Chapter 1

INTRODUCTION

1.1 Motivation

Autonomous robotic systems have the potential for profound impact on our society — legged and wheeled robots for search and rescue missions, drones for wildfire management, self-driving cars for improved mobility, and robotic space missions for exploration and repair of spacecraft. These systems are expected to *correctly reason* about and execute tasks in vast operational environments, including interactions with other agents, both human and autonomous. Furthermore, these systems are incredibly complex: they comprise of several subsystems which are designed under different algorithmic paradigms (e.g., learning-based to model-based) and operate at different timescales and abstractions (e.g., high-level reasoning and decision making to low-level control) to accomplish the different functionalities (e.g., perception, behavior prediction, planning, and control) necessary for correct system-level behavior.

In light of these complexities, formal guarantees of system behavior during the design phase alone is not sufficient; mainstream deployment of these systems requires principled *theoretical* and *algorithmic* frameworks for test and evaluation, and verification and validation, not just to validate software and hardware implementations of the system, but also to complement *formal guarantees* derived during system design. How do we derive a small number of tests that can provide high confidence that the system can operate safely? These operational tests should ideally cover salient features of the operating environment such as disturbance and uncertainty, discrete and continuous inputs, closed-loop behavior of agents in the environment among others. Furthermore, designing and testing systems for guarantees relies on definitions of *correct behavior*, *success*, or *good performance*, which differs for each subsystem and might not be easily identifiable. How do we evaluate subsystems with respect to system-level task requirements?

Driven by these questions, this thesis is focused on testing and evaluating high-level reasoning and decision making algorithms in safety-critical robotic systems. We will draw from fundamentals in control and systems theory, convex and combinatorial optimization, formal methods, and to address challenges in specification,

testing, and evaluation of safety-critical autonomous systems.

1.2 Challenges

This thesis considers the following challenges in that are currently bottlenecks to safe deployment of complex autonomous systems, especially in safety-critical applications. We will take the example of self-driving to illustrate these challenges due to the richness of the example, but these challenges translate to other robotic applications as well.

Challenge 1: Evaluating Perception Performace with Respect to System-level Requirements

Consider the high-level overview of a classical software stack in a self-driving car as shown in Figure 1.1. Variations of this software stack differ in the neat separation of the perception and planning modules. Typically, the perception and planning modules are developed under different computational paradigms. The backbone of perception models is deep learning, while approaches to planning have traditionally included rulebooks and formal methods, sampling based planners, planning over occupancy grids, and model-based approaches such as model predictive control for mid-level planning. Due to this, these sub-systems are designed differently, often optimizing for different performance metrics. Therefore, it becomes important to establish a *safety case* that accounts for the interaction between perception and planning modules, and its impact on system-level safety. In its document, "A Blueprint for AV Safety: Waymo's Toolkit For Building a Credible Safety Case" [2, 3], Waymo defines a safety case as follows:

"A safety case for fully autonomous operations is a formal way to explain how a company determines that an AV system is safe enough to be deployed on public roads without a human driver, and it includes evidence to support that determination."

In an effort to establish such a safety case, we need to formally and quantitatively reason about how each subsystem contributes to the overall safety of the system. Advancements in perception models is often made along metrics that are not clearly aligned with system-level behavior. Yet, these state-of-the-art models are directly used in robotic systems such as self-driving cars, without standard methods of assessing whether it is indeed suited for the downstream planning and control task. For example, in object detection tasks, *recall* or *sensitivity* is a metric that quantifies how well a model can correctly classify a sample with a certain class label given all

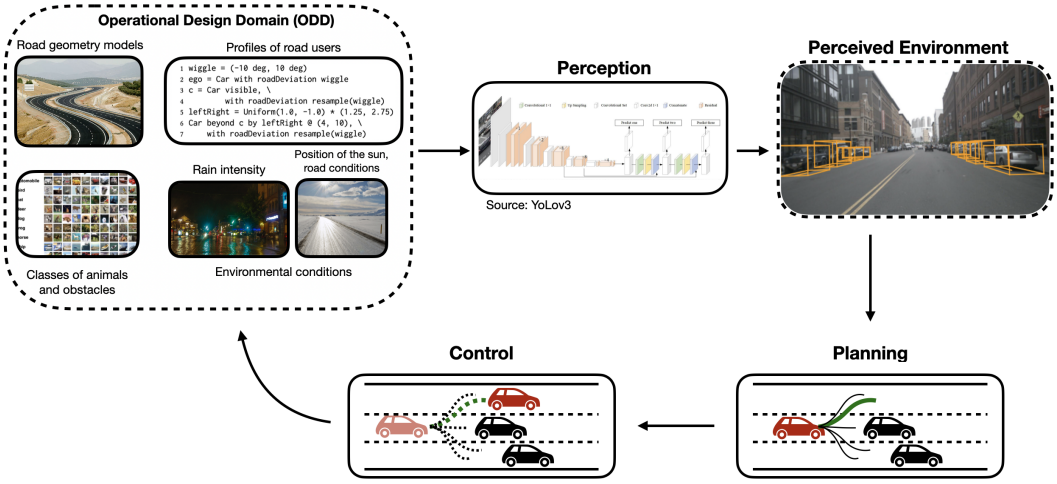


Figure 1.1: Typical software stack in a safety-critical system such as a self-driving vehicle.

relevant samples with that true class label. However, as we will see in this thesis, optimizing models with high *recall* with respect to pedestrians does not necessarily translate to better safety guarantees in all scenarios. Thus, we need new theoretical tools to formalize the interaction of perception errors, including detection and classification errors, localization errors, tracking errors, among others, on downstream planning tasks.

Challenge 2: Test and Evaluation, and Verification and Validation of Safety-Critical Autonomous Systems

For mainstream deployment of safety-critical systems, we need rigorous test and evaluation protocols to certify that autonomous systems comply with certain requirements. Testing can impact the certification process in by guiding regulators and designers to aspects of the design that need more careful evaluation.

Current approaches to safety certification can be broadly categorized as follows. The first category comprises of analysis techniques (e.g., fault tree analysis (FTA) and hazard analysis and risk assessment (HARA)), which cannot scale with the complexity in system design and in operational environments. The second category covers simulation-based testing such as Monte Carlo sampling, simulation-based falsification, and regression testing. These approaches typically sample continuous test parameters, and even if discrete parameters are sampled, they are typically kept fixed for the duration of the test (e.g., color of environment car) as opposed to a discrete test strategy that is reactive to system behavior. The third category involves collecting real-world experimental data (e.g., miles driven without disen-

(a) Static national qualifying test at 2007 Darpa Urban Challenge.

(b) Dynamic national qualifying test at 2007 Darpa Urban Challenge.

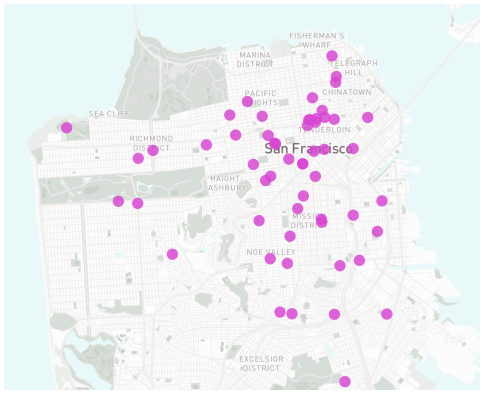
Figure 1.2: National Qualifying Events (NQEs) from the 2007 DARPA Urban Challenge. The photos are from the perspective of Alice, Caltech's entry in the competition, during the track tests.

gagement) to build statistical confidence that the system is safe. This approach can be extremely inefficient in time and cost, and would have to be repeated after each design iteration [4]. The final approach of manual constructing tests requires test engineers to rely on their expertise to specify the high-level scenario as well as design the test harness (e.g., specifying the number and locations of obstacles, dynamic agents and their strategies). In the application of autonomous vehicles, there is ongoing effort to standardize requirement specification, and test and evaluation procedures [5–8]. Standards such as "ISO 21448:2022 Safety of the Intended Functionality (SOTIF)" [6] provide guidance on verification and validation methods to demonstrate that self-driving. Listed below are some approaches to testing in the self-driving industry today.

Track-testing at the DARPA Urban Challenge: The 2007 DARPA Urban Challenge ushered interest in autonomous driving in urban environments [9]. Participating vehicles had to pass three small-scale operational test-courses, national qualifying events or NQEs, that were designed to evaluate the autonomous car's ability to satisfy safety, basic and advanced navigation requirements, and basic and advanced traffic scenarios [10]. Exhaustive verification for such complex safety-critical systems is prohibitive, creating a need for a formal operational testing framework to certify reliability of these systems [11]. Figure 1.2 shows Caltech's entry, Alice, in the test tracks corresponding to a completely static test environment and a dynamic test environment with other live vehicles. These tests were designed by entirely by test engineers.

AV companies have long relied on testing on urban roads to demonstrate to gather

(a) Cruise vehicle driving in the path of a fire truck.



(b) Map by Will Jarrett at Mission Local [12] using data from the San Francisco Fire Department. This map shows locations where Cruise vehicles violated traffic rules during on-road testing in their interactions with fire trucks.

Figure 1.3: Cruise vehicles driving in the path of emergency responders. In just the first half of 2023, 55 such incidents were reported in the city of San Francisco.

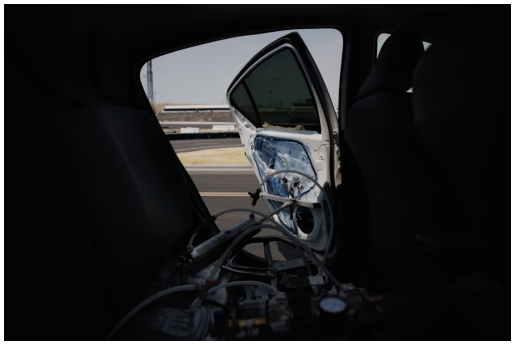
data for test and evaluation, and to demonstrate technological readiness. However, even test driving for millions of miles is not sufficient to demonstrate safety guarantees. In California in the year 2023 alone, six companies with permits for driverless testing have completed 3,267,792 miles in autonomous driving mode at SAE Level 4. However, it is still not sufficient to demonstrate required levels of safety. For example, in the first half of 2023, there were 55 incidents of Cruise vehicles driving in the path of emergency vehicles [12] (also see Figure 1.3). Recently, issues such as these have led to driverless permits being suspended by the California DMV.

In addition to road testing, the AV industry heavily relies on track testing and simulation-based testing to ensure the safety of its vehicles. Waymo's safety methodology [5] lists the following methods to evaluate autonomous driving behavior on its vehicles: i) hazard analysis that tests for robustness against user-defined hazards, ii) scenario-based testing on an instrumented track and in simulation, and iii) extensive simulation testing that aggregates driving performance across several simulations. Aside from manually specified scenarios, the industry also relies on police reports to test its software in challenging scenarios [5]. The self-driving car company, Zoox, also released a highlight video demonstrating its approach to track testing, snapshots of which are shown in Figure 1.4. First, scenarios that are difficult are identified by test engineers, and these scenarios are recreated in simulation and on the closed-loop track.

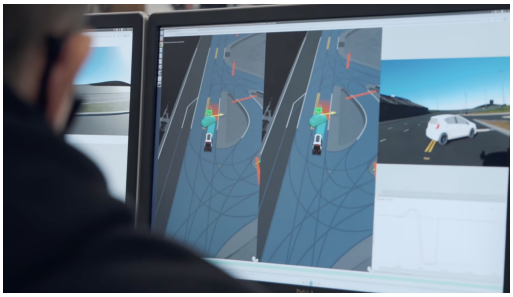
(a) Road condition: bumpy

(b) Road condition: damp

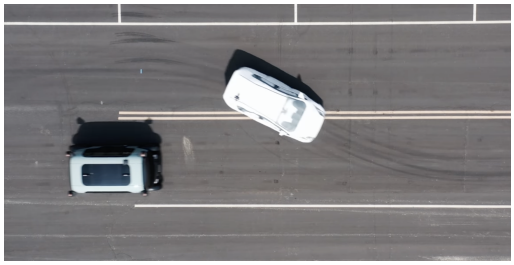
(c) Testing high-speed maneuverability: obstacle course in simulation



(d) Instrumented door to test whether Zoox car can properly detect and avoid collision.



(e) Scenario design by test engineers prior to track test shown in Figure 1.4f.



(f) Reactive test scenario in which Zoox car must respond correctly in reaction to the environment agent.

Figure 1.4: Instrumented track testing at Zoox. These images are taken from "Putting Zoox to the Test" [1].

These case studies illustrate the need for rigorous approaches to test and evaluation of these systems. Existing approaches do not provide a definitive answer to the certification of autonomous systems in safety-critical settings. Now, we will cover related work that is motivated by these challenges.

1.3 Related Work

Task-Relevant Evaluation of Perception

As discussed in the Challenges, perception and planning modules are typically designed under different computational paradigms. At the NVIDIA AV Team, empirical studies on how perception design choices affect overall system-level safety have been studied in a pedestrian jay-walking scenario [13]. These empirical studies reflect the need for studying this problem more rigorously. The design paradigms for planning and control submodules are usually backed by guarantees of safety and stability. For this related work, we will take the example of formal methods as a paradigm for control system design, but these insights can extend to other planning and control frameworks that provide guarantees of correctness.

Formal methods have been employed to construct provably correct planners and controllers given a system model and temporal logic specifications [14–18]. The correctness guarantee, typically specified using a temporal logic formula, relies heavily on the assumption that the input (i.e., the perceived world reported by perception) is perfect. Perception is important for *state estimation*, which is necessary for the downstream control and planning logic to effectively react to the environment. For example, if the perception component only reports the most likely class of each object, the control component assumes that the reported class is correct. Unfortunately, this assumption may not hold in most real-world systems, and the correctness guarantees might no longer hold.

In recent years, verifying neural networks with respect to safety and robustness properties has grown into an active research area [19–22]. Often, these methods apply to specific neural net architectures, such as those with piece-wise linear activation functions [19], or might require knowledge of the safe set in the output space of the neural network [20, 21]. Furthermore, these methods have been demonstrated on learning-based controllers with smaller input dimensions, and are not yet deployed for analysis of perception models. One reason for this is the difficulty in formally characterizing properties of ML-based perception models, as elaborated below.

First, recent work demonstrates that it is not realistically feasible to formally specify properties reflecting human-level perception for perception models, in particular, classification ML models, due to the high dimensional nature of the input, such as pixels in an image [23]. Finally, not all perception errors are equally safety-critical. Dreossi *et al.* reason that not all misclassifications are the same; some are more likely to result in system-level failure, and therefore, it is necessary to adopt system-level specifications and contextual semantics in developing a framework for quantitative analysis and verification of perception models [23, 24]. This has led to work on compositional analysis of perception models in finding system-level counter-examples [25]. The work in [26] introduced the concept of interaction zones using Hamilton-Jacobi reachability theory, and illustrated that perception errors in the interaction zone were more likely to result in system-level violations than those outside of it. This observation was further backed in [27], which demonstrated instances of both small perception errors (for the task of segmentation over RGB images) leading to closed-loop system-level failure, and large perception errors still resulting in safe system-level failures.

While there is work on evaluating performance of perception with temporal logic, those formal specifications are defined over image data streams, and must be manually formalized for each scenario / data stream [28, 29]. Often, there is high variability in the performance of perception models in seemingly similar environments, such as variations in sun angle [30]. Therefore, for any given scenario, it can be challenging to specify all realizations of the environment that a perception system might encounter. On the other hand, it is simpler, and more accurate, to define system-level specifications, such as "maintain a safe distance of 5 m from obstacles" [23, 31–33].

Testing for Autonomous Systems

As described in the Challenges section, tests are often manually designed by test engineers. This was seen in the DARPA Urban Challenge, and in current practices at AV companies such as Waymo and Zoox. Test scenarios are often constructed first in simulation using tools such as CARLA [34] and Scenic [35]. For example, Scenic is a probabilistic programming language to model environments of autonomous cyber-physical systems. A single Scenic program describes a distribution of environments by declaring random variables (e.g., position of parked car, location of pedestrians, color of obstacles) and specifying distributions of each of these random variables. A compiled Scenic program can be sampled to provide

concrete scenes, and these concrete scenarios are related by the high-level scenario (e.g., number of cars and their approximate locations) used to define the Scenic program. However, Scenic cannot handle the generation of these Scenic programs from high-level specifications. The automated, reactive test synthesis framework in Chapters 3– 4 addresses this, and can potentially be interfaced to Scenic to automatically construct scenarios at all levels of the planning stack.

In the formal methods community, research on falsification aims to uncover bugs in the software of cyber-physical systems with access to just black-box models, and without any knowledge of the control design [36–40]. Oftentimes, specifications for these cyber-physical systems are characterized in metric temporal logic (MTL) and signal temporal logic (STL), which allow for specifying timed requirements and also lend themselves to quantitative metrics of robustness to characterize the degree to which a specification is satisfied or violated. The goal of falsification is search over a specified input domain (typically continuous) to identify an input that maximizes the degree of violation of the specified requirement. The community has introduced several toolboxes, e.g., Breach [41] and S-TaLiRo [36, 42], among others [43] for this effort. These falsification toolboxes can be interfaced with scenario definition programs such as Scenic to automatically construct test scenarios, and an example of such a tool is VerifAI [44]. Note that the user still needs to define the high-level scenario in Scenic — interfacing with the falsifier returns the worst-case concrete scenario from the distribution of scenarios.

Aside from traditional black-box optimization methods such as Bayesian optimization, cross-entropy method, reinforcement learning has been used to identify falsifying inputs [45–47]. Oftentimes, falsification algorithms are applied over continuous domains and metrics, and often cannot handle discrete input spaces. However, complex cyber-physical systems are expected to handle both continuous and discrete inputs, and reason over continuous and discrete state spaces [48]. Additionally, falsifying inputs are often open-loop signals that generate the worst-case trajectory in simlation. However, feedback is a fundamental principle in control theory that allows us to design systems that are robust to unmodeled dynamics, uncertainties, and disturbances. The contributions in thesis complements falsification — our focus is on synthesizing high-level test environments and reactive test strategies that operate over discrete state spaces. In future work, we can search over the continuous parameters of the synthesized test environment (e.g., continuous pose values of test agents, friction coefficients, exact timing of events) using falsification

algorithms for further concretizing the test scenario.

1.4 Thesis Overview and Contributions

The principles underlying my past and current work are *reactive* test plans, *modular* test and evaluation of subsystems and interfaces between subsystems, and choosing relevant *specifications* and *evaluation criteria* at the system and subsystem levels by accounting for interactions between subsystems and their impact on system-level behavior. The theoretical contributions as well as its applications, in both algorithms and hardware, are outlined below.

Part I: System-level Reasoning for deriving Task-Relevant Metrics of Perception

Chapter 2 focuses on introducing task-relevant evaluation metrics for object detection and classification models for perception. This work identifies evaluation metrics of perception tasks that are useful in providing probabilistic guarantees on system-level behavior. At a high-level, the main contribution of this work is in identifying standard perception metrics that can be used in a quantitative system-level analysis, and in proposing new perception metrics that are relevant to the downstream planner and the system-level task.

First, we identify popularly used metrics in computer vision confusion matrices as a candidate model for sensor error, and leverage probabilistic model checking to quantify the probability of the overall system satisfying its requirements. Prior work [49] has shown how to leverage a probabilistic model of sensor error in probabilistic model-checking of the overall system with respect to system-level temporal logic specifications. The work in this chapter was the first to identify confusion matrices as a model of sensor error for detection and classification tasks, rigorously define probabilities of misdetection from the confusion matrix, and show how it can be leveraged in probabilistically model-checking system-level task specifications.

Confusion matrices are popularly used in computer vision to compare and evaluate models for detection tasks, and a wide-variety of metrics such as accuracy, precision, recall, among others, can be derived from the confusion matrix. The key idea was in identifying confusion matrices as a candidate for capturing requirements on detection tasks, and in rigorously defining probabilities to relate the confusion matrix to system-level performance with respect to temporal logic specifications. Even on simple examples, our approach highlighted fundamental insights: performance tradeoffs (e.g., precision-recall tradeoff) in detection tasks get reflected in

system-level performance, and our method gives sanity checks – both qualitative and quantitative guidelines on selecting detection models and high-level planners, which in combination have probabilistic system-level guarantees.

For example, consider a car-pedestrian scenario in which the autonomous car needs to contend with multiple safety requirements — to stop for a pedestrian at a crosswalk and to not stop unnecessarily if there are no pedestrians at the crosswalk. While engineers training perception algorithms might optimize for high recall (i.e., to never miss a pedestrian even at the cost of false negatives), this will lead to the car stopping frequently. This intuition was captured quantitatively in my framework. Furthermore, if we have probabilistic system-level guarantees (e.g., meet a safety requirement to 99%) and given a specific planning logic, we can derive lower bounds on elements of the confusion matrix such as minimum true positive rate, minimum false negative rate, and encode requirements on perception tasks in this manner. The practical impact of this method is the ability to communicate quantitative requirements via confusion matrices, rather than temporal logic specifications, to engineers training perception algorithms for detection tasks.

The second contribution is in defining new metrics for detection tasks, informed by the system-level specification as well as the downstream planning logic. This work stemmed from the insight that not all perception errors are equally safety-critical, and that current methods to evaluate perception models do not account for this distinction. For instance, in evaluating models for object detection tasks in computer vision, all misdetections are given equal weight in the confusion matrix. However, not all misclassifications or misdetections will have the same impact on system-level safety. To account for this, we introduced a distance-parametrized, proposition-labeled confusion matrix, which: i) placed higher weight on correct detection of objects closer to the ego, and ii) replaced the object class labels of confusion matrices with atomic propositions that are more relevant to system-level safety specification.

Guarantees from the distance-parametrized, proposition-labeled confusion matrices is less conservative than the analysis that used the traditional class-based confusion matrix. Further extensions of the proposition-labeled confusion matrix, in which predictions are grouped according to the same level of abstraction used by the high-level planner, result in system-level satisfaction probabilities that are neither too relaxed and nor too conservative. Finally, the proposed metrics are used to evaluate a PointPillars on the real-world nuScenes dataset. The core message of this work

is that metrics for evaluating perception tasks need to be carefully informed by both the system-level specification as well as the downstream planning and control logic. This work is being packaged as a Python toolbox, TRELPy: Task-Relevant Evaluation of Perception.

Part II: Reactive Test Synthesis

Chapters 3– 4 focus on reactive test synthesis. These chapters address the problem of synthesizing tests for high-level reasoning and decision-making in autonomous robotic applications. Instead of having the entire test be manually designed, we presume that it is easier for a test engineer to provide a formal description of the objective of the test. My work focused on automated construction of test scenarios from these high-level test objectives specified by the user/test engineer. Chapter 5 introduced preliminary directions on compositional test synthesis from unit tests via assume-guarantee contracts.

Chapters 3– 4: The first contribution of this work is introducing the notion of a test specification: a high-level description of the objective of the test. This test objective is not revealed to the system under test, but is consistent with safety and liveness assumptions the system has on its environment (e.g., there will always exist a path to the goal, the environment agents will not adversarially collide).

The second contribution is in automatic construction of a reactive test that is consistent with the test objective as well as minimally restrictive to the system. In particular, the constructed test harness involves placement of static and reactive constraints to system actions, and the smallest number of restrictions needed for the test objective are found. These restrictions on system actions are such that if the system under test is successful in meeting its requirements, the test objective is also met. The third contribution is in automatically mapping these reactive constraints to synthesize a reactive strategy of a dynamic test agent. Finally, we also prove that the reactive test synthesis problem is NP-hard via a reduction from 3-SAT.

Algorithms: Chapter 4 provides algorithms to automate each of the aforementioned tasks. First, leveraging automata theory and combinatorial graph algorithms, we formulate a network flow optimization to identify static and/or reactive test constraints for the system. The problem data for this algorithm includes the specifications the system is expected to satisfy, the test objective, and a discrete-state abstraction model of the system. Note that the system model is non-deterministic and does not carry knowledge of the system control; it is just a high-level abstraction

representing all possible actions a system can take from any given state. Although this is a combinatorial problem, we take advantage of the structure in the resulting product graph to formulate a mixed-integer linear program with discrete variables representing interdiction of edges in the network that correspond to reactive test constraints. The choice of using network flows allows for the optimization to handle medium sized problems (5000 integer variables) with a runtime of around 30s to a few minutes. We prove that the optimal solution corresponds to a set of restrictions with the following guarantees: any trajectory of the system that satisfies the system objective will also satisfy the test objective.

Furthermore, it is easy to augment additional optimization constraints (e.g., some system actions cannot be constrained). This becomes prominent when synthesizing a test agent strategy to match the restrictions returned by the optimization. The optimization is solved offline, and the resulting solution is automatically mapped to a reactive test strategy for a given dynamic agent. If the solution is not dynamically feasible for the test agent, we use an efficient counterexample-guided approach to resolve the MILP. For this, the test constraints are mapped as safety formulas that the dynamic test agent is expected to satisfy. Additional safety formulas are found to ensure that the dynamic test agent does not restrict system actions other than the test constraints. Furthermore, we address livelocks by automatically identifying potential livelock states, and specify that the test agent, if it occupies these states, should only transiently occupy it. Finally, the synthesized test strategy chooses from a set of possible initial conditions and realizes the reactive test constraints found by the optimization.

Hardware Demos: This framework was demonstrated in hardware using quadrupeds for robot navigation examples such as search and rescue, and testing motion primitives. In addition to demonstrating the usefulness of this approach to real robotic systems, the hardware experiments were repeatable and successful immediately after the test strategy was generated in simulation. These experiments demonstrated that our framework can handle test objectives beyond simple abstractions of robot position (e.g., go to a particular cell), but can also capture more complex behaviors such as dynamic motion primitives (e.g., jump then stand). Our test synthesis framework had no knowledge of the control architecture for low-level motion primitives (e.g., standing, walking, jumping) or even the mid-level planning framework (e.g., waypoint following) on the quadruped. Despite this, the high-level, reactive test strategy resulted in successful demonstrations in hardware. This experimen-

tal success points to promising future directions in decoupling test synthesis for high-level reasoning and low-level control.

Finally, we also provide an argument for why traditional GR(1) synthesis techniques cannot be used to directly synthesize tests that are not overly-restrictive. There are two reasons. First, the synthesis of test constraints cannot be cast into an LTL synthesis problem. In reactive synthesis for LTL or similar temporal logics, synthesis assumes worst-case behavior of the other player, which is not consistent with our objectives. Our test harness is not fully cooperative nor fully adversarial: we do not help the system achieve its requirements yet ensure that there always exists a path for success. It is possible that this can be cast as a synthesis problem in a different temporal logic that reasons over path properties (e.g., computational tree logic (CTL), or hyperLTL). However, the synthesis in those specification languages is known to be computationally intractable. Second, our optimization finds the least restrictive set of test constraints, which traditional synthesis methods cannot provide.

Chapter 5: The main contributions of this chapter are as follows. We establish a mathematical framework for merging two unit test scenarios using assume-guarantee contracts. The merged test can be optimized according to an arbitrary difficulty metric, and we use a receding horizon approach to synthesize winning sets that guide the test strategy to optimize for the metric.

Bibliography

[1] Zoox, "Putting Zoox to the test: preparing for the challenges of the road," 2021. https://zoox.com/journal/structured-testing/, Last accessed on 2024-04-11.

[2] Waymo, "A blueprint for av safety: Waymo's toolkit for building a credible safety case," 2020. https://waymo.com/blog/2023/03/a-blueprint-for-av-safety-waymos/#:~:text=A%20safety%20case%20for%20fully,evidence%20to%20support%20that%20determination., Last accessed on 2024-05-05.

[3] F. Favarò, L. Fraade-Blanar, S. Schnelle, T. Victor, M. Peña, J. Engstrom, J. Scanlon, K. Kusano, and D. Smith, "Building a credible case for safety: Waymo's approach for the determination of absence of unreasonable risk," 2023. www.waymo.com/safety.

[4] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?," *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 182–193, 2016.

[5] N. Webb, D. Smith, C. Ludwick, T. Victor, Q. Hommes, F. Favaro, G. Ivanov, and T. Daniel, "Waymo's safety methodologies and safety readiness determinations," 2020.

[6] I. S. Organization, "Road vehicles: Safety of the intended functionality (ISO Standard No. 21448:2022)," 2022. https://www.iso.org/standard/77490.html, Last accessed on 2024-04-11.

[7] L. Li, W.-L. Huang, Y. Liu, N.-N. Zheng, and F.-Y. Wang, "Intelligence testing for autonomous vehicles: A new approach," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 2, pp. 158–166, 2016.

[8] H. Winner, K. Lemmer, T. Form, and J. Mazzega, "Pegasus—first steps for the safe introduction of automated driving," in *Road Vehicle Automation 5*, pp. 185–195, Springer, 2019.

[9] "DARPA Urban Challenge." https://www.darpa.mil/about-us/timeline/darpa-urban-challenge.

[10] "Technical Evaluation Criteria." https://archive.darpa.mil/grandchallenge/rules.html.

[11] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[12] J. Eskenazi and W. Jarett, "Explore: See the 55 reports — so far — of robot cars interfering with SF fire dept.," 2023. https://missionlocal.org/2023/08/cruise-waymo-autonomous-vehicle-robot-taxi-driverless-car-reports-san-francisco/, Last accessed on 2024-04-11.

[13] H. Zhao, S. K. Sastry Hari, T. Tsai, M. B. Sullivan, S. W. Keckler, and J. Zhao, "Suraksha: A framework to analyze the safety implications of perception design choices in avs," in *2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE)*, pp. 434–445, 2021.

[14] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[15] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[16] M. Lahijanian, S. B. Andersson, and C. Belta, "A probabilistic approach for control of a stochastic system from LTL specifications," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 2236–2241, IEEE, 2009.

[17] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *53rd IEEE Conference on Decision and Control*, pp. 81–87, IEEE, 2014.

[18] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, 2012.

[19] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *International Conference on Computer Aided Verification*, pp. 97–117, Springer, 2017.

[20] M. Fazlyab, M. Morari, and G. J. Pappas, "Probabilistic verification and reachability analysis of neural networks via semidefinite programming," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 2726–2731, IEEE, 2019.

[21] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, 2020.

[22] H.-D. Tran, X. Yang, D. M. Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson, "NNV: The neural network verification tool for

deep neural networks and learning-enabled cyber-physical systems," in *International Conference on Computer Aided Verification*, pp. 3–17, Springer, 2020.

[23] T. Dreossi, S. Jha, and S. A. Seshia, "Semantic adversarial deep learning," in *International Conference on Computer Aided Verification*, pp. 3–26, Springer, 2018.

[24] S. A. Seshia, A. Desai, T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, S. Shivakumar, M. Vazquez-Chanlatte, and X. Yue, "Formal specification for deep neural networks," in *International Symposium on Automated Technology for Verification and Analysis*, pp. 20–34, Springer, 2018.

[25] T. Dreossi, A. Donzé, and S. A. Seshia, "Compositional falsification of cyber-physical systems with machine learning components," *Journal of Automated Reasoning*, vol. 63, no. 4, pp. 1031–1053, 2019.

[26] S. Topan, K. Leung, Y. Chen, P. Tupekar, E. Schmerling, J. Nilsson, M. Cox, and M. Pavone, "Interaction-dynamics-aware perception zones for obstacle detection safety evaluation," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1201–1210, IEEE, 2022.

[27] K. Chakraborty and S. Bansal, "Discovering closed-loop failures of vision-based controllers via reachability analysis," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2692–2699, 2023.

[28] A. Dokhanchi, H. B. Amor, J. V. Deshmukh, and G. Fainekos, "Evaluating perception systems for autonomous vehicles using quality temporal logic," in *International Conference on Runtime Verification*, pp. 409–416, Springer, 2018.

[29] A. Balakrishnan, A. G. Puranic, X. Qin, A. Dokhanchi, J. V. Deshmukh, H. B. Amor, and G. Fainekos, "Specifying and evaluating quality metrics for vision-based perception systems," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1433–1438, IEEE, 2019.

[30] B. Bauchwitz and M. Cummings, "Evaluating the reliability of Tesla model 3 driver assist functions," 2020.

[31] H. Kress-Gazit, D. C. Conner, H. Choset, A. A. Rizzi, and G. J. Pappas, "Courteous cars," *IEEE Robotics & Automation Magazine*, vol. 15, no. 1, pp. 30–38, 2008.

[32] H. Kress-Gazit and G. J. Pappas, "Automatically synthesizing a planning and control subsystem for the DARPA Urban Challenge," in *2008 IEEE International Conference on Automation Science and Engineering*, pp. 766–771, IEEE, 2008.

[33] T. Wongpiromsarn, S. Karaman, and E. Frazzoli, "Synthesis of provably correct controllers for autonomous vehicles in urban environments," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1168–1173, IEEE, 2011.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conference on Robot Learning*, pp. 1–16, PMLR, 2017.

[35] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and scene generation," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 63–78, 2019.

[36] Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan, "S-taliro: A tool for temporal logic falsification for hybrid systems," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 254–257, Springer, 2011.

[37] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, vol. 410, no. 42, pp. 4262–4291, 2009.

[38] G. E. Fainekos, S. Sankaranarayanan, K. Ueda, and H. Yazarel, "Verification of automotive control applications using s-taliro," in *2012 American Control Conference (ACC)*, pp. 3567–3572, IEEE, 2012.

[39] S. Sankaranarayanan and G. Fainekos, "Falsification of temporal properties of hybrid systems using the cross-entropy method," in *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, pp. 125–134, 2012.

[40] S. Bak, S. Bogomolov, A. Hekal, N. Kochdumper, E. Lew, A. Mata, and A. Rahmati, "Falsification using reachability of surrogate koopman models," in *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '24, (New York, NY, USA), Association for Computing Machinery, 2024.

[41] A. Donzé, "Breach, a toolbox for verification and parameter synthesis of hybrid systems," in *International Conference on Computer Aided Verification*, pp. 167–170, Springer, 2010.

[42] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski, "Simulation-based adversarial test generation for autonomous vehicles with machine learning components," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1555–1562, IEEE, 2018.

[43] C. Menghi, P. Arcaini, W. Baptista, G. Ernst, G. Fainekos, F. Formica, S. Gon, T. Khandait, A. Kundu, G. Pedrielli, *et al.*, "Arch-comp 2023 category report: Falsification," in *10th International Workshop on Applied Verification of Continuous and Hybrid Systems. ARCH23*, vol. 96, pp. 151–169, 2023.

[44] T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, H. Ravanbakhsh, M. Vazquez-Chanlatte, and S. A. Seshia, "Verifai: A toolkit for the formal design and analysis of artificial intelligence-based systems," in *International Conference on Computer Aided Verification*, pp. 432–442, Springer, 2019.

[45] A. Corso, P. Du, K. Driggs-Campbell, and M. J. Kochenderfer, "Adaptive stress testing with reward augmentation for autonomous vehicle validatio," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 163–168, IEEE, 2019.

[46] S. Feng, H. Sun, X. Yan, H. Zhu, Z. Zou, S. Shen, and H. X. Liu, "Dense reinforcement learning for safety validation of autonomous vehicles," *Nature*, vol. 615, no. 7953, pp. 620–627, 2023.

[47] X. Qin, N. Arechiga, J. Deshmukh, and A. Best, "Robust testing for cyber-physical systems using reinforcement learning," in *Proceedings of the 21st ACM-IEEE International Conference on Formal Methods and Models for System Design*, MEMOCODE '23, (New York, NY, USA), p. 36–46, Association for Computing Machinery, 2023.

[48] S. A. Seshia, D. Sadigh, and S. S. Sastry, "Toward verified artificial intelligence," *Commun. ACM*, vol. 65, p. 46–55, jun 2022.

[49] B. Johnson and H. Kress-Gazit, "Probabilistic analysis of correctness of high-level robot behavior with sensor error," 2011.

[50] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[51] X. Wang, R. Li, B. Yan, and O. Koyejo, "Consistent classification with generalized metrics," 2019.

[52] P. Antonante, H. Nilsen, and L. Carlone, "Monitoring of perception systems: Deterministic, probabilistic, and learning-based fault detection and identification," *arXiv preprint arXiv:2205.10906*, 2022.

[53] M. Hekmatnejad, S. Yaghoubi, A. Dokhanchi, H. B. Amor, A. Shrivastava, L. Karam, and G. Fainekos, "Encoding and monitoring responsibility sensitive safety rules for automated vehicles in signal temporal logic," in *Proceedings of the 17th ACM-IEEE International Conference on Formal Methods and Models for System Design*, pp. 1–11, 2019.

[54] T. Wongpiromsarn and E. Frazzoli, "Control of probabilistic systems under dynamic, partially known environments with temporal logic specifications," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 7644–7651, 2012.

[55] A. Badithela, T. Wongpiromsarn, and R. M. Murray, "Leveraging classification metrics for quantitative system-level analysis with temporal logic specifications," in *2021 60th IEEE Conference on Decision and Control (CDC)*, (Austin, TX, USA (virtual)), pp. 564–571, IEEE, 2021.

[56] C. S. Pasareanu, R. Mangal, D. Gopinath, S. G. Yaman, C. Imrie, R. Calinescu, and H. Yu, "Closed-loop analysis of vision-based autonomous systems: A case study," *arXiv preprint arXiv:2302.04634*, 2023.

[57] S. Beland, I. Chang, A. Chen, M. Moser, J. Paunicka, D. Stuart, J. Vian, C. Westover, and H. Yu, "Towards assurance evaluation of autonomous systems," in *Proceedings of the 39th International Conference on Computer-Aided Design*, pp. 1–6, 2020.

[58] Y. V. Pant, H. Abbas, K. Mohta, R. A. Quaye, T. X. Nghiem, J. Devietti, and R. Mangharam, "Anytime computation and control for autonomous systems," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 2, pp. 768–779, 2021.

[59] P. Karkus, B. Ivanovic, S. Mannor, and M. Pavone, "Diffstack: A differentiable and modular control stack for autonomous vehicles," in *Proceedings of The 6th Conference on Robot Learning* (K. Liu, D. Kulic, and J. Ichnowski, eds.), vol. 205 of *Proceedings of Machine Learning Research*, pp. 2170–2180, PMLR, 14–18 Dec 2023.

[60] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[61] O. Koyejo, N. Natarajan, P. Ravikumar, and I. S. Dhillon, "Consistent multilabel classification.," in *NeurIPS*, vol. 29, (Palais des Congrès de Montréal, Montréal CANADA), pp. 3321–3329, Advances in Neural Information Processing Systems, 2015.

[62] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: Verification of probabilistic real-time systems," in *International conference on computer aided verification*, pp. 585–591, Springer, 2011.

[63] C. Dehnert, S. Junges, J.-P. Katoen, and M. Volk, "A Storm is coming: A modern probabilistic model checker," in *International Conference on Computer Aided Verification*, pp. 592–600, Springer, 2017.

[64] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11621–11631, 2020.

[65] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 12689–12697, IEEE Computer Society, jun 2019.

[66] M. Contributors, "MMDetection3D: OpenMMLab next-generation platform for general 3D object detection." https://github.com/open-mmlab/mmdetection3d, 2020.

[67] S. Gupta, J. Kanjani, M. Li, F. Ferroni, J. Hays, D. Ramanan, and S. Kong, "Far3det: Towards far-field 3d detection," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, (Los Alamitos, CA, USA), pp. 692–701, IEEE Computer Society, jan 2023.

[68] I. Incer, A. Badithela, J. Graebener, P. Mallozzi, A. Pandey, S.-J. Yu, A. Benveniste, B. Caillaud, R. M. Murray, A. Sangiovanni-Vincentelli, *et al.*, "Pacti: Scaling assume-guarantee reasoning for system analysis and design," *arXiv preprint arXiv:2303.17751*, 2023.

[69] A. Badithela, T. Wongpiromsarn, and R. M. Murray, "Evaluation metrics of object detection for quantitative system-level analysis of safety-critical autonomous systems," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Detroit, MI, USA), p. To Appear., IEEE, 2023.

[70] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *International Conference on Formal Modeling and Analysis of Timed Systems*, pp. 92–106, Springer, 2010.

[71] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Falsification of ltl safety properties in hybrid systems," *International Journal on Software Tools for Technology Transfer*, vol. 15, no. 4, pp. 305–320, 2013.

[72] G. Chou, Y. E. Sahin, L. Yang, K. J. Rutledge, P. Nilsson, and N. Ozay, "Using control synthesis to generate corner cases: A case study on autonomous driving," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 11, pp. 2906–2917, 2018.

[73] T. Wongpiromsarn, M. Ghasemi, M. Cubuktepe, G. Bakirtzis, S. Carr, M. O. Karabag, C. Neary, P. Gohari, and U. Topcu, "Formal methods for autonomous systems," *arXiv preprint arXiv:2311.01258*, 2023.

[74] G. Fainekos, H. Kress-Gazit, and G. Pappas, "Hybrid controllers for path planning: A temporal logic approach," in *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 4885–4890, 2005.

[75] R. Majumdar, A. Mathur, M. Pirron, L. Stegner, and D. Zufferey, "Paracosm: A language and tool for testing autonomous driving systems," *arXiv preprint arXiv:1902.01084*, 2019.

[76] L. Tan, O. Sokolsky, and I. Lee, "Specification-based testing with linear temporal logic," in *Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, 2004. IRI 2004.*, pp. 493–498, IEEE, 2004.

[77] G. Fraser and F. Wotawa, "Using LTL rewriting to improve the performance of model-checker based test-case generation," in *Proceedings of the 3rd International Workshop on Advances in Model-Based Testing*, pp. 64–74, 2007.

[78] G. Fraser and P. Ammann, "Reachability and propagation for LTL requirements testing," in *2008 The Eighth International Conference on Quality Software*, pp. 189–198, IEEE, 2008.

[79] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[80] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[81] C. Menghi, C. Tsigkanos, P. Pelliccione, C. Ghezzi, and T. Berger, "Specification patterns for robotic missions," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2208–2224, 2019.

[82] R. Bloem, G. Fey, F. Greif, R. Könighofer, I. Pill, H. Riener, and F. Röck, "Synthesizing adaptive test strategies from temporal logic specifications," *Formal methods in system design*, vol. 55, no. 2, pp. 103–135, 2019.

[83] J. Tretmans, "Conformance testing with labelled transition systems: Implementation relations and test generation," *Computer Networks and ISDN Systems*, vol. 29, no. 1, pp. 49–79, 1996.

[84] B. K. Aichernig, H. Brandl, E. Jöbstl, W. Krenn, R. Schlick, and S. Tiran, "Killing strategies for model-based mutation testing," *Software Testing, Verification and Reliability*, vol. 25, no. 8, pp. 716–748, 2015.

[85] R. Hierons, "Applying adaptive test cases to nondeterministic implementations," *Information Processing Letters*, vol. 98, no. 2, pp. 56–60, 2006.

[86] A. Petrenko and N. Yevtushenko, "Adaptive testing of nondeterministic systems with FSM," in *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering*, pp. 224–228, IEEE, 2014.

[87] A. Pnueli and R. Rosner, "On the synthesis of a reactive module," in *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pp. 179–190, 1989.

[88] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Sa'ar, "Synthesis of reactive (1) designs," *Journal of Computer and System Sciences*, vol. 78, no. 3, pp. 911–938, 2012.

[89] M. Yannakakis, "Testing, optimization, and games," in *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004.*, pp. 78–88, IEEE, 2004.

[90] L. Nachmanson, M. Veanes, W. Schulte, N. Tillmann, and W. Grieskamp, "Optimal strategies for testing nondeterministic systems," *ACM SIGSOFT Software Engineering Notes*, vol. 29, no. 4, pp. 55–64, 2004.

[91] A. David, K. G. Larsen, S. Li, and B. Nielsen, "Cooperative testing of timed systems," *Electronic Notes in Theoretical Computer Science*, vol. 220, no. 1, pp. 79–92, 2008.

[92] E. Bartocci, R. Bloem, B. Maderbacher, N. Manjunath, and D. Ničković, "Adaptive testing for specification coverage in CPS models," *IFAC-PapersOnLine*, vol. 54, no. 5, pp. 229–234, 2021.

[93] T. Marcucci, J. Umenberger, P. Parrilo, and R. Tedrake, "Shortest paths in graphs of convex sets," *SIAM Journal on Optimization*, vol. 34, no. 1, pp. 507–532, 2024.

[94] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science Robotics*, vol. 8, no. 84, p. eadf7843, 2023.

[95] H. Zhang, M. Fontaine, A. Hoover, J. Togelius, B. Dilkina, and S. Nikolaidis, "Video game level repair via mixed integer linear programming," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, pp. 151–158, 2020.

[96] M. Fontaine, Y.-C. Hsu, Y. Zhang, B. Tjanaka, and S. Nikolaidis, "On the Importance of Environments in Human-Robot Coordination," in *Proceedings of Robotics: Science and Systems*, (Virtual), July 2021.

[97] J. R. Büchi, *On a Decision Method in Restricted Second Order Arithmetic*, pp. 425–435. New York, NY: Springer New York, 1990.

[98] A. Duret-Lutz, A. Lewkowicz, A. Fauchille, T. Michaud, É. Renault, and L. Xu, "Spot 2.0 — a framework for ltl and omega-automata manipulation," in *Automated Technology for Verification and Analysis* (C. Artho, A. Legay, and D. Peled, eds.), (Cham), pp. 122–129, Springer International Publishing, 2016.

[99] F. Fuggitti, "Ltlf2dfa," June 2020.

[100] S. Bansal, Y. Li, L. Tabajara, and M. Vardi, "Hybrid compositional reasoning for reactive synthesis from finite-horizon specifications," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 9766–9774, Apr. 2020.

[101] N. Klarlund and A. Møller, *MONA Version 1.4 User Manual*. BRICS, Department of Computer Science, University of Aarhus, January 2001. Notes Series NS-01-1. Available from `http://www.brics.dk/mona/`.

[102] D. Goktas and A. Greenwald, "Convex-concave min-max Stackelberg games," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[103] I. Tsaknakis, M. Hong, and S. Zhang, "Minimax problems with coupled linear constraints: computational complexity, duality and solution methods," *arXiv preprint arXiv:2110.11210*, 2021.

[104] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, and D. L. Woodruff, *Pyomo–optimization modeling in python*, vol. 67. Springer Science & Business Media, third ed., 2021.

[105] V. V. Vazirani, *Approximation algorithms*, vol. 1. Springer, 2001.

[106] M. Fischetti and M. Monaci, "A branch-and-cut algorithm for mixed-integer bilinear programming," *European Journal of Operational Research*, vol. 282, no. 2, pp. 506–514, 2020.

[107] J. B. Graebener, A. S. Badithela, D. Goktas, W. Ubellacker, E. V. Mazumdar, A. D. Ames, and R. M. Murray, "Flow-based synthesis of reactive tests for discrete decision-making systems with temporal logic specifications," *arXiv preprint arXiv:2404.09888*, 2024.

[108] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, and R. M. Murray, "Tulip: a software toolbox for receding horizon temporal logic planning," in *Proceedings of the 14th international conference on Hybrid systems: computation and control*, pp. 313–314, 2011.

[109] I. Filippidis, S. Dathathri, S. C. Livingston, N. Ozay, and R. M. Murray, "Control design for hybrid systems with tulip: The temporal logic planning toolbox," in *2016 IEEE Conference on Control Applications (CCA)*, pp. 1030–1041, IEEE, 2016.

[110] S. Maoz and J. O. Ringert, "Gr (1) synthesis for ltl specification patterns," in *Proceedings of the 2015 10th joint meeting on foundations of software engineering*, pp. 96–106, 2015.

[111] S. A. Cook, "The complexity of theorem-proving procedures," in *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*, pp. 143–152, 2023.

[112] C. H. Papadimitriou, *Computational complexity*, p. 260–265. GBR: John Wiley and Sons Ltd., 2003.

[113] W. Ubellacker and A. D. Ames, "Robust locomotion on legged robots through planning on motion primitive graphs," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12142–12148, 2023.

[114] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2023.

[115] E. W. Dijkstra, "Guarded commands, nondeterminacy and formal derivation of programs," *Communications of the ACM*, vol. 18, no. 8, pp. 453–457, 1975.

[116] L. Lamport, "win and sin: Predicate transformers for concurrency," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 12, no. 3, pp. 396–428, 1990.

[117] B. Meyer, "Applying 'design by contract'," *Computer*, vol. 25, no. 10, pp. 40–51, 1992.

[118] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis, "Multiple viewpoint contract-based specification and design," in *Formal Methods for Components and Objects: 6th International Symposium, FMCO 2007, Amsterdam, The Netherlands, October 24-26, 2007, Revised Lectures* (F. S. de Boer, M. M. Bonsangue, S. Graf, and W.-P. de Roever, eds.), (Berlin, Heidelberg), pp. 200–225, Springer Berlin Heidelberg, 2008.

[119] A. L. Sangiovanni-Vincentelli, W. Damm, and R. Passerone, "Taming Dr. Frankenstein: Contract-based design for cyber-physical systems," *Eur. J. Control*, vol. 18, no. 3, pp. 217–238, 2012.

[120] P. Nuzzo, A. L. Sangiovanni-Vincentelli, D. Bresolin, L. Geretti, and T. Villa, "A platform-based design methodology with contracts and related tools for the design of cyber-physical systems," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2104–2132, 2015.

[121] I. Incer, *The Algebra of Contracts*. PhD thesis, EECS Department, University of California, Berkeley, May 2022.

[122] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. L. Sangiovanni-Vincentelli, W. Damm, T. A. Henzinger, K. G. Larsen, *et al.*, "Contracts for system design," *Foundations and Trends in Electronic Design Automation*, vol. 12, no. 2-3, pp. 124–400, 2018.

[123] I. Incer, A. L. Sangiovanni-Vincentelli, C.-W. Lin, and E. Kang, "Quotient for assume-guarantee contracts," in *16th ACM-IEEE International Conference on Formal Methods and Models for System Design*, MEMOCODE'18, pp. 67–77, October 2018.

[124] R. Passerone, Í. Íncer Romeo, and A. L. Sangiovanni-Vincentelli, "Coherent extension, composition, and merging operators in contract models for system design," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–23, 2019.

[125] R. Negulescu, "Process spaces," in *CONCUR 2000 — Concurrency Theory* (C. Palamidessi, ed.), (Berlin, Heidelberg), pp. 199–213, Springer Berlin Heidelberg, 2000.

[126] J. B. Graebener^*, A. Badithela^*, and R. M. Murray, "Towards better test coverage: Merging unit tests for autonomous systems," in *NASA Formal Methods* (J. V. Deshmukh, K. Havelund, and I. Perez, eds.), (Cham), pp. 133–155, Springer International Publishing, 2022. A. Badithela and J.B. Graebener contributed equally to this work.

[127] R. Bloem, B. Könighofer, R. Könighofer, and C. Wang, "Shield synthesis," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 533–548, Springer, 2015.

[128] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *European conference on machine learning*, pp. 282–293, Springer, 2006.

[129] I. Incer, L. Mangeruca, T. Villa, and A. Sangiovanni-Vincentelli, "The quotient in preorder theories," *arXiv:2009.10886*, 2020.

[130] O. Hussien, A. Ames, and P. Tabuada, "Abstracting partially feedback linearizable systems compositionally," *IEEE Control Systems Letters*, vol. 1, no. 2, pp. 227–232, 2017.

[131] P. Tabuada, G. J. Pappas, and P. Lima, "Composing abstractions of hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 436–450, Springer, 2002.

[132] S. Coogan and M. Arcak, "Efficient finite abstraction of mixed monotone systems," in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15, (New York, NY, USA), p. 58–67, Association for Computing Machinery, 2015.

[133] J. Liu and N. Ozay, "Abstraction, discretization, and robustness in temporal logic control of dynamical systems," in *Proceedings of the 17th international conference on Hybrid systems: computation and control*, pp. 293–302, 2014.