-(n)$
$\mathsf{SO}^-(n)$	$\mathsf{SO}^-(n)$	$\mathsf{SO}(n)$

(b)

$\times$	$+1$	-1
$+1$	$+1$	-1
-1	-1	$+1$

EXAMPLE 15.5 (Group nuisance). Define $\text{sample}_G : \mathcal{X} \rightrightarrows \mathcal{X}$ as the one-to-many function that maps a point x to the orbit $G \cdot x$. This function can represent a random nuisance acting on the data. Here, we are using a one-to-many function to model nondeterministic uncertainty. Figure 15.3a shows the dossier for sample_G. Line (1) expresses the fact that the output of the function is an orbit of G. Line (2) expresses the fact that this function is also invariant to G:

$$\text{sample}_G(g \cdot x) = G \cdot g \cdot x = (Gg) \cdot x = G \cdot x = \text{sample}_G(x).$$

We note in passing that the right inverse has the same group-spectral dossier (Figure 15.3b).

sample_G

$\mathcal{X}$	$\rightrightarrows$	$\mathcal{X}$	
Id	$\overset{\star}{\to}$	G	(1)
G	$\overset{0}{\to}$	Id	(2)

(a)

$(\text{sample}_G)^{-1}$

$\mathcal{X}$	$\rightrightarrows$	$\mathcal{X}$
Id	$\overset{*}{\to}$	G
G	$\overset{0}{\to}$	Id

(b)

Figure 15.3. The group-spectral dossier of a random nuisance. Note that the function and its inverse have the same signature.

15.3. Dossier for the Inverse of a Mapping

Suppose that we have a group-spectral dossier of the mapping f; what can we say about the group-spectral dossier of the right inverse f^{-1}? What we find is that we just need to "reverse" the table.

$\mathrm{Domain}(f)$	$\overset{f}{\to}$	$\mathrm{Codomain}(f)$
G	$\overset{m}{\to}$	H
G	$\overset{0}{\to}$	Id
Id	$\overset{*}{\to}$	H
G	$\overset{\mathsf{Id}}{\to}$	G
G	$\overset{\sim}{\to}$	H
G	$\overset{\sim}{\to}$	$\backsim$

$\overset{-1}{\Rightarrow}$

$\mathrm{Codomain}(f)$	$\overset{f^{-1}}{\rightrightarrows}$	$\mathrm{Domain}(f)$
H	$\overset{m^{-1}}{\to}$	$G/\ker m$
Id	$\overset{*}{\to}$	G
H	$\overset{0}{\to}$	Id
G	$\overset{\mathsf{Id}}{\to}$	G
H	$\overset{\sim}{\to}$	G
	n/a	

Figure 15.1. Relation between the group-spectral dossier of a function f and of its right inverse f^{-1}.

PROPOSITION 15.6 (Group-spectral dossier for right-inverse). *Let $f : \mathcal{X} \rightrightarrows \mathcal{Y}$. Then the group-spectral dossier of f^{-1} can be described from the group-spectral dossier of f, according to the following rules:*

(1) *Domain*$(f^{-1}) =$ *Codomain*(f).

(2) *Codomain*$(f^{-1}) =$ *Domain*$(f)/f$.

(3) *As for* HomMaps(f):

   (a) *If* $\langle G, m, H\rangle \in$ *HomMaps*(f), *then* $\langle H, m^{-1}, G/\ker m\rangle \in$ *HomMaps*(f^{-1}).

   (b) *If* $\langle G, \sim, H\rangle \in$ *HomMaps*(f), *then* $\langle H, \sim, G\rangle \in$ *HomMaps*(f^{-1}).

   (c) *In general,* $\langle G, \backsim, \backsim\rangle \in$ *HomMaps*(f) *does not allow to conclude anything about* HomMaps(f).

PROOF. Points 1 and 2 follow directly from the definition of right-inverse. Point 3a follows from Lemma 15.7 □

LEMMA 15.7. *Suppose $f : \mathcal{X} \rightrightarrows \mathcal{Y}$, the group G acts on $\mathcal{X}$, the group H acts on $\mathcal{Y}$, and there exists a homomorphism $m : G \to H$ such that*

$$f(g \cdot x) = m(g) \cdot f(x), \qquad \forall g \in G, x \in \mathcal{X}. \tag{15.1}$$

Then the group $G/\ker m$ acts on the equivalence classes $\mathcal{X}/f$ by an operation "$\circ : (G/\ker m) \times (\mathcal{X}/f) \to (\mathcal{X}/f)$.

Moreover, call $m^{-1} : H \to G/\ker m$ the right-inverse of m. Then it holds that

$$f^{-1}(h \cdot y) = m^{-1}(h) \circ f^{-1}(y), \qquad \forall h \in H, y \in \mathit{Codomain}(f). \tag{15.2}$$

PROOF. Because m is a homomorphism, the group $N = \ker m$ is well defined, and it is a normal subgroup (Lemma C.21). This makes the quotient group $Q = G/\ker m$ well defined. Formally, the group Q is the set of cosets of $N = \ker m$:

$$Q = \{Ng \mid g \in G\}.$$

The group operation is $(Ng)(Nh) = NNgh = (Ngh)$. (By Lemma C.12, left and right cosets coincide.)

The set $\mathcal{X}/f$ is the set of equivalence classes induced by f. A value $y \in \text{Codomain}(f)$ can be used to index the equivalence classes. We define $c_y = \{x \in \mathcal{X} \mid f(x) = y\} \in \mathcal{X}/f$.

We want to show that Q acts on $\mathcal{X}/f$. We already know that G acts on $\mathcal{X}$, and have well defined $g \cdot x$. We need to find an operation $\circ : Q \times \mathcal{X}/f \to \mathcal{X}/f$ which satisfies the properties in Definition C.35 (closedness, identity, transitivity). Consider the operation $\circ$

---

normal subgroup: A normal subgroup is a subgroup that commutes with all elements of the group. See Definition C.11.

coset: A subset of a group that can be written as the product of a subgroup and an element of the group. See Definition C.6.

defined as

$$\begin{aligned} \circ : (G/\ker m) \times (\mathcal{X}/f) &\rightarrow \mathcal{X}/f, \\ (Ng, c_y) &\mapsto g \cdot c_y. \end{aligned}$$

Identity The identity property is satisfied because the equivalence classes are invariant with respect to the action of N: $N \otimes c_y = c_y$. From (15.1), it follows that f is invariant to $N = \ker m$:

$$f(n \cdot x) = f(x), \quad \forall n \in N, x \in \mathcal{X}. \tag{15.3}$$

Let $x \in c_y$. Then $f(N \otimes x) = f(x) = y$ from (15.3), which implies that $N \otimes x \in c_y$.

Closedness The operation must be closed (i.e., the codomain must be $\mathcal{X}/f$), which means that

$$(Ng) \circ c_y = g \cdot c_y \in \mathcal{X}/f.$$

All elements of $g \cdot c_y$ map to the same value:

$$\begin{aligned} f(g \cdot c_y) &= f\left(g \cdot \{x \in \mathcal{X} \mid f(x) = y\}\right) \\ &= f\left(\{g \cdot x \in \mathcal{X} \mid f(x) = y\}\right) \\ &= \{f(g \cdot x) \mid x \in \mathcal{X} \wedge f(x) = y\} \\ &= \{m(g) \cdot f(x) \mid x \in \mathcal{X} \wedge f(x) = y\} \\ &= m(g) \cdot y. \end{aligned}$$

Therefore, $(Ng) \circ c_y = c_{m(g) \cdot y} \in \mathcal{X}/f$.

Transitivity Then, we show transitivity:

$$\begin{aligned}
(Ng) \circ \big((Nh) \circ c_y\big) &= (Ng) \circ c_{m(h)\cdot y} \\
&= c_{m(g)\cdot m(h)\cdot y} \\
&= c_{(m(g)m(h))\cdot y} \\
&= c_{m(gh)\cdot y} \\
&= (Ngh) \circ c_y.
\end{aligned}$$

This establishes the first part.

Let m^{-1} be the right inverse of m: $m^{-1}(h) = \{g \in G \mid m(g) = h\}$. By the definition of kernel, it follows that

$$m^{-1}(h) = Nm^{-1}(h). \tag{15.4}$$

At this point one can verify that

$$\begin{aligned}
f^{-1}(h \cdot y) &= c_{h\cdot y} \\
&= (Nm^{-1}(h)) \circ c_y \\
&\qquad \text{(From the equation above)} \\
&= (m^{-1}(h)) \circ c_y \\
&= m^{-1}(h) \circ f^{-1}(y),
\end{aligned}$$

which proves (15.2). □

CHAPTER 16

Pontifical Features and Canonization Operators

In this section we study pontifical features, *which are useful to define invariant representations. With respect to previous work, we make a more careful classification of such features and the* canonization operators *they induce. We discuss several classes of features, each characterized by their strength: unstructured* $\Leftarrow$ *weak* $\Leftarrow$ *mild* $\Leftarrow$ *bold* $\Leftarrow$ *strong. Each feature class induces a different canonization operators, which are described by their group-spectral dossiers.*

16.1. A Hierarchy of Features

In general, a *feature* for the space $\mathcal{X}$ is simply a function φ on $\mathcal{X}$ that has some special properties. One way that features are used are to identify a subset of the space which is "canonical" in some way.

DEFINITION 16.1 (Canonical point). A point $x \in X$ is *canonical* for a feature φ if it satisfies $\varphi(x) = 0$.

Features are mainly used to describe interesting subsets of the space. Sometimes we use the name of the feature as a proxy for the subset $\{x \in \mathcal{X} \mid \varphi(x) = 0\}$. For example, given a feature φ on $\mathcal{X}$, and a subset $\mathcal{S} \subset \mathcal{X}$, $\varphi \cap S$ is the set $\{x \in \mathcal{S} \mid \varphi(x) = 0\}$.

Strong pontifical features determine a *strong canonization operator*that is able to canonize the action of a group. However, these are particular rare to find, especially for large groups, or the computation involved might be prohibitive. This motivates the study of weaker features.

The *structured* pontifical features *(bold, mild, and weak)* determine the group nuisance which acted on the data only up to a heresy subgroup $H \leq G$. There is a further classification according to the relations between H and G.

- For weak pontifical features, we only know $H \leq G$. The weak canonization operator maps a point to the orbit of the entire H coset.
- For mild pontifical features, H is a normal subgroup of G. This allows to define the quotient group G/H which is used to define a 1-to-1 mild canonization operator.
- For bold pontifical features, we know that $G \cong H \times G_0$. These induce a *bold canonization operator* that commutes with the action of H.

Unstructured pontifical features normalize only a subgroup of G, but do not have the structure of a weak pontifical feature.

The features classes are summarized in Table 16.2.

Table 16.1. Symbols introduced in Chapter 16

φ	$: \mathcal{X} \to \mathbb{R}$	Symbol used for features.
$\hat{g}_x^\varphi$	$\in G$	The solution of the feature equation $\varphi(g \cdot x) = 0$.
$\hat{G}_x^\varphi$	$\subset G$	The set of solutions, if the solution is not unique.
SCan_G^φ		strong canonization operator
$\mathrm{BCan}_{G/H}^\varphi$		bold canonization operator
$\mathrm{MCan}_{G/H}^\varphi$		mild canonization operator
$\mathrm{WCan}_{G,H}^\varphi$		weak canonization operator
UCan^φ		unstructured canonization operator

feature class	*symmetries of* φ
strong pontifical feature	Id
bold pontifical feature	$H \triangleleft G$, $G/H \triangleleft G$
mild pontifical feature	$H \triangleleft G$
weak pontifical feature	$H \leq G$
unstructured pontifical feature	*unstructured*

Table 16.2. Classes of pontifical features and relative canonization operators.

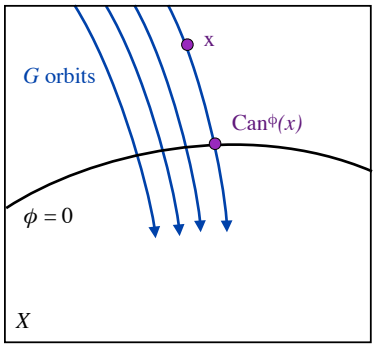


Figure 16.1. Geometry of a strong pontifical feature. The feature is transversal to the orbits of the group action.

16.2. Strong Pontifical Features

Strong pontifical features are one class of features that can be used to find a canonical representation of the data in spite of a group acting on the data as a nuisance.

DEFINITION 16.2 (Strong pontifical feature [52]). Let G be a group acting on a set $\mathcal{X}$. A *strong pontifical feature* for G is a function $\varphi : \mathcal{X} \to \mathbb{R}$ such that, for all $x \in \mathcal{X}$, the equation $\varphi(g \cdot x) = 0$ has exactly one solution for $g \in G$, which we denote as $\hat{g}_x^\varphi$.

16.2.1. *Strong Canonization Operators*

A strong pontifical feature gives a way to remove the effect of a group nuisance by inducing a well-defined canonization operator.

DEFINITION 16.3 (Strong canonization operator). Given a strong pontifical feature $\varphi : \mathcal{X} \to \mathbb{R}$ for the group G acting on $\mathcal{X}$, the *strong canonization operator* for φ is the map

$$\begin{aligned} \mathrm{SCan}_G^\varphi : \mathcal{X} &\to \mathcal{X} \\ x &\mapsto \hat{g}_x^\varphi \cdot x. \end{aligned}$$

LEMMA 16.4. $\hat{g}_{h\cdot x}^\varphi = \hat{g}_x^\varphi h^{-1}$.

PROOF. Consider the equation $\varphi(g \cdot (h \cdot x)) = 0$. Because φ is a strong pontifical feature, this has one solution for g, given by $g = \hat{g}_{h\cdot x}^\varphi$. The equation can be written also as $\varphi((gh) \cdot x) = \varphi(k \cdot x) = 0$, which has only one solution for k, given by $k = \hat{g}_x^\varphi$. We obtain $gh = \hat{g}_x^\varphi$, hence $g = \hat{g}_x^\varphi h^{-1}$. From $g = \hat{g}_{h\cdot x}^\varphi$ we obtain the result. □

PROPOSITION 16.5 (*Properties of the strong canonization operator*).

(1) It is invariant to G.

(2) The output is canonical: $\varphi \circ \mathrm{SCan}_G^\varphi = 0$.

(3) It is the identity on $\varphi \cap X$.

Figure 16.1 shows the tabular representation of the group-spectral properties of a strong canonization operator.

16.2.2. *Examples of Strong Pontifical Features*

In this first example, we suppose that the objects in $\mathcal{X}$ are sequences of real numbers and the group is $\mathbb{R}$ with addition (Definition D.1). This can be thought as an unknown bias which is added to the sequence. The first feature considered is the expected value of the sequence: $\varphi(x) = \mathbb{E}\{x\}$.

PROPOSITION 16.6. *For any $c \in \mathbb{R}$, $\mathbb{E}\{x\} = c$ is a strong pontifical feature for $G = (\mathbb{R}, +)$.*

PROOF. The equation $\varphi(g \cdot x) = c$ is $\mathbb{E}\{x + g\} = c$, which has only one solution for $g = c - \mathbb{E}\{x\}$. Therefore, the feature is a strong pontifical feature. The relative strong canonization operator consists in removing the mean of the sequence. □

The next two examples consider the group $(\mathbb{R}_\circ^+, \times)$ of strictly positive numbers (Definition D.1) acting on sequences of real numbers. The examples show two different strong pontifical features for the same group on the same space.

$$\begin{array}{ccc} & \mathrm{SCan}_G^\varphi & \\ \mathcal{X} & \to & \varphi \cap \mathcal{X} \\ \varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\ \hline G & \overset{0}{\to} & \mathsf{Id} \\ \hline & \| \; G & \end{array}$$

(a) Strong canonization operator.

$$\begin{array}{ccc} & (\mathrm{SCan}_G^\varphi)^{-1} & \\ \varphi \cap \mathcal{X} & \rightrightarrows & \mathcal{X} \\ \varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\ \hline \mathsf{Id} & \overset{\star}{\to} & G \end{array}$$

(b) Right inverse of strong canonization operator

Figure 16.1. The group-spectral dossier of a strong canonization operator.

PROPOSITION 16.7. *For any $c \in \mathbb{R}_\circ^+$, $\mathbb{E}\{x^2\} = c$ is a strong pontifical feature for $G = (\mathbb{R}_\circ^+, \times)$.*

PROPOSITION 16.8. *For any $c \in \mathbb{R}_\circ^+$, $\mathsf{std}\{x\} = c$ is a strong pontifical feature for $G = (\mathbb{R}_\circ^+, \times)$.*

16.3. Strong Canonization Operators Cannot be Simply Composed

The problem with strong pontifical features is that it is difficult to find one feature to canonize the action of a large group G. A simple idea would be to find features for smaller factors of the group G, derive the canonization operator for each subgroup, and then apply the canonization operators in series. This, however, does not always work.

PROPOSITION 16.9. *Suppose φ_1 is a strong pontifical feature for G_1 and that φ_2 is a strong pontifical feature for G_2. Then, the series of the strong canonization operators for φ_1 and φ_2 does not necessarily make the output canonical with respect to $\varphi_1 \wedge \varphi_2$.*

Some counterexamples are shown in Table 16.3, which enumerates different cases for the series of the features defined previously in Proposition 16.6, Proposition 16.8, and Proposition 16.7. For example, the table shows that, if one first normalizes the mean according to $\mathbb{E}\{x\} = 1$, and then scales x to obtain $\mathsf{std}\{x\} = 1$, the final results does not satisfy $\mathbb{E}\{x\} = 1$ in general.

This problem motivates further development of the theory. In the next sections, several classes of "weaker" features are considered.

first step		*second step*		
φ_1	G_1	φ_2	G_2	$\varphi_1 \wedge \varphi_2$ *respected?*
$\mathbb{E}\{x\} = 0$	$(\mathbb{R}, +)$	$\text{std}\{x\} = 1$	$(\mathbb{R}^+_\circ, \times)$	✓
$\mathbb{E}\{x\} = 0$	$(\mathbb{R}, +)$	$\mathbb{E}\{x^2\} = 1$	$(\mathbb{R}^+_\circ, \times)$	✓
$\mathbb{E}\{x\} = 1$	$(\mathbb{R}, +)$	$\text{std}\{x\} = 1$	$(\mathbb{R}^+_\circ, \times)$	✗
$\mathbb{E}\{x\} = 1$	$(\mathbb{R}, +)$	$\mathbb{E}\{x^2\} = 1$	$(\mathbb{R}^+_\circ, \times)$	✗
$\text{std}\{x\} = 1$	$(\mathbb{R}^+_\circ, \times)$	$\mathbb{E}\{x\} = 0$	$(\mathbb{R}, +)$	✓
$\mathbb{E}\{x^2\} = 1$	$(\mathbb{R}^+_\circ, \times)$	$\mathbb{E}\{x\} = 0$	$(\mathbb{R}, +)$	✓
$\text{std}\{x\} = 1$	$(\mathbb{R}^+_\circ, \times)$	$\mathbb{E}\{x\} = 1$	$(\mathbb{R}, +)$	✓
$\mathbb{E}\{x^2\} = 1$	$(\mathbb{R}^+_\circ, \times)$	$\mathbb{E}\{x\} = 1$	$(\mathbb{R}, +)$	✓

Table 16.3. The series of two strong canonization operators does not necessarily produce an output canonical with respect to both φ_1 and φ_2.

16.4. Weak Pontifical Features

A weak pontifical feature is a function that does not "detect" the action of the entire group G, but it is invariant to a subgroup $H \leq G$, which we call "heresy subgroup".

DEFINITION 16.10 (Weak pontifical feature). Let G be a group acting on $\mathcal{X}$. A *weak pontifical feature* is a map $\varphi : \mathcal{X} \to \mathbb{R}$ such that, for all $x \in \mathcal{X}$, the set $\hat{G}^\varphi_x = \{g \mid \varphi(g \cdot x) = 0\}$ is a left coset of G for a subgroup $H \leq G$, called "heresy subgroup", which does not depend on x; that is, $\hat{G}^\varphi_x = H\hat{g}^\varphi_x$.

A weak pontifical feature is not strong enough to determine exactly which group element acted as a nuisance on the data. But it can be used to define a "weak canonization

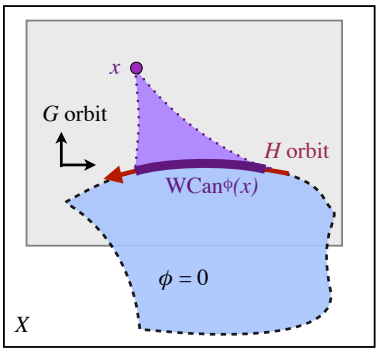


Figure 16.1. The geometry of a weak pontifical feature. A weak pontifical feature determines the nuisance only up to a subgroup H. Because we do not have further information to choose a particular element of H, we define the weak canonization operator as a one-to-many map that maps a point to an H-orbit.

operator", a one-to-many function which maps a point $x \in \mathcal{X}$ to a whole orbit of H.

DEFINITION 16.11 (Weak canonization operator). Given a weak pontifical feature $\varphi : \mathcal{X} \to \mathbb{R}$ for the group G acting on $\mathcal{X}$ with heresy group H, the *weak canonization operator* for φ is the map $\mathrm{WCan}^{\varphi}_{G,H}$, defined as

$$\begin{aligned} \mathrm{WCan}^{\varphi}_{G,H} : \mathcal{X} &\rightrightarrows \mathcal{X} \\ x &\mapsto \hat{G}^{\varphi}_{x} \cdot x. \end{aligned}$$

PROPOSITION 16.12 (*Properties of the weak canonization operator*).

(1) It is invariant to G.

(2) The output is canonical: $\varphi \circ \mathrm{WCan}^{\varphi}_{G,H} = 0$.

16.4.1. *Examples of Weak Pontifical Features*

In this next example, we still work with sequences of real numbers. The group acting on the sequences is $(\mathbb{R}_{\circ}, \times)$ (nonzero real numbers; see Definition D.1). The group acts by scaling a sequence x: $g \cdot x = gx$.

LEMMA 16.13. *For any $c > 0$, $\mathsf{std}\{x\} = \mathbb{E}\{(x - \mathbb{E}\{x\})^2\}^{1/2} = c$ is a weak pontifical feature for $G = (\mathbb{R}_{\circ}, \times)$ with heresy subgroup $H = (\pm 1, \times)$.*

$\mathrm{WCan}^{\varphi}_{G}$

$\mathcal{X}$	$\rightrightarrows$	$\varphi \cap \mathcal{X}$
$\varphi \cap \mathcal{X}$	$\equiv$	$\varphi \cap \mathcal{X}$
G	$\xrightarrow{0}$	Id
Id	$\xrightarrow{\star}$	H

(a) Weak canonization operator.

$(\mathrm{WCan}^{\varphi}_{G})^{-1}$

$\varphi \cap \mathcal{X}$	$\rightrightarrows$	$\mathcal{X}$
$\varphi \cap \mathcal{X}$	$\equiv$	$\varphi \cap \mathcal{X}$
Id	$\xrightarrow{\star}$	G
H	$\xrightarrow{0}$	Id

(b) ... and its right inverse.

Figure 16.2. The group-spectral dossier for the weak canonization operator and its right inverse.

PROOF. The equation $\varphi(g \cdot x) = 0$ is written as

$$\begin{aligned} \mathbb{E}\{(gx - \mathbb{E}\{gx\})^2\} &= \\ g^2\mathbb{E}\{(x - \mathbb{E}\{x\})^2\} &= 1. \end{aligned}$$

Therefore, there are two solutions for g: the scale is not completely constrained by the variance. Therefore, the feature is not a strong pontifical feature. The heresy subgroup H is easily seen to be $H = (\pm 1, \times)$. See also Proposition 16.21 for additional properties of this feature. □

LEMMA 16.14 (Power). *For any $c > 0$, $\mathbb{E}\{x^2\} = c$ is a weak pontifical feature for $G = (\mathbb{R}_\circ, \times)$ with heresy subgroup $H = (\pm 1, \times)$.*

PROOF. The equation $\varphi(g \cdot x) = 0$ is written as $g^2\mathbb{E}\{x^2\} = c$. Therefore, there are, again, two solutions for g. The heresy subgroup H is again $H = \{\pm 1, \times\}$. □

LEMMA 16.15. *$\mathbb{E}\{x\} = 1$ is a weak pontifical feature for $G = \mathsf{Aff}(1, \mathbb{R})$, with heresy subgroup $H = \left\{ \left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix} \right) \mid s \in \mathbb{R}_\circ \right\} \leq G$.*

PROOF. The equation $\varphi(g \cdot x) = 0$ is written as $\mathbb{E}\{ax + b\} = 1$, from which it follows that

$$a\mathbb{E}\{x\} + b = 1.$$

One particular solution is $g = (1, -\mathbb{E}\{x\} + 1)$, corresponding to translation. The complete set of solutions can be written as Hg, where H is of the form

$$H = \{(1, +1)(s, 0)(1, -1) \mid s \in \mathbb{R}_\circ\}.$$

This corresponds to translating to the origin, scaling, and then translating back. More formally, we can show that the set H is a subgroup of $\mathsf{Aff}(\mathbb{R})$. We already know $H \subset$

$\mathsf{Aff}(\mathbb{R})$, so we just need to show that it is closed with respect to composition. Elements of H can be written as $\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)$, for some $s \in \mathbb{R}_\circ$:

$$\begin{aligned} H &= \left\{ \left(\begin{smallmatrix} 1 & +1 \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} s & 0 \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} 1 & -1 \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ \right\} \\ &= \left\{ \left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ \right\}. \end{aligned}$$

This set can be verified to be a subgroup of $\mathsf{Aff}(\mathbb{R})$ (Lemma D.14). □

LEMMA 16.16. $\mathbb{E}\{x\} = 0$ *is a weak pontifical feature for* $G = \mathsf{Aff}(1, \mathbb{R})$*, with heresy subgroup* $H = (\mathbb{R}_\circ, \times) \leq G$.

PROOF. The equation $\varphi(g \cdot x) = 0$ is written as $\mathbb{E}\{ax + b\} = 0$, from which it follows that

$$a\mathbb{E}\{x\} + b = 0.$$

Therefore, the set of solutions are of the form

$$\hat{G} = \{(a, -a\mu) \mid a \in \mathbb{R}_\circ\}.$$

Note that $\hat{G}$ itself is not a subgroup of $\mathsf{Aff}(1, \mathbb{R})$. Consider two of these elements: $(a_1, -a_1\mu)$ and $(a_2, -a_2\mu)$. The composition gives $(a_1a_2, a_2(-a_1\mu - a_2\mu))$, which is not of the form $(a, -a\mu)$, unless for $\mu = 0$. Therefore, $\hat{G}$ is not a subgroup of $\mathsf{Aff}(1, \mathbb{R})$.

However, it is a left coset, because it can be written as $\hat{G} = H \cdot g$, with $g \in (0, -\mu)$ and $H = \{(a, 0), a \in \mathbb{R}_\circ\} \cong (\mathbb{R}_\circ, \times)$. Because $(\mathbb{R}_\circ, \times) \leq \mathsf{Aff}(1, \mathbb{R})$ but not $(\mathbb{R}_\circ, \times) \triangleleft \mathsf{Aff}(1, \mathbb{R})$ (Lemma D.11), the feature is only a weak pontifical feature. □

LEMMA 16.17 (Continued from Lemma 16.28). $\mathbb{E}\{x^2\} = 1$ *is <u>not</u> a weak pontifical feature for* $G = \mathsf{Aff}(1, \mathbb{R})$.

PROOF. The equation $\varphi(g \cdot x) = 0$ is written as

$$\begin{aligned} \mathbb{E}\{(ax+b)^2\} &= 1 \\ \mathbb{E}\{a^2x^2 + b^2 + 2axb\} &= 1 \\ a^2\mathbb{E}\{x^2\} + b^2 + 2ab\mathbb{E}\{x\} &= 1. \end{aligned}$$

One particular solution is a pure scaling $g = (a, 0)$, with $a = 1/\mathbb{E}\{x^2\}$. Now, assuming $\mathbb{E}\{x^2\} = 1$, examine the set of elements which maintain the feature constant:

$$\begin{aligned} a^2\mathbb{E}\{x^2\} + b^2 + 2ab\mathbb{E}\{x\} &= 1 \\ & \qquad \text{(Assuming the feature is satisfied.)} \\ a^2 + b^2 + 2ab\mathbb{E}\{x\} &= 1 \end{aligned}$$

If we fix a, to find b we need to solve a second-order polynomial $b^2(1) + b(2a\mu) + (a^2 - 1) = 0$, whose solutions are

$$b = -a\mu \pm \sqrt{a^2\mu^2 - 2(a^2 - 1)}.$$

For $a = 1$ (do not scale the signal) one obtains the two solutions $b = 0$ (the identity) as well as $b = -2\mu$ (which corresponds to "mirroring" the signal). In the case $\mu = 0$, the two solutions coincide. For $a = 2$, we obtain the two solutions $b = -2\mu \pm \sqrt{4\mu^2 - 6}$. But for $\mu = 0$, there is no real solution for b. Therefore, this set of solution cannot be written as a coset of a subgroup H independent of x. □

16.5. Mild Pontifical Features

DEFINITION 16.18 (Mild pontifical feature). A *mild pontifical feature* is a weak pontifical feature whose heresy subgroup H is a normal subgroup of G.

There is some more structure that we can use for defining a canonization operator. We recall some basic facts of group theory. See Appendix C for more formal definitions.

Because H is a normal subgroup (written as $H \triangleleft G$, Definition C.11), then the quotient group G/H is well-defined (Definition C.30). Moreover, G is homomorphic to the semidirect product of H and a subgroup $Q \leq G$ which is homomorphic to the quotient G/H (Lemma C.28):

$$G \cong H \rtimes Q \qquad \text{for some } Q \leq G, \text{ with } Q \cong G/H.$$

A graphical representation of the groups G, H, Q and their actions is shown in Figure 16.1.

Any element $g \in G$ can be written uniquely as a product $g = hq$, for $h \in H$ and $q \in Q$ (Lemma C.29). This means that we can define a projection map $\gamma : G \to Q$ as $\gamma : hq \mapsto q$ (Definition C.33). The mild canonization operator is based on this projection map.

DEFINITION 16.19 (Mild canonization operator). Consider a mild pontifical feature $\varphi : \mathcal{X} \to \mathbb{R}$ for the group G acting on $\mathcal{X}$ with heresy subgroup $H \triangleleft G$ and the relative projection

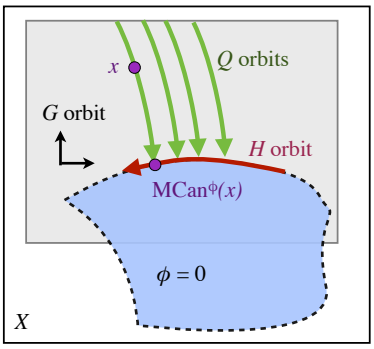


Figure 16.1. Geometry of a mild pontifical feature and relative canonization operator. In this case, we know that the heresy subgroup H is a normal subgroup of G. Therefore, we can define the quotient group Q such that $G \cong H \rtimes Q$. This factorization allows to define the mild canonization operator as an action of Q which brings each point x to the feature surface $\varphi(x) = 0$.

map $\gamma : G \to Q$ (Definition C.33). The *mild canonization operator* for φ is the map

$$\begin{aligned} \mathrm{MCan}^{\varphi}_{G/H} : \mathcal{X} &\to \mathcal{X} \\ x &\mapsto \gamma(\hat{G}^{\varphi}_x) \cdot x. \end{aligned}$$

PROPOSITION 16.20. Properties of $\mathrm{MCan}^{\varphi}_{G/H}$:

(1) It is the action of some group element $q_x \in Q$.

(2) The output is canonical: $\varphi \circ \mathrm{MCan}^{\varphi}_{G/H} = 0$.

(3) It is the identity on $\varphi \cap \mathcal{X}$.

(4) It is invariant to the action of Q.

(5) There exists an endomorphism $m^{\varphi}_x : H \to H$ such that

$$\mathrm{MCan}^{\varphi}_{G/H}(h \cdot x) = m^{\varphi}_x(h) \cdot \mathrm{MCan}^{\varphi}_{G/H}(x).$$

That is, while the operator does *not* commute with the action of H, it transforms its action in a somewhat regular way.

PROOF. We first prove some preliminary results about $\hat{G}^{\varphi}_x$:

- (a) $\hat{G}^{\varphi}_x$ can be written uniquely as $\hat{G}^{\varphi}_x = H\hat{q}^{\varphi}_x$ for some $\hat{q}^{\varphi}_x \in Q$.
- (b) $\hat{q}^{\varphi}_x$ is contra-variant to Q: $\hat{q}^{\varphi}_{q\cdot x} = \hat{q}^{\varphi}_x q^{-1}$.

$$\begin{array}{rcl} & \mathrm{MCan}^{\varphi}_{G/H} & \\ \mathcal{X} & \to & \varphi \cap \mathcal{X} \\ \varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\ \hline G/H & \xrightarrow{0} & \mathsf{Id} \\ H & \xrightarrow{m_x} & H \\ \hline & \| \; G/H & \end{array}$$

(a) Mild canonization operator.

$$\begin{array}{rcl} & (\mathrm{MCan}^{\varphi}_{G/H})^{-1} & \\ \varphi \cap \mathcal{X} & \rightrightarrows & \mathcal{X} \\ \varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\ \hline \mathsf{Id} & \xrightarrow{*} & G/H \\ H & \xrightarrow{m_x^{-1}} & H \end{array}$$

(b) ... and its right inverse.

Figure 16.2. The group-spectral dossier of the mild canonization operator.

- (c) $\hat{q}^{\varphi}_{x}$ is invariant to H: $\hat{q}^{\varphi}_{h\cdot x} = \hat{q}^{\varphi}_{x}$.

Now we prove points 1 and 5 of the Proposition.

(1) This follows from the fact that $\gamma(\hat{G}^{\varphi}_{x}) \in Q$.

(2) Let us compute

$$\begin{aligned}
\varphi(\mathrm{BCan}^{\varphi}_{G/H}(x)) &= \varphi(\gamma(\hat{G}^{\varphi}_{x}) \cdot x) \\
&\quad (\text{Property (a) above.}) \\
&= \varphi(\gamma(H\hat{q}^{\varphi}_{x}) \cdot x) \\
&\quad (\text{Property (1) of Lemma C.34.}) \\
&= \varphi(\gamma(\hat{q}^{\varphi}_{x}) \cdot x) \\
&\quad (\text{Property (2) of Lemma C.34.}) \\
&= \varphi(\hat{q}^{\varphi}_{x} \cdot x) \\
&\quad (\hat{q}^{\varphi}_{x} \in \hat{G}^{\varphi}_{x}) \\
&= 0.
\end{aligned}$$

(3) If $x \in \varphi \cap \mathcal{X}$, then it is already canonical: $\varphi(x) = 0$. Therefore $e \in \hat{G}^{\varphi}_{x}$. By property (a) above we know that every element $g \in \hat{G}^{\varphi}_{x}$ can be written as $g = h\hat{q}^{\varphi}_{x}$, for some $h \in H$ and $\hat{q}^{\varphi}_{x} \in Q$. For $g = e$, we obtain $e = h\hat{q}^{\varphi}_{x}$, which implies $\hat{q}^{\varphi}_{x} = h^{-1} \in Q$, and because Q is a group, $h \in Q$. Therefore $h \in H \cap Q$. But we know that $Q \cap H = e$, we obtain $h = e$, and $\hat{q}^{\varphi}_{x} = h^{-1} = e$. So $\gamma(H\hat{q}^{\varphi}_{x}) = \gamma(\hat{q}^{\varphi}_{x}) = \gamma(e) = e$.

(4) This follows easily from property (b) above. Let us compute:

$$\begin{aligned}
\mathrm{BCan}^{\varphi}_{G/H}(q \cdot x) &= \gamma(\hat{G}^{\varphi}_{q\cdot x}) \cdot (q \cdot x) \\
&\qquad \text{(Property (a) above.)} \\
&= \gamma(H\hat{q}^{\varphi}_{q\cdot x}) \cdot (q \cdot x) \\
&\qquad \text{(Property (b) above.)} \\
&= \gamma(H\hat{q}^{\varphi}_{x} q^{-1}) \cdot (q \cdot x) \\
&\qquad \text{(Property (3) of Lemma C.34.)} \\
&= \gamma(H\hat{q}^{\varphi}_{x}) q^{-1} \cdot (q \cdot x) \\
&= \gamma(H\hat{q}^{\varphi}_{x}) \cdot x \\
&= \mathrm{BCan}^{\varphi}_{G/H}(x).
\end{aligned}$$

(5) Let us compute:

$$\begin{aligned}
\mathrm{MCan}^{\varphi}_{G/H}(h \cdot x) &= \gamma(\hat{G}^{\varphi}_{h\cdot x}) \cdot (h \cdot x) \\
&\qquad \text{(Property (a) above.)} \\
&= \gamma(H\hat{q}^{\varphi}_{h\cdot x}) \cdot (h \cdot x) \\
&\qquad \text{(Property (c) above.)} \\
&= \gamma(H\hat{q}^{\varphi}_{x}) \cdot (h \cdot x) \\
&= \hat{q}^{\varphi}_{x} \cdot (h \cdot x) \\
&= (\hat{q}^{\varphi}_{x} h) \cdot x.
\end{aligned}$$

At this point, if $Q \triangleleft G$ (that is, $G \cong Q \times H$) then the two elements would commute: $\hat{q}_x^\varphi h = h\hat{q}_x^\varphi$. In general, however, this is not true; what we can say is that, because $H \triangleleft G$, then $\hat{q}_x^\varphi h = h'\hat{q}_x^\varphi$ for *some* $h \in H$. In general this h' depends on $\hat{q}_x^\varphi$, and therefore on x: $h' = m_x^\varphi(h) = \hat{q}_x^\varphi h \left(\hat{q}_x^\varphi\right)^{-1}$. That is, it is a conjugation, and therefore an homomorphism of G into G. Because $H \triangleleft G$, H is invariant to conjugation (Lemma C.15), and so $m_x^\varphi : H \to H$ is an Endomorphism. What we obtain is

$$\mathrm{MCan}_{G/H}^\varphi(h \cdot x) = (m_x^\varphi(h)\hat{q}_x^\varphi) \cdot x = m_x^\varphi(h) \cdot \mathrm{MCan}_{G/H}^\varphi(x).$$

□

16.5.1. *Examples of Mild Pontifical Features*

PROPOSITION 16.21 (Continued from Proposition 16.13). *For any $c > 0$, $\mathsf{std}\{x\} = \mathbb{E}\{(x - \mathbb{E}\{x\})^2\}^{1/2} = c$ is a mild pontifical feature for $G = (\mathbb{R}_\circ, \times)$ with heresy subgroup $H = (\pm 1, \times) \triangleleft G$.*

PROOF. From Proposition 16.13 we know that this is a weak pontifical feature. Because $H = (\pm 1, \times) \triangleleft (\mathbb{R}_\circ, \times)$, the feature is also a mild pontifical feature. (See also Proposition 16.25 for additional properties of this feature.) □

PROPOSITION 16.22. *For any $c > 0$, $\mathsf{std}\{x\} = c$ is a mild pontifical feature for $G = \mathsf{Aff}(\mathbb{R})$. The heresy subgroup is $H = (\mathbb{R}, +) \rtimes (\pm 1, \times) \triangleleft G$, and $G/H \cong (\mathbb{R}_\circ^+, \times) \leq G$.*

endomorphism: An homomorphism from a group to itself. See Definition C.19.

PROOF. The equation $\varphi(g \cdot x) = 0$ becomes

$$\begin{aligned} \mathbb{E}\{(ax + b - \mathbb{E}\{ax + b\})^2\} &= \\ \mathbb{E}\{(ax + b - a\mathbb{E}\{x\} - b)^2\} &= \\ a^2\mathbb{E}\{(x - \mathbb{E}\{x\})^2\} &= 1, \end{aligned}$$

which implies that $a = \pm 1/\sigma^2$. Any translation will also conserve the feature. Therefore, the heresy subgroup is $H = (\mathbb{R}, +) \vee (\pm 1, \times)$, which is a normal subgroup of $\mathsf{Aff}(1, \mathbb{R})$ (Lemma D.10). Therefore, this is a mild pontifical feature. However, $\mathsf{Aff}(1, \mathbb{R})/(\mathbb{R}, +) \cong (\mathbb{R}_\circ, \times)$ is not a normal subgroup, so the feature is not a bold pontifical feature. □

16.6. Bold Pontifical Features

We define *bold* pontifical feature as a particular case of mild features.

DEFINITION 16.23 (Bold pontifical feature). A *bold pontifical feature* is a mild pontifical feature for which both the heresy subgroup H and G/H are normal subgroups of G.

In other words, the group G is isomorphic to the direct product $Q \times H$ (Proposition **??**). In this case, we can show that the mild canonization operator is equivariant to the action of H.

DEFINITION 16.24 (Bold canonization operator). The *bold canonization operator* $\mathrm{BCan}^{\varphi}_{G/H}$ is the mild canonization operator for a bold pontifical feature. It enjoys all properties given by Proposition 16.20, plus equivariance:

$$\mathrm{BCan}^{\varphi}_{G/H}(h \cdot x) = h \cdot \mathrm{BCan}^{\varphi}_{G/H}(x).$$

16.6.1. *Examples of Bold Pontifical Features*

PROPOSITION 16.25 (Continued from Proposition 16.21). $\mathsf{std}\{x\} = \mathbb{E}\{(x - \mathbb{E}\{x\})^2\}$ *is a bold pontifical feature for* $G = (\mathbb{R}_\circ, \times)$ *with heresy subgroup* $H = (\pm 1, \times) \triangleleft G$.

PROOF. The quotient group $G/H = (\mathbb{R}_\circ^+, \times)$ is a normal subgroup of $G = (\mathbb{R}_\circ, \times)$ (Lemma D.8). Consequently, this is a bold pontifical feature. □

$$
\begin{array}{rcl}
 & \mathrm{BCan}^{\varphi}_{G/H} & \\
\mathcal{X} & \to & \varphi \cap \mathcal{X} \\
\varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\
\hline
G/H & \overset{0}{\to} & \mathsf{Id} \\
H & \overset{\mathsf{Id}}{\to} & H \\
 & \| & G/H
\end{array}
\qquad\qquad
\begin{array}{rcl}
 & (\mathrm{BCan}^{\varphi}_{G/H})^{-1} & \\
\varphi \cap \mathcal{X} & \rightrightarrows & \mathcal{X} \\
\varphi \cap \mathcal{X} & \equiv & \varphi \cap \mathcal{X} \\
\hline
\mathsf{Id} & \overset{*}{\to} & G/H \\
H & \overset{\mathsf{Id}}{\to} & H
\end{array}
$$

Figure 16.1. The group-spectral dossier of the *bold canonization operator.*

16.7. Unstructured Pontifical Features

Finally, we relax the definition of weak pontifical feature to define "unstructured" features.

DEFINITION 16.26 (Unstructured pontifical feature). An *unstructured pontifical feature* for a group G acting on a set $\mathcal{X}$ is a strong pontifical feature for a subgroup of G.

It is easy to verify that

$$\text{unstructured} \Leftarrow \text{weak} \Leftarrow \text{mild} \Leftarrow \text{bold} \Leftarrow \text{strong pontifical feature.}$$

DEFINITION 16.27 (Unstructured canonization operator). An *unstructured canonization operator* for a group G is the strong canonization operator associated to a strong pontifical feature of a subgroup of G.

16.7.1. *Examples of unstructured pontifical features*

LEMMA 16.28. *For any $c > 0$, $\mathbb{E}\{x^2\} = c$ is an unstructured pontifical feature for $G = \mathsf{Aff}(1, \mathbb{R})$.*

PROOF. By Proposition 16.7, the feature is a strong pontifical feature for $(\mathbb{R}_\circ^+, \times) \leq \mathsf{Aff}(1, \mathbb{R})$. See Lemma 16.17 for further properties of this feature. □

CHAPTER 17

Algebra of Pontifical Features

This chapter studies two composition properties: the intersection of pontifical features and the series of canonization operators. These are the two tools that we can use to obtain invariance with respect to a larger group, starting from smaller groups. The intersection of pontifical features creates more powerful features in an easy way, but implies solving a more complicated problem. Just putting a set of canonization operators in series is computationally easy, but it produces a canonical output only if certain conditions are satisfied. Here, the classification of features in unstructured, weak, mild, etc., becomes relevant.

17.1. Intersection of Pontifical Features

DEFINITION 17.1 (Intersection of features). Given two features φ_1 and φ_2, denote by $\varphi_1 \wedge \varphi_2$ their intersection:

$$(\varphi_1 \wedge \varphi_2)(x) = \varphi_1^2(x) + \varphi_2^2(x).$$

This section gives several results regarding the intersection of features. The first result is a negative result concerning unstructured features.

PROPOSITION 17.2 (Unstructured features do not mix well). *Suppose that $\varphi_1 = 0$ is an unstructured pontifical feature for a group G_1 acting on a set $\mathcal{X}$ and that $\varphi_2 = 0$ is an unstructured pontifical feature for a group $G_2 \neq G_1$ acting on $\mathcal{X}$. Then $\varphi_1 \wedge \varphi_2$ is not necessarily an unstructured pontifical feature for the group $G_1 \vee G_2$.*

PROOF. This can be seen in the simplest case of linear actions. For example, let $\mathcal{X} = \mathbb{R}^2$, let $G_1 = \{[0, b] \mid a \in \mathbb{R}\}$ with addition and $G_2 = \{[0, b] \mid b \in \mathbb{R}\}$. Then $\varphi_1(x) = x_2 - \sin x_1 = 0$ is an unstructured pontifical feature for G_1. This is just because the orbits of G_1 are vertical lines that intersect the curve $x_2 - \sin x_1 = 0$ only once. Take now $\varphi_2(x) = x_1 - 2\sin x_2 = 0$. This is a pontifical feature for G_2. However, the intersection of these two curves are not an unstructured pontifical feature for $G_1 \vee G_2 = \mathbb{R}^2$, because the intersection of $\varphi_1 = 0$ and $\varphi_2 = 0$ is not a single point.

□

The rest of the results are positive results regarding the various classes of structured features.

PROPOSITION 17.3 (Intersection of two weak features is a weak feature). *Let G act on $\mathcal{X}$. Let φ_1 and φ_2 be two weak pontifical features for G with heresy subgroup H_1 and H_2. Suppose that the set $\{x \in \mathcal{X} \mid \varphi_1(x) = 0 \wedge \varphi_2(x) = 0\}$ is non empty. Then $\varphi_1 \wedge \varphi_2$ is a weak pontifical feature for G with heresy subgroup $H_1 \cap H_2$.*

PROOF. Given a point $x \in \mathcal{X}$, define $S_x \subset G$ as the set of canonization elements:

$$S_x = \{g \in G \mid (\varphi_1(g \cdot x) = 0) \wedge (\varphi_2(g \cdot x) = 0)\}.$$

We need to prove that S_x can be written as a coset:

$$S_x = (H_1 \cap H_2)g_x$$

for some element $g_x \in G$. The proof consists of two steps. First we prove that $S_x = (H_1 \cap H_2)S_x$, then we prove that if $S^x = HS^x$, necessarily $H = H_1 \cap H_2$.

For the first step, let g be an element of S_x (assumed nonempty). This implies $\varphi_1(g \cdot$

$x) = 0$, $\varphi_2(g \cdot x) = 0$. If you take hg, where $h \in H_1 \cap H_2$, then $\varphi_1(hg \cdot x) = 0$ because $h \in H_1$, and equivalently for the other feature.

For the second step, suppose $S^x = HS^x$ for some subgroup $H \leq G$. This implies that, for each $h \in H$, $\varphi_1(hg \cdot x) = 0$ and $\varphi_2(hg \cdot x) = 0$. Therefore $h \in H_1$ and $h \in H_2$. □

PROPOSITION 17.4 (Intersection of two mild pontifical features). *If φ_1 and φ_2 are mild pontifical features, then $\varphi_1 \wedge \varphi_2$ is a mild pontifical feature as well.*

PROOF. Because a mild pontifical feature is also a weak pontifical feature, from Proposition 17.3 it follows that $\varphi_1 \wedge \varphi_2$ is a weak pontifical feature with heresy group $H_1 \cap H_2$, and from Lemma C.13 is a normal subgroup. □

17.2. Series of Canonization Operators

This section considers the use of canonization operators in series. This is the case that is most important for applications: the intersection of features, studied in the previous section, implies that we can solve a more complicated problem, because normalizing with respect to $\varphi_1 \wedge \varphi_2$ implies that we solve simultaneously for $\varphi_1(g \cdot x) = 0$ and $\varphi_2(g \cdot x) = 0$, which might be complicated.

Instead, we want to find sufficient conditions under which we can just use in series the canonization operators for φ_1 and φ_2 and still obtain an output which is canonical with respect to both.

PROPOSITION 17.5 (Series of weak and mild canonization operators). *Suppose φ_1 is a weak pontifical feature for $G = G_1 \vee G_2$ with heresy subgroup H_1 and that φ_2 is a mild canonical feature for G with heresy group H_2. Then a sufficient condition for the output of the series of the two canonization operator to be canonical with respect to both φ_1 and φ_2 is that $G/H_2 \leq H_1$.*

PROOF. A mild canonization operator is the action of G/H_2. If $G/H_2 \leq H_1$, then the value of φ_1 is conserved. □

Figure 17.1 shows the situation using group-spectral dossiers.

$\mathrm{WCan}^{\varphi_1}_{G_1 \vee G_2}$

$\mathcal{X}$	$\rightrightarrows$	$\varphi_1 \cap \mathcal{X}$
$\varphi_1 \cap \mathcal{X}$	$\equiv$	$\varphi_1 \cap \mathcal{X}$
$G_1 \vee G_2$	$\overset{0}{\to}$	Id
Id	$\overset{\star}{\to}$	H_1
$\Vert\ (G_1 \vee G_2)$		

(a) First canonization operator

$\mathrm{MCan}^{\varphi_2}_{(G_1 \vee G_2)/H_2}$

$\mathcal{X}$	$\to$	$\varphi_2 \cap \mathcal{X}$
$\varphi_2 \cap \mathcal{X}$	$\equiv$	$\varphi_2 \cap \mathcal{X}$
$(G_1 \vee G_2)/H_2$	$\overset{0}{\to}$	Id
H_2	$\overset{m_x}{\to}$	H_2
$\Vert\ (G_1 \vee G_2)/H_2$		

(b) Second canonization operator

$\mathrm{WCan}^{\varphi_1}_{G_1 \vee G_2} \circ \mathrm{MCan}^{\varphi_2}_{(G_1 \vee G_2)/H_2}$

$\mathcal{X}$	$\rightrightarrows$	$(\varphi_2 \cap \varphi_1) \cap \mathcal{X}$
$(\varphi_2 \cap \varphi_1) \cap \mathcal{X}$	$\equiv$	$(\varphi_2 \cap \varphi_1) \cap \mathcal{X}$
$G_1 \vee G_2$	$\overset{0}{\to}$	Id
Id	$\overset{\star}{\to}$	$H_1 \cap H_2$
$\Vert\ G_1 \vee G_2$		

(c) Dossier for the composition

Figure 17.1. Illustration for Proposition 17.5.

17.3. Examples

Table 17.1 summarizes the properties of the various features considered so far through the chapter:

$$\mathbb{E}\{x\} = 0, \qquad \mathbb{E}\{x\} = 1, \qquad \mathbb{E}\{x^2\} = 1, \qquad \mathsf{std}\{x\} = 1.$$

The table shows what class of pontifical features they belong to, with respect to the group $G = \mathsf{Aff}(\mathbb{R})$ and its subgroups $(\mathbb{R}_\circ, \times)$ and $(\mathbb{R}, +)$.

Table 17.2 shows the results of composing in series the canonization operators of two of the features. These are the same results already shown in Table 16.3, but this time the

	G		
φ	$(\mathbb{R}, +)$	$(\mathbb{R}_\circ, \times)$	$\mathsf{Aff}(\mathbb{R})$
$\mathbb{E}\{x\} = 0$	*strong*	✗	*weak* $H = (\mathbb{R}_\circ, \times) \leq G$
$\mathbb{E}\{x\} = 1$	*strong*	*strong*	*weak* $H = \left\{ \left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ \right\} \leq G$
$\mathrm{std}\{x\} = 1$	✗	*bold* $H = (\pm 1, \times) \triangleleft G$ $G/H \cong (\mathbb{R}_\circ^+, \times) \triangleleft G$	*mild* $H = (\mathbb{R}, +) \rtimes (\pm 1, \times) \triangleleft G$ $G/H \cong (\mathbb{R}_\circ^+, \times) \leq G$
$\mathbb{E}\{x^2\} = 1$	✗	*bold* $H = (\pm 1, \times) \triangleleft G$ $G/H \cong (\mathbb{R}_\circ^+, \times) \triangleleft G$	*unstructured*

Table 17.1. Strength of three features for various groups.

table shows also the classification of the features and their heresy subgroups. We are now able to predict most of these results.

For example, case (a) can be predicted to work by using Proposition 17.5, as the two features satisfy the preconditions of the proposition: they are weak and mild, respectively, and $G/H_2 \leq H_1$. Instead, in case (c), in which there is still a weak and a mild feature, the condition $G/H_2 \leq H_1$ fails because $(\mathbb{R}_\circ^+, \times)$ is not a subgroup of $\left\{ \left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right) \right\}$.

	first step			*second step*			
	φ_1	*type*	H_1	φ_2		H_2	$\varphi_1 \wedge \varphi_2$*?*
a)	$\mathbb{E}\{x\} = 0$	weak	$(\mathbb{R}_\circ, \times)$	$\mathrm{std}\{x\} = 1$	mild	$(\mathbb{R}, +) \rtimes (\pm 1, \times)$ $G/H_2 = (\mathbb{R}_\circ^+, \times)$	✓
b)	$\mathbb{E}\{x\} = 0$	weak	$(\mathbb{R}_\circ, \times)$	$\mathbb{E}\{x^2\} = 1$	unstr.	n/a	✓
c)	$\mathbb{E}\{x\} = 1$	weak	$\left\{\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\right\}$	$\mathrm{std}\{x\} = 1$	mild	$(\mathbb{R}, +) \rtimes (\pm 1, \times)$ $G/H_2 = (\mathbb{R}_\circ^+, \times)$	✗
d)	$\mathbb{E}\{x\} = 1$	weak	$\left\{\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\right\}$	$\mathbb{E}\{x^2\} = 1$	unstr.	n/a	✗
e)	$\mathrm{std}\{x\} = 1$	mild	$(\mathbb{R}, +) \rtimes (\pm 1, \times)$ $G/H_1 = (\mathbb{R}_\circ^+, \times)$	$\mathbb{E}\{x\} = 0$	weak	$(\mathbb{R}_\circ, \times)$	✓
f)	$\mathbb{E}\{x^2\} = 1$	unstr.	n/a	$\mathbb{E}\{x\} = 0$	weak	$(\mathbb{R}_\circ, \times)$	✓
g)	$\mathrm{std}\{x\} = 1$	mild	$(\mathbb{R}, +) \rtimes (\pm 1, \times)$ $G/H_1 = (\mathbb{R}_\circ^+, \times)$	$\mathbb{E}\{x\} = 1$	weak	$\left\{\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\right\}$	✓
h)	$\mathbb{E}\{x^2\} = 1$	unstr.	n/a	$\mathbb{E}\{x\} = 1$	weak	$\left\{\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\right\}$	✓

Table 17.2. Results for the series of two canonization operators

CHAPTER 18

Some Pontifical Features for Bootstrapping

This chapter gives some example of canonization procedures based on pontifical features in the context of bootstrapping.

18.1. Legend

preconditions on format: These are assumptions on the format of the data.

assumptions on the system: These are assumptions on the physical system.

group: The group that acts as a nuisance.

action: The action of the group.

feature: The pontifical feature being examined.

feature type: The type of the feature (strong, weak, etc.).

heresy subgroup: The heresy subgroup, in the case of weak features.

canonization operator: The canonization operator associated to the feature.

18.2. Whitening

preconditions on format: $\mathcal{Y} = \mathbb{R}^{n_y}$

group: $\mathsf{GL}(n_y)$

set acted on: $\mathcal{D}(\mathcal{Y}; \mathcal{U})$

action: $\boldsymbol{y}_t \mapsto \mathbf{A}\boldsymbol{y}_t$

feature: $\text{cov}\{\boldsymbol{y}\} = \mathbf{I}_{n_y}$

feature type: weak

heresy subgroup: $\mathsf{O}(n_y)$

canonization operator: $\boldsymbol{y}_t \mapsto \text{cov}\{\boldsymbol{y}\}^{-1/2}\boldsymbol{y}_t$

Whitening is the simplest example of canonization operator that can be explained as a pontifical feature. Suppose that there is a linear nuisance acting on the data, represented by a nonsingular matrix $\mathbf{A}$:

$$\boldsymbol{y}_t \mapsto \mathbf{A}\boldsymbol{y}_t, \qquad \mathbf{A} \in \mathsf{GL}(n_y).$$

One way to obtain an invariant representation is to use the covariance matrix as a covariant statistics. Under this nuisance, the covariance matrix $\mathbf{P} = \text{cov}\{\boldsymbol{y}\}$ transforms as

$$\mathbf{P} \mapsto \mathbf{A}\mathbf{P}\mathbf{A}^T.$$

(a) y_t **(b)** $-1.5\, y_t$

Figure 18.1. Effect of a linear transformation on a signal.

The feature

$$\mathbf{P} = \mathbf{I}_{n_y}$$

is a *weak* pontifical feature for this nuisance (Lemma E.10).

The heresy subgroup is the group of orthogonal transformations $\mathsf{O}(n_u)$, because they leave the feature invariant:

$$\mathbf{X}\mathbf{I}_{n_y}\mathbf{X}^T = \mathbf{I}_{n_y} \qquad \text{for all } \mathbf{X} \in \mathsf{O}(n_u).$$

The canonization operator is

$$\boldsymbol{y} \mapsto \mathbf{P}^{-1/2}\boldsymbol{y}.$$

The group spectral dossier (Table 18.1) of this canonization operator shows that the nuisance group $\mathsf{GL}(n_y)$ is reduced to $\mathsf{O}(n_y)$.

Table 18.1. Group-spectral dossier for whitening (Subsection 18.2).

	whitening	
$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$	$\rightrightarrows$	$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$
$\mathsf{GL}(n_y)$	$\overset{0}{\to}$	Id
Id	$\overset{\star}{\to}$	$\mathsf{O}(n_y)$

18.3. Contrast Transformation

preconditions on format: $\mathcal{Y} = [0,1]^{n_y}$

group: $\mathsf{Homeo}_+([0,1])^{n_y}$

set acted on: $\mathcal{D}(\mathcal{Y};\mathcal{U})$

action: $y_t^i \mapsto f_i(y_t^i), \qquad f_i \in \mathsf{Homeo}_+([0,1])$

feature: $\mathbb{P}(y_t^i < x) = x$ (the sensels have uniform pdf)

feature type: strong

heresy subgroup: None.

This is the simplest example of a nonlinear nuisance acting on the data. For simplicity, we assume that the domain of each sensel is $[0,1]$ (this is not a restrictive condition, because we can find a bijection between $\mathbb{R}$ and $[0,1]$).

The nuisance $f_i \in \mathsf{Homeo}_+([0,1])$ is a monotonic transformation of the the sensel values. A pontifical feature can be found using as a statistic the cumulative distribution function c_i of each sensel:

$$\begin{aligned} c_i : [0,1] &\rightarrow [0,1], \\ x &\mapsto \mathbb{P}(y_t^i < x). \end{aligned}$$

Note that the domain of c_i is $[0,1]$ because it is the assumed to be the domain of the sensel values, and that the codomain is $[0,1]$ because it is a probability. It is also easy to see that

(a) Original function y_t **(b)** $f(y_t)$, with $f \in \mathsf{Homeo}_+(\mathbb{R})$

Figure 18.1. Effect of a homeomorphism nuisance.

$c_i(0) = 0$, $c_i(1) = 1$ and that it is monotonically increasing. In other words the cdf is a homeomorphism: $c_i \in \mathsf{Homeo}_+([0,1])$.

Consider the action of a nuisance $f_i \in \mathsf{Homeo}_+([0,1])$ and call $\tilde{y}_t^i = f_i(y_t^i)$ the perturbed values. The perturbed cdf $\tilde{c}$ is

$$\begin{aligned}
\tilde{c}_i(x) &= \mathbb{P}(\tilde{y}_t^i < x) \\
&= \mathbb{P}(f_i(y^i) < x) \\
&= \mathbb{P}(y^i < f_i^{-1}(x)) \\
&= c_i(f_i^{-1}(x)).
\end{aligned}$$

This means that the cdf is contravariant with respect to the action of the nuisance. Summarizing in a diagram:

$$\begin{array}{ccc}
y^i & & f_i \circ y^i \\
\text{cdf} \downarrow & \xmapsto{f_i} & \text{cdf} \downarrow \\
c_i & & c_i \circ f_i^{-1}
\end{array}$$

This information can be used to find a canonization operator. Define z^i, the canonical version of y^i, as $z^i \triangleq c_i(y^i)$. It follows that $\mathbb{P}(z^i < x) = c_i \circ c_i^{-1}(x) = x$, which means that z^i is uniformly distributed in the interval $[0,1]$. This implies that $\mathbb{P}(y_t^i < x) = x$ is a strong pontifical feature for the action of $\mathsf{Homeo}_+([0,1])$.

---

cdf: Cumulative distribution function. See Definition B.5.

Table 18.2. Group-spectral dossier for canonization to $\mathsf{Homeo}_+([0,1])$

$\mathcal{D}([0,1];\mathcal{U})$	$\overset{c}{\rightrightarrows}$	$\mathcal{D}([0,1];\mathcal{U})$
$\mathsf{Homeo}_+([0,1])$	$\overset{0}{\to}$	Id

18.4. Unlabeled Sensels

preconditions on format: $\mathcal{Y} = \overline{\mathcal{Y}}^{n_y}$

assumptions on the system: Field-sampler with equispaced observations.

group: $\mathsf{Perm}(n_y)$

set acted on: $\mathcal{D}(\mathcal{Y};\mathcal{U})$

action: $y_t^i \mapsto y_t^{\pi(i)}$, for $\pi \in \mathsf{Perm}(n_y)$

feature: given by (18.1)

feature type: weak

heresy subgroup: $(\pm 1, \times)$

canonization operator: (described below)

This is the simplest example of sensor geometry reconstruction. Assume that the agent has a simple field-sampler whose sensels are equispaced (Figure 18.1):

$$y_t^i = h(s^i + \boldsymbol{q}), \qquad s^i = i\Delta.$$

Suppose that the nuisance is a permutation π that scrambles the sensels:

$$\tilde{y}_t^i = y_t^{\pi(i)}.$$

Can the original order be reconstructed?

This is a simplified version of the sensor geometry reconstruction problem, because

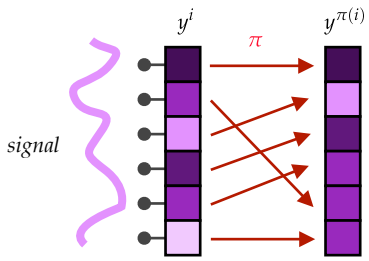


Figure 18.1. The nuisance scrambles the sensels of a one-dimensional field-sampler.

it assumes that the sensels positions are fixed at regular intervals, and only the order is unknown. Consequently, it is possible to use a relatively simple feature.

Consider three arbitrary sensels labeled $i, j, k \in \{1, n\}$, and the corresponding sensel positions $s^i, s^j, s^k \in \mathbb{R}$. If the sensels were picked in the right spatial order ($s^i < s^j < s^k$), then one expects that the values of y^i are more "similar" to the values of y^j than to the values of y^k. Let R be a similarity measure that satisfies such property:

$$i < j < k \qquad \Leftrightarrow \qquad \begin{array}{l} R(y^j, y^i) > R(y^i, y^k) \\ R(y^j, y^k) > R(y^i, y^k) \end{array} . \tag{18.1}$$

In this one-dimensional problem, the order can be recovered by a simple algorithm. Consider the sensel that was the j-th sensel in the original sequence, and count the number N of pairs (i, k) for which the previous constraint holds, which is the number of sensel triplets for which j is the middle sensel. This is equal to $N(j) = (j-1) \times (n-j)$, because there are exactly $(j-1)$ sensels on the left, and $(n-j)$ on the right. If $N(j) = 0$, the sensel is either the first ($j = 1$) or the last ($j = n$). One can choose (arbitrarily) one of the two possibilities, and continue counting. After the first arbitrary choice, the constraints uniquely determine the rest of the sensel indices.

The first ambiguous choice determines whether the sensels are ordered left-to-right or right-to-left. One solution can be transformed in the other using the permutation

$$\hat{\pi} = (n \quad n-1 \quad \dots \quad 2 \quad 1).$$

Applying this permutation twice corresponds to the identity: $\hat{\pi} \cdot \hat{\pi} = e$, so the set $\{e, \hat{\pi}\}$ is a subgroup of $\mathsf{Perm}(n)$. This subgroup is the heresy subgroup of the feature (18.1). The subgroup is isomorphic to $(\pm 1, \times)$, which is used as a placeholder in Table 18.3.

Table 18.3. Group-spectral dossier for reconstruction from unlabeled sensels

$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$	$\rightrightarrows$	$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})\cap\varphi$
$\mathsf{Perm}(n_y)$	$\xrightarrow{0}$	Id
Id	$\xrightarrow{\star}$	$(\pm 1,\times)$

In this example, the group action of $(\pm 1,\times)$ corresponds to reversing the order of the sensels.

18.5. Sensel Space Diffeomorphisms

preconditions on format: $\mathcal{Y} = \mathsf{Differentiable}(\mathbb{R}; \mathbb{S}^1)$

preconditions on format: Assumption 6 (*The observations correspond to a spatial field*)

group: $\mathsf{Diff}(\mathbb{S}^1)$

set acted on: $\mathcal{D}(\mathcal{Y}; \mathcal{U})$

action: $y_t(s) \mapsto y_t(\varphi(s))$, for $\varphi \in \mathsf{Diff}(\mathbb{S}^1)$

feature: $\mathsf{var}\{\nabla y(\theta)\} = \text{const}$

feature type: weak

heresy subgroup: $\mathsf{Isom}(\mathbb{S}^1) \cong \mathsf{O}(2)$

canonization operator: (see (18.2) below)

Suppose that an agent has a one-dimensional vision sensor (Figure 18.1). The previous example assumes that the sensor had a discrete set of sensels. In this example, the observations are a differentiable function from $\mathbb{S}^1$ to $\mathbb{R}$. The nuisance $\mathsf{Diff}(\mathbb{S}^1)$ deforms the field of view for the agent, similarly to the effect of an unknown optics for the sensor.

One way to find a canonical representation is to impose that the observations have uniform statistics across the visual field, by constraining the variance of the image gradient to be constant:

$$\mathsf{var}\{\nabla y(\theta)\} = \alpha, \quad \text{for some } \alpha > 0. \tag{18.1}$$

In this one-dimensional case, the canonization operator can be found in a closed form.

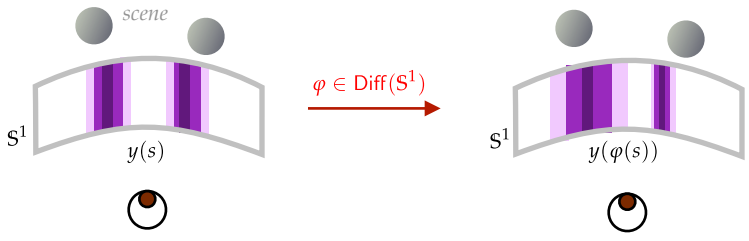


Figure 18.1. A diffeomorphism nuisance acting on the domain of the signal.

The nuisance is a diffeomorphism that acts on the domain of the function. The domain is the unit circle, so write the observations as a function of an angle θ:

$$y'(\theta) = y(\varphi(\theta)).$$

The gradient is $\nabla y'(\theta) = \frac{\partial \varphi}{\partial \theta} y(\varphi(\theta))$. The constraint (18.1) is written as

$$\left(\frac{\partial \varphi}{\partial \theta}\right)^2 \mathsf{var}\{\nabla_\theta y(\theta)\} = \alpha.$$

The Jacobian $\partial\varphi/\partial\theta$ is nonzero for all θ because φ is a diffeomorphism, and it has the same sign everywhere because the domain is connected. We can choose either a positive or negative sign for $\partial\varphi/\partial\theta$:

$$\frac{\partial \varphi}{\partial \theta} = \pm \frac{\alpha}{\sqrt{\mathsf{var}\{\nabla_\theta y(\theta)\}}}$$

The diffeomorphism φ must also satisfy

$$\int_0^{2\pi} \frac{\partial \varphi}{\partial \theta} = 2\pi,$$

because the circle must be mapped onto itself. This constraints allows to find the constant α. The final solution for φ, representing the canonization operator, is

$$\varphi^\star(\theta) = c + \pm\alpha \int_0^\theta \frac{1}{\sqrt{\mathsf{var}\{\nabla y(\beta)\}}}\, d\beta, \qquad \alpha = \frac{1}{2\pi} \int_0^{2\pi} \frac{1}{\sqrt{\mathsf{var}\{\nabla y(\theta)\}}}\, d\theta. \tag{18.2}$$

There are two ambiguities: the sign ($\pm$) and the integration constant c. These corresponds to reflections and rotations of the circle.

Table 18.4. Group-spectral dossier for the canonization operator

$\mathcal{D}(\mathsf{Differentiable}(\mathbb{R};\mathbb{S}^1);\mathcal{U})$	$\overset{\varphi^\star}{\rightrightarrows}$	$\mathcal{D}(\mathsf{Differentiable}(\mathbb{R};\mathbb{S}^1);\mathcal{U})$
$\mathsf{Diff}(\mathbb{S}^1)$	$\overset{0}{\to}$	Id
Id	$\overset{\star}{\to}$	$\mathsf{Isom}(\mathbb{S}^1) \simeq \mathsf{SO}(2)$

CHAPTER 19

An Example of Compositional Analysis

This chapter presents a longer example of the use of pontifical features and group-spectral dossiers that demonstrates how they simplify reasoning about compositional properties of bootstrapping agents.

19.1. Building Blocks

This example uses several building blocks, most of which have already been encountered.

19.1.1. *BGDS agent*

Table 19.1 shows the group-spectral dossier for the BGDS agent described in 12. More precisely, this is the map $\mathsf{WtoB}_{\mathcal{A}}$ from world to behavior (Definition 5.6), but for clarity it is just denoted "BGDSagent". The dossier says that the agent is invariant to any diffeomorphisms of the image, and it is contravariant to linear nuisances of the commands.

Table 19.1. Group-spectral dossier for BGDSagent

$\mathcal{D}(\mathsf{Images}(\mathcal{S}); \mathbb{R}^{n_u})$	$\overset{\mathsf{BGDSagent}}{\longrightarrow}$	*behavior*
$\mathsf{Diff}(\mathcal{S})$	$\xrightarrow{0}$	Id
$\mathsf{Aff}(\mathbb{R})$	$\xrightarrow{0}$	Id
$\mathsf{GL}(n_u)$	$\xrightarrow{-1}$	$\mathsf{GL}(n_u)$

19.1.2. *CalibA and CalibB*

The BGDS agent assumes that the observations are a spatially coherent field on a manifold. If the observations are just a set of scrambled pixels, one possibility is to add a sensor geometry reconstruction stage, as studied in 10.

Table 19.2 shows the group-spectral dossier for the algorithm CalibA (Algorithm 1). The calibration algorithm is seen as a map that takes a world with scrambled sensels (an element of $\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$) to a world world whose observations are spatially coherent (an element of $\mathcal{D}(\mathbb{S}^2 \to \mathbb{R};\mathcal{U})$). The algorithm is invariant to any scrambling ($\mathsf{Perm}(n_y)$) and any contrast transformation ($\mathsf{Homeo}_+(\mathbb{R})$). The absolute orientation of the sensor is unobservable, which is modeled by the introduction of the nuisance $\mathsf{Isom}(\mathbb{S}^2)$.

Table 19.2. Group-spectral dossier for CalibA

$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$	$\overset{\mathsf{CalibA}}{\rightrightarrows}$	$\mathcal{D}(\mathsf{Images}(\mathbb{S}^2);\mathcal{U})$
$\mathsf{Perm}(n_y)$	$\xrightarrow{0}$	Id
$\mathsf{Homeo}_+(\mathbb{R})$	$\xrightarrow{0}$	Id
Id	$\xrightarrow{\star}$	$\mathsf{Isom}(\mathbb{S}^2)$

The calibration algorithm described in Chapter 10 produces a coherent image on $\mathbb{S}^2$. The absolute orientation is unobservable, thus the group $\mathsf{Isom}(\mathbb{S}^2)$ appears as a nuisance. The algorithm is invariant to both permutations of the sensels as well as nonlinear transformations of the values such as contrast transformations ($\mathsf{Homeo}_+(\mathbb{R})$).

CalibB is the calibration operator using the simpler algorithm based on MDS . It is not able to compensate contrast transformations. The solution is still topologically correct, but the solution is deformed. This is modeled by an unknown diffeomorphism acting on the solution.

MDS: Multidimensional scaling is a classic embedding algorithm that recovers the positions of a set of points given the interpoint distance matrix [115] .

Table 19.3. Group-spectral dossier for CalibB

$\mathcal{D}(\mathbb{R}^{n_y};\mathcal{U})$	$\overset{\mathsf{CalibB}}{\rightrightarrows}$	$\mathcal{D}(\mathsf{Images}(\mathbb{S}^2);\mathcal{U})$
$\mathsf{Perm}(n_y)$	$\xrightarrow{0}$	Id
$\mathsf{Homeo}_+(\mathbb{R})$	$\xrightarrow{\sim}$	$\mathsf{Diff}(\mathbb{S}^2)$
Id	$\xrightarrow{\star}$	$\mathsf{Isom}(\mathbb{S}^2)$

The calibration algorithm based on MDS is not invariant to nonlinear transformations of the values. A contrast transformation ($\mathsf{Homeo}_+(\mathbb{R})$) will change the similarity values and this will perturb the solution. The result will still be topologically correct. This is indicated by the nuisance $\mathsf{Diff}(\mathbb{S}^2)$ introduced.

19.1.3. *Smoothing*

Smoothing is a common operation on images.

DEFINITION 19.1. Spherical smoothing with a kernel $k : \mathbb{R}^+_\bullet \to \mathbb{R}^+_\bullet$ on a manifold $\mathcal{S}$ is defined as

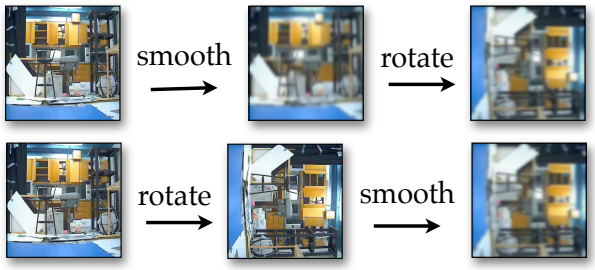
$$\mathsf{Smooth}_k(\boldsymbol{y})(s) = \int_v y(s)k(d(s,v))\,\mathrm{d}\mathcal{S}.$$

The definition of smoothing uses the metric of the manifold. Smoothing commutes with isometries because they keep the metric invariant. Smoothing does not commute with general diffeomorphisms. This is explicitly noted in Table 19.4 using the "unstructured" notation "$\sim$".

Table 19.4. Group-spectral dossier for Smooth_k

$\mathcal{D}(\mathsf{Images}(\mathcal{S});\mathcal{U})$	$\overset{\mathsf{Smooth}_k}{\to}$	$\mathcal{D}(\mathsf{Images}(\mathcal{S});\mathcal{U})$
$\mathsf{Isom}(\mathcal{S})$	$\overset{\mathsf{Id}}{\longrightarrow}$	$\mathsf{Isom}(\mathcal{S})$
$\mathsf{Diff}(\mathcal{S})$	$\overset{\sim}{\longrightarrow}$	$\sim$
$\mathsf{Aff}(\mathbb{R})$	$\overset{\mathsf{Id}}{\longrightarrow}$	$\mathsf{Aff}(\mathbb{R})$

The smoothing operation commutes with isometries of the domain ($\mathsf{Isom}(\mathcal{S})$), but not more general transformations, such as diffeomorphisms. Smoothing also commutes with an affine transformation of the values.



19.2. Analysis of Compositions

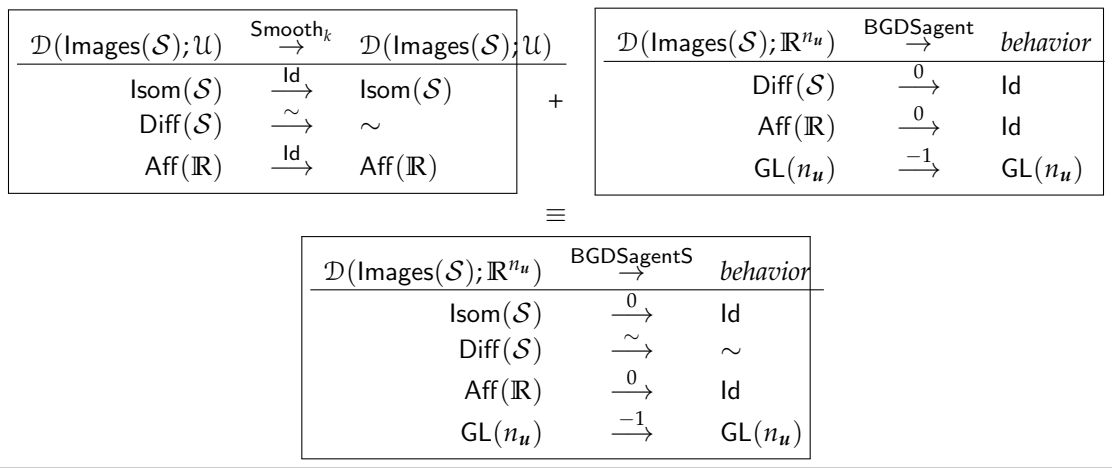
19.2.1. *Series of Smooth_k and BGDSagent*

Let us define a new agent by adding smoothing before the input image to the BGDS agent:

$$\mathsf{BGDSagentS} = \mathsf{BGDSagent} \circ \mathsf{Smooth}_k.$$

Intuitively, preprocessing the data with a smoothing operation makes the agent more robust to noise, and therefore the resulting agent is "better" than the one without smoothing. However, from the bootstrapping perspective, the agent is less powerful because it is less invariant. As can be easily seen using the group-spectral dossiers (19.1), the composition is only invariant to isometries of the domain, because smoothing has an unstructured response to diffeomorphisms.

Figure 19.1. Invariance analysis for the composition of smoothing and the BGDS agent. The resulting agent is less powerful because it is not invariant to diffeomorphisms.

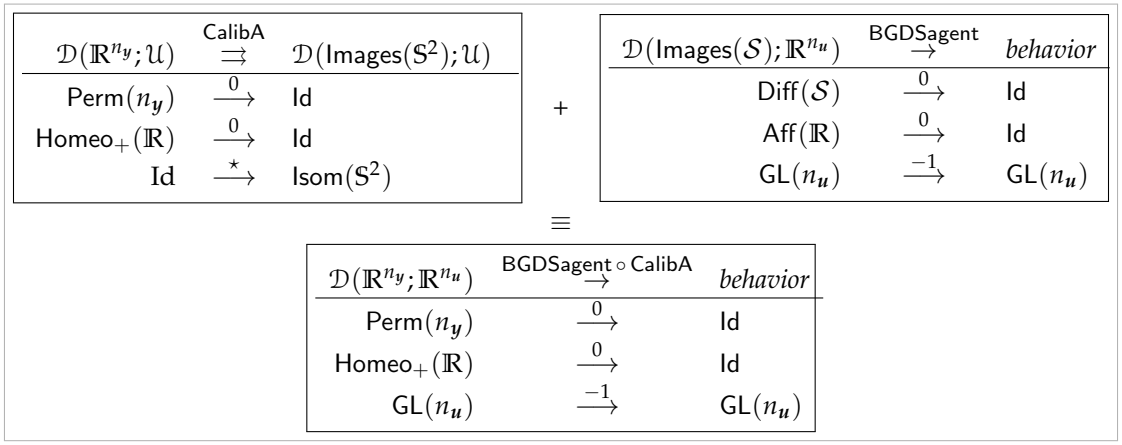


19.2.2. *Calibration and BGDSagent*

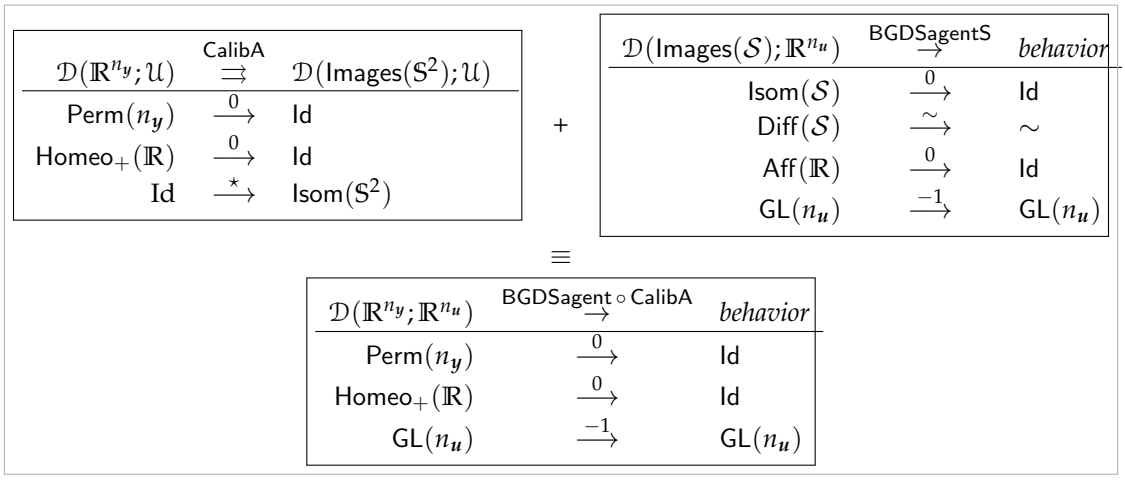
So far two agents have been defined: "BGDSagent" and "BGDSagentS", which adds a smoothing operation. Two calibration algorithms ("CalibA" and "CalibB") allow these agents to work with scrambled sensels. Which combinations of agents and calibration algorithm work best?

This question can be answered easily by computing the group-spectral dossier of the compositions. Note that at this point we have abstracted away everything about these components, except their assumptions on the format of the data and their invariance properties.

Figures 19.2a–19.3b show the results of the analysis. The least powerful agent is the combination of CalibB, which produces a diffeomorphism nuisance, and BGDSagentS, whose smoothing stage does not make the agent invariant to diffeomorphisms. All the other combinations have the same invariance properties. CalibB plus BGDSagent works well because BGDSagent can compensate the diffeomorphism. CalibA plus BGDSagentS works well because no diffeomorphism nuisance is introduced.

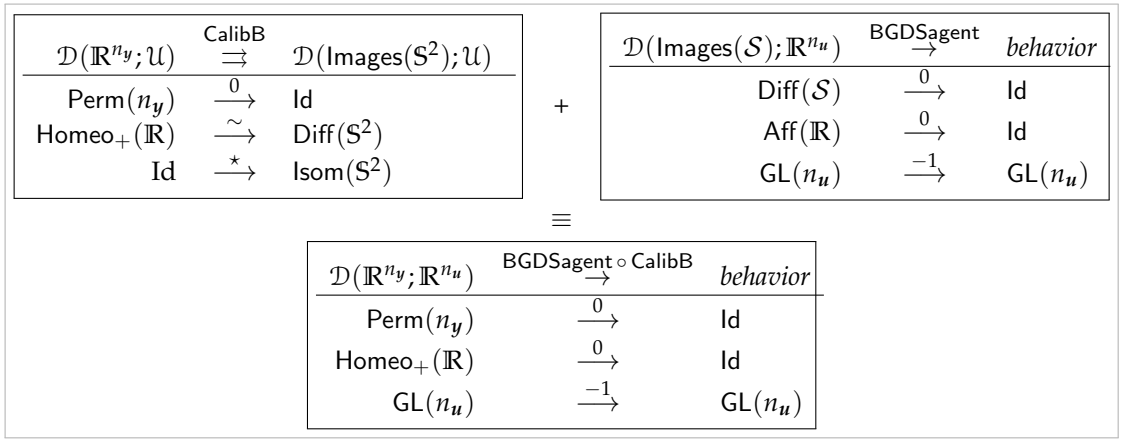


(a) CalibA and BGDSagent

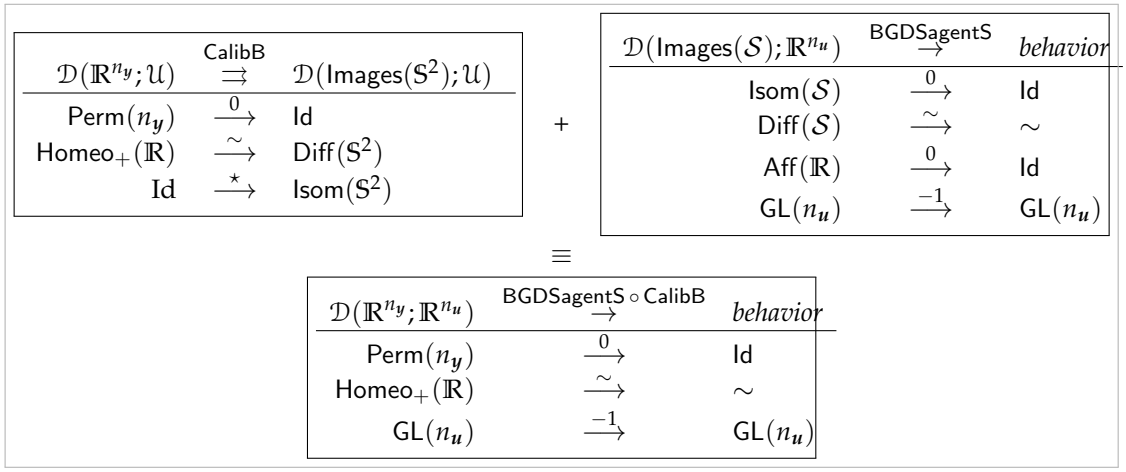


(b) CalibA and BGDSagentS

Figure 19.2. Combinations of the two calibration algorithms and the two agents (with and without smoothing)



(a) CalibB and BGDSagent



(b) CalibB and BGDSagentS

Figure 19.3. Combinations of the two calibration algorithms and the two agents (with and without smoothing)

Conclusions

CHAPTER 20

Conclusions

This dissertation has two main messages. The *optimistic* message is that it is possible to create agents that can work with a large variety of robotic sensors, with little to zero prior information on the sensor. The *pessimistic* message is that agents should be evaluated on their invariance properties with respect to the representation nuisances, and models usually come with very little built-in invariance (e.g., the BDS is closed only with respect to linear transformations). Another *optimistic* message—so that the final balance is positive—is that thinking about bootstrapping as a canonization problem might allow designing agents as an interconnection of modular components that can be composed together to obtain invariance to large representation nuisances.

20.1. Future Work

20.1.1. *Completing the enumeration of the robot set*

This dissertation only considered exteroceptive sensors with simple kinematics models (i.e., the Vehicles universe). Eventually one wishes to create agents that can work with any robot. There are three main features that were outside of the formalization:

hidden states The state space for articulated robots includes the variable describing the position of each joint (rotation or translation). The state of each joint influences the commands-observations dynamics. If the robot has proprioception abilities, meaning that it can observe the position of its joints, then it is possible to extend the techniques presented (BDS/BGDS/DDS models) by learning one of such models for each joint state. The more challenging scenario is if the state

are not observable.

actuator dynamics Kinematic models have limited applicability, but second-order models, in which the commands are forces/torques instead of velocities, capture most situations of interest.

other sensors There are several other sensors that should be considered to complete the enumeration of the set of all robots. Sensors that observe orientation, velocities, or accelerations should be easily incorporated. Unconventional sensors such as artificial touch might prove an interesting application for bootstrapping techniques.

Rather than defining larger and larger model classes, it would be most interesting to keep a modular approach to the problem, in which the set of robots is described as the Cartesian product of a certain sensor dynamics, a certain actuator dynamics, certain latent hidden states, as well as a certain class of representation nuisance:

$$\text{robots} = \underbrace{\mathcal{D}^\star(\mathcal{U};\mathcal{Y})}_{\text{representation}} \times \underbrace{\begin{matrix}\mathsf{BDS}(n;k)\\ \mathsf{BGDS}(\mathcal{S};k)\\ \mathsf{DDS}(\mathcal{S};\mathcal{U})\end{matrix}}_{\text{sensor dynamics}} \times \underbrace{\mathrm{CLTI}(n_u)}_{\text{actuator dynamics}} \times \text{joint dynamics},$$

and create techniques that use this structure. For example, in the case of hidden states, it would be desirable to derive a solution that joins the still-present BDS/BGDS/DDS structures with techniques that also learn a latent state space (e.g., [89]).

20.1.2. *Useful subgroup of representation nuisances*

The group of representation nuisances and its action on dynamical systems should be object of further study. It would be desirable to have a more systematic way to derive features and canonization operators.

The class $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ is quite large, as it contains extremely nonlinear transformations such as symmetric encryption. The "ideal" encryption algorithm destroys all possible statistics of the data, producing bit-streams that have (approximately) uniform statistics. The interesting question is understanding what is a subgroup of $\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$ that covers all situations of interests for robotics, yet it allows for a tractable inference.

20.1.3. *Data, regularization, and noise*

The analysis of the learning algorithms in 2 used simple asymptotic properties, in which we assumed that the statistics computed converged to the expected value. This does not answer the question of how much data is needed to learn a certain class of model. For example, the BDS class assumes a generic bilinear model, while the BGDS class assumes that the dynamics is bilinear in the *gradient* of the observations. How much less data is needed with this assumption?

Another issue that has been avoided is model *regularization* or other priors. Imposing soft or hard constraints on the parameters (e.g., smoothness of the tensor fields) is not computationally difficult, but it should be done in a way that maintains the symmetries of the agent.

As a simple example, consider the estimation of the tensor $\mathbf{M}$ in a BDS model (Definition 11.1). At first sight, it would be quite intuitive to introduce a penalty function to have a more stable estimation, perhaps by penalizing the norm $\|\mathbf{M}\|_2$. However, the BDS family is closed with respect to linear transformations of the commands, of the kind $\boldsymbol{u} \mapsto \mathbf{A}\boldsymbol{u}$, for $\mathbf{A} \in \mathsf{GL}(n_{\boldsymbol{u}})$. The corresponding action on the tensor is $\mathbf{M} \mapsto \mathbf{A}^{-1}\mathbf{M}$, but the norm $\|\mathbf{M}\|_2$ is only invariant to orthogonal transformations. This shows that, ultimately, model regularization leads to reduced agent symmetries.

Back Matter

APPENDIX A

Sets, functions, sequences

This appendix contains basic notions about sets, functions, sequences, and other related concepts. The notation and definitions are mostly standard, except for the notation $\rightrightarrows$ for one-to-many functions (Definition A.18).

A.1. Logic and sets

The notation used for logic and sets is standard:

$\wedge$	Logical "and".
$\vee$	Logical "or".
$\neg$	Logical "not".
$\cup$	Set union.
$\cap$	Set intersection.
$\subseteq$	Subset inclusion.
$\subset$	Strict subset inclusion (sometimes written as "$\subsetneq$").
$\mathcal{X} \times \mathcal{Y}$	Cartesian product of two sets.
$\emptyset$	Empty set.
$\mathbb{N}$	Natural numbers.
$\mathbb{Q}$	Rational numbers.
$\mathbb{R}$	Real numbers.

The notation used for subsets of $\mathbb{R}$ uses the mnemonics that the empty circle "$\circ$" means that zero is not included, and the filled circle "$\bullet$" means that zero is included.

$\mathbb{R}_\circ$	$= \{a \in \mathbb{R} \mid a \neq 0\}$ (nonzero reals).

$\mathbb{R}_{\circ}^{+}$ $= \{a \in \mathbb{R} \mid a > 0\}$ (positive reals).

$\mathbb{R}_{\bullet}^{+}$ $= \{a \in \mathbb{R} \mid a \geq 0\}$ (nonnegative reals).

$*\mathbb{R}$ Hyper-real numbers.

DEFINITION A.1 (Infinite binary strings). $\{\square, \boxdot\}^{\mathbb{N}}$ is the set of infinite binary strings, i.e., a mapping from the natural numbers $\mathbb{N}$ to a set with two elements $\{\square, \boxdot\}$.

DEFINITION A.2 (Power set). $\mathsf{powerset}(\mathcal{X})$ is the set of all subsets of $\mathcal{X}$, including the empty set and $\mathcal{X}$ itself. If $\mathcal{X} = \{a, b, c\}$, then

$$\mathsf{powerset}(\mathcal{X}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{c, a\}, \{a, b, c\}\}.$$

A.2. Topology

DEFINITION A.3 (Topological space). A topological space is a tuple $(\mathcal{X}, \mathcal{S})$ composed of a set $\mathcal{X}$ and a collection $\mathcal{S} \subset \mathsf{powerset}(\mathcal{X})$ of its subsets called *open sets*, satisfying the properties:

(1) The empty set and $\mathcal{X}$ itself are open sets: $\{\emptyset, \mathcal{X}\} \subset \mathcal{S}$.

(2) The union of open sets is open.

(3) The intersection of a finite number of open sets is open.

$\mathcal{S}$ is called the "topology" assigned to $\mathcal{X}$.

DEFINITION A.4 (Topological definition of continuous functions). A function between two topological spaces $\mathcal{X}$ and $\mathcal{Y}$ is *continuous* if the inverse image of every open set in $\mathcal{Y}$ is an open set of $\mathcal{X}$. Equivalently, the inverse image of every closed set is closed.

See also the metric definition of continuity (Definition E.4).

A.3. Relations and their properties

DEFINITION A.5 (Relation). A relation $\sim$ between $n \geq 2$ sets $\{\mathcal{A}, \mathcal{B}, \dots\}$ is a subset of the cartesian product $\mathcal{A} \times \mathcal{B} \times \cdots$. In the case of a binary relation $\sim$, we write $a \sim b$ if $\langle a, b \rangle \in \sim$.

DEFINITION A.6 (Types of relations). A binary relation $\sim$ is called

antisymmetric if $(x \sim y) \Rightarrow \neg(y \sim x)$;

symmetric if $(x \sim y) \Rightarrow (y \sim x)$;

transitive if $(x \sim y) \wedge (y \sim z) \Rightarrow (x \sim z)$;

reflexive if $(x \sim x)$ always holds;

irreflexive if $(x \sim x)$ never holds;

total if $(x \sim y) \vee (y \sim x)$ holds;

left-total *if for all x, there exists y such that $(x \sim y)$;*

functional (right-unique) if $(x \sim y_1) \wedge (x \sim y_2) \Rightarrow y_1 = y_2$;

injective (left-unique) if $(x_1 \sim y) \wedge (x_2 \sim y) \Rightarrow (x_1 = x_2)$.

A.4. Special classes of relations

DEFINITION A.7 (Equivalence relation). A reflexive, symmetric, and transitive binary relation.

---

reflexive relation: $(x \sim x)$ always holds. See Definition A.6.

symmetric relation: $x \sim y \Rightarrow y \sim x$ See Definition A.6.

transitive relation: $(x \sim y) \wedge (y \sim z) \Rightarrow (x \sim z)$ See Definition A.6.

DEFINITION A.8 (Partial order). A *partial order* is an antisymmetric, transitive, and reflexive binary relation. A *partially ordered set* (or *poset*) is a set equipped with a partial order relation.

DEFINITION A.9 (Total order). A *total order* is an antisymmetric, transitive, and total binary relation. An *ordered set* is a set equipped with a total order relation.

REMARK A.10 (A total order induces a topology). A total order on a set induces a topology using the sets $\{x \mid a < x < b\}$ as the basis of the topology.

DEFINITION A.11 (Lattice). A *lattice* is a partially ordered set $(\mathcal{X}, \leq)$ that contains two elements $\top, \bot \in \mathcal{X}$ such that each element $x \in \mathcal{X}$ is comparable with both of them, and it holds that $\bot \leq x \leq \top$.

EXAMPLE A.12 (Set of subgroups is a lattice). Let $\leq$ denote the subgroup relation. Consider the set of subgroup of a group G: $\mathcal{X} = \{H \mid H \leq G\}$. Then $(\mathcal{X}, \leq)$ is a lattice, with $\top = G$ and $\bot = \{e\}$.

A.5. Functions

We do give an explicit definition of "function"; not because of being pedantic, but because we use it when describing the agent semantics (Chapter 3).

---

antisymmetric relation: $(x \sim y) \Rightarrow \neg(y \sim x)$ See Definition A.6.

total relation: At least one of $(x \sim y)$ and $(y \sim x)$ holds. See Definition A.6.

DEFINITION A.13 (Function). A function f from the set $\mathcal{A}$ to the set $\mathcal{B}$, is a binary relation on $\mathcal{A} \times \mathcal{B}$ that is functional and left-total.

The *domain* of f is $\text{Domain}(f) = \mathcal{A}$ and the *codomain* is $\text{Codomain}(f) = \mathcal{B}$.

The *image* of f is the subset of the codomain that correspond to some element of the domain:

$$\text{Image}(f) = \{ f(a) \mid a \in \mathcal{A} \}.$$

$\mathsf{Functions}(\mathcal{B}; \mathcal{A})$ is the set of all functions from $\mathcal{A}$ to $\mathcal{B}$ (note the order "$\mathcal{B}; \mathcal{A}$"). The notation "$f : \mathcal{A} \to \mathcal{B}$" is equivalent to $f \in \mathsf{Functions}(\mathcal{B}; \mathcal{A})$.

REMARK A.14. Note that a function is usually not properly defined or somewhat ambiguous. For example, according to Definition A.13, the two functions

$$\begin{aligned} f_1 : \mathbb{R} &\to \mathbb{R}, \\ x &\mapsto x^2 \end{aligned}$$

and

$$\begin{aligned} f_2 : \mathbb{R} &\to \mathbb{R}^+_\bullet, \\ x &\mapsto x^2 \end{aligned}$$

are different objects, because f_1 is a relation on $\mathbb{R} \times \mathbb{R}$ and f_2 is a relation on $\mathbb{R} \times \mathbb{R}^+_\bullet$, but in many contexts these would be considered the same function.

---

functional relation: $(x \sim y_1) \wedge (x \sim y_2) \Rightarrow y_1 = y_2$ See Definition A.6.

left-total relation: For each x, there is at least a y such that $(x \sim y)$. See Definition A.6.

DEFINITION A.15 (Identity function). $\mathsf{Id}_{\mathcal{X}} : \mathcal{X} \to \mathcal{X}$ is the identity function on the set $\mathcal{X}$. Sometimes just written as Id when the set $\mathcal{X}$ is understood.

DEFINITION A.16 (Set automorphisms). A set endomorphism is a function from a set to itself. A set *automorphism* is an invertible endomorphism.

$\mathsf{Aut}(\mathcal{X})$ is the set of all automorphisms of the set $\mathcal{X}$.

DEFINITION A.17 (Types of functions). Functions inherit the definitions of relation properties given in Definition A.6:

injective function $f(x) = f(y) \;\Rightarrow\; x = y$

surjective function $\text{Image}(f) = \text{Codomain}(f)$

bijective (1-to-1) function injective and surjective

DEFINITION A.18 (Notation for 1-to-many functions). The notation $f : \mathcal{X} \rightrightarrows \mathcal{Y}$ is a shorthand for $f : \mathcal{X} \to \mathsf{powerset}(\mathcal{Y})$.

The "joker" function $\star$ maps any element of a set $\mathcal{X}$ to the whole set $\mathcal{Y}$.

DEFINITION A.19 (Joker function). Given two spaces $\mathcal{X}, \mathcal{Y}$, define $\star : \mathcal{X} \rightrightarrows \mathcal{Y}$ as $\star(x) = \mathcal{Y}$.

A.6. Inverting functions

LEMMA A.20. *If $g \circ f$ is bijective, then f is injective and g is surjective.*

LEMMA A.21 (Left inverse implies right inverse for endomorphisms). *If $f, g : \mathcal{X} \to \mathcal{X}$ and $g \circ f = \mathsf{Id}_{\mathcal{X}}$, and $\mathcal{X}$ has finite cardinality, then $f \circ g = \mathsf{Id}_{\mathcal{X}}$. But if $\mathcal{X}$ is not finite, this is false in general.*

EXAMPLE A.22. We give a simple counterexample for this Lemma. Suppose $\mathcal{X} = \{0,1\}^{\mathbb{N}}$, and let

$$\begin{aligned} f\left(\langle x_1, x_2, x_3, \ldots \rangle\right) &= \langle 0, x_1, x_2, x_3, \ldots \rangle, \\ g\left(\langle x_1, x_2, x_3, \ldots \rangle\right) &= \langle x_2, x_3, \ldots \rangle. \end{aligned}$$

The function g shifts the sequence one step to the right, and f shifts it to the left (losing information). We can verify that $g \circ f = \mathsf{Id}_{\mathcal{X}}$. From Lemma A.20, it follows that f is injective and g is surjective, which can be verified easily. However, g is not injective. Therefore $f \circ g$ cannot be a bijection.

A left inverse exists only for a bijective function.

DEFINITION A.23 (Left inverse). A *left inverse* for a bijective function $f : \mathcal{X} \to \mathcal{Y}$ is a function $f^{-1} : \mathcal{Y} \to \mathcal{X}$ such that $f^{-1}(f(x)) = x$ for all $x \in \mathcal{X}$.

A right inverse always exists, if it is defined as follows.

DEFINITION A.24 (Right inverse). A (1-to-many) *right inverse* of $f : \mathcal{X} \to \mathcal{Y}$ on $\mathcal{Z} \subset \mathcal{Y}$ is a function $f^{-1} : \mathcal{Z} \rightrightarrows \mathcal{X}$ such that $f(f^{-1}(z)) = z$ for all $z \in \mathcal{Z}$.

A.7. Sorting vectors

DEFINITION A.25. $\mathsf{differ}(\mathcal{X}, n)$ is the subset of $\mathcal{X}^n$ for which there are no repetitions ($x_i \neq x_j$ for $i \neq j$).

DEFINITION A.26 (Weakly sorted vector). Given an ordered set $(\mathcal{X}, <)$, $\mathsf{weaksortedSeq}(\mathcal{X}^n)$ is the subset of $\mathcal{X}^n$ for which $x_i \leq x_{i+1}$ for all $i \in \{1, \ldots, n\}$.

DEFINITION A.27 (Sorted vector). Given an ordered set $(\mathcal{X}, <)$, $\mathsf{sortedSeq}(\mathcal{X}^n)$ is the subset of $\mathsf{weaksortedSeq}(\mathcal{X}^n)$ for which $x_i < x_{i+1}$ for all $i \in \{1, \dots, n\}$.

DEFINITION A.28 (Order (or *rank*) of a vector). The function $\mathsf{order}(\boldsymbol{x})$ gives the order of each element of a vector:

$$\begin{aligned} \mathsf{order} : \mathbb{R}^n &\rightarrow \mathsf{Perm}(n), \\ \mathsf{order}(\boldsymbol{x})^i &= \sum_{j=1}^{n} (x^i \leq x^j). \end{aligned}$$

If counting elements from zero, replace "$\leq$" with "$<$".

DEFINITION A.29 (Sorting a vector). $\mathsf{sorted}(\boldsymbol{x})$ applies a permutation to $\boldsymbol{x}$ such that its entries are ordered in ascendent order. Sorting a vector without repetitions produces a strongly sorted sequence:

$$\mathsf{sorted} : \mathsf{differ}(\mathbb{R}, n) \rightarrow \mathsf{sortedSeq}(n).$$

Sorting a vector with repetitions produces a weakly sorted sequence:

$$\mathsf{sorted} : \mathbb{R}^n \rightarrow \mathsf{weaksortedSeq}(n).$$

A.8. Sequences

A *sequence* is a map from $\mathbb{N}$ to elements of some set $\mathcal{A}$. Bold symbols ($\boldsymbol{a}, \boldsymbol{b}, \dots$) denote sequences. Elements of the sequence are written with normal case ($\boldsymbol{a} = \langle a_0, a_1, a_2, \dots \rangle$), and numbering starts at 0. The notation "$\boldsymbol{a}_{:k}$" denotes subsequences of $\boldsymbol{a}$ up to and including the k-th term:

$$\boldsymbol{a}_{:k} = \langle a_0, \dots, a_k \rangle \in \mathcal{X}^{k+1}.$$

To make the notation compact, denote by $\boldsymbol{a}_{:-1} \doteq \emptyset$ the empty sequence.

DEFINITION A.30 (Sequences). $\mathsf{Sequences}(\mathcal{X})$ denotes the set of all sequences of all lengths of elements in $\mathcal{X}$, including the empty sequence (which we write as $\emptyset$):

$$\mathsf{Sequences}(\mathcal{X}) = \emptyset \cup \mathcal{X} \cup \mathcal{X}^2 \cup \mathcal{X}^3 \cup \dots \quad .$$

For example, the set $\mathsf{Sequences}(\mathbb{R})$ contains the elements $\emptyset, \{1\}, \{2\}, \{40, 41, 42\}$.

APPENDIX B

Probability and Statistics

General references

Loève [136, Chapter 3] contains the standard constructions for measure-theoretic probability. Lehmann and Casella [137] and Casella and Berger [138] are accessible introductions to mathematical statistics and point estimation. Maybeck [139] is a classic engineering text for stochastic estimation. Kailath, Sayed, and Hassibi [140] is a modern reference for linear estimation.

B.1. Probability Measures

We skip over the details of how to construct probability spaces, and we just focus on the operative properties of the objects that inhabit those spaces. See, e.g., Loève [136, Chapter 3] for a more rigorous construction.

DEFINITION B.1 (Probability measure). A *probability measure* μ on a set $\mathcal{X}$ is a map $\mu : \mathsf{powerset}(\mathcal{X}) \to [0,1]$ that associates a value in the $[0,1]$ interval to each subset of $\mathcal{X}$. Two conditions must be satisfied:

(1) The measure of the entire set is 1: $\mu(\mathcal{X}) = 1$.

(2) The measure of a countably finite intersection of disjoint subsets $\{\mathcal{A}_i\}_{i\in\mathbb{N}}$ is the sum of the measures:

$$\mu\left(\bigcup_i \mathcal{A}_i\right) = \sum_i \mu\left(\mathcal{A}_i\right).$$

$\mathsf{ProbMeasures}(\mathcal{X})$ is the set of all probability measures on $\mathcal{X}$.

DEFINITION B.2 (Support of a probability measure). The *support* of a measure is the subset that has nonzero measure:

$$\mathsf{Support} : \mathsf{ProbMeasures}(\mathcal{X}) \to \mathsf{powerset}(\mathcal{X}).$$

DEFINITION B.3 (Conditional distribution). A *conditional distribution* γ between two spaces $\mathcal{A}$ and $\mathcal{B}$ is a map that associates a probability measure on $\mathcal{B}$ to each element of $\mathcal{A}$:

$$\gamma : \mathcal{A} \to \mathsf{ProbMeasures}(\mathcal{B}).$$

$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$ is the set of all conditional distributions from $\mathcal{A}$ to $\mathcal{B}$.

For $\gamma \in \mathsf{Conditional}(\mathcal{B};\mathcal{A})$, the following composition rules hold:

- If $a \in \mathcal{A}$, then $\gamma a \in \mathsf{ProbMeasures}(\mathcal{B})$.
- If $\mu \in \mathsf{ProbMeasures}(\mathcal{A})$, then $\gamma\mu \in \mathsf{ProbMeasures}(\mathcal{B})$.
- If $\boldsymbol{a} \in \mathsf{Sequences}(\mathcal{A})$, then $\gamma\boldsymbol{a} \in \mathsf{StocProcesses}(\mathcal{B})$.
- If $\varphi \in \mathsf{Conditional}(\mathcal{C};\mathcal{B})$, then $\varphi\gamma \in \mathsf{Conditional}(\mathcal{C};\mathcal{A})$.

DEFINITION B.4 (Impulse). $\delta_x \in \mathsf{ProbMeasures}(\mathcal{X})$ is the impulse centered at $x \in \mathcal{X}$.

B.2. Basic Operators

DEFINITION B.5 (Cumulative density function (cdf)). Let $\mu \in \mathsf{ProbMeasures}(\mathbb{R})$. Then the cumulative density function of μ is a function $F : \mathbb{R} \to [0,1]$ defined as $F(x) = \mu(\,(-\infty, x)\,)$.

DEFINITION B.6 (Probability density function (pdf)). Let $\mu \in \mathsf{ProbMeasures}(\mathbb{R})$. The probability density function of μ is the object f whose integral is F, the cdf of μ:

$$F(b) - F(a) = \int_a^b f \, \mathrm{d}\mu.$$

According to this definition, if f is a function $f : \mathbb{R} \to \mathbb{R}_\circ^+$, then the cdf only exists if F is continuous. For a discontinuous function, the pdf always exists in the sense of a distribution.

B.3. Stochastic Processes

DEFINITION B.7 (Stochastic process). The set of stochastic processes on the space $\mathcal{A}$ is denoted $\mathsf{StocProcesses}(\mathcal{A})$. A stochastic process on the space $\mathcal{A}$ is a probability measure on the sequences of $\mathcal{A}$ that satisfies the consistency conditions of the Kolmogorov extension theorem:

$$\mathsf{StocProcesses}(\mathcal{A}) \subset \mathsf{ProbMeasures}(\mathcal{A}^{\mathbb{N}}).$$

DEFINITION B.8 (The Final operator). For the stochastic processes that converges in distribution ("weak convergence"), we use the operator Final to extract the limit distribution:

$$\mathsf{Final} : \mathsf{StocProcesses}(\mathcal{A}) \to \mathsf{ProbMeasures}(\mathcal{A}).$$

B.4. Statistics

DEFINITION B.9 (Basic operators). Standard notation is used for these basic functions:

$\mathbb{E}\{x\}$ Expectation with respect to some (implicit) measure. Often used to indicate the empirical mean of some stochastic process.

$\mathsf{corr}\{x\}$	Correlation matrix.
$\mathsf{corr}(x, y)$	Correlation between two random variables.
$\mathrm{cov}\{x\}$	Covariance matrix.
$\mathcal{H}(x)$	Entropy of a random variable.
$\mathcal{I}(x; y)$	Mutual information of two random variables.

DEFINITION B.10 (Spearman correlation). The *Spearman correlation* is the correlation of the rank statistics:

$$\mathsf{spear}(x, y) = \mathsf{corr}(\mathsf{order}(x), \mathsf{order}(y)).$$

LEMMA B.11 (Properties of Spearman correlation). *The Spearman correlation is invariant to monotonic transformations of the arguments. For all* $f \in \mathsf{Homeo}_+(\mathbb{R})$, $\mathsf{spear}(f(x), y) = \mathsf{spear}(x, y)$.

REMARK B.12 (Invariance to parametrization of entropy and derived quantities). In the discrete case, the entropy of a random variable is invariant with respect to any automorphism of the values. For all $f \in \mathsf{Aut}(\mathcal{X})$, $\mathcal{H}(x) = \mathcal{H}(f(x))$. In the continuous case, there are analogous invariance properties, but there is a subtle issue. Consider a one-dimensional random variable. Entropy is defined as an integral of the probability measure $\mu \in \mathsf{ProbMeasures}(\mathbb{R})$. According to the definition of integral that is being used, the push-forward of the probability measure $f_*\mu$ might or might not be defined for all invertible maps $f \in \mathsf{Aut}(\mathbb{R})$. For example, consider the invertible map

$$\begin{array}{rcl} f : \mathbb{R} & \to & \mathbb{R} \\ x & \mapsto & \begin{cases} +x & \text{if } x \in \mathbb{Q}, \\ -x & \text{if } x \notin \mathbb{Q}. \end{cases} \end{array}$$

For the case of Lebesgue–Stieltjes integration, we will say that the entropy is invariant to piecewise continuous bijections $\mathsf{PieceHomeo}(\mathbb{R})$ (Definition E.7).

The mutual information is not a metric because it does not satisfy the triangle inequality. The following related quantity is a proper metric on probability measures [141].

DEFINITION B.13 (Variation of information). The *variation of information* for two random variables X and Y is the function $\mathcal{V}(X;Y)$ defined by

$$\mathcal{V}(X;Y) = \mathcal{H}(\langle X, Y\rangle) - \mathcal{I}(X;Y).$$

It is also useful to define the *normalized variation of information* $\mathcal{V}_1(X;Y)$, which is bounded by 1:

$$\mathcal{V}_1(X;Y) = \frac{\mathcal{V}(X;Y)}{\mathcal{H}(\langle X, Y\rangle)} \leq 1.$$

APPENDIX C

Glossary of Basic Group Theory

General references

This appendix summarizes basic definitions and results from group theory that are referenced in the text. This makes this dissertation self-contained; however, the reader is strongly encouraged to read one of the superb introductions to the field.

A graduate-level introductory text to group theory is Rotman [53]; some of the definitions below report the page number where they can be found in that text. Also Robinson [142] is very accessible.

For probability applied to groups and other algebraic structures, see Grenander [143] and Diaconis [144]. See Chirikjian [145, 146] for an hands-on approach on how to compute on Lie groups.

C.1. Groups

DEFINITION C.1 (Semigroup). A *semigroup* is a set $\mathcal{X}$ with an associative operation $*$.

DEFINITION C.2 (Monoid). A *monoid* is a semigroup with an identity element $e \in \mathcal{X}$ such that, for all $x \in \mathcal{X}$, $x * e = e * x = x$.

LEMMA C.3. *If an element of a monoid has a left and a right inverse, then they coincide.*

DEFINITION C.4 (Group). A group $(G, \circ)$ is a set G equipped with an operation $\circ : G \times G \to G$ that satisfies four properties*:

*Note that these four conditions are slightly redundant.

closure For any $g, h \in G$, $g \circ h$ belongs to G.

associativity For any $g, h, k \in G$, $(g \circ h) \circ k = g \circ (h \circ k)$.

identity There exists a unique element $e \in G$ such that, for all $g \in G$, $e \circ g = g \circ e = g$.

invertibility For all $g \in G$, there exists a unique element g^{-1} in G such that $g \circ g^{-1} = g^{-1} \circ g = e$.

Equivalently, a group is a monoid where each element has an inverse. Using the multiplicative notation, the group operation is omitted, writing "$g\,h$" in place of "$g \circ h$".

DEFINITION C.5 (Subgroup). A subset H of a group G is a *subgroup* of G (written "$H \leq G$") if it is closed with respect to the group operation and inversion:

$$h_1, h_2 \in H \quad \Rightarrow \quad h_1 \circ h_2 \in H,$$

$$h \in H \quad \Rightarrow \quad h^{-1} \in H.$$

$\mathsf{AllSubgroups}(G)$ is the lattice of all subgroups of G.

DEFINITION C.6 (Coset). Given an element g of a group G and a subgroup $H \leq G$, the set

$$g\,H = \{\, g\,h \mid h \in H \}$$

is called a *left coset of H in G*. Symmetrically, the set

$$H\,g = \{\, h\,g \mid h \in H \}$$

is called a *right coset*. In general, $g\,H \neq H\,g$.

DEFINITION C.7 (Product of subgroups). Given $H, K \leq G$, the product $H\,K$ is defined as $\{\, h\,k \mid h \in H,\, k \in K \,\}$. In general, the set $H\,K$ is not a subgroup of G.

DEFINITION C.8 (Generated subgroup $\langle\cdot\rangle$). Given a group G and a subset $X \subset G$, define $\langle X\rangle$ as the smallest subgroup of G containing X [53, p. 22].

DEFINITION C.9 (Group join "$\vee$"). For two subgroups $H, K \leq G$, define the "$\vee$" operator as $H \vee K = \langle H \cup K\rangle$.

DEFINITION C.10 (Complement of a subgroup [53, p. 167]). The *complement* of a subgroup $K \leq G$ is a subgroup $Q \leq G$ such that $K \cap Q = e$ and $KQ = G$. In general, a subgroup need not have a complement, and if one exists, it is not necessarily unique.[†]

C.2. Normal Subgroups

DEFINITION C.11 (Normal subgroup). A subgroup $K \leq G$ is a *normal subgroup* of G (written "$K \triangleleft G$") if $g^{-1} K g = K$ for every $g \in G$.

LEMMA C.12 (Left and right cosets of a normal subgroup). *An alternative definition is that if $K \triangleleft G$, then $Kg = gK$, therefore there is no distinction between left and right cosets for a normal subgroup.*

LEMMA C.13 (Normality is preserved by intersection). *$K_1, K_2 \triangleleft G$ implies $K_1 \cap K_2 \triangleleft G$.*

LEMMA C.14 (Normality is not a transitive relation). *$H \triangleleft K$ and $K \triangleleft G$ do not imply $H \triangleleft G$.*

LEMMA C.15 (Normality is preserved by conjugation). *$K \triangleleft G \Leftrightarrow gKg^{-1} \triangleleft G$.*

LEMMA C.16 (Normality is preserved by subgroup product). *The product of two normal subgroups is a normal subgroup: if $K_1 K_2 \triangleleft G$ then $K_1 K_2 = K_2 K_1 \triangleleft G$.*

[†]Groups for which subgroups always have a complement are called *complemented* or *completely factorizable.*

C.3. Homomorphisms

DEFINITION C.17 (Group homomorphism). A *homomorphism* is a map $\varphi : G \to H$ between two groups $(G, \circ)$ and $(H, \diamond)$ that satisfies, for all $g, h \in G$,

$$\varphi(g \circ h) = \varphi(g) \diamond \varphi(h).$$

REMARK. A "homomorphism" is different from a "homeomorphism".

DEFINITION C.18 (Group isomorphism). A *group isomorphism* is a homomorphism which is also a bijection. Two groups G and H are *isomorphic* (written $G \cong H$) if there exists a group isomorphism between the two groups.

DEFINITION C.19 (Group endomorphism). An *endomorphism* is a homomorphism from G to itself.

DEFINITION C.20 (Kernel of a group homomorphism). The *kernel* of a group homomorphism $\varphi : G \to H$, written as $\ker \varphi$, is the subset of G that maps to the identity of H:

$$\ker \varphi = \{g \in G \mid \varphi(g) = e\}.$$

LEMMA C.21. *The kernel of a homomorphism is a normal subgroup.*

DEFINITION C.22 (Zero homomorphism). Given two groups G, H, the "zero homomorphism" $0 : G \to H$ maps an entire group to the identity: $0(g) = e$.

DEFINITION C.23 (Conjiugation). Let G be a group and fix an element $x \in G$. Then the *conjiugation* operation is defined as:

$$\begin{aligned} \mathsf{conj}_x : G &\mapsto G \\ g &\mapsto x\, g\, x^{-1}. \end{aligned}$$

Conjugation is a homomorphism:

$$\mathsf{conj}_x(g)\,\mathsf{conj}_x(h) = (x\, g\, x^{-1})(x\, h\, x^{-1}) = \mathsf{conj}_x(g\, h).$$

LEMMA C.24 (Conjugation preserves subgroups). *If* $H \leq G$, *then* $\mathsf{conj}_x(H) = \{\, x\, h\, x^{-1} \mid h \in H \,\}$ *is a subgroup of* G.

DEFINITION C.25 (Class function). A *class function* $f : G \to \mathcal{Y}$ is a function on a group G that is constant on the conjugacy classes:

$$f(x\, h\, x^{-1}) = f(h).$$

C.4. Quotients

DEFINITION C.26 (Direct product). Given two groups H, K, the *direct product* $H \times K$ is a group formed by the ordered pairs (h, k), with $h \in H$, $k \in K$, and with operation given by $(h_1, k_1)(h_2, k_2) = (h_1 h_2, k_1 k_2)$.

Note that, in general, only one of them being normal ($H \leq G$, $K \triangleleft G$) is not sufficient for G being homomorphic to the direct product. But in that case, G is homomorphic to the semidirect product (see later).

DEFINITION C.27 (Semidirect product). G is the *semidirect product* of $N \triangleleft G$ and $H \leq G$, written as $G = N \rtimes H$, if $HN = G$ (or $NH = G$) and $H \cap N = e$.

LEMMA C.28. *If $H \triangleleft G$, then $G \cong H \rtimes Q$ for a subgroup $Q \cong G/H$.*

LEMMA C.29. *If $G \cong N \rtimes H$, then every element of G can be written uniquely as the product of one element of N and one of H (or, uniquely as the product of one element from H and one from N).*

DEFINITION C.30 (Quotient group). If $N \triangleleft G$, then the set of cosets of N in G form a group called *quotient group*, which is indicated as $G \ / \ N$.

DEFINITION C.31 (Simple group). A group is *simple* if its only normal subgroups are the trivial group $\{e\}$ and the group itself.

If a group G is not simple, then it has a nontrivial normal subgroup $N \triangleleft G$, and it can be factored as a semidirect product $G \cong N \rtimes Q$. In some sense, the adjective "simple" is a misnomer, because if a group is "simple", then it has some irreducible complexity.

C.5. Natural Projections

If N is a normal subgroup, we can define the "natural projection map" π which maps an element of G to the corresponding coset of N in G.

DEFINITION C.32 (Natural projection map). Let $N \triangleleft G$. Then the natural projection map defined as

$$\begin{aligned} \pi : G &\to G/N, \\ g &\mapsto Ng, \end{aligned}$$

is a surjective homomorphism with kernel N.

It is also useful to define a different kind of projection from G to G.

DEFINITION C.33. Given a subgroup $Q \leq G$ isomorphic to the quotient ($Q \cong G/N$) we can define the projection map

$$\begin{aligned} \gamma : G &\to Q, \\ g = nq &\mapsto q. \end{aligned}$$

The definition takes into account that because $N \triangleleft G$, each element of G can be written in a unique way as nq with $n \in N$ and $q \in Q$ (Lemma C.29).

LEMMA C.34. *Properties of the projection map γ:*

(1) *γ is constant on N-cosets: if $n \in N$, then $\gamma(ng) = \gamma(g)$.*

(2) *γ is the identity on Q: if $q \in Q$, then $\gamma(q) = q$.*

(3) *For any $q \in Q$, $\gamma(gq) = \gamma(g)q$.*

C.6. Group Actions

DEFINITION C.35 (Right action of a group). A group G *acts* on a set $\mathcal{X}$ if there is a binary operation $\cdot : G \times \mathcal{X} \to \mathcal{X}$ that satisfies:

(1) For all $x \in \mathcal{X}$, $e \cdot x = x$ (identity).

(2) For all $x \in \mathcal{X}$ and $g, h \in G$, $g \cdot (h \cdot x) = (gh) \cdot x$ (associativity).

DEFINITION C.36 (Orbit). The *orbit* of x under G is the set

$$G \cdot x \triangleq \{g \cdot x \mid g \in G\},$$

DEFINITION C.37 (Types of group actions). Let G be a group acting on a non-empty set $\mathcal{X}$. Then the action can have several attributes:

Transitive The action is *transitive* if, for any $x, y \in \mathcal{X}$, there exists a $g \in G$ (not necessarily unique) such that $g \cdot x = y$. This implies that $G \cdot x = \mathcal{X}$ for some $x \in \mathcal{X}$; and also that $G \cdot x = \mathcal{X}$ for any x.

Faithful (or effective) The action is *faithful* if, for any two distinct $g, h \in G$, there exists an $x \in \mathcal{X}$ such that $g \cdot x \neq h \cdot x$. Equivalently, for any $g \neq e \in G$ there exists an $x \in \mathcal{X}$ such that $g \cdot x \neq x$.

Free (or semiregular) The action is *free* if $g \cdot x = h \cdot x$ for all $x \in \mathcal{X}$ implies that $g = h$. Equivalently: if there exists an $x \in \mathcal{X}$ such that $g \cdot x = x$, then necessarily $g = e$. (There are no fixed points for any g, except the identity).

Regular (or simply transitive) The action is *regular* if it is both free and transitive. This implies that, for any two $x, y \in \mathcal{X}$, there exists exactly one group element $g \in G$ such that $g \cdot x = y$.

These properties are summarized in Table C.1.

LEMMA C.38 (Faithful action of the factor subgroup). *Suppose that G acts on $\mathcal{X}$. Define the set $N = \{g \in G \mid g \cdot x = x \text{ for all } x \in \mathcal{X}\}$. Then N is a normal subgroup of G, and the action of the factor group G/N, defined as*

$$(Ng) \cdot x = g \cdot x,$$

is a faithful action on $\mathcal{X}$.

Table C.1. Types of group actions

type of group action	*property*
transitive	$G \cdot x = \mathcal{X}$ for any $x \in \mathcal{X}$.
faithful	If $g \neq e$, then $g \cdot x \neq x$ for some $x \in \mathcal{X}$.
free	$g \cdot x = x$ implies $g = e$.
regular	For any $x, y \in \mathcal{X}$, there exists exactly one g such that $g \cdot x = y$.

C.7. Invariance of Sets and Functions

DEFINITION C.39 (Stabilizer). Suppose the group G acts on $\mathcal{X}$. Then the *stabilizer* of a point $x \in \mathcal{X}$ is the subset of G that leave x fixed:

$$\mathsf{stab}^G(x) = \{g \in G \mid g \cdot x = x\}.$$

Likewise, the stabilizer of a subset $\mathcal{A} \subset \mathcal{X}$ is the set of elements that leave the subset invariant:

$$\mathsf{stab}^G(\mathcal{A}) = \{g \in G \mid g \cdot \mathcal{A} = \mathcal{A}\}.$$

The stabilizer is a subgroup of G, but not necessarily a normal subgroup.

Given a set $\mathcal{X}$ and a group G acting on $\mathcal{X}$, a function $f : \mathcal{X} \to \mathcal{Y}$ is G-invariant if $f(x) = f(g \cdot x)$ for all $x \in \mathcal{X}$ and $g \in G$.

DEFINITION C.40 (Symmetries of a function). Given a group G acting on $\mathcal{X}$, and a function f with domain $\mathcal{X}$, $\mathsf{Sym}^G(f)$ is the largest subgroup of G to which the function is invariant:

$$\begin{aligned} \mathsf{Sym}^G : \mathsf{Functions}(\mathcal{Y}; \mathcal{X}) &\to \mathsf{AllSubgroups}(G) \\ f &\mapsto \{\, g \in G \mid \forall x :\ f(g \cdot x) = f(x) \,\}. \end{aligned}$$

DEFINITION C.41 (Equivariant function). Given a set $\mathcal{X}$ and a group G acting on $\mathcal{X}$, a function $f : \mathcal{X} \to \mathcal{X}$ is *G-equivariant* if it commutes with the action of G:

$$f(g \cdot x) = g \cdot f(x),$$

for all $x \in \mathcal{X}$ and $g \in G$.

DEFINITION C.42 (Contravariant function). Given a set $\mathcal{X}$ and a group G acting on $\mathcal{X}$, a function $f : \mathcal{X} \to \mathcal{X}$ is *G-contravariant* if

$$f(g \cdot x) = g^{-1} \cdot f(x),$$

for all $x \in \mathcal{X}$ and $g \in G$.

C.8. Lie Groups

DEFINITION C.43 (Topological group). A *topological group* is group endowed with a topology such that the group operation and inversions are continuous maps.

DEFINITION C.44 (Lie group). A *Lie group* is a group which is also a smooth manifold, and in which the group operation and inversion are smooth.

APPENDIX D

Group Bestiary

DEFINITION D.1 (Simple examples of groups). Standard notation is used to denote the following groups.

Id	The identity group.
$(\mathbb{R}, +)$	Additive group.
$(\mathbb{R}_\circ, \times)$	Nonzero reals with multiplication.
$(\pm 1, \times)$	$\{-1, +1\}$.
$(\mathbb{R}^+_\circ, \times)$	Positive reals.
$\mathsf{Perm}(n)$	Permutations of n elements.

D.1. Matrix Groups

See Figure E.2 for a summary of the relations among these matrix groups.

DEFINITION D.2 (Linear transformations and subgroups). Standard notation is used to indicate:

$\mathsf{GL}(n)$ The group $\mathsf{GL}(n)$ is the set of $n \times n$ matrices that have nonzero determinants:

$$\mathsf{GL}(n) = \{\mathbf{A} \in \mathbb{R}^{n \times n} \mid \det(\mathbf{A}) \neq 0\}.$$

$\mathsf{GL}_+(n)$ The subgroup $\mathsf{GL}_+(n)$ is the component with positive determinant:

$$\mathsf{GL}_+(n) = \{\mathbf{A} \in \mathsf{GL}(n) \mid \det(\mathbf{A}) > 0\}.$$

$\mathsf{SL}(n)$ Special linear group:

$$\mathsf{SL}(n) = \{\mathbf{A} \in \mathbb{R}^{n\times n} \mid \det(\mathbf{A}) = 1\}.$$

$\mathsf{O}(n)$ Orthogonal group:

$$\mathsf{O}(n) = \{\mathbf{A} \in \mathbb{R}^{n\times n} \mid \mathbf{A}\mathbf{A}^T = \mathbf{A}^T\mathbf{A} = \mathbf{I}_n\}.$$

$\mathsf{SO}(n)$ Special orthogonal group (rotations):

$$\mathsf{SO}(n) = \{\mathbf{A} \in \mathsf{O}(n) \mid \det \mathbf{A} = 1\}.$$

Nonstandard notation is used for the subgroups:

$\mathsf{D}(n)$ This is the group of diagonal matrices with nonzero elements:

$$\mathsf{D}(n) = \left\{ \begin{pmatrix} \alpha_1 & 0 & 0 \\ & \ddots & 0 \\ 0 & 0 & \alpha_n \end{pmatrix} \mid \alpha_i \in \mathbb{R}_\circ \right\}.$$

$\mathsf{D}_\pm(n)$ This is the group of diagonal matrices with elements equal to ± 1:

$$\mathsf{D}_\pm(n) = \left\{ \begin{pmatrix} \alpha_1 & 0 & 0 \\ & \ddots & 0 \\ 0 & 0 & \alpha_n \end{pmatrix} \mid \alpha_i \in \{-1, +1\} \right\}.$$

$\mathsf{Sc}(n)$ This group contains the multiples of the identity:

$$\mathsf{Sc}(n) = \{\alpha\, \mathbf{I}_n \mid \alpha \in \mathbb{R}_\circ\}.$$

DEFINITION D.3 (Affine transformations). $\mathsf{Aff}(\mathbb{R}^n)$ The affine group is the set of affine

transformations (translations and invertible scaling) of $\mathbb{R}^n$, which can be represented using $(n+1) \times (n+1)$ matrices:

$$\left\{ \begin{bmatrix} \mathbf{A} & b \\ 0 & 1 \end{bmatrix}, \ \mathbf{A} \in \mathsf{GL}(n), \ b \in \mathbb{R}^n \right\}.$$

Write an element of the group as a tuple $(\mathbf{A}, b)$, with $\mathbf{A} \in \mathsf{GL}(n)$ and $b \in \mathbb{R}^n$. The group operation is written as $(\mathbf{A}_1, b_1) \cdot (\mathbf{A}_2, b_2) = (\mathbf{A}_1\mathbf{A}_2, \mathbf{A}_2 b_1 + b_2)$.

$\mathsf{T}(n)$ Translations of $\mathbb{R}^n$:

$$\mathsf{T}(n) = \left\{ \begin{bmatrix} 0 & b \\ 0 & 1 \end{bmatrix}, \ b \in \mathbb{R}^n \right\} \cong (\mathbb{R}, +)(\mathbb{R}^n).$$

$\mathsf{Aff}(\mathbb{R}^n)$ is the semidirect product of $\mathsf{T}(n)$ and $\mathsf{GL}(n)$:

$$\mathsf{Aff}(\mathbb{R}^n) \ \cong \ \mathsf{T}(n) \rtimes \mathsf{GL}(n).$$

$\mathsf{E}(n)$ The Euclidean group:

$$\mathsf{E}(n) = \left\{ \begin{bmatrix} \mathbf{A} & b \\ 0 & 1 \end{bmatrix}, \ \mathbf{A} \in \mathsf{O}(n), b \in \mathbb{R}^n \right\}.$$

$\mathsf{SE}(n)$ The Special Euclidean group:

$$\mathsf{SE}(n) = \left\{ \begin{bmatrix} \mathbf{A} & b \\ 0 & 1 \end{bmatrix}, \ \mathbf{A} \in \mathsf{SO}(n), \ b \in \mathbb{R}^n \right\}.$$

D.1.1. *Supporting lemmas*

DEFINITION D.4 (Commuting group of a matrix). Fixed a matrix $\mathbf{A} \in \mathbb{R}^n$ (not necessarily invertible), the set of matrices commuting with $\mathbf{A}$ form a group

$$\mathsf{Com}_{\mathbf{A}} = \{\mathbf{X} \in \mathsf{GL}(n) \mid \mathbf{XA} = \mathbf{AX}\} \leq \mathsf{GL}(n).$$

PROOF. In fact, the group if closed with respect to composition: if $\mathbf{XA} = \mathbf{AX}$ and $\mathbf{YA} = \mathbf{AY}$, then

$$\begin{aligned}(\mathbf{XY})\mathbf{A} &= \mathbf{X}(\mathbf{YA}) = \mathbf{X}(\mathbf{AY}) \\ &= (\mathbf{XA})\mathbf{Y} = (\mathbf{AX})\mathbf{Y} = \mathbf{A}(\mathbf{XY}).\end{aligned}$$

Moreover, if $\mathbf{XA} = \mathbf{AX}$, then by multiplying by $\mathbf{X}^{-1}$ on the left and right, we obtain $\mathbf{X}^{-1}\mathbf{A} = \mathbf{AX}^{-1}$. □

LEMMA D.5. *The function $f(y) = \|\mathbf{A}y\|_2$, for $\det \mathbf{A} \neq 0$, is invariant to the action of the subgroup*

$$\alpha_{\mathbf{A}}(\mathsf{O}(n)) = \{\mathbf{A}^{-1}\mathbf{XA} \mid \mathbf{X} \in \mathsf{O}(n)\} \leq \mathsf{GL}(n).$$

PROOF. The set $\alpha_{\mathbf{A}}(\mathsf{O}(n))$ is a subgroup of $\mathsf{GL}(n)$ because it is the conjugated subgroup of $\mathsf{O}(n)$ (Lemma C.24). For $\mathbf{X} \in \mathsf{O}(n)$, consider the map $\varphi_{\mathbf{X}} : y \mapsto \mathbf{A}^{-1}\mathbf{XA}y$. Then $f(\varphi_{\mathbf{X}}(y)) = \|\mathbf{A}\varphi_{\mathbf{X}}(y)\|_2 = \|\mathbf{AA}^{-1}\mathbf{XA}y\|_2 = \|\mathbf{XA}y\|_2 = \|\mathbf{A}y\|_2 = f(y)$. □

LEMMA D.6 (Sylvester). *The number of positive and negative eigenvalues are a complete set of invariants of the action of $\mathsf{GL}(n)$ on $\mathbb{R}^{n\times n}$ defined as $\mathbf{X} \mapsto \mathbf{AXA}^T$ [147, example 2.26].*

LEMMA D.7. *A* symmetric function *of n values is any function which is invariant to the permutation of the values. The action of $\mathsf{O}(n)$ on $\mathbb{R}^{n\times n}$ preserves all symmetric function of the eigenvalues. [147, example 2.31].*

LEMMA D.8. *The group* $(\mathbb{R}_\circ, \times)$ *can be factorized to the direct product* $(\pm 1, \times) \times (\mathbb{R}_\circ^+, \times)$. *Equivalently, both* $(\pm 1, \times)$ *and* $(\mathbb{R}_\circ^+, \times)$ are normal subgroups of $(\mathbb{R}_\circ, \times)$.

PROOF. It is easy to see $(\pm 1, \times) \vee (\mathbb{R}_\circ^+, \times)$, as each nonzero number can be factorized uniquely in sign times magnitude. Because $(\mathbb{R}_\circ, \times)$ is Abelian, all subgroups are normal subgroups. □

All of the following proofs are elementary and based on the definition of normal subgroup: a subgroup $K \leq G$ is a normal subgroup of G if, for any $g \in G$, $gKg^{-1} = K$.

LEMMA D.9. *The group* $(\pm 1, \times)$ *is not a normal subgroup of* $\mathsf{Aff}(\mathbb{R})$.

PROOF. In this case, $K = \{\left(\begin{smallmatrix} \pm 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)\}$, $g = \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)$:

$$\begin{aligned} gKg^{-1} &= \left\{ \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} \pm 1 & 0 \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \right\} \\ &= \left\{ \left(\begin{smallmatrix} \pm & \mp b + b \\ 0 & 1 \end{smallmatrix}\right) \right\} \neq K. \end{aligned}$$

□

LEMMA D.10. *The group* $(\mathbb{R}, +) \rtimes (\pm 1, \times)$ *is a normal subgroup of* $\mathsf{Aff}(\mathbb{R})$.

PROOF. By Lemma D.13, $(\mathbb{R}, +) \vee (\pm 1, \times) \cong (\mathbb{R}, +) \rtimes (\pm 1, \times)$. In this case, the subgroup K is $K = \{\left(\begin{smallmatrix} \pm 1 & t \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R}\}$. Let $g = \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right), a \in \mathbb{R}_\circ, b \in \mathbb{R}$ be any fixed element of G.

Then

$$\begin{aligned} gKg^{-1} &= \left\{ \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} \pm 1 & t \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \mid t \in \mathbb{R} \right\} \\ &= \left\{ \left(\begin{smallmatrix} \pm 1 & \mp b + at + b \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R} \right\} \\ &= \left\{ \left(\begin{smallmatrix} 1 & at \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R} \right\} \cup \left\{ \left(\begin{smallmatrix} -1 & at - 2b \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R} \right\} \\ &= \left\{ \left(\begin{smallmatrix} 1 & t \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R} \right\} \cup \left\{ \left(\begin{smallmatrix} -1 & t \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R} \right\} \\ &= K. \end{aligned}$$

So $(\mathbb{R}, +) \rtimes (\pm 1, \times)$ is a normal subgroup. Its quotient is $(\mathbb{R}_\circ^+, \times)$, which is not normal in $\mathsf{Aff}(\mathbb{R})$ (Lemma D.12). □

LEMMA D.11. *$(\mathbb{R}_\circ, \times)$ is not a normal subgroup of $\mathsf{Aff}(\mathbb{R})$.*

PROOF. This would mean that for $K = \{ \left(\begin{smallmatrix} s & 0 \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ \}$,

$$\begin{aligned} gKg^{-1} &= \left\{ \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} s & 0 \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \mid s \in \mathbb{R}_\circ \right\} \\ &= \left\{ \begin{pmatrix} s & b(1-s) \\ 0 & 1 \end{pmatrix} \mid s \in \mathbb{R}_\circ \right\} \neq K. \end{aligned}$$

□

LEMMA D.12. *$(\mathbb{R}_\circ^+, \times)$ is not a normal subgroup of $\mathsf{Aff}(\mathbb{R})$.*

PROOF. This would mean that for $K = \{ \left(\begin{smallmatrix} s & 0 \\ 0 & 1 \end{smallmatrix}\right) \mid s > 0 \}$,

$$\begin{aligned} gKg^{-1} &= \left\{ \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} s & 0 \\ 0 & 1 \end{smallmatrix}\right) \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \mid s > 0 \right\} \\ &= \left\{ \begin{pmatrix} s & b(1-s) \\ 0 & 1 \end{pmatrix} \mid s > 0 \right\} \neq K. \end{aligned}$$

This also follows directly from Lemma D.11 □

LEMMA D.13. $(\mathbb{R}, +)$ is a *normal subgroup* of $(\mathbb{R}, +) \vee (\pm 1, \times)$.

PROOF. In this case, $K = \{\left(\begin{smallmatrix} 1 & t \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R}\}$, and $g = \left(\begin{smallmatrix} \pm 1 & b \\ 0 & 1 \end{smallmatrix}\right)$:

$$\begin{aligned} gKg^{-1} &= \left\{\left(\begin{smallmatrix} \pm 1 & b \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & t \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} \pm 1 & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \mid t \in \mathbb{R}\right\} \\ &= \{\left(\begin{smallmatrix} 1 & \pm t \\ 0 & 1 \end{smallmatrix}\right) \mid t \in \mathbb{R}\} = K. \end{aligned}$$

□

LEMMA D.14. $\left\{\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ\right\}$ is a subgroup of $\mathsf{Aff}(\mathbb{R})$, but not a normal subgroup.

PROOF. Verify that H is closed with respect to composition:

$$\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} x & -x+1 \\ 0 & 1 \end{smallmatrix}\right) = \left(\begin{smallmatrix} sx & s(1-x))-s+1 \\ 0 & 1 \end{smallmatrix}\right) = \left(\begin{smallmatrix} sx & -sx+1 \\ 0 & 1 \end{smallmatrix}\right) \in H$$

and closed with respect to inversion:

$$\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)^{-1} = \left(\begin{smallmatrix} 1/s & -1/s+1 \\ 0 & 1 \end{smallmatrix}\right) \in H.$$

However, it is not a normal subgroup of $\mathsf{Aff}(\mathbb{R})$:

$$\begin{aligned} gHg^{-1} &= \left\{\left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} s & -s+1 \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^{-1} \mid s \in \mathbb{R}_\circ\right\} \\ &= \left\{\left(\begin{smallmatrix} s & (-s+1)(a+b) \\ 0 & 1 \end{smallmatrix}\right) \mid s \in \mathbb{R}_\circ\right\} \neq H. \end{aligned}$$

□

LEMMA D.15. $\det : \mathsf{GL}(n) \to (\mathbb{R}_\circ, \times)$ is a group homomorphism.

LEMMA D.16. *The symmetry group of* $\|\cdot\|_1$ is $\mathsf{D}_\pm(n) \times \mathsf{Perm}(n)$.

LEMMA D.17. *The symmetry group of* $\|\cdot\|_\infty$ is $\mathsf{D}_\pm(n) \times \mathsf{Perm}(n)$.

APPENDIX E

Geometry

General references

"Geometry and the imagination" [148], written by Hilbert himself, will convince you of the beauty of differential geometry as well as groups

A simple introduction to differential geometry is given by Do Carmo [149]. Spivak [150] is a commonly used text. Abraham, Marsden, and Ratiu give the definitive formal treatment [151]. For geometry and mechanics, see Marsden and Ratiu [152].

For information geometry (manifolds of probability distributions), see Amari [153], Amari and Nagaoka [154], and Murray and Rice [155].

E.1. Metric Spaces

DEFINITION E.1 (Metric space). A metric space $(\mathcal{M}, d)$ is a set $\mathcal{M}$ equipped with a function $d : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+_\bullet$ (called *metric* or *distance*), that satisfies the properties:

(1) $d(a,b) = 0 \Leftrightarrow a = b$,

(2) $d(a,b) = d(b,a)$ (symmetry),

(3) $d(a,c) \leq d(a,b) + d(b,c)$ (triangle inequality).

DEFINITION E.2. The *radius* of a set $\mathcal{S} = \{s_i\}$ in a metric space is defined as

$$\mathsf{rad}(\mathcal{S}) \triangleq \min_i \max_j d(s_i, s_j). \tag{E.1}$$

The *diameter* is twice the radius.

DEFINITION E.3 (Isometries). An *isometry* of a metric space $(\mathcal{M}, d)$ is a mapping $f : \mathcal{M} \to \mathcal{M}$ that preserves the metric: $d(x, y) = d(f(x), f(y))$.

$\mathsf{Isom}(\mathcal{M})$ is the group of all isometries.

DEFINITION E.4 (ϵ–δ definition of continuous functions). A function $f : \mathcal{X} \to \mathcal{Y}$ between two metric spaces $\mathcal{X}$ and $\mathcal{Y}$ is *continuous* at a point $\overline{x}$ if, for all $\epsilon > 0$, there exists a $\delta(\epsilon) > 0$ such that $d^{\mathcal{X}}(x, \overline{x}) < \delta(\epsilon)$ implies $d^{\mathcal{Y}}(f(x), f(\overline{x})) < \epsilon$.

$\mathsf{Continuous}(\mathcal{Y}; \mathcal{X})$ is the set of all continuous functions from $\mathcal{X}$ to $\mathcal{Y}$. $\mathsf{Continuous}(\mathcal{X})$ is the set of all continuous functions from $\mathcal{X}$ to itself.

See also the topological definition of continuity (Definition A.4).

DEFINITION E.5 (Homeomorphism). A *homeomorphism* of a set $\mathcal{M}$ is a continuous bijection $f : \mathcal{M} \to \mathcal{M}$ such that its inverse f^{-1} is continuous.

$\mathsf{Homeo}(\mathcal{M})$ is the set of all homeomorphisms of $\mathcal{M}$.

REMARK E.6. A homeomorphism is different from a homomorphism.

DEFINITION E.7. $\mathsf{PieceHomeo}(\mathbb{R})$ is the set of piecewise continuous bijections of $\mathbb{R}$.

E.2. Linear Algebra

Bhatia [156] is a good reference for linear algebra.

DEFINITION E.8 (Angle function). We define the map $\angle$ as follows:

$$\begin{aligned} \angle : \mathbb{R}^n &\to \mathbb{S}^{n-1}, \\ \boldsymbol{x} &\mapsto \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2}. \end{aligned}$$

For $n = 2$, we recover the map $\angle : \mathbb{R}^2 \to \mathbb{S}^1$ from the plane to the unit circle $\mathbb{S}^1$.

The exponential of a square matrix is defined as

$$\begin{aligned} \mathsf{mexp} : \mathbb{R}^{n\times n} &\to \mathsf{GL}_+(n), \\ \mathbf{A} &\mapsto \sum_{k=0}^{\infty} \frac{1}{k!}\mathbf{A}^k. \end{aligned}$$

The first few terms of the expansion are

$$\mathsf{mexp}(\mathbf{A}) = \mathbf{I} + \mathbf{A} + \frac{1}{2}\mathbf{A}^2 + \frac{1}{6}\mathbf{A}^3 + \cdots. \tag{E.1}$$

The exponential of a matrix satisfies the properties:

(1) $\det(\mathsf{mexp}(\mathbf{A})) = e^{\mathsf{Tr}\mathbf{A}}$.

(2) $\mathsf{mexp}(\mathbf{A}+\mathbf{B}) = \mathsf{mexp}(\mathbf{A})\,\mathsf{mexp}(\mathbf{B})$ if and only if $\mathbf{A}$ and $\mathbf{B}$ commute.

DEFINITION E.9 (Solving linear ODEs with the matrix exponential). The solution to $\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t)$ is $\boldsymbol{x}(t) = \mathsf{mexp}(\mathbf{A}t)\,\boldsymbol{x}(0)$.

LEMMA E.10 (Whitening). $\mathrm{cov}\{\boldsymbol{y}\} = \mathbf{I}_{n_y}$ *is a weak pontifical feature for the action of* $\mathsf{GL}(n_{\boldsymbol{y}})$ *on* $\mathcal{D}(\mathbb{R}^{n_{\boldsymbol{y}}}; \mathcal{U})$ *given by* $\boldsymbol{y} \mapsto \mathbf{A}\boldsymbol{y}$*, the corresponding heresy subgroup is the orthogonal group* $\mathsf{O}(n_{\boldsymbol{y}})$*, the canonization operator is* $\boldsymbol{y} \mapsto (\mathrm{cov}\{\boldsymbol{y}\}^{-1/2}\mathsf{O}(n_{\boldsymbol{y}}))\boldsymbol{y}$*, where* $\mathbf{A}^{-1/2}$ *is any solution of* $\mathbf{X}\mathbf{X}^T = \mathbf{A}$.

PROOF. Clearly this is just an application of whitening [157]. Let $\mathbf{P}$ be the covariance of $\boldsymbol{y}$. Then it transforms as $\mathbf{P} \mapsto \mathbf{A}\mathbf{P}\mathbf{A}^T$. We can verify that

$$\mathrm{cov}\{\mathrm{cov}\{\boldsymbol{y}\}^{-1/2}\boldsymbol{y}\} = \mathrm{cov}\{\boldsymbol{y}\}^{-1/2}\mathrm{cov}\{\boldsymbol{y}\}\mathrm{cov}\{\boldsymbol{y}\}^{-1/2} = \mathbf{I}_{n_y}.$$

□

REMARK E.11. Recall that the square root of a positive definite matrix is not unique; for example the square root is 2 is $\pm\sqrt{2}$. For a matrix, one can show that if $\mathbf{Q}\mathbf{Q}^T = \mathbf{P}$, then for any $\mathbf{X} \in \mathsf{O}(n)$, $(\mathbf{Q}\mathbf{X})(\mathbf{Q}\mathbf{X})^T = (\mathbf{Q}\mathbf{X}\mathbf{X}^T\mathbf{Q}^T) = (\mathbf{Q}\mathbf{Q}^T) = \mathbf{P}$. This ambiguity is perfectly captured by the heresy subgroup.

E.2.1. *Vector operations*

DEFINITION E.12. The "hat map" maps a vector in $\mathbb{R}^3$ to a skew-symmetric matrix:

$$\hat{\boldsymbol{a}} = \begin{pmatrix} 0 & -a^3 & a^2 \\ a^3 & 0 & -a^1 \\ -a^2 & a^1 & 0 \end{pmatrix}.$$

DEFINITION E.13 (Cross product). Given two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3$, the cross product $\boldsymbol{a} \times \boldsymbol{b}$ is defined as

$$\boldsymbol{a} \times \boldsymbol{b} \triangleq \hat{\boldsymbol{a}}\,\boldsymbol{b}.$$

E.2.2. *Linear algebra tricks*

LEMMA E.14. $a^T \hat{b}\, c = a^T(b \times c) = (b \times a)^T c$.

LEMMA E.15. For a rotation matrix $\mathbf{R} \in \mathsf{SO}(n)$, $\mathbf{R}^T = \mathbf{R}^{-1}$.

E.3. Manifolds

DEFINITION E.16 (Manifold nomenclature). See [151] for a formal definition of differentiable manifold. We use the following symbols:

$d\mathcal{M}$	This is the *volume form* associated to the manifold $\mathcal{M}$.
$\nabla_d f$	This is the *gradient tensor* for the differentiable function $f : \mathcal{M} \to \mathbb{R}$.
	$\nabla_d y$ is a $(0, \dim(\mathcal{M}))$ tensor at each point of $\mathcal{M}$.

DEFINITION E.17 (Simple manifolds). $\mathbb{S}^1$ Unit circle.

$\mathbb{S}^2$ Unit sphere.

$\mathbb{H}^n$ Hyperbolic space.

DEFINITION E.18 (Geodesic curve). A *geodesic curve* $g(A, B, t)$ from point A to point B, for $t \in [0, 1]$, is the curve on the manifold such that

$$\begin{aligned} d(g(A, B, t), A) &= t\, d(A, B), \\ d(g(A, B, t), B) &= (1 - t) d(A, B). \end{aligned}$$

In particular, $g(A, B, 0) = A$ and $g(A, B, 1) = B$.

DEFINITION E.19 (Geodesic convexity). A function $f : \mathcal{M} \to \mathbb{R}$ is *geodesically convex* if, for any geodesic curve $g(A, B, t)$, the composition $f \circ g : \mathbb{R} \to \mathbb{R}$ is a convex function.

E.4. Diffeomorphisms

DEFINITION E.20 (Diffeomorphism). A *diffeomorphism* of $\mathcal{M}$ is a continuously differentiable homeomorphism $f : \mathcal{M} \to \mathcal{M}$ such that its inverse f^{-1} is continuously differentiable. (Most often the degree of differentiability required is left implicit.)

$\mathsf{Diff}(\mathcal{M})$ is the set of all diffeomorphisms of $\mathcal{M}$. This is a topological group but not a Lie group.*

*Thanks to Ari Stern for clarifying this point.

DEFINITION E.21 (Conformal map). A diffeomorphism $f : \mathcal{M} \to \mathcal{M}$ on a Riemannian manifold $\mathcal{M}$ is *conformal* if the pulled back metric is conformally equivalent to the original metric. Roughly speaking, a conformal map preserves angles between geodesics on a manifold. A map is conformal if and only if its Jacobian is proportional to a rotation matrix.

$\mathsf{Conformal}(\mathcal{M})$ is the set of all conformal maps on $\mathcal{M}$.

DEFINITION E.22 (Orientation-preserving diffeomorphism). An orientation-preserving diffeomorphism is one for which $\det J > 0$.

$\mathsf{Diff}_+(\mathcal{M})$ is the group of all orientation-preserving diffeomorphisms.

DEFINITION E.23 (Point-fixing diffeomorphism). $\mathsf{Diff}_{[m]}(\mathcal{M})$ is the set of diffeomorphisms that fix $m \in \mathcal{M}$.

DEFINITION E.24 (Volume-preserving diffeomorphisms). $\mathsf{Diff}_{\mathrm{vol}}(\mathcal{M})$ is the set of volume-preserving diffeomorphisms [158].

Nomenclature

$\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots, \mathcal{M}, \mathcal{X}, \mathcal{Y}, \mathcal{Z}$	Symbols denoting sets.	
$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots$	Symbols used for denoting sequences.	
$a_k, b_k, c_k, \ldots$	Symbols denoting the elements of sequences.	
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$	Symbols denoting matrices.	
$\mathrm{A}, \mathrm{B}, \mathrm{C}, \ldots$	Symbols denoting matrix elements.	
$\mathsf{A}, \mathsf{B}, \mathsf{C}, \ldots$	Symbols denoting tensors.	
$\mathsf{A}_{ijk}, \mathsf{B}_{ijk}, \mathsf{C}_{ijk}, \ldots$	Symbols denoting tensors elements.	
$\mathrm{G}, \mathrm{H}, \mathrm{K}, \mathrm{N}, \ldots$	Symbols denoting groups.	
$\boldsymbol{D}, \boldsymbol{E}, \boldsymbol{F}, \ldots$	Names for dynamical systems.	
$\cdot^{-1}$	Inverse function.	→Definition A.24
$\star$	Joker function.	→Definition A.19
$(\pm 1, \times)$	+1/-1 multiplication group.	→Definition D.1
$\sim$	Unstructured symbol.	→Definition 15.1
$\angle$	Angle function.	→Definition E.8
$\mathbb{III}$	Accumulator system.	→Definition 4.22
$\square, \boxdot$	Uninterpreted bits values.	
近	The Chinese character corresponding to “close” or “near”.	→Section 3.3
$\boldsymbol{a} \times \boldsymbol{b}$	Cross product.	→Definition E.13
$\sim$	Distributed as.	
$i, j, k, s, v, \ldots$	Symbols usually used as indices.	
$\emptyset$	Empty sequence.	→Definition A.30

$f \circ g$	Function composition.	
$\sim$	Generic relation.	
$G \cong H$	Isomorphism relation.	→Definition C.18
$G \vee H$	Group join.	→Definition C.9
$G \,/\, H$	Group quotient.	→Definition C.30
$G \rtimes H$	Semidirect product of G and H.	→Definition C.27
$\xrightarrow{-1}$	Contravariance.	→Chapter 15
$\xrightarrow{\mathsf{Id}}$	Equivariance.	→Chapter 15
$\xrightarrow{\star}$	Nuisance introduced.	→Chapter 15
$\xrightarrow{0}$	Invariance.	→Chapter 15
$\xrightarrow{\sim}$	Unstructured result.	→Chapter 15
$f \parallel G$	The function f can be expressed as a group action of G.	→Definition 15.1
$\{\square, \boxdot\}^{\mathbb{N}}$	Set of infinite binary strings.	→Definition A.1
$[a, b]$	Closed interval $\{x \mid a \leq x \leq b\}$.	
(a, b)	Open interval $\{x \mid a < x < b\}$.	
$G \leq H$	Subgroup relation.	→Definition C.5
$\wedge$	Logic and.	→Section A.1
$\neg$	Logic not.	→Section A.1
$\vee$	Logic or.	→Section A.1
$G \lhd H$	Normal subgroup relation.	→Definition C.11
$\rightrightarrows$	Symbol for set functions (one-to-many).	→Definition A.18
$\mathcal{A}$	An agent.	→Definition 5.4
$\mathcal{A}_{\text{cmd}}$	Anti-distance between two commands.	→Chapter 13
act	Agent's action phase.	→Definition 5.4
$\mathsf{Aff}(\mathbb{R}^n)$	Affine transformations on $\mathbb{R}^n$.	→Definition D.3

$\mathsf{Agents}(\mathcal{Y};\mathcal{U})$	All agents with given formats.	
$\mathsf{AllOutcomes}(\boldsymbol{D})$	All sequences that can be produced by the $\mathsf{Loop}(\boldsymbol{FD})$ for any system $\boldsymbol{F}$.	
$\mathsf{AllOutputs}(\boldsymbol{D})$	All sequences that can be produced by the system $\boldsymbol{D}$.	→Definition 4.13
$\mathsf{AllSubgroups}(G)$	All subgroups of a group G.	→Definition C.5
$\mathsf{Aut}(\mathcal{X})$	Automorphisms of the set $\mathcal{X}$.	→Definition A.16
β	Nonlinear function in range-finder tensors.	
$\mathbf{B}$	Affine part of dynamics.	
$\mathsf{BDS}(n;k)$		→Definition 11.1
BGDS	Bilinear gradient dynamics system.	→Definition 12.1
BGDSagent		→Chapter 19
BGDSagentS		→Chapter 19
$\mathsf{BGDS}(\mathcal{S};k)$		→Definition 12.1
$C_{\mathcal{A}}$		→Definition 5.14
c	CDF of one sensel.	
$\mathbf{C}$	Learned affine part of dynamics.	
CalibA		→Chapter 19
CalibB		→Chapter 19
$\mathrm{CLTI}(n;k)$	Continuous-time linear time-invariant systems.	
$\mathsf{Codomain}(f)$	Co-domain of a function f.	→Definition A.13
$\mathsf{Com}_{\mathbf{A}}$	Subgroup of $\mathsf{GL}(n)$ commuting with $\mathbf{A}$.	→Definition D.4
$\mathsf{Conditional}(\mathcal{B};\mathcal{A})$	All conditional distributions from the $\mathcal{A}$ to set $\mathcal{B}$.	
$\mathsf{Conformal}(\mathcal{M})$	Conformal transformations of the manifold $\mathcal{M}$.	→Definition E.21
conj_x	Conjugation by the element x.	→Definition C.23
$\mathsf{Continuous}(\mathcal{B};\mathcal{A})$	Continuous functions from $\mathcal{A}$ to $\mathcal{B}$.	

$\mathsf{ContSequences}(\mathcal{X})$	Sequences on the space $\mathcal{X}$.	→Definition A.30
$\mathsf{corr}\{x\}$	Correlation matrix of x.	→Definition B.9
$\mathsf{cov}\{x\}$	Covariance matrix of x.	→Definition B.9
$\mathcal{D}^\star(\mathcal{U})$	Representation nuisances on observations.	→Definition 4.29
$\mathcal{D}^\star(\mathcal{Y})$	Representation nuisances on commands.	→Definition 4.29
$\mathcal{D}^\star(\mathcal{Y};\mathcal{U})$	Representation nuisances.	→Definition 4.36
$\mathcal{D}^\star_{\mathsf{fm}}(\mathcal{A})$	Systems with finite memory and invertible.	→Definition 4.31
$\mathcal{D}^\star_{\mathsf{inst}}(\mathcal{A})$	Invertible and instantaneous systems.	→Definition 4.30
$\mathcal{D}_{\mathsf{det}}(\mathcal{B};\mathcal{A})$	Deterministic systems.	→Definition 4.17
$\mathcal{D}_{\mathsf{fm}}(\mathcal{B};\mathcal{A})$	Systems with finite memory.	→Definition 4.19
$\mathcal{D}_{\mathsf{inst}}(\mathcal{B};\mathcal{A})$	Set of instantaneous systems.	
$\mathcal{D}^\circ(\mathcal{Y};\mathcal{U})$	Systems up to representation.	→Definition 4.38
$\mathcal{D}(\mathcal{B};\mathcal{A})$	Set of all black box systems with input in $\mathcal{A}$ and output in $\mathcal{B}$.	
d^{Diff}	Distance between two diffeomorphism.	→Chapter 13
$\mathbf{D}$ D_{ij}	Distance matrix.	→Chapter 10
d	Infinitesimal.	
$\mathsf{D}_\pm(n)$	Diagonal matrices with ± 1 on the diagonal.	→Definition D.2
$\mathcal{D}_{\mathsf{cmd}}$	Distance between two commands.	→Chapter 13
$d_{\max}$	Bound on the maximum diffeomorphism in a DDS.	→Chapter 13
$d^{\mathcal{S}}$	Metric on $\mathcal{S}$.	→Chapter 13
$\mathsf{DDSL}(\mathcal{S},\mathcal{V};\mathcal{U})$		→Definition 13.2
$\mathsf{DDS}(\mathcal{S};\mathcal{U})$		→Definition 13.1
Δ	The one-step delay system.	→Definition 4.21
$\mathsf{D}(n)$	Diagonal matrices with non-zero elements.	→Definition D.2
$\mathsf{Diff}_+(\mathcal{M})$	Diffeomeorphisms from $\mathcal{M}$ to itself that preserve the orientation.	→Definition E.22

$\mathsf{Diff}_{[x]}(\mathcal{M})$	Diffeomeorphisms from $\mathcal{M}$ to itself that fix a point x.	→Definition E.23
$\mathsf{Diff}(\mathcal{M})$	Diffeomeorphisms from $\mathcal{M}$ to itself.	→Definition E.20
$\mathsf{Differentiable}(\mathcal{B};\mathcal{A})$	Differentiable functions from $\mathcal{A}$ to $\mathcal{B}$.	
$\mathsf{differ}(\mathbb{R}, n)$	Subset of $\mathbb{R}^n$ for which they are all different.	→Definition A.25
$\mathsf{Diff}_{\mathrm{vol}}(\mathcal{M})$	Diffeomorphism that preserve the volume element.	
$\mathsf{Domain}(f)$	Domain of a function f.	→Definition A.13
$d^{\overline{\mathcal{Y}}}$	Metric on $d^{\overline{\mathcal{Y}}}$.	
$\mathrm{DSMPLTI}(n;k)$	Discrete-time stable minimum-phase linear time-invariant systems.	
$d^{\mathcal{Y}}$	Metric on $d^{\mathcal{Y}}$.	
$\mathsf{E}(n)$	Euclidean group on $\mathbb{R}^n$.	→Definition D.3
$\mathbb{E}\{x\}$	Expected value of x.	→Definition B.9
e	Identity of a group.	→Definition C.4
e_{pr}	Procrustes score.	→Chapter 10
e_{r}		→Chapter 10
e_{sr}		→Chapter 10
$\mathsf{EqSet}(f)$	Fixed points of a function f.	→Definition 15.1
expl	Agent's exploration phase.	→Definition 5.4
f	Distance to similarity function.	→Chapter 10
$\mathcal{F}$	Field sampled by the field sensor.	
Final	Stationary distribution of a stochastic process.	→Definition B.8
FOV	field of view.	→Chapter 10
$\mathsf{Functions}(\mathcal{B};\mathcal{A})$	All maps from a space $\mathcal{A}$ to $\mathcal{B}$.	
$\mathsf{G}_{\mathcal{A}}$		→Definition 5.14
$\mathsf{G}^{\mathcal{U}}_{\mathcal{A}}$		→Definition 5.14
$\mathsf{G}^{\mathcal{Y}}_{\mathcal{A}}$		→Definition 5.14

$\mathcal{G}$	The agent's goal (a subset of $\mathsf{StocProcesses}(\mathcal{Y} \times \mathcal{U})$).	→Definition 5.12
Γ	Uncertainty of estimated diffeomorphism.	→Chapter 13
$\mathbf{G}$	$\boldsymbol{y}$ gradient dynamics.	
$\mathsf{GL}_+(n)$	Linear transformations preserving orientation.	→Definition D.2
$\mathsf{GL}(n)$	General linear group.	→Definition D.2
GrAct	If the function is the action of a group.	→Definition 15.1
$\mathcal{H}(x)$	Entropy of two variables.	→Definition B.4
$\mathbb{H}^n$		→Definition E.17
$\mathbf{H}$	$\boldsymbol{y}$ gradient learned tensor.	
$\mathsf{Homeo}_+(\mathbb{R})$	Orientation-preserving homeomorphisms (of the real line).	→Definition E.22
$\mathsf{Homeo}(\mathcal{M})$	All homeomorphisms of $\mathcal{M}$.	→Definition E.5
$\mathrm{HomMaps}(f)$	Homomorphism induced by a function f.	→Definition 15.2
$\mathbf{I}_n$	Identity matrix of size $n \times n$.	
$\mathcal{I}(x;y)$	Mutual information between two variables.	→Definition B.4
$\mathsf{Id}_{\mathcal{X}}$	The identity function on the set $\mathcal{X}$.	→Definition A.15
Id	The trivial group with identity only.	→Definition D.1
$\mathsf{Id}_{\mathcal{S}}$	Identity diffeomorphisms.	→Chapter 13
$\mathrm{Image}(f)$	Image of a function f.	→Definition A.13
$\mathsf{Images}(\mathcal{S})$	Images on physical space $\mathcal{S}$.	
$\mathsf{infr}(f)$	Informative radius of f.	→Definition 10.4
$\mathsf{Isom}(\mathcal{M})$	Isometries of the metric space $\mathcal{M}$.	→Definition E.3
J	An element of the Jacobian of φ.	→Section 12.4
$\mathbf{J}$	Jacobian of φ.	→Section 12.4
k		→Chapter 19
$\mathcal{L}$	Laplace transform.	

Loop	Closes the loop around a system.	→Definition 4.12
$\mathcal{M}$	Agent's model space.	→Definition 5.4
$\boldsymbol{m}$	Agent representation.	→Definition 5.4
$\overline{\mu}$	Average nearness.	→Chapter 11
μ	Nearness.	
m	Inner product bilinear form.	→Chapter 9
m	Order of a generic semantic relations.	
$\mathcal{M}$	Target manifold.	→Chapter 10
$\mathbf{M}$ M^s_{vi}	Bilinear tensor in BDS dynamics.	→Chapter 11
Maps	Maps set $\mathsf{Maps} = \mathsf{Shapes} \times \mathrm{SE}(3)$.	
mexp	Matrix exponential.	→Definition E.8
$n_{\boldsymbol{u}}$	Number of actuators.	
$n_{\boldsymbol{y}}$	Number of sensels.	
$\mathbb{N}$	Natural numbers.	→Section A.1
$\mathcal{O}$	Big-O notation.	
$\mathcal{O}$	Observation output space.	→Chapter 9
$\mathsf{O}(n)$	Orthogonal group of order n.	→Definition D.2
$\mathsf{order}(x)$	Order (or rank) of the elements of a vector x.	→Definition A.28
$\boldsymbol{p}$	Map pose.	
$\mathsf{powerset}(\mathcal{X})$	Power set of a set $\mathcal{X}$.	→Definition A.2
$\mathbf{P}$ P^{sv}	Covariance tensor of $\boldsymbol{y}$ (BDS models).	→Chapter 11
p_{T}	Training distribution.	→Definition 9.9
$\mathsf{Perm}(n)$	Permutations of a set of n elements.	→Definition D.1
φ	Mapping between $\mathcal{S}$ and $\mathcal{Z}$.	→Section 12.4
$\mathsf{PieceHomeo}(\mathbb{R})$	Piece-wise continuous invertible functions on $\mathbb{R}$.	

$\mathsf{ProbMeasures}(\mathcal{X})$	All probability measures on the set $\mathcal{X}$.	→Definition B.1
$\mathsf{StocProcesses}(\mathcal{X})$	Stochastic processes on the space $\mathcal{X}$.	→Definition B.7
$\mathbb{Q}$	Pose space, subgroup of $\mathrm{SE}(3)$.	→Chapter 9
$\boldsymbol{q}$	Robot pose $\boldsymbol{q} = (\boldsymbol{t}, \mathbf{R}) \in \mathbb{Q} \subset \mathrm{SE}(3)$.	→Chapter 9
$\mathbb{Q}$	Rational numbers.	→Section A.1
$\mathbf{Q}$ Q^{ij}	Covariance tensor of $\boldsymbol{u}$ (BDS models).	→Chapter 11
$\mathbb{R}_\circ$	Nonzero reals.	→Section A.1
$\mathbb{R}_\circ^+$	Strictly positive reals.	→Section A.1
$(\mathbb{R}, +)$	Addition group.	→Definition D.1
ρ	Resolution of the sensor in a DDS.	→Chapter 13
$\mathcal{R}$	A generic semantic relation.	
R	A generic similarity measure.	
$*\mathbb{R}$	Hyper-real numbers.	→Section A.1
$(\mathbb{R}_\circ, \times)$	Multiplication group.	→Definition D.1
$(\mathbb{R}_\circ^+, \times)$	Positive multiplication group.	→Definition D.1
$\mathbb{R}$	Real numbers.	→Section A.1
R	Reward function.	
$\mathbf{R}$	Rotation matrix representing orientation in the world frame.	→Chapter 9
$\mathbf{R}$	$\boldsymbol{y}$ gradient covariance.	
rad	Radius of a distribution.	→Chapter 10
ρ	The map from a command to its reverse.	→Subsection 6.2
ρ_{sp}^*	Normalized Spearman performance measure.	→Chapter 10
ρ_{sp}	Spearman performance measure.	→Chapter 10
$\mathbb{R}_\bullet^+$	Non negative reals.	→Section A.1
$\mathbb{S}^1$	Unit circle.	→Definition E.17

$\mathbb{S}^2$	Unit sphere.	→Definition E.17
$\lvert\mathcal{S}\rvert$	Area of the manifold $\mathcal{S}$.	→Chapter 13
$\mathbf{S}$	Directions stacked in a matrix.	→Chapter 10
$\mathcal{S}$	Set of directions.	→Chapter 10
σ	Distance to obstacle.	
$s \in \mathcal{S}$	Spatial index for the sensels.	→Chapter 9
$\boldsymbol{s}$	Map shape.	
$\mathcal{S}$	Observation physical space.	
$\mathsf{Sc}(n)$	Multiples of the identity.	→Definition D.2
$\mathsf{SE}(n)$	Special Euclidean group on $\mathbb{R}^n$.	→Definition D.3
$\mathsf{Sequences}(\mathcal{X})$	Sequences on the space $\mathcal{X}$.	→Definition A.30
sgn	Sign function.	
Shapes	Shape space.	
SK	Shepard-Kruscall algorithm.	→Chapter 10
SKv	An extension to the SK algorithm (without warping).	→Chapter 10
$SKv + w$	An extension to the SK algorithm.	→Algorithm 1
$\mathsf{SL}(n)$	Special linear group.	→Definition D.2
Smooth_k		→Chapter 19
$\mathsf{so}(n)$	Lie algebra for SO.	
$\mathsf{SO}^-(n)$	Elements of the orthogonal group $\mathsf{SO}(n)$ with negative determinant (not a group).	→Example 15.4
$\mathsf{SO}(n)$	Special Orthogonal group on $\mathbb{R}^n$.	→Definition D.2
$\mathsf{sorted}(x)$	Sorted version of a vector x.	→Definition A.29
$\mathsf{sortedSeq}(\mathcal{X}, n)$	Subset of $\mathcal{X}^n$ such that $x_i < x_{i+1}$.	→Definition A.27
$\mathsf{spear}(x; y)$	Spearman correlation between two variables.	→Definition B.10

$\mathsf{stab}^G(\mathcal{S})$	Stabilizer of a set $\mathcal{S}$ with respect to the group G.	→Definition C.39
$\mathsf{std}(x)$	Standard deviation of x.	→Definition B.4
$\mathsf{success}^{\mathcal{G}}_{\mathcal{A}}$	Success set for the agent $\mathcal{A}$ and goal $\mathcal{G}$.	→Definition 5.13
$\mathsf{Support}$	Support of a probability measure.	→Definition B.2
$\mathsf{Sym}(\mathcal{R})$	Symmetries of the semantic relation.	
$\mathsf{Sym}(p_{\mathrm{T}})$	Symmetry group of p_{T}.	→Definition 9.10
$\mathsf{Sym}^G(\mathcal{S})$	Symmetries of a set $\mathcal{S}$ with respect to the group G.	→Definition C.40
T	Discretization interval.	→Chapter 11
$\boldsymbol{t}$	Position in the world frame.	→Chapter 9
$\mathsf{T}(n)$	Translation group.	→Definition D.3
$\mathbf{T}$ T^{isv}	Learned statistics for a BDS agent.	→Chapter 11
$\mathsf{TF}(\boldsymbol{D})$	Transfer function for the linear system $\boldsymbol{D}$.	
Tr	Trace of a matrix.	
$\overline{\mathcal{U}}$	Domain of a single actuator $\mathcal{U} = \overline{\mathcal{U}}^{n_u}$.	
u^i	One entries of the commands.	
$\boldsymbol{u}^{\mathsf{nop}}$	Command corresponding to "resting".	→Assumption 12
$\boldsymbol{u}^\star$	The optimal command.	
$\mathcal{U}$	Commands space.	
$\boldsymbol{u}$	Commands vector.	
$\lvert\mathcal{U}\rvert$	Number of commands words.	→Chapter 13
UCan^{φ}	Unstructured canonization operator.	→Definition 16.27
$\mathsf{Uniform}$	Uniform distribution.	
$\mathcal{V}$	Field of view for DDS.	→Definition 13.2
$\boldsymbol{v}$	Linear velocity.	→Chapter 9
$\mathcal{V}(x;y)$	Variation of information.	→Definition B.13

$\mathcal{V}_1(x;y)$	Variation of information (normalized).	→Definition B.13
$\mathsf{var}(x)$	Variance of x.	→Definition B.4
vec	Matrix-to-vector rearrangement.	
$\hat{\boldsymbol{\omega}}$	Angular velocity (as skew-symmetric matrix).	→Chapter 9
$\boldsymbol{\omega}$	Angular velocity (as vector).	→Chapter 9
$\boldsymbol{w}$	The world, an element of $\mathcal{D}(\mathcal{Y};\mathcal{U})$.	
$\mathsf{weaksortedSeq}(\mathcal{X}, n)$	Subset of $\mathcal{X}^n$ such that $x_i \leq x_{i+1}$.	→Definition A.26
$\mathsf{WtoB}_{\mathcal{A}}$		→Definition 5.6
$\mathsf{WtoR}_{\mathcal{A}}$	Map from the world to the result for the agent $\mathcal{A}$.	→Definition 5.7
x		→Chapter 19
$\boldsymbol{x}$		→Chapter 19
$\check{y}$	Goal observations (element).	
$\check{\boldsymbol{y}}$	Goal observations.	
$\overline{\mathcal{Y}}$	Domain of a single sensel $\mathcal{Y} = \overline{\mathcal{Y}}^{n_{\mathcal{Y}}}$.	
y	Observations element.	
$\mathcal{Y}$	Observation space.	
$\boldsymbol{y}$	Observations vector.	
$\mathbf{Y}$ Y_{ij}	Similarity matrix.	→Chapter 10
$\check{z}$	Goal observations (element).	
z	Observation logical space element.	→Section 12.4
$\mathcal{Z}$	Observation logical space.	→Section 12.4
$\boldsymbol{z}$	Observations in logical space.	→Section 12.4

Bibliography

[1] I. Asimov. *I, Robot*. Voyager, 1968. ISBN: 0586025324. URL: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0586025324.

[2] D. A. Grier. *When computers were humans*. Princeton University Press., 2007. ISBN: 9780691133829.

[3] B. Siciliano and O. Khatib, eds. *Handbook of Robotics*. Springer, 2008.

[4] N. J. Nilsson. *The Quest for AI*. Cambridge University Press, 2010.

[5] *iRobot Roomba*. URL: http://www.irobot.com/.

[6] *Evolution Robotics Mint*. URL: http://mintcleaner.com/.

[7] H. Durrant-Whyte, D. Pagac, B. Rogers, M. Stevens, and G. Nelmes. "An autonomous straddle carrier for movement of shipping containers: From Research to Operational Autonomous Systems". In: *IEEE Robotics and Automation Magazine* 14.3 (2007). ISSN: 1070-9932. DOI: 10.1109/MRA.2007.901316.

[8] S. Vasudevan, F. Ramos, E. Nettleton, and H. Durrant-Whyte. "A Mine on Its Own". In: *IEEE Robotics and Automation Magazine* 17.2 (2010). ISSN: 1070-9932. DOI: 10.1109/MRA.2010.936960.

[9] *Kiva's Automated Material Handling Order Fulfillment System*. URL: http://www.kivasystems.com/.

[10] D. Fitzgerald. "Amazon to Buy Robot Company Kiva for $775 Million". In: *The Wall Street Journal* March 19 (2012).

[11] R. C. Arkin. *Governing Lethal Behavior: Embedding Ethics in a Hybrid Deliberative/Reactive Robot Architecture*. Tech. rep. U.S. Army Research Office, 2007.

[12] D. H. Wilson. *How To Survive a Robot Uprising: Tips on Defending Yourself Against the Coming Rebellion*. Bloomsbury USA, 2005. ISBN: 1582345929.

[13] *The VisLab Intercontinental Autonomous challenge*. URL: http://viac.vislab.it/.

[14] R. Calo. *Nevada governor signs driverless car bill into law*. The Center for Internet and Society at Stanford Law School. 2011. URL: http://cyberlaw.stanford.edu/node/6688.

[15] L. Feng, J. Borenstein, and H. Everett. *"Where am I?" Sensors and Methods for Autonomous Mobile Robot Positioning*. Tech. rep. UM-MEAM-94-21. University of Michigan, 1994.

[16] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005. ISBN: 0262201623.

[17] S. Pinker. *The Language Instinct*. New York: Harper Perennial Modern Classics, 1994.

[18] V. Kumar and M. Mason. *Berlin Summit on Robotics – Conference Report 2011*. Chapter 3, "Are we even in the game?". 2011. URL: http://berlinsummit.org/images/Files/berlin_summit_2011.pdf.

[19] E. W. Dijkstra. "The humble programmer". In: *Communications of the ACM* 15.10 (1972). ISSN: 0001-0782. DOI: 10.1145/355604.361591.

[20] J. Bongard, V. Zykov, and H. Lipson. "Resilient Machines Through Continuous Self-Modeling". In: *Science* 314.5802 (2006). DOI: 10.1126/science.1133687.

[21] D. Floreano and C. Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. The MIT Press, 2008. ISBN: 0262062712, 9780262062718.

[22] R. Dawkins. *The Selfish Gene*. Popular Science. Oxford University Press, 1989. ISBN: 9780192860927. URL: http://books.google.com/books?id=WkHO9HI7koEC.

[23] A. Winfield and M. Erbas. "On embodied memetic evolution and the emergence of behavioural traditions in Robots". In: *Memetic Computing* 3 (4 2011). 10.1007/s12293-011-0063-x. ISSN: 1865-9284. URL: http://dx.doi.org/10.1007/s12293-011-0063-x.

[24] L. Ljung. *System Identification: Theory for the User*. 2nd ed. Prentice Hall, 1999. ISBN: 0136566952.

[25] T. Katayama. *Subspace methods for system identification*. Springer, 1999.

[26] P. Ioannou. *Adaptive Control Tutorial (Advances in Design and Control)*. SIAM, 2006. ISBN: 0898716152.

[27] J. Ko and D. Fox. "Learning GP-BayesFilters via Gaussian process latent variable models". English. In: *Autonomous Robots* 30.1 (2010). DOI: 10.1007/s10514-010-9213-0.

[28] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Ver, 1995.

[29] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2001.

[30] T. Poggio and S. Smale. "The Mathematics of Learning: Dealing with Data". In: *Notices of the American Mathematical Society* 50.5 (2005).

[31] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998. ISBN: 0262193981.

[32] J. Kolter and P. Abbeel. "Hierarchical apprenticeship learning with application to quadruped locomotion". In: *Advances in Neural Information Processing Systems (NIPS)*. 2008.

[33] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the theory of neural computation*. Westview, 1991.

[34] C. M. Bishop. *Neural Network for Pattern Recognition*. Oxford University Press, 1995.

[35] G. E. Hinton, S. Osindero, and Y.-W. Teh. "A Fast Learning Algorithm for Deep Belief Nets". In: *Neural Computation* 18.7 (2006). DOI: 10.1162/neco.2006.18.7.1527.

[36] Y. Bengio. "Learning Deep Architectures for AI". In: *Foundations and Trends in Machine Learning* (2009). DOI: 10.1561/2200000006.

[37] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida. "Cognitive Developmental Robotics: A Survey". In: *IEEE Transactions on Autonomous Mental Development* 1.1 (2009). ISSN: 1943-0604. DOI: 10.1109/TAMD.2009.2021702.

[38] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini. "Developmental robotics: a survey". In: *Connection Science* 15 (2003). DOI: 10.1080/09540090310001655110.

[39] G. Metta, G. Sandini, L. Natale, L. Craighero, and L. Fadiga. "Understanding mirror neurons: a bio-robotic approach". In: *Interaction Studies* 7.2 (2006).

[40] D. Pierce and B. Kuipers. "Map learning with uninterpreted sensors and effectors". In: *Artificial Intelligence* 92.1-2 (1997). DOI: 10.1016/S0004-3702(96)00051-3.

[41] Cohen *et al.* "Functional relevance of cross-modal plasticity in blind humans". In: *Nature* 389.6647 (1997). DOI: 10.1038/38278.

[42] O. Collignon, P. Voss, M. Lassonde, and F. Lepore. "Cross-modal plasticity for the spatial processing of sounds in visually deprived subjects." English. In: *Experimental brain research* 192.3 (2009). DOI: 10.1007/s00221-008-1553-z.

[43] B. Kuipers. "Drinking from the firehose of experience". In: *Artificial Intelligence in Medicine* 44.2 (2008).

[44] G. Tononi. "An information integration theory of consciousness". In: *BMC Neuroscience* 5.1 (2004). ISSN: 1471-2202. DOI: 10.1186/1471-2202-5-42.

[45] D. George and J. Hawkins. "Towards a Mathematical Theory of Cortical Micro-circuits". In: *PLoS Computational Biology* 5.10 (2009). DOI: 10.1371/journal.pcbi.1000532.

[46] M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Berlin: Springer, 2004. URL: http://www.hutter1.net/ai/uaibook.htm.

[47] V. Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. The MIT Press, 1984. ISBN: 0262521121.

[48] A Baranes and P. Y. Oudeyer. "R-IAC: Robust Intrinsically Motivated Exploration and Active Learning". In: *IEEE Transactions on Autonomous Mental Development* 1.3 (2009). DOI: 10.1109/TAMD.2009.2037513.

[49] S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg. "Intrinsically Motivated Reinforcement Learning: An Evolutionary Perspective". In: *IEEE Transactions on Autonomous Mental Development* 2.2 (2010). DOI: 10.1109/TAMD.2010.2051031.

[50] C. M. Vigorito and A. G. Barto. "Intrinsically motivated hierarchical skill learning in structured environments". In: *IEEE Transactions on Autonomous Mental Development* (2010).

[51] T. Poggio. "The computational magic of the ventral stream". In: *Nature Precedings* (2011). URL: http://hdl.handle.net/10101/npre.2011.6117.2.

[52] S. Soatto. "On the Distance Between Non-stationary Time Series". In: *Modeling, Estimation and Control*. Ed. by A. Chiuso, S. Pinzoni, and A. Ferrante. Vol. 364. Lecture Notes in Control and Information Sciences. Springer Berlin / Heidelberg, 2007. DOI: 10.1007/978-3-540-73570-0_22.

[53] J. Rotman. *An introduction to the theory of groups*. Springer-Verlag, 1995. ISBN: 0387942858.

[54] G. W. Erickson and J. A. Fossa. *Dictionary of paradox*. University Press of America, 2008.

[55] E. Haghverdi, P. Tabuada, and G. Pappas. "Bisimulation Relations for Dynamical and Control Systems". In: *Electronic Notes in Theoretical Computer Science* 69 (2003).

[56] M. Denham. "Canonical forms for the identification of multivariable linear systems". In: *IEEE Transactions on Automatic Control* 19.6 (1974).

[57] R. Kalman. "Mathematical System Theory". In: 1974. Chap. On invariants, canonical forms, moduli for linear constant final dimensional dynamical systems.

[58] R. Gardner. *The Method of Equivalence and its Applications*. SIAM, 1989.

[59] M. van Nieuwstadt, M. Rathinam, and R. M. Murray. "Differential Flatness And Absolute Equivalence Of Nonlinear Control Systems". In: *SIAM Journal on Control and Optimization* 36 (1994).

[60] P. Martin, R. M. Murray, and P. Rouchon. *Flat Systems, Equivalence and Trajectory Generation*. 2003.

[61] J. P. Byrnes, D. C. Miller, and W. D. Schafer. "Gender differences in risk taking: A meta-analysis." In: 125.3 (1999). URL: http://psycnet.apa.org/journals/bul/125/3/367.

[62] C. Koch. *Biophysics of Computation: Information Processing in Single Neurons (Computational Neuroscience)*. 1st ed. Oxford University Press, 1998. ISBN: 0195104919.

[63] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *Int. J. Comput. Vision* 60.2 (2004). ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000029664.99615.94.

[64] A. Censi, M. Hakansson, and R. M. Murray. "Fault detection and isolation from uninterpreted data in robotic sensorimotor cascades". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Saint Paul, MN, 2012. URL: http://purl.org/censi/2011/fault.

[65] B. Kuipers. "The Spatial Semantic Hierarchy". In: *Artificial Intelligence* 119.1–2 (2000).

[66] B. Kuipers. "An intellectual history of the Spatial Semantic Hierarchy". In: *Robotics and cognitive approaches to spatial mapping* 38 (2008).

[67] M. J. Milford. *Robot navigation from nature: Simultaneous localisation, mapping, and path planning based on hippocampal models*. Springer, 2008.

[68] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. 1994.

[69] B. Siciliano, L. Villani, L. Sciavicco, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2008.

[70] H. Le and D. G. Kendall. "The Riemannian Structure of Euclidean Shape Spaces: A Novel Environment for Statistics". In: *Annals of Statistics* 21.3 (1993).

[71] P. W. Michor and D. Mumford. "Riemannian geometries on spaces of plane curves". In: *Journal of the European Mathematics Society* 8 (2006).

[72] T. Lochmatter and A. Martinoli. "Theoretical analysis of three bio-inspired plume tracking algorithms". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe, Japan, 2009. ISBN: 978-1-4244-2788-8.

[73] J.-S. Gutmann, G. Brisson, E. Eade, P. Fong, and M. Munich. "Vector field SLAM". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2010. DOI: 10.1109/ROBOT.2010.5509509.

[74] S. Soatto. "Steps Towards a Theory of Visual Information: Active Perception, Signal-to-Symbol Conversion and the Interplay Between Sensing and Control". In: *CoRR* abs/1110.2053 (2011). URL: http://arxiv.org/abs/1110.2053.

[75] M. Gevers. "A personal view of the development of system identification: A 30-year journey through an exciting field". In: *IEEE Control Systems Magazine* 26.6 (2006). DOI: 10.1109/MCS.2006.252834.

[76] V. Verdult. "Nonlinear System Identification: A State-space Approach". PhD thesis. The Netherlands: University of Twente, 2002.

[77] M. O. Franz and B. Schölkopf. "A unifying view of Wiener and Volterra theory and polynomial kernel regression". In: *Neural Computation* 18.12 (12 2006). ISSN: 0899-7667. DOI: 10.1162/neco.2006.18.12.3097.

[78] C. E. Rasmussen. *Gaussian processes for machine learning*. MIT Press, 2006.

[79] R Memisevic and G Hinton. "Unsupervised learning of image transformations". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007.

[80] R Memisevic and G. Hinton. "Learning to represent spatial transformations with factored higher-order Boltzmann machines". In: *Neural Computation* 22.6 (2010). DOI: 10.1162/neco.2010.01-09-953.

[81] M. Ranzato and G. Hinton. "Modeling pixel means and covariances using factorized third-order boltzmann machines". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010. DOI: 10.1109/CVPR.2010.5539962.

[82] H Larochelle and G Hinton. "Learning to combine foveal glimpses with a third-order Boltzmann machine". In: *Advances in Neural Information Processing Systems (NIPS)*. Vol. 1. 2010.

[83] R. Roberts, C. Potthast, and F. Dellaert. "Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009. DOI: 10.1109/CVPRW.2009.5206538.

[84] S. Siddiqi, B. Boots, and G. J. Gordon. "Reduced-Rank Hidden Markov Models". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS-2010)*. 2010.

[85] D. Hsu, S. M. Kakade, and T. Zhang. "A spectral algorithm for learning Hidden Markov Models". In: *Journal of Computer and System Sciences* 78.5 (2012). ISSN: 0022-0000. DOI: 10.1016/j.jcss.2011.12.025.

[86] M. L. Littman, R. S. Sutton, and S. Singh. "Predictive Representations of State". In: *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2001.

[87] S Singh and M. James. "Predictive state representations: A new theory for modeling dynamical systems". In: *International Conference on Uncertainty in Artificial Intelligence*. 2004. URL: http://portal.acm.org/citation.cfm?id=1036905.

[88] B. Boots, S. Siddiqi, and G. Gordon. "Closing the Learning Planning Loop with Predictive State Representations". In: *International Journal of Robotics Research* 30 (2011). DOI: 10.1177/0278364911404092.

[89] B. Boots, S. Siddiqi, and G. Gordon. "An Online Spectral Learning Algorithm for Partially Observable Nonlinear Dynamical Systems". In: *Proceedings of the 25th National Conference on Artificial Intelligence (AAAI-2011)*. 2011.

[90] T. A. Clarke and J. G. Fryer. "The Development of Camera Calibration Methods and Models". In: *The Photogrammetric Record* 16.91 (1998). DOI: 10.1111/0031-868X.00113.

[91] P. Sturm, S. Ramalingam, J.-P. Tardif, S. Gasparini, and J. Barreto. "Camera Models and Fundamental Concepts Used in Geometric Computer Vision". In: *Foundations and Trends in Computer Graphics and Vision* 6.1–2 (2011). DOI: 10.1561/0600000023.

[92] J.-Y. Bouguet. *The Matlab Calibration Toolbox*. URL: http://www.vision.caltech.edu/bouguetj/calib_doc/.

[93] Z. Zhang. "A flexible new technique for camera calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (2002). DOI: 10.1109/34.888718.

[94] D. Gennery. "Generalized camera calibration including fish-eye lenses". In: *International Journal of Computer Vision* 68.3 (2006). DOI: 10.1007/s11263-006-5168-1.

[95] D. Scaramuzza, A. Martinelli, and R. Siegwart. "A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion". In: *Proceedings of IEEE International Conference of Vision Systems (ICVS)*. 2006. DOI: 10.1109/ICVS.2006.3.

[96] D. Scaramuzza, A. Martinelli, and R. Siegwart. "A Toolbox for Easy Calibrating Omnidirectional Cameras". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2006. DOI: 10.1109/IROS.2006.282372.

[97] D. Scaramuzza and R. Siegwart. "Vision Systems: Applications". In: ed. by G. Obinata and A. Dutta. inTech, 2007. Chap. A Practical Toolbox for Calibrating Omnidirectional Cameras. ISBN: 978-3-902613-01-1. URL: http://www.intechopen.com/articles/show/title/a_practical_toolbox_for_calibrating_omnidirectional_cameras.

[98] C. Mei and P. Rives. "Single View Point Omnidirectional Camera Calibration from Planar Grids". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Rome, Italy, 2007. DOI: 10.1109/ROBOT.2007.364084.

[99] C. Mei. *Omnidirectional camera calibration toolbox for MATLAB*. URL: http://homepages.laas.fr/~cmei/index.php/Toolbox.

[100] J. Kannala and S. S. Brandt. "A generic camera model and calibration method for conventional, wide-angle and fish-eye lenses". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.8 (2006). DOI: 10.1109/TPAMI.2006.153.

[101] J. Kannala. *Camera calibration Toolbox for Generic Lenses for MATLAB*. URL: http://www.ee.oulu.fi/~jkannala/calibration/.

[102] D. Scaramuzza. *OCamCalib: Omnidirectional Camera Calibration Toolbox for Matlab*. URL: https://sites.google.com/site/scarabotix/ocamcalib-toolbox.

[103] M. Grossberg and S. Nayar. "The Raxel Imaging Model and Ray-Based Calibration". In: *International Journal of Computer Vision* 61.2 (2005). DOI: 10.1023/B:VISI.0000043754.56350.10.

[104] F. Espuny and J. B. Gil. "Generic self-calibration of central cameras from two "real" rotational flows". In: *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*. 2008. URL: http://hal.inria.fr/inria-00325325.

[105] E. Grossmann, J. A. Gaspar, and F. Orabona. "Discrete camera calibration from pixel streams". In: *Computer Vision and Image Understanding* 114.2 (2010). ISSN: 1077-3142. DOI: 10.1016/j.cviu.2009.03.009.

[106] M. Boerlin, T. Delbruck, and K. Eng. "Getting to know your neighbors: unsupervised learning of topography from real-world, event-based input". In: *Neural computation* 21.1 (2009). DOI: 10.1162/neco.2009.06-07-554.

[107] J. Stober, L. Fishgold, and B. Kuipers. "Sensor Map Discovery for Developing Robots". In: *AAAI Fall Symposium on Manifold Learning and Its Applications*. 2009. URL: http://www.cs.utexas.edu/~stober/pdf/stoberFSS09.pdf.

[108] J. Modayil. "Discovering sensor space: Constructing spatial embeddings that explain sensor correlations". In: *Proceedings of the International Conference on Development and Learning (ICDL)*. 2010. DOI: 10.1109/DEVLRN.2010.557885.

[109] J. W. Sammon. "A Nonlinear Mapping for Data Structure Analysis". In: *IEEE Transactions on Computers* 18 (5 1969). DOI: 10.1109/T-C.1969.222678.

[110] R. C. T. Lee, J. R. Slagle, and H. Blum. "A Triangulation Method for the Sequential Mapping of Points from N-Space to Two-Space". In: *IEEE Transactions on Computers* 26 (3 1977). ISSN: 0018-9340. DOI: 10.1109/TC.1977.1674822.

[111] R. Shepard. "The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function (Part I)". In: *Psychometrika* 27.3 (1962). DOI: 10.1007/BF02289630.

[112] R. Shepard. "The analysis of proximities: Multidimensional scaling with an unknown distance function (Part II)". In: *Psychometrika* 27.3 (1962). ISSN: 0033-3123. DOI: 10.1007/BF02289621.

[113] J. B. Kruskal. "Multidimensional scaling by optimizing goodness of fit to a nonparametric hypothesis". In: *Psychometrika* 29.1 (1964). DOI: 10.1007/BF02289565.

[114] S. L. France and J. J. Carroll. "Two-Way Multidimensional Scaling: A Review". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 99 (2010). DOI: 10.1109/TSMCC.2010.2078502.

[115] T. Cox and M. Cox. *Multidimensional Scaling*. Boca Raton, FL: Chapman & Hall / CRC, 2001. ISBN: 1-58488-094-5.

[116] Y. Shang, W. Rumi, Y. Zhang, and M. Fromherz. "Localization from connectivity in sensor networks". In: *IEEE Transactions on Parallel and Distributed Systems* 15.11 (2004). DOI: 10.1109/TPDS.2004.67.

[117] S. Agarwal, J. Wills, L. Cayton, G. Lanckriet, D. Kriegman, and S. Belongie. "Generalized Non-metric Multidimensional Scaling". In: *Eleventh International Conference on Artificial Intelligence and Statistics*. 2007. URL: http://www.cs.washington.edu/homes/sagarwal/nmds.pdf.

[118] J. Ratcliffe. *Foundations of hyperbolic manifolds*. Vol. 149. Graduate Texts in Mathematics. Springer, 2006. DOI: 10.1007/978-0-387-47322-2.

[119] J. C. Gower and G. B. Dijksterhuis. *Procrustes problems*. Vol. 30. Oxford Statistical Science Series. Oxford, UK: Oxford University Press, 2004. ISBN: 978-0-19-851058-1.

[120] J. C. Platt. "FastMap, MetricMap, and Landmark MDS are all Nystrom algorithms". In: *In Proceedings of 10th International Workshop on Artificial Intelligence and Statistics*. 2005. URL: http://research.microsoft.com/en-us/um/people/jplatt/nystrom2.pdf.

[121] Pure Digital Technologies. *The Flip MINO HD website*. URL: http://www.theflip.com/.

[122] Woodman Labs. *The GOPRO Camera website*. URL: http://www.gopro.com/.

[123] C. Gimkiewicz, C. Urban, E. Innerhofer, P. Ferrat, S. Neukom, G. Vanstraelen, and P. Seitz. "Ultra-miniature catadioptrical system for an omnidirectional camera". In: ed. by H. Thienpont, P. V. Daele, J. Mohr, and M. R. Taghizadeh. Vol. 6992. 1. Strasbourg, France: SPIE, 2008. DOI: 10.1117/12.779988.

[124] S. Weiss, D. Scaramuzza, and R. Siegwart. "Monocular-SLAMÐbased navigation for autonomous micro helicopters in GPS-denied environments". In: *Journal of Field Robotics* 28.6 (2011). ISSN: 1556-4967. DOI: 10.1002/rob.20412.

[125] M. Rufli, D. Scaramuzza, and R. Siegwart. "Automatic Detection of Checkerboards on Blurred and Distorted Images," in: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Nice, France, 2008. DOI: 10.1109/IROS.2008.4650703.

[126] P. Lichtsteiner, C. Posch, and T. Delbruck. "A 128 $\times$ 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor". In: *IEEE Journal of Solid-State Circuits* 43.2 (2008). ISSN: 0018-9200. DOI: 10.1109/JSSC.2007.914337.

[127] E. Margolis. "Reconstruction of periodic bandlimited signals from nonuniform samples". MA thesis. Technion, 2004. URL: http://webee.technion.ac.il/people/YoninaEldar/Download/main.pdf.

[128] J. Yen. "On Nonuniform Sampling of Bandwidth-Limited Signals". In: *IRE Transactions on Circuit Theory* 3.4 (1956). ISSN: 0096-2007. DOI: 10.1109/TCT.1956.1086325.

[129] D. L. Elliott. *Bilinear control systems: matrices in action*. Springer, 2009. DOI: 10.1023/b101451.

[130] S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Berlin: Springer-Verlag, 1999.

[131] R. W. Brockett. "Asymptotic Stability and Feedback Stabilization". In: *Differential Geometric Control Theory*. Ed. by R. W. Brockett, R. S. Millman, and H. J. Sussmann. Boston: Birkhauser, 1983.

[132] Ceriani *et al.* "Rawseeds ground truth collection systems for indoor self-localization and mapping". In: *Autonomous Robots* 27.4 (2009). ISSN: 0929-5593. DOI: 10.1007/s10514-009-9156-5.

[133] E. Gawlik, P. Mullen, D. Pavlov, J. E. Marsden, and M. Desbrun. "Geometric, Variational Discretization of Continuum Theories". In: *Physica D: Nonlinear Phenomena* 240.21 (2011). DOI: 10.1016/j.physd.2011.07.011.

[134] L. Kneip, F. T. G. Caprari, and R. Siegwart. "Characterization of the compact Hokuyo URG-04LX 2D laser range scanner". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe, Japan, 2009. DOI: 10.1109/ROBOT.2009.5152579.

[135] M. L. Eaton. *Group Invariance Applications in Statistics*. English. Vol. 1. 1989. URL: http://www.jstor.org/stable/4153172.

[136] M. Loéve. *Probability Theory I*. Springer-Verlag, 1977.

[137] E. Lehmann and G. Casella. *Theory of Point Estimation*. Springer Texts in Statistics. Springer, 1998. ISBN: 9780387985022. URL: http://books.google.com/books?id=9St7DCbu9AUC.

[138] G. Casella and R. L. Berger. *Statistical Inference*. Duxbury Press, 1990.

[139] P. S. Maybeck. *Stochastic models, estimation and control*. Vol. 1. Academic Press, 1979.

[140] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, 2000. ISBN: 0130224642.

[141] M. Meila. "Comparing clusterings: an information based distance". In: *Journal of Multivariate Analysis* 98.5 (2007). ISSN: 0047-259X. DOI: 10.1016/j.jmva.2006.11.013.

[142] D. J. S. Robinson. *A Course in the Theory of Groups*. Springer, 1982. ISBN: 9780387906003.

[143] U. Grenander. *Probabilities on Algebraic Structures*. Dover Publications, 1965.

[144] P. Diaconis. *Group representations in probability and statistics*. Ed. by S. S. Gupta. 11. Institute of Mathematical Statistics, 1988.

[145] G. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 1: Classical Results and Geometric Methods*. Applied and Numerical Harmonic Analysis. Birkhäuser, 2009. ISBN: 9780817648022.

[146] G. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Applied and Numerical Harmonic Analysis. Birkhäuser, 2011. ISBN: 9780817649432.

[147] P. J. Olver. *Classical Invariant Theory*. Cambridge University Press, 2002.

[148] D. Hilbert and S. Cohn-Vossen. *Geometry and the imagination*. Chelsea Scientific Books: Geometry. Chelsea, 1990. ISBN: 9780828410878.

[149] M. do Carmo. *Riemannian Geometry*. Birkhauser, 1994. ISBN: 3-540-20493-8.

[150] M. Spivak. *A comprehensive introduction to differential geometry. Vol. I*. Second. Wilmington, Del.: Publish or Perish Inc., 1979. ISBN: 0-914098-83-7.

[151] R. Abraham, J. E. Marsden, and T. Ratiu. *Manifolds, tensor analysis, and applications*. Second. Vol. 75. Applied Mathematical Sciences. New York: Springer-Verlag, 1988. URL: `http://www.ams.org/mathscinet-getitem?mr=960687`.

[152] J. E. Marsden and T. S. Ratiu. *Introduction to Mechanics and Symmetry: a basic exposition of classical mechanical systems*. Second Edition. Springer, 2008.

[153] S. Amari. *Differential-Geometrical Methods in Statistics*. Springer Verlag, 1985.

[154] S. Amari and H. Nagaoka. *Methods of information geometry*. Vol. 191. Oxford University Press, 2000.

[155] M. K. Murray and J. W. Rice. *Differential Geometry and Statistics*. Chapman & Hall/CRC, 1993.

[156] R. Bhatia. *Matrix Analysis*. Springer, 1997.

[157] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, 2001.

[158] D. G. Ebin and J. Marsden. "Groups of Diffeomorphisms and the Motion of an Incompressible Fluid". English. In: *The Annals of Mathematics*. Second Series 92.1 (1970). ISSN: 0003486X. URL: `http://www.jstor.org/stable/1970699`.

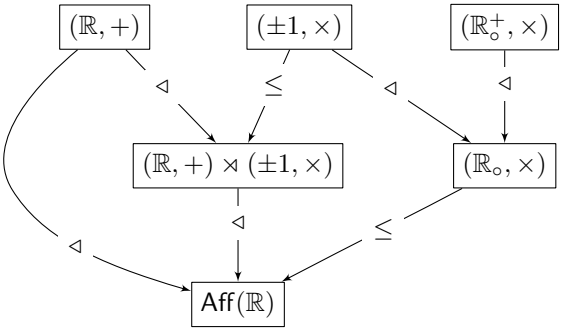


Figure E.1. Relations between simple groups acting on $\mathbb{R}$.

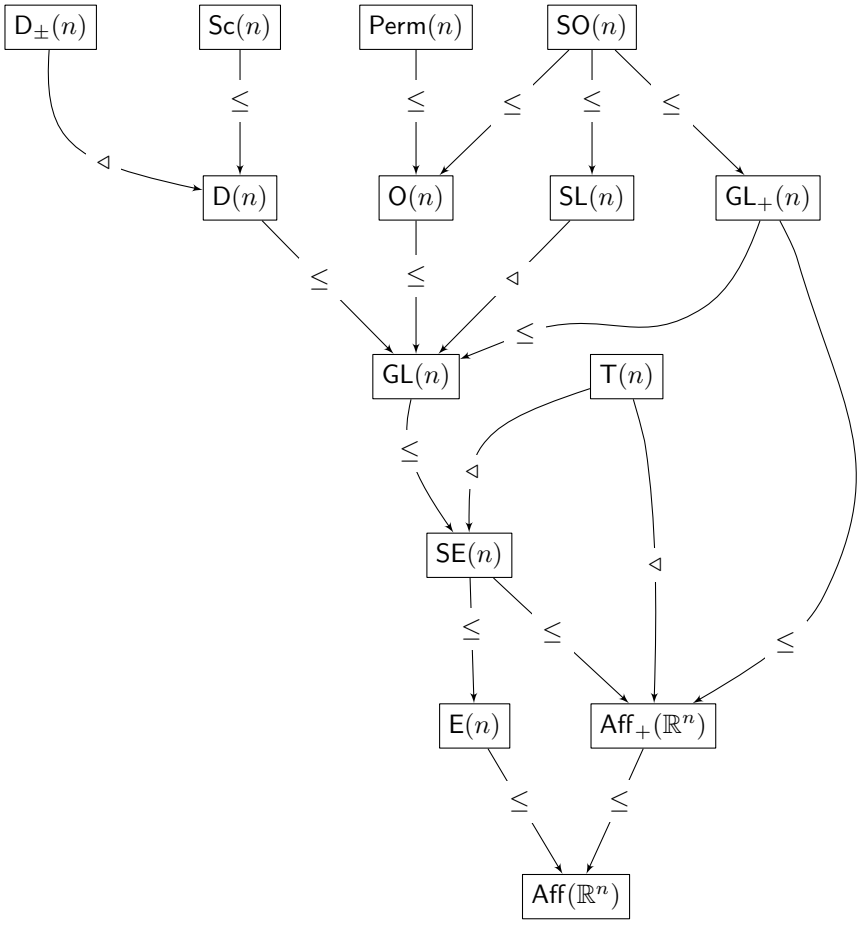


Figure E.2. Relations between simple Lie groups used as examples.

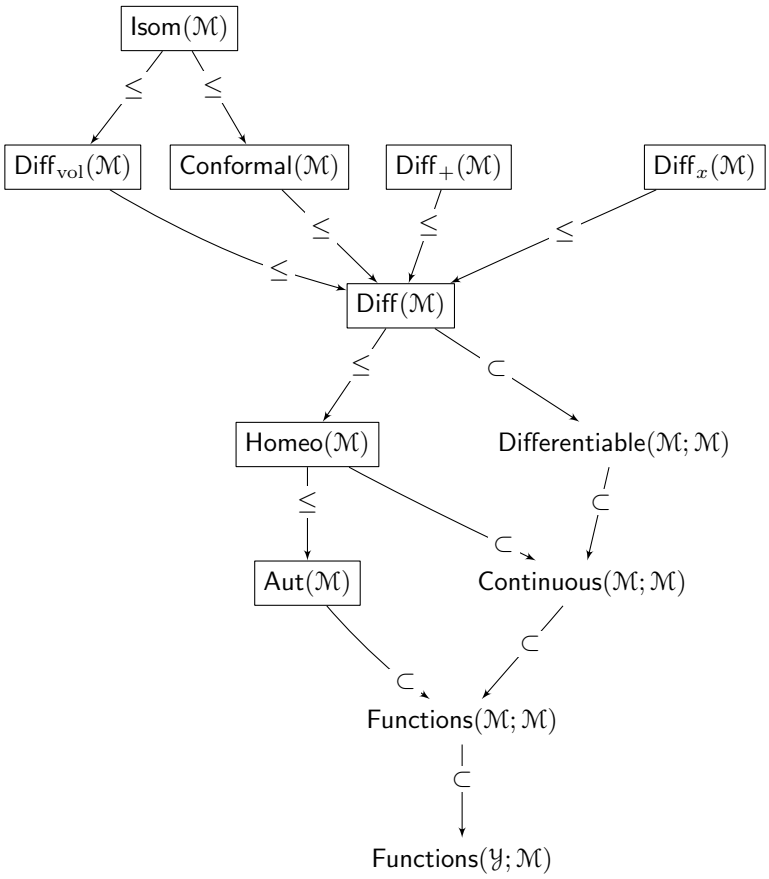


Figure E.3. Relations between classes of mappings

Figure E.4. Relations between some of the classes of invertible systems, defined in Chapter 4.