Kinematics and Local Motion Planning for Quasi-static Whole-Body Mobile Manipulation

Thesis by

Krishna Shankar

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2016

(Defended April 4)

Acknowledgements

I am deeply grateful to Joel Burdick. He is an exceptional adviser, mentor and human being. His patience and kindness were unwavering throughout my years at Caltech. He has taught me a great deal about how to be a good roboticist, and has shown me how to be a good person by example.

A big thank you to Nicolas Hudson whose frustrations with prior work motivated much of this thesis, and who saw promise in my toy two-dimensional examples. Working together with him was a highlight of my PhD, and I hope we can continue to collaborate for years to come.

Thanks to Richard Murray who always made time for me in his busy schedule; Richard was always able to quickly relate to my work and ideas, and provide useful insights and directions that I would not have encountered otherwise. I am thankful to Joel Tropp and Melany Hunt for being on my thesis committee, making time to meet with me, reading my work, and providing useful feedback.

Thanks to Maria Koeper, Sonya lincoln, Cheryl Geer, Lynn Seymour and Chris Silva for their travel, scheduling and organizational support. Their efforts behind the scenes were crucial to making this work possible.

The experiments in this thesis were all performed at NASA/JPL. Thanks to the many supervisors there, including Issa Nesnas, Paul Backes and Brett Kenedy, for welcoming me into their projects and labs. A special thanks to Brett who generously allowed me to put his robots' lives on the line on multiple occasions.

Thanks to Paul Hebert, Jeremy Ma, James Borders and Max Bajracharya for patiently helping me learn and navigate the complexities of making a real robotic system function. The time I've spent working and interacting with them has fundamentally changed my view of robotics and manipulation. Their work ethic, humility and passion are inspiring. Without their help, I would never have moved beyond Matlab simulations.

Thanks to all members of the Burdick group. Special thanks to Tom Allen for his friendship and the many helpful discussions on robotics, coffee, diet, exercise and life.

The time I spent in Pasadena went by very quickly, mostly due to the many great friends I made. My first friends at Caltech were made during orientation – Alex, Wael, Carlos, Ruby, Christophe, and Christos. I am grateful to them for the countless coffees, lunches and dinners we have had together, that have taken my mind off (pretending to) work. Thanks to Ryan and Abhilash for being the world's greatest roommates. Thanks to Ivan and Anandh for many conversations on control and optimization.

I almost gave up on pursuing tennis after my first year at Caltech, but luckily I met Karthik, Simon, Arnav and Dylan. We formed close friendships from watching and playing hours of tennis, and from being ardent admirers of Roger Federer (a huge thanks to RF for being my hero).

Last but not least, a big thank you to my parents and my brother for their love and support throughout my PhD. I could not have completed this work without their constant encouragement and belief.

Abstract

This thesis studies mobile robotic manipulators, where one or more robot manipulator arms are integrated with a mobile robotic base. The base could be a wheeled or tracked vehicle, or it might be a multi-limbed locomotor. As robots are increasingly deployed in complex and unstructured environments, the need for mobile manipulation increases. Mobile robotic assistants have the potential to revolutionize human lives in a large variety of settings including home, industrial and outdoor environments.

Mobile Manipulation is the use or study of such mobile robots as they interact with physical objects in their environment. As compared to fixed base manipulators, mobile manipulators can take advantage of the base mechanisms added degrees of freedom in the task planning and execution process. But their use also poses new problems in the analysis and control of base system stability, and the planning of coordinated base and arm motions. For mobile manipulators to be successfully and efficiently used, a thorough understanding of their kinematics, stability, and capabilities is required. Moreover, because mobile manipulators typically possess a large number of actuators, new and efficient methods to coordinate their large numbers of degrees of freedom are needed to make them practically deployable. This thesis develops new kinematic and stability analyses of mobile manipulation, and new algorithms to efficiently plan their motions.

I first develop detailed and novel descriptions of the kinematics governing the operation of multi-limbed legged robots working in the presence of gravity, and whose limbs may also be simultaneously used for manipulation. The fundamental stance constraint that arises from simple assumptions about friction and the ground contact and feasible motions is derived. Thereafter, a local relationship between joint motions and motions of the robot abdomen and reaching limbs is developed. Based on these relationships, one can define and analyze local kinematic qualities including limberness, wrench resistance and local dexterity. While previous researchers have noted the similarity between multi-fingered grasping and quasi-static manipulation, this thesis makes explicit connections between these two problems.

The kinematic expressions form the basis for a local motion planning problem that that determines the joint motions to achieve several simultaneous objectives while maintaining stance stability in the presence of gravity. This problem is translated into a convex quadratic program entitled the balanced priority solution, whose existence and uniqueness properties are developed. This problem is related in spirit to the classical redundancy resoxlution and task-priority approaches. With some simple modifications, this local planning and optimization problem can be extended to handle a large variety of goals and constraints that arise in mobile-manipulation. This local planning problem applies readily to other mobile bases including wheeled and articulated bases. This thesis describes the use of the local planning techniques to generate global plans, as well as for use within a feedback loop. The work in this thesis is motivated in part by many practical tasks involving the Surrogate and RoboSimian robots at NASA/JPL, and a large number of examples involving the two robots, both real and simulated, are provided.

Finally, this thesis provides an analysis of simultaneous force and motion control for multi- limbed legged robots. Starting with a classical linear stiffness relationship, an analysis of this problem for multiple point contacts is described. The local velocity planning problem is extended to include generation of forces, as well as to maintain stability using force-feedback. This thesis also provides a concise, novel definition of static stability, and proves some conditions under which it is satisfied.

Published Content and Contributions

- K. Shankar and J.W. Burdick. “Kinematics and methods for Combined Quasi-Static Stance/Reach planning in Multi-Limbed Robots”. In: *IEEE International Conference Robotics and Automation.* Hong Kong. 2014. `http://dx.doi.org/10.1109/ICRA.2014.6907286`
- K. Shankar, J.W. Burdick, and N. H. Hudson. “A Quadratic Programming Approach to Quasi-Static Whole-Body Manipulation”. In: *Workshop on Algorithmic Foundations of Robotics.* Istanbul, 2014. `http://authors.library.caltech.edu/50894/`
- K. Shankar and J.W. Burdick. “Kinematics for Combined Quasi-Static Force and Motion Control in Multi-Limbed Robots”. In: *IEEE International Conference on Robotics and Automation.* Seattle, 2015. `http://dx.doi.org/10.1109/ICRA.2015.7139830`

K.S. led the development of theory and examples in all of the above works.

Contents

List of Figures

List of Algorithms

Notation

Used Throughout	
A^+	Moore-penrose inverse of A.
$\mathcal{W}$	World-fixed coordinate frame.
$\mathcal{B}$	Coordinate frame attached to the robot's body or abdomen.
$\mathcal{E}$	Coordiante frame at the reaching end-effector.
$\mathfrak{C}$	Center-of-mass coordinate frame.
B_i	Wrench basis at i^{th} contact.
C_i	Coordinate frame attached to the i^{th} contact.
F_i	Coordinate frame attached to the i^{th} supporting 'foot'.
S_i	Coordinate frame attached to the i^{th} shoulder.
g_{ab}	Homogenous transformation from coordinate frame b to coordinate frame a.
$\mathrm{Ad}_{g_{ab}}$	Adjoint transformation. Transforms a rigid body velocity as seen in frame b to one seen in frame a.
V^a_{bc}	Twist velocity of a rigid body. To be read as "the velocity of coordinate frame c with respect to coordinate frame b as seen in coordinate frame a.
$\vec{\theta}_i$	Joint angles in the i^{th} s limb.
$\vec{\theta}$	Joint angles in all supporting limbs.
$\vec{\Theta}$	All robot joint angles.
f_i	Contact forces at the i^{th} contact.
J_i	The Spatial Jacobian for the i^{th} limb.
S	The Stance map.
$J_{\mathcal{S}}$	Stance Jacobian.
$J_{\mathcal{R}}$	Reach Jacobian.
$J_{\mathfrak{C}}$	Center of mass Jacobian.
FC	Friction cone.

Stance and Reach Kinematics for Multi-Limbed Robots	
F	Net wrench on the robot body.
(i, j)	The j^{th} link in the i^{th} limb.
ξ	spatial joint twist.
$\bar{J}_i$	mass-weighted jacobian for the i^{th} limb.

Kinematic Local Planning	
F_i^g	Reference frame attached to the robot, associated with a motion goal.
F_a^r, F_b^r	Reference frames attached to colliding bodies a and b.
$J_{F_i^g}$	Jacobian associated with F_i^g.
L_F	$S^+ J_F$ for a legged robot, J_F otherwise.

Combined Force and Velocity Kinematics and Local Planning for Multi-limbed Robots	
D	Twist basis.
K_i	Stiffness matrix for the i^{th} contact.
$K_{\mathcal{S}}$	Stance stiffness matrix.
$\mathcal{G}$	Gravity wrench.

Chapter 1

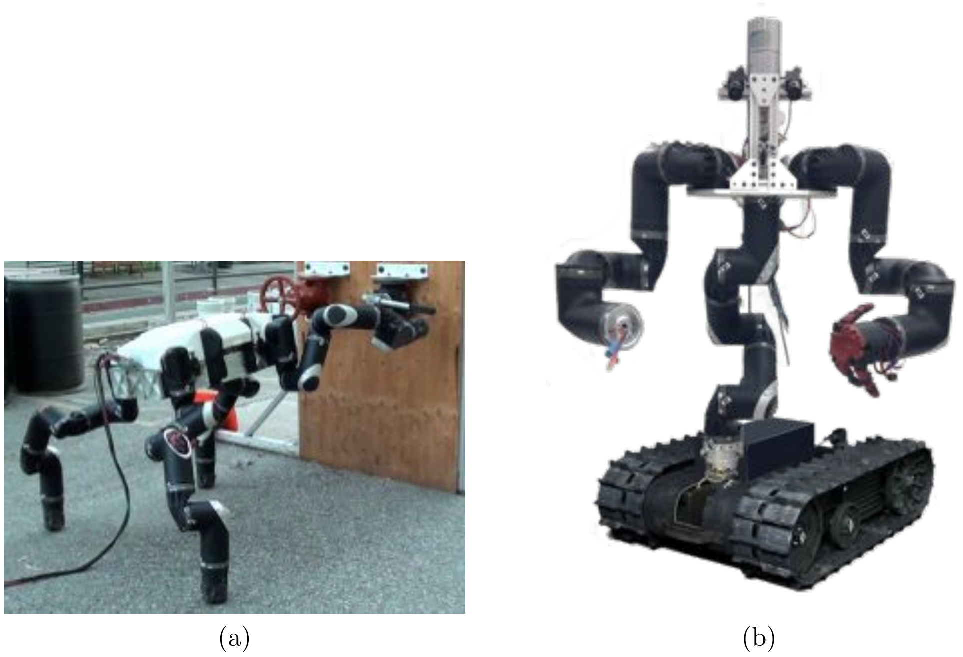
Introduction

1.1 Motivation

Robots have long been considered for their potential ability to perform tasks that may be too dangerous, expensive, or repetitive for humans. Commercial robots have been successfully deployed for many applications in highly structured environments, such as factory automation (see figure 2.2 a) and vacuum cleaning (see figure 2.2 b). However, more complex tasks in large and unstructured environments remain at the boundary of existing robotic systems.

Figure 1.1: Examples of successful commercial robots (a). iRobot Roomba — a robotic vacuum cleaner (b) Kuka robot arms designed for palletizing products.

In general, unstructured tasks involve a multitude of simultaneous goals and constraints, and therefore require robots with a high degree of dexterity and mobility with a large and whole workspace. Although it is relatively straightforward to design and build robots with great dexterity and mobility, understanding and planning their motions is algorithmically and mathematically challenging. Existing models, theories, and techniques do not adequately address many of the issues associated with mobile robots capable of manipulation.



(a) (b)

Figure 1.2: (a) RoboSimian turning a valve mounted on a wall. (b) SURROGATE. Both robots designed by NASA\JPL-Caltech.

The theories developed in this thesis are motivated by two dexterous, mobile robots, designed and built at the NASA Jet Propulsion Laboratory. Figure 1.2 (a) depicts the *RoboSimian* robot, a competitor in the DARPA Robotics Challenge (DRC, see [28]). Each of this robot's four limbs can be used either as a supporting leg, or as a manipulator (a 3-fingered hand is attached to the distal end of each limb). To locomote over rough terrain, this vehicle can use standard quasi-static gait planning and coordination techniques. However, many tasks which this robot must accomplish involve using some limbs as legs to provide a stable stance, while one or more other limbs simultaneously carry out a manipulation task, such as turning a water valve, or picking up a fire hose.

Figure 1.2 (b) depicts the *SURROGATE* robot. It has a tracked base, an articulated torso, and two limbs — both the torso and the limbs are identical to the limbs of RoboSimian. Though the mobility of Surrogate is considerably simpler than that of Robosimian, there are many tasks that require coordinated motion of the manipulating limb(s) the articulated torso, as well as the base. The torso and arm movements can cause the base to tip over, and so their motions must be carefully managed.

In practice, these robots will initially be used in hazardous environments for activities like disaster relief, where there is a human operator likely to be in the loop. This removes the need for complete high level autonomy. However, the joint motions required to move an end-effector as desired can be extremely counter-intuitive due to the complex workspace of such mechanisms, their high number of mechanical degrees of freedom, and the need to account for stance stability and other safety-critical constraints throughout the task.

This thesis presents new tools, methods, and analysis to address many tasks that arise in mobile manipulation. Many of the ideas in the thesis are demonstrated in physical and simulated experiments using the RoboSimian and Surrogate robots.

1.2 Problem Statement

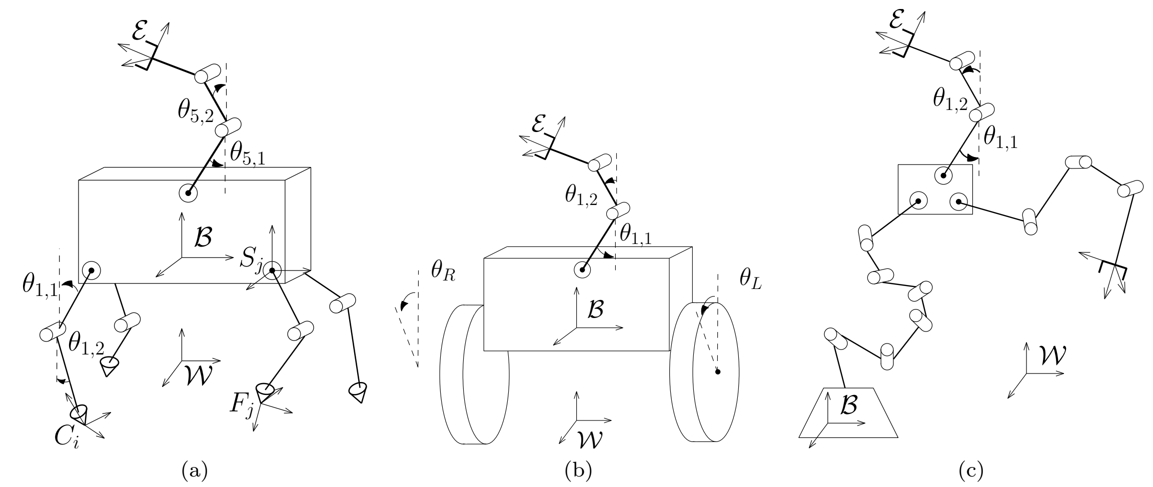


Figure 1.3: Key reference frames for (a) A Legged-base Robot (b) A Wheeled-base robot (c) An Articulated-base robot

A generalized problem is formulated to precisely motivate such methods: consider one or more (possibly redundant) serial chain manipulator arms mounted on a mobile robot base. The base could be a wheeled or tracked vehicle, or it may be a multi-legged walker (see Figures 1.3 (a) and (b)). The mechanism may also include an articulated neck upon which visual and range sensors are mounted (see Figure 1.3 (c)). Of particular interest are the cases where the arms have sufficient reach and mass such that the mobile vehicle may tip over in the presence of gravity when they are extended too far during a manipulation task.

Suppose the robot must complete a manipulation task that can be described by a tool frame location and a desired wrench applied to the environment, or a sequence of such locations and wrenches.

1. It is clear that motions of the end effector or robot body may arise from either motions of the

mobile base, or motions of joints in a serial chain torso – how does one account for and model this for a general mobile-manipulator?

2. What arm and base configurations satisfy the manipulation constraints?
3. How should base and limb motions be apportioned to achieve the goal?
4. How can vehicle tip-over be prevented — can the limbs be moved so as to keep the system center of mass over a safe region of support? Can motions be chosen to *improve* stability with respect to gravitational forces?
5. How are natural task constraints, such as avoiding mechanism self-collision, obstacle avoidance, preferred camera gaze direction, and joint movement limits, incorporated?
6. When the robot is making contact with the world, how are forces generated?
7. How are motions and forces related when the end-effector is in contact with a compliant surface?
8. Can quasi-static stability in the presence of gravity be defined succinctly in terms of robot geometry and forces?

These problems form a generalized inverse kinematic problem, where the distal end of the manipulator(s) must be placed at specified locations applying known wrenches, while incorporating numerous constraints as well as gravitational stability, and optimality criteria are used to resolve ambiguities in the case of multiple possible solutions. The optimality criteria also endow the solution with desirable properties.

In this thesis this problem is addressed both practically and theoretically, for a general class of mobile manipulators.

1.3 Review of Existing Work

In this section, related work is presented broadly as it applies to mobile manipulation in general. Subsequent chapters will provide more a detailed review of related work as necessary.

1.3.1 Mobile Manipulation

Mobile manipulation is the study of robots that have moving bases mounted with one or more limbs with end-effectors that interact with the world. One of the earliest works to consider mobile-manipulation was by Lynch [49], modeling a simple 2 DoF (degree-of-freedom) mobile-manipulator as a linear system locally, and analyzing stability. Since then, a great deal of work has been done on many topics related to mobile manipulation using wheeled bases.Yamamoto and Yun [100, 103] were the first to consider the problem of tracking a trajectory by coordinating base and manipulator motions for a simple mobile-manipulator, using nonlinear control techniques for base motions to maximize a manipulability measure locally. Yamamoto elaborated on this work in his thesis [102].

Seraji introduced an augmented Jacobian for mobile-rovers with manipulators, and related control schemes [85, 86]. Khatib [41] provides a torque-control method for coordinating arm motions with the motions of a holonomic base by projecting the full robot dynamics into the workspace, and using feedback linearization. Holmberg and Khatib [30] take a similar approach for torque-control of a robot with holonomic wheels and other desirable design features. All of these approaches depend on differentiating local parametrizations of $SE(3)$ (see Chapter 2), and are therefore susceptible to coordinate singularities.

Some work has also been done on planning motions globally for mobile manipulation. Berenson, Kuffner and Choset [3] provide a method for moving an object from a start to a goal configuration by generating initial and final robot configurations using evolutionary algorithms, and then finding a path between them using a Rapidly exploring Random Tree (RRT) [46]. Yang and Brock [104] provide a method that combines workspace control like that of [41] with heuristic sampling for global waypoints that satisfy constraints, and using local control techniques and potential functions to establish connectivity between milestones or waypoints. These methods are inefficient, and rely heavily on the quality of samples for success.

Though the early work on mobile-manipulation was for robots with wheeled bases on earth, mobile manipulation in space has also been well studied. Manipulation and coordinated base/limb motion in space are very different, as conservation of momentum plays a major role. Papadopoulos and Dubowsky [16, 67, 68] explore dynamic singularities that arise from conservation of angular momentum, and show that control approaches for manipulation on earth can also be applied in space under mild assumptions. Umetani and Yoshida [95] address the problem of controlling free-floating manipulators by generalizing the traditional fixed-base Manipulator Jacobian to include the dynamic constraints imposed by conservation of linear and angular momentum.

Manipulation by humanoids is much more challenging than that by robots with wheeled bases, as there are generally many additional degrees of freedom, complex kinematic constraints imposed by the two legs maintaining contact and dynamic constraints on stability. Traditional planning techniques do not work for humanoids, as they have many degrees of freedom. Kuffner et al. [43] provide sampling-based approaches that rely on known inverse kinematics solutions for manipulation and whole-body motions; this class of approaches cannot be used if a final robot configuration is not known. Humanoid control, though still challenging, can be achieved in some regimes using linearization and approximation. Inoue et al. [32] provide a method that combines linear feedback controllers to ensure arm trajectory-tracking and balancing. Sugihara and Nakamura [94] introduce the Center-of-Gravity Jacobian, and use it to track trajectories while rejecting disturbances. These works depend on simple, approximate models (based on pendulums) for stability. Sentis and Khatib [82] extend the 'operation space' approach taken in [41] to humanoids – this work does not handle contact or stability as it cannot handle inequality constraints.

1.3.2 Manipulator Kinematics and Control

A key feature of the robots being considered in this thesis is the presence of one or more *serial chain manipulators* — robot arms with a series of single DOF joints; the kinematics, modeling, and analysis of manipulators is covered formally in great detail by Murray, Li and Sastry [58], and Chapter 2 includes a short discussion. A fundamental problem associated with serial chain manipulators is the *inverse kinematics problem*: find all of the manipulator joint positions that result in a specified position and orientation of the robot's hand or tool. When the robot has six joints or less, this problem can be solved algebraically for all possible solutions. For robots with six degrees of freedom or less, the inverse kinematics problem has been solved explicitly [51, 69, 70] and there are always finitely many solutions.

When a robot has seven or more joints there is no general explicit solution to the inverse kinematics problem, and there are potentially infinitely many solutions. Such manipulators are known as *redundant*, as they have more degrees of freedom than necessary in some cases. For redundant manipulators, one can attempt to solve the inverse-kinematics problem incrementally or numerically. This class of techniques involves moving *locally* in the *direction* of the goal, by using a linear model. This idea was introduced by Whitney [97], who also suggested using the pseudo-inverse of the Jacobian to resolve redundancy. These techniques for solving the serial-chain inverse kinematics problem incrementally are now classical results; a broad survey is given by Buss in [11].

Nakamura and Hanafusa [61] generalized these techniques to include additional constraints under a framework known as *Task-Priority Redundancy Control.* This approach imposes one task rigidly (task), and encourages another by minimizing a cost (priority). This work was generalized to include an arbitrary number of sequential tasks by Siciliano and Slotine [92], where a subsequent task does not interfere with a previous one. A great deal of effort has gone into efficient algebraic and numerical solutions of this problem and modest extensions thereof, and this class of approaches has come to be known as Hiearchical Quadratic Programming [17, 35, 36, 52]. Whereas these works impose a strict hierarchy between narrowly defined tasks, the novel local problem formulated in this thesis allows balancing of any number of goals and provides a much more general weighting and task framework. These works have several other shortcomings and differences that are discussed in section 4.1.2.

Many manipulation tasks, like painting walls or turning valves require the simultaneous regulation of forces and motions. One of the earliest works to describe simultaneous force/position control is [71]. This work was preceded by efforts to model compliance relationships, including a linear stiffness model [76, 77] as well as dynamic compliance models [29]. Whitney [98] provides a survey on the topic of Robot Force Control. Khatib [40] introduced the approach of transforming idealized robot dynamics into the *task-space*, so that standard linear-control techniques (e.g. PD-control) could be applied in cartesian coordinates, rather than at the joint level. This approach has been extended to a vast number of settings for different robots (e.g. [30, 41, 82]) . Three limitations with this approach are that it is susceptible to coordinate singularities associated with the forward kinematics map to $SE(3)$, that it relies on a known and idealized *dynamic* model of joint motions (many robots are very stiff with high friction, and their dynamics cannot be easily characterized), and that it cannot accommodate inquality constraints meaning that a large number of practical constraints like contact friction and collision avoidance cannot be handled.

1.3.3 Multi-Fingered Hands: Kinematics and Control

A major contribution of this thesis is the derivation and analysis of the kinematics of a Multi-Legged Robot's Stance and Reach (see Chapter 3). It builds on the kinematics of Multi-Fingered hands. Intuitively, the parallel between a hand manipulating an object and a legged robot adjusting its stance is clear, and has been noted by many [8, 27, 33, 45, 96]. Hand kinematics were introduced by Kerr and Roth [39], and extended by Montana [57]. Control and geometry for manipulating objects in a hand have been studied carefully by many, including Murray and Sastry [59] as well as Han and Trinkle [21, 22]. This thesis makes a formal connection between the dexterous hand kinematics and the kinematics of a legged robot's stance.

1.4 Contributions of This Thesis

This thesis provides an explicit kinematic formulation describing the adjustment of a legged robot's *stance*. The derivation is general with respect to the type of contact or hold at each leg, as well as the number of limbs or joints that the robot has. This derivation allows a precise and explicit comparison to the kinematics of multi-fingered hands. With the formulation of Stance kinematics, many interesting local properties with analogues in dexterous manipulation are introduced and analyzed. The kinematics of *reaching* is also introduced, and simple conditions on when a legged robot can reach in any direction are discussed. Towards the use of geometric definitions of static stability, the *stance-constrained Center of Mass Jacobian* is defined and derived. Stance, Reach and Center-of-Mass kinematics are also derived for robots with wheeled and articulated bases.

A general local planning problem, associated with an arbitrary mobile manipulation task, is formulated as a simple constrained minimization problem whose properties are analyzed. This problem is generalized to include a vast number of geometric goals and constraints that occur naturally in the context of mobile manipulation. Simple linear programs to certify feasibility of a set of constraints, and to generate feasible constraints are introduced. These optimization problems can be solved iteratively to generate trajectories, or within a feedback loop in real time. A number of examples demonstrating the different classes of goals and constraints are given using the RoboSimian and Surrogate robots at NASA-JPL.

Many Mobile-Manipulation tasks involve producing motions and forces simultaneously. Classical linear-stiffness models are extended to multi-limbed robots by introducing *stance stiffness*, that relates motions of the robot to forces on the robot when in contact with the world. Stability in the presence of gravity is defined in terms of contact forces. The local planning problem for a manipulation task involving desired forces is formulated and analyzed. Simple planar examples are provided.

1.5 Structure of the Thesis

Chapter 2 provides mathematical preliminaries and background. The third chapter derives stance, reach, and center-of-mass kinematics for legged robots, and provides results that are analagous to key results in dexterous manipulation. Chapter 4 introduces a local planning problem for mobile manipulation, as well as a characterization of its solution, generalizations, and numerous examples. Chapter 5 describes the kinematics associated with simultaneous force and motion goals, formulates a new local planning problem, and provides a novel definition for static equlibrium. Finally, the thesis concludes in chapter 6, and provides directions for future work.

Chapter 2

Background and Preliminaries

This chapter reviews some of the basic results in manipulator kinematics and optimization used in this thesis. The material introduced here is covered by a number of textbooks.

2.1 Rigid Body and Manipulator Kinematics

This section reviews basic geometric and kinematic conventions that will be used throughout the thesis. The notation follows [58]. The position and orientation of a reference frame b with respect to a reference frame a is given by a homogeneous transformation

$$g_{ab} = \begin{pmatrix} R_{ab} & p_{ab} \\ 0 & 1 \end{pmatrix} \quad \in \quad \mathrm{SE}(3),$$

where $R_{ab} \in \mathrm{SO}(3)$ is a rotation matrix from frame b to frame a, and $p_{ab} \in \mathbb{R}^3$ is the position of frame b's origin with respect to frame a's origin. The matrix g_{ab} transforms homogeneous vectors in frame b to corresponding vectors in frame a. The rigid-body velocity of frame b with respect to frame a as seen in frame b (body velocity) is written as a *twist*,

$$\hat{V}^b_{ab} = g_{ab}^{-1}\dot{g}_{ab} = \begin{pmatrix} R_{ab}^T\dot{R}_{ab} & R_{ab}^T\dot{p}_{ab} \\ 0 & 0 \end{pmatrix} \quad \in \quad \mathrm{se}(3).$$

The skew symmetric matrix $R_{ab}^T\dot{R}_{ab}$ can be interpreted as an angular velocity. In other words, for some $\omega^b = \begin{pmatrix}\omega_1 & \omega_2 & \omega_3\end{pmatrix}^T$,

$$R_{ab}^T \dot{R}_{ab} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \quad \in \quad \mathrm{so}(3)$$

and w is defined in terms of $R_{ab}^T \dot{R}_{ab}$ by the operation $(\cdot)^\vee$, i.e.

$$\omega^b = (R_{ab}^T \dot{R}_{ab})^\vee.$$

Twists can be written as *twist vectors*, in the form

$$V_{ab}^b = \begin{bmatrix} R_{ab}^T \dot{p}_{ab} \\ (R_{ab}^T \dot{R}_{ab})^\vee \end{bmatrix}.$$

A rigid body velocity V_{ab}^c can be expressed with respect to a new frame d by using the *Adjoint Mapping*, Ad_{dc},:

$$V_{ab}^d \triangleq \mathrm{Ad}_{g_{dc}} V_{ab}^c = \begin{bmatrix} R_{dc} & \hat{p}_{dc} R_{dc} \\ 0 & R_{dc} \end{bmatrix} V_{ab}^c.$$

A *unit twist* is a twist with magnitude 1. The instantaneous motion of an end-effector due to a prismatic or revolute joint (with respect to a reference frame) can be written as a unit twist ξ. Suppose that there is a single revolute joint between reference frames a and b, and that at the joint's zero configuration, the transformation between frame b and frame a is $g_{ab}(0)$. Then, the transformation after rotation by an amount θ around twist axis ξ is given by

$$g_{ab}(\theta) = \exp(\hat{\xi}\theta) g_{ab}(0).$$

$exp(\cdot)$ is the standard matrix exponential; it always produces a homogeneous transformation when applied to a unit twist. The special structure of the exponential of a twist is detailed in [58].

Wrenches are force-torque pairs. A wrench written is written with respect to a coordinate frame a as $F_a = (f_a, \tau_a)$ – this wrench is applied at the origin of a and its components are defined in terms of the orientation and position of a. A wrench written with respect to a coordinate system a can be written with respect to a coordinate system c according to

$$F_c = \mathrm{Ad}_{g_{ac}^T} F_a.$$

For a serial-chain manipulator (a robot arm), the base or stationary frame is denoted s, while the frame associated with the end-effector or tool is denoted t. Suppose a manipulator has n revolute joints, and that the transformation between the tool and the stationary frame in the zero-configuration is $g_{st}(0)$. Then for arbitrary joint angles $\theta_1, \ldots, \theta_n$,

$$g_{st}(\theta_1, \cdots, \theta_n) = e^{\hat{\xi}_1 \theta_1} e^{\hat{\xi}_2 \theta_2} \cdots e^{\hat{\xi}_n \theta_n} g_{st}(0).$$

The velocity of the end-effector of a serial-chain manipulator is generated by velocities of its joints. In general, a linear map between joint velocities and end-effector rigid-body velocity is called a *Manipulator Jacobian Matrix*. Maps from joint velocities to *twists* of the end-effector generally take two forms: the *Spatial Jacobian* and the *Body Jacobian*. The Spatial Jacobian $J_{st}^s(\theta)$ is a map from joint velocities to the twist velocity of the end-effector with respect to the stationary frame as seen in the stationary frame, V_{st}^s. It is defined by

$$J_{st}^t(\theta) = \begin{bmatrix} \xi_1 & \xi_2' & \cdots & \xi_n' \end{bmatrix},$$

where

$$\xi_i' = \mathrm{Ad}_{\left(e^{\hat{\xi}_1 \theta_1} \cdots e^{\hat{\xi}_{i-1} \theta_{i-1}}\right)} \xi_i.$$

The spatial velocity of a manipulator is then given by $V_{st}^s = J_{st}^s(\theta)\dot{\theta}$ where $\theta = \begin{pmatrix} \theta_1, \cdots, \theta_n \end{pmatrix}$. The Body Jacobian is defined by

$$J_{st}^t(\theta) = \begin{bmatrix} \xi_1 & \xi_2^\dagger & \cdots & \xi_n^\dagger \end{bmatrix},$$

where

$$\xi_i^\dagger = \mathrm{Ad}_{\left(e^{\hat{\xi}_i \theta_i} \cdots e^{\hat{\xi}_n \theta_n}\right)} \xi_i.$$

The velocity of the tool as seen in the tool frame is then given by $V_{st}^t = J_{st}^t(\theta)\dot{\theta}$.

Two pieces of notation are introduced for convenience. First, whenever the superscript of a velocity is omitted, the velocity is a body velocity, i.e.

$$V_{bc} \triangleq V_{bc}^c.$$

I will write the spatial Jacobian of the i^{th} limb with respect its shoulder (the point where it attaches to the body) as $J_i(\theta_i)$, where

$$J_i(\theta_i) \triangleq J^{s_i}_{s_i f_i}(\theta_i).$$

Later chapters will consider the case of multiple arms attached to a commmon body.

The instantaneous power of a rigid body moving with velocity V and sustaining wrench F is given by $V \cdot F$ (where V is a body velocity). This knowledge can be used to relate joint torques to wrenches. The end-effector of a manipulator moves according to $V^t_{st} = J^t_{st} \cdot \theta$, and it undergoes a wrench F_t. The instantaneous power in this situation is given by $V^t_{st} \cdot F_t$. Define τ to be the vector of joint-torques, with the same dimension as $\dot{\theta}$. Assuming that energy is conserved, it does not matter whether power is measured in the joints or at the end-effector; therefore

$$\begin{aligned} \dot{\theta}^T \tau &= V^t_{st} \cdot F_t \\ &= \dot{\theta}^T (J^t_{st})^T F_t \end{aligned}$$

This implies that

$$\tau = (J^t_{st})^T F_t.$$

A wrench basis is a matrix whose columns span a sub-space of wrenches. If a wrench, expressed with respect to coordinate frame A, is known to lie in a k dimensional sub-space, then it may be written as a linear combination of the elements in a wrench basis,

$$F_a = B_a f,$$

where $B_a \in \mathbb{R}^{6 \times k}$ is a wrench basis, and f are the coordinates for F_a in the basis B_a. Wrench bases arise in the context of kinematics when local motions or forces are restricted to subspaces.

2.2 Convex Optimization

The work of this thesis includes the use of Linear and Quadratic Programs. These are relatively simple classes of constrained optimization problems. These classes of problems arise often because they are somewhat general, the mathematical functions that they contain are simple, and they can be solved very efficiently in many cases. The efficiency and simplicity of analysis of these problems arise from a property of functions known as *convexity.*

This section will provide an introduction to the main components of a constrained optimization problem, the classes of objective and constraint functions that arise in this thesis, and a short discussion of the numerical methods that are used to solve these problems in practice. The material in this section is covered in detail by a number of texts [5,6,48,63]. The use of Linear and Quadratic Programs in the context of Robotics will be reviewed in Chapter 4.

2.2.1 Convex, Linear and Quadratic Functions

A set X is *convex* if for all $x, y \in X$, all the points on the line segment between x and y are also contained in X. In other words, for all θ with $0 \leq \theta \leq 1$,

$$\theta x + (1-\theta)y \in X.$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if for all $x, y \in \mathbb{R}^n$ and all θ satisfying $0 \leq \theta \leq 1$,

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y). \tag{2.1}$$

In other words, the graph of a convex function is a convex set — this means that the line segment between any two points on the graph of f lie above the graph of f.

Suppose that f is differentiable on $\mathbb{R}^n$, so that $\nabla f(x)$ exists for all x. Then f is convex if and only

$$f(y) \geq f(x) + \nabla f(x)^T(y-x).$$

One final definition of the convexity of a function on $\mathbb{R}^n$ requires the introduction of some simple matrix properties. Let $M \in \mathbb{R}^{n\times n}$ be a matrix. Then M is:

- Positive definite if for all $x \in \mathbb{R}^n$, $x^T M x > 0$, written as $M \succ 0$.
- Positive semi-definite if for all $x \in \mathbb{R}^n$, $x^T M x \geq 0$ written as $M \succeq 0$.
- Negative definite if for all $x \in \mathbb{R}^n$, $x^T M x < 0$, written as $M \prec 0$.
- Negative semi-definite if for all $x \in \mathbb{R}^n$, $x^T M x \leq 0$, written as $M \preceq 0$.
- Indefinite otherwise.

If f is twice differentiable on $\mathbb{R}^n$, so that $\nabla^2 f(x)$ exists for all x, then f is convex if and only if the hessian is *positive semi-definite* everywhere. In other words, for all x,

$$\nabla^2 f(x) \succeq 0. \tag{2.2}$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is *linear* if for all $x,\ y \in \mathbb{R}^n$ and for all $\alpha,\ \beta \in \mathbb{R}$,

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y).$$

It is immediately clear that all linear functions are convex because they satisfy the convexity condition (2.1) with equality. Such an f can be written in the form

$$f(x) = a^T x + b$$

for some $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$.

Note that a vector-valued function of the form $f(x) = Ax + b$, where $f : \mathbb{R}^n \to \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, can be written as m component scalar valued linear functions $f_i(x) = a_i^T x + b_i$, where $a_i \in \mathbb{R}^n$ is the i^{th} row of A and b_i is the i^{th} element of b.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is quadratic if, for any $x \in \mathbb{R}^n$, for some $Q \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$, it can be written in the form

$$f(x) = x^T Q x + x^T b + c. \tag{2.3}$$

It is clear from the convexity condition for twice differentiable functions (2.2) that a quadratic function of the form (2.3) is convex if and only if $Q \succeq 0$.

2.2.2 Optimization Problems

A *Mathematical Optimization problem* is of the form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq b_i \qquad i = 1 \ldots m, \end{array} \tag{2.4}$$

where $x \in \mathbb{R}^n$ is *the decision variable*, the quantity being sought. $f_0 : \mathbb{R}^n \to \mathbb{R}$ is known as the *objective function,* and the functions $f_i : \mathbb{R}^n \to \mathbb{R} \quad i = 1 \ldots m$ are known as *constraint functions.* The set of all x satisfying the constraints $\{x \in \mathbb{R}^n | f_i(x) \leq 0 \quad i = 1 \ldots n\}$ is known as the *feasible set.* It is easy to show that this is a convex set if the constraint functions are convex. The quantity

x^* is called the *solution* to this problem if, for all the x in the feasible set, the inequality

$$f_0(x^*) \leq f_0(x)$$

is satisfied. An optimization problem may have a unique solution, many solutions, or no solutions. The optimal level of the objective $f_0(x^*)$ is called the *optimal value.*

A huge variety of problems can be written in the form (2.4). Note that the constraints may represent equality constraints ($f(x) \leq b$ and $f(x) \geq b$ imply $f(x) = b$), and strict inequalities of the form $f(x) > 0$ can be written as non-strict inequalities for all practical purposes [1]. Note also that a problem that requires maximization of the objective is equivalent to the minimization problem with the same constraints but the objective multiplied by -1.

An *Convex Optimization* problem is one that can be written in the form (2.4), where the objective and all of the constraint functions $f_0, f_1, \cdots f_n$ are convex. Convex optimization problems are relatively straightforward to analyze and manipulate because the convexity of the feasible set and objective function are powerful properties. The convexity of the objective also implies that a local minimum is a global minimum — this allows for relatively simple and very efficient numerical techniques to solve real Convex Optimization problems.

2.2.3 Linear and Quadratic Programs

An optimization problem of the form (2.4) is called a *linear program* (LP) if the objective and constraint functions $f_0, f_1, \cdots, f_m$ are all linear functions. A linear programming problem in *standard form* is written as

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ & x \succeq 0, \end{aligned}$$

where $x \succeq 0$ means that x is positive element-wise. All linear programs can be written in this form using simple coordinate transformations and the introduction of additional variables. Linear programs are widely used in science and engineering for their generality, ease of analysis, and efficient

[1]when the optimal value is on the boundary of a strict inequality, it does not make practical sense, and from an implementation perspective is not meaningful. Instead, a perturbed problem with a non-strict inequality can be posed.

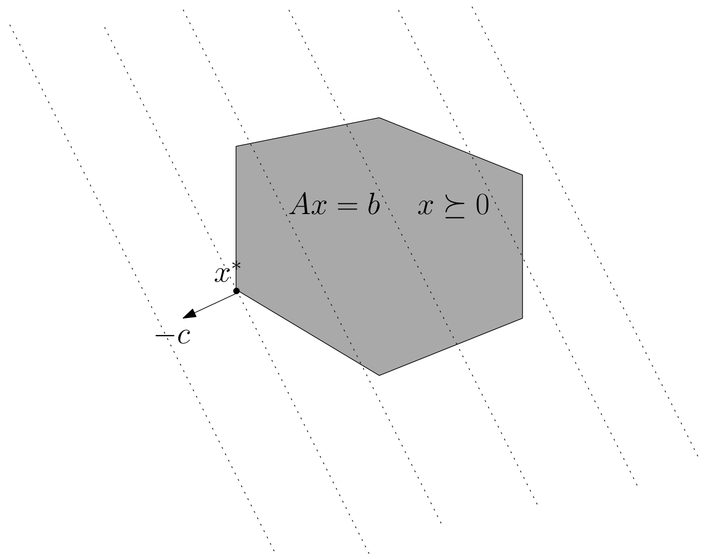


Figure 2.1: A pictorial representation of the solution to an LP in Standard Form. Dotted lines represent level sets of the objective. Gray shaded regions represent the feasible set. The solution to the LP lies at a corner of the feasible region, where the objective cannot be reduced further in the direction of c without violating the constraints.

numerical solutions. They are used in this thesis to efficiently check whether a set of given constraints are feasible in section 4.6. One of the first methods for solving linear programs was known as the *simplex method* by Dantzig [13]. This method has an exponential worst-case complexity. Interior point methods, introduced by Karmarkar [38], were shown to succeed efficiently, with polynomial-time worst-case complexity. Interior point methods will be reviewed in the following section.

A *Quadratic Program* (QP) is an optimization problem with a convex, quadratic objective, and linear constraints. In other words, it is a problem of the form (2.4) where f_0 is a quadratic function, and $f_1 \dots f_n$ are linear functions. A Quadratic Program can be written as

$$\begin{aligned} \text{minimize} \quad & x^T Q x + a^T x + b \\ \text{subject to} \quad & Bx = c \\ & Gx \preceq d, \end{aligned}$$

where $Q \succeq 0$, $B \in \mathbb{R}^{m \times n}$, $c \in \mathbb{R}^m$, $G \in \mathbb{R}^{p \times n}$, and $d \in \mathbb{R}^p$. Quadratic Programs can also be solved efficiently (in polynomial time) using Interior Point Methods (see [6, 63]). They are used extensively in chapter 4 to solve a local planning problem for mobile manipulation.

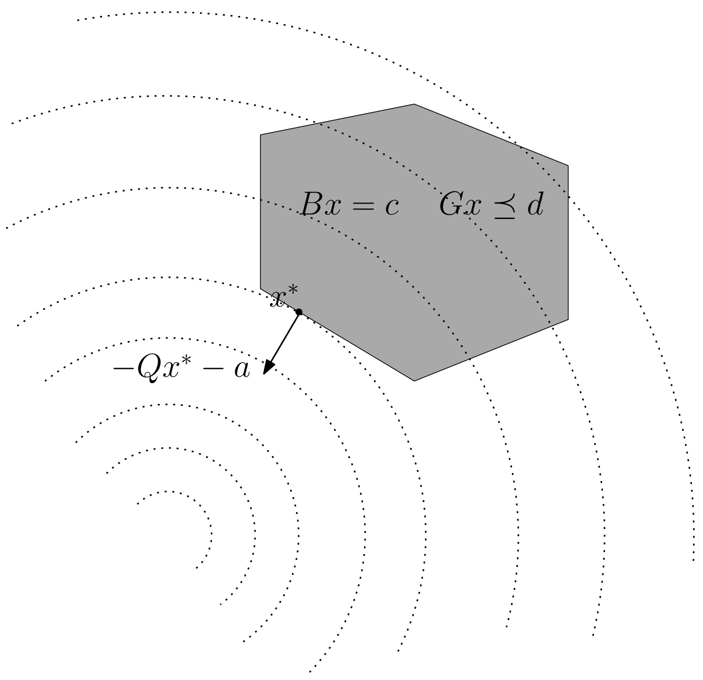


Figure 2.2: A pictorial representation of the solution to an QP in Standard Form. Dotted lines represent level sets of the objective. Gray shaded regions represent the feasible set. The solution to the QP lies at a point where the objective cannot be reduced in the direction of steepest descent $-Qx^* - a$ without leaving the feasible set.

2.2.4 Analytical and Numerical Solution of Convex Optimization Problems

The solution to Convex Optimization problems is obtained by solving a set of equations known as *Karush-Kuhn-Tucker (KKT) Conditions* [6] analytically or numerically. This sub-section briefly introduces and motivates these conditions, and provides examples of when an analytical solution is possible, and very briefly describes the class of numerical methods known as *interior point methods.*

An *Optimization Problem in Standard Form* is written as

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0 \qquad i = 1 \ldots m \\ & h_i(x) = 0 \qquad i = 0 \ldots p. \end{aligned} \tag{2.5}$$

The *Lagrangian* associated with this problem is a function $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ defined by

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x). \tag{2.6}$$

This is the function obtained by augmenting the objective with a weighted sum of the constraints. The weights $\lambda \in \mathbb{R}^m$ and $\nu \in \mathbb{R}^p$ are known as *Lagrange multipliers*, while x is referred to as a *primal variable*, and (2.5) is the *primal problem.* λ and ν are also called *dual variables.*

The *Lagrange Dual Function*, $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$, is obtained minimizing [2] the Lagrangian (2.6) with respect to x, i.e.

$$g(\lambda, \nu) = \min_x L(x, \lambda, \nu).$$

Any value of the Lagrange Dual function is clearly a lower bound on the optimal objective value of the primal problem. To get the largest such bound, one solves the *(Lagrange) Dual Problem*:

$$\begin{aligned} &\text{maximize} && g(\lambda, \nu) \\ &\text{subject to} && \lambda \succeq 0. \end{aligned} \tag{2.7}$$

The solution to the Dual Problem (2.7) with respect to the dual variables λ, ν provides the greatest-lower bound on the optimal value of the optimization problem. In the case of Linear and Quadratic Programs, this bound is tight and *Strong Duality* – which occurs when the primal and dual problems have the same optimal objective value – holds.

The KKT conditions arise from the fact that the optimal x^* and dual variables λ^*, ν^* satisfy the original constraints, minimize the Lagrangian and maximize the Lagrange Dual Function. For the Optimization Problem written in the form (2.5), the KKT conditions are as follows: let $\tilde{x} \in \mathbb{R}^n, \tilde{\lambda} \in$

[2]It is assumed that the optimization problem is feasible, the domain is $\mathbb{R}^n$ and the objective is convex; in general however, the dual is obtained by *infimizing* the Lagrangian with respect to x as the minimum may not be attained in the problem domain, i.e., more generally, $g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$ [6]

$\mathbb{R}^m, \tilde{\nu} \in \mathbb{R}^p$. Then, if

$$\begin{aligned}
f_i(\tilde{x}) &\le 0, & i &= 1, \ldots, m \\
h_i(\tilde{x}) &= 0, & i &= 0, \ldots, p \\
\tilde{\lambda}_i &\ge 0, & i &= 0, \ldots, m \\
\tilde{\lambda}_i f_i(\tilde{x}) &= 0, & i &= 0, \ldots, m \\
\nabla f_0(\tilde{x}) + \sum_{i=1}^{m} \tilde{\lambda}_i \nabla f_i(\tilde{x}) + \sum_{i=1}^{p} \tilde{\nu}_i \nabla h_i(\tilde{x}) &= 0,
\end{aligned} \tag{2.8}$$

$\tilde{x}, \tilde{\lambda}, \tilde{\nu}$ are primal and dual optimal, and $f_0(\tilde{x})$ is the optimal value.

An example of a Convex Optimization problem that can be solved explicitly by a direct application of the KKT conditions is the following equality-constrained Quadratic Program:

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2} x^T Q x + c^T x + d \\
\text{subject to} \quad & Ax = b,
\end{aligned} \tag{2.9}$$

where $Q \succeq 0$. If the (relevant) KKT conditions are applied to this problem, one sees that the optimal x^* satisfies

$$Ax^* = b \quad Qx^* + c + A^T \nu^* = 0,$$

which can be rewritten as a single linear system of equations,

$$\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} x^* \\ \nu^* \end{pmatrix} = \begin{pmatrix} -c \\ b \end{pmatrix}.$$

The matrix $\begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix}$ is called the *KKT Matrix*. When there is no solution to this system of equations, the problem is either infeasible or unbounded. When the matrix is invertible, there is a unique solution (x^*, ν^*). If the KKT matrix is singular, but the system of equations has a solution, every solution is optimal.

This system of equations has a unique solution whenever the KKT Matrix on the left-hand-side is invertible. When A is full rank, this matrix is invertible when $\mathcal{N}(A) \cap \mathcal{N}(Q) = \{0\}$; this is true, for example, when $Q \succ 0$.

Many optimization problems that are easy to formulate may not be easily solved analytically. In such cases, particular instances of the problem may be solved efficiently by using a class of numerical techniques known as *interior point methods.* The general idea of Interior Point Methods is to generate a sequence of solutions or feasible points (in a similar fashion to *Newton's Method for Minimization*) that are *interior* to the feasible set. Successive points in the sequence are generated by using the solution to an approximate problem at the previous point as a step-direction.

This sequence is generated by starting with a feasible point (there are number of methods for finding feasible points for convex problems known as *Phase I Methods* [6]). At each point in the sequence, the optimization problem is approximated locally as a Quadratic Program. This is done by replacing the objective with a second order (Taylor) approximation (much like *Newton's Method*), and replacing inequality constraints with *Barrier* or penalty functions added to the objective. Thus, for each point in the sequence, a problem of the form (2.9) is solved. Each such problem instance provides a *direction* Δx. If one moves along the approximately optimal direction Δx by an appropriate amount (as is the case for Newton's Method), it can be shown that such methods converge arbitrarily close to the optimal value in polynomial time [63].

Chapter 3

Stance and Reach Kinematics for Multi-Limbed Robots

3.1 Introduction

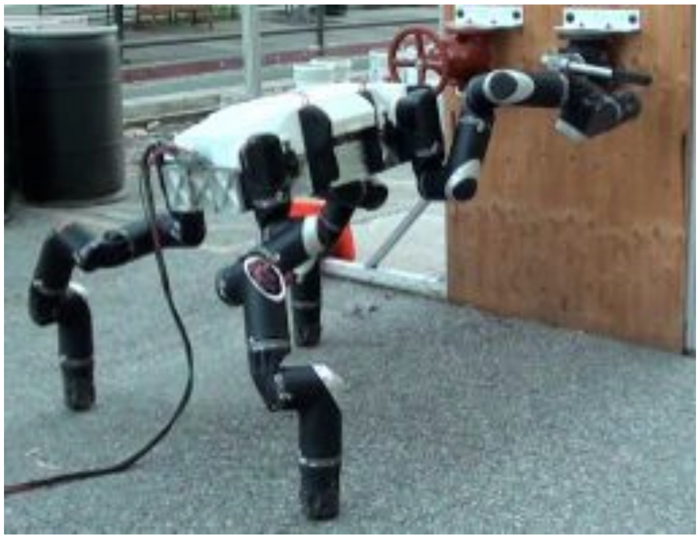


Figure 3.1: RoboSimian turning a valve mounted on a wall. Designed by NASA\JPL-Caltech

This chapter studies the local motions of multi-legged robots with articulated limbs that must support their own weight, and may use all of their degrees of freedom for a given task. An example of such a robot is RoboSimian, shown in Figure 3.1. Each of RoboSimian's four limbs has seven revolute joints, and a 3-fingered hand at the end of each limb. The majority of the tasks that RoboSimian may be charged with require that it maintains frictional contact (without slipping), and stability with respect to gravity using three of its four limbs, while the remaining limb reaches out in the world. This generic task description applies both to manipulation tasks like turning valves as in Figure 3.1, as well as to quasi-static locomotion, where the robot reaches out to take a new step while remaining in static stability as part of a quasi-static gait.

RoboSimian has a total of 28 degrees of freedom, which makes it highly redundant (see Section 2.1 for why this can be challenging). Furthermore, it must remain in contact with the ground at its supporting limbs. As a result, the joint motions required to move the end-effector in the desired direction can be extremely counter-intuitive. To overcome this problem one must have a general and thorough understanding of which motions are possible, and among possible motions, how motions of the various joints in the robot correspond to motions of the Robot's abdomen or body and its manipulating hand(s). In particular, one must formulate the kinematics by assuming the most general form of contact at each supporting limb that allows for models of friction to be applied if necessary, as well as to account for more interesting contacts that can sustain some forces and torques but will slip or break under others.

In order to be able to discuss such local motions for a legged robot, the notion of a Robot's *stance*, and a *hold* must be introduced. As these definitions are crucial but non-technical, they are introduced here:

Definition 1. *A robot **stance** is a combined description of the position and orientation of a legged robot's body, in addition to the joint angles in its supporting limbs, and the properties and normal vectors of the supporting ground contacts. Informally, one may think of a Stance as the 'way a robot is standing'.*

Definition 2. *A **hold** generalizes a contact to include the possibility of forces in the direction of the contact normal; intuitively, one may think of a hold on a climbing wall that can sustain pushing* and *pulling forces, as well as torques.*

3.1.1 Problem Statement

Suppose that a multi-legged robot such as RoboSimian in Figure 3.1 is at a known initial *stance*, with three of its supporting legs making contact with the world at known contacts or *holds*, and the remaining limb free to perform a task or to take a step as part of a gait cycle. Moving towards a thorough understanding of the local kinematics of these situations, consider the following questions:

1. Given a general class of contacts or *holds*, which may sustain pushing and pulling forces, as well as torques, including the possibility of coulomb friction, how are the possible contact velocities and wrenches mapped to possible velocities and wrenches at the robots body or manipulating end-effector?

2. Given a characterization of admissible robot abdomen velocities in terms of contacts, what are the motions of the robot body or that can be achieved by the robot with its joints at their current configuration?

3. Given a full understanding of possible robot motions as a function of admissible contact velocities or forces, and possible local joint motions, what are the possible motions of the manipulating or reaching end effector?

4. What are general conditions under which the robot body or robot end-effector can be moved in any way?

5. What are the cases in which the robot body's motion is restricted, but the end-effector is can still achieve any motion locally?

6. How does the center-of-mass of the robot move as a function of possible robot motions and admissible contact velocities?

These questions motivate the key kinematics and tools that enable the analysis and planning of local motions of mult-limbed robots for achieving a large variety of manipulation tasks. This chapter answers all of these questions in a formal mathematical and kinematic framework.

3.1.2 Contributions

This chapter provides an explicit kinematic formulation to locally describe the adjustment of a legged robot's *stance*. The possibility for each contact having different properties and wrench bases is accounted for directly, using the same framework that is applied to multi-fingered hand kinematics in [58]. Using this formulation, important local properties are defined for a stance (*limberness*), a stance-constrained reach (*local dexterity*). In addition, a property related to the ability of a stance to respond to forces is introduced (*wrench resistance*). These properties provide a direct analogy between dexterous manipulation and legged stance adjustment, and the analogy is formalized in Section 3.3 to include direct parallels between force closure and *wrench resistance*. This chapter also describes and proves certain conditions under which each of these properties hold, including simple cases in which a legged robot is limber, locally dexterous, and wrench-resistant. To allow motions that are stable in the presence of gravity, the *Stance-Constrained Center of Mass Jacobian* is defined and derived in a novel fashion that accounts for all of the links and joints in the robot explicitly, and includes the effects of the supporting holds.

3.1.3 Relation to Prior Work

Multi-limbed standing manipulation has many similarities to dexterous manipulation by multi-fingered robotic hands, and this work takes advantage of that fact. Many authors [8, 27, 33, 45, 96] have previously noted these similarities. One of the contributions of this chapter is to make these analogies explicit. Multi-fingered hand kinematics (velocity and wrench) were first introduced by Salisbury and Roth [75], and soon thereafter elaborated on by Kerr and Roth [39]. Now considered classical, this material is also derived and covered in detail using the same notation and formalism as this thesis in [58].

There has been significant work to model the differential geometry of contact curvature that generalizes simple notions of frictional contact. Much of the modeling effort was made by Montana [57], and his work was extended in [59]. The effects of curvature on quasi-static stability of rigid bodies in the presence of gravity have also been studied by Mason et al. [53]. Although this thesis does not account for and include the effects of curvature, they are straightforward to include in the same fashion as curvature is applied to multi-fingered kinematics.

Quasi-static multi-legged locomotion has been well-studied, the foundation for which was laid in [18, 55] and built upon in many works, for example [56, 96]. The analysis of static equilibrium and quasi-static stability for legged robots with attention to contacts (friction, curvature, etc.) has been approached from both theoretical [53, 65, 66, 73] and computational [8, 66] standpoints. The local center-of-mass kinematics introduced in this Chapter is motivated by the use of prior characterizations of stance equilibria, as one seeks to determine those motions which satisfy manipulation criteria while also attempting to maintain the center of mass within the support region defined by the prior works.

The relationship between the center of mass motion and joint motion is central to the problem of free floating space manipulators (e.g. [67, 68, 95]). In contrast to the space manipulator problem, the center of mass location for the systems studied here is not conserved, and its motion must also take into account the reaction forces generated by the stance limbs.

Manipulation from a legged robot platform has not been explored deeply, but some work has been done for humanoids [25, 26, 43]. Accounting for the position of the center of mass is very important for humanoids, and [94] details the relationship between joint velocities and center of mass motion in relation to notions of dynamic disturbance rejection; a number of other works build on this result. [90] provides methods for dynamic locomotion, and describes the construction of a whole body Jacobian that is applied as part of a hierarchical redundancy-resolution based controller. The present work is novel in that the concepts provided apply to a much larger class of mobile bases (see Chapter 4); moreover, for the case of legged robots, this thesis provides a formal development of the local kinematics and center-of-mass kinematics for legged robots that accounts for arbitrary contacts and mechanism topologies (i.e. number and placement of links and joints). Finally, planning manipulation for legged robots has not been explored in a way that accounts for redundancy explicitly— for example, [43, 78] assume a known goal configuration. In general, such goal configurations cannot be found using analytical inverse kinematics as the degree of redundancy (number of free joint variables) is prohibitively large. The kinematics of this chapter combined with the methods of 4 produce plans without any knowledge of the goal configuration.

Preliminary versions of this work appeared in [89], which describes kinematics and methods for legged stance and reach tasks.

3.2 Structure of the Chapter

Section 3.3 derives the kinematics relevant to stance and reach tasks for legged robots. In Section 3.4, key properties of a stance are introduced, and are related to the analogous properties and kinematics of multi-fingered hands. Finally, the kinematics of the Stance-Constrained Center-of-Mass are defined and derived in Section 3.5. The chapter is summarized in 3.6.

3.3 Stance and Reach Kinematics

Figure 3.2 depicts an n-limbed robot with one limb used as a manipulator, while the other limbs provide a (potentially) quasi-statically stable stance. The goal of this section is to derive the kinematic relationships which relate end-effector motions to mechanism joint motions. Section 3.5 will derive analogous relationships for center of mass motion. While this thesis focuses on the case of a single manipulation limb, the methods can be extended to multiple manipulating limbs.

Let a *world* (fixed) reference frame be denoted by $\mathcal{W}$. Assume that a terrain model is available from which one can derive the location, height, and contact normals associated to each foothold, and this information is contained in a parameter x_0. A user-selected frame attached to the robot's main body, or abdomen, is denoted by $\mathcal{B}$ (see Fig. 3.2). For each limb, the *shoulder* reference frame, S_j, is located at the point where the j^{th} limb attaches to the body, and is *fixed* with respect to $\mathcal{B}$. Without loss of generality, the stance legs are indexed $j = 1, \dots, n-1$, while the reaching limb has index n. The end effector frame at the distal end of the n^{th} limb is denoted by $\mathcal{E}$. For limbs that are positioned

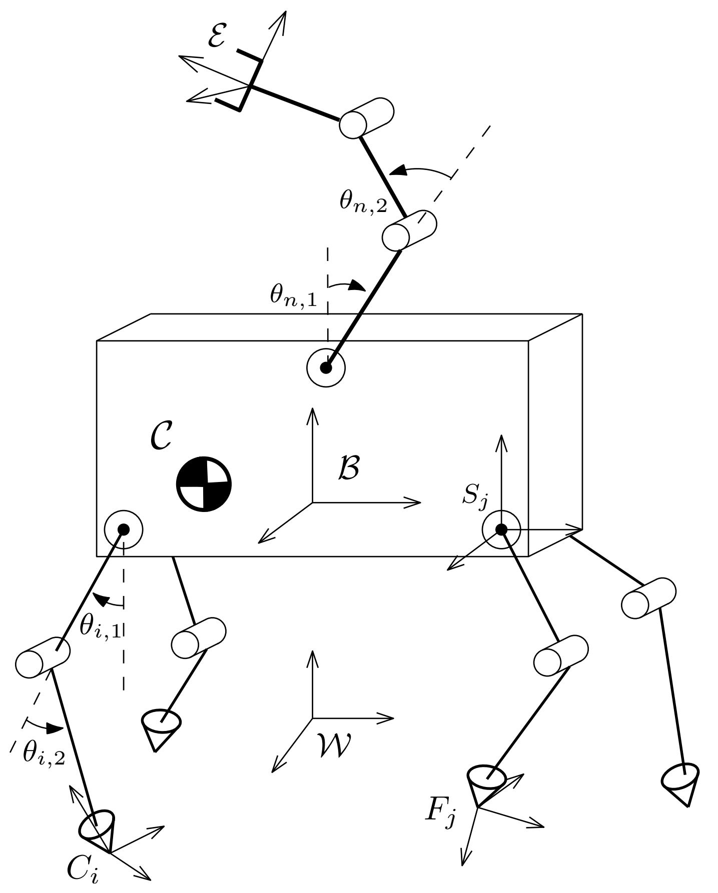


Figure 3.2: Key Reference Frames for Stance and Reach.

on holds, the *foot frame*, denoted F_j, is rigidly attached to the foot body, with origin at the point of ground contact [1] and the $z-$ axis oriented with the normal of the foot. The i^{th} *contact frame*, C_i, is fixed with origin at the contact between the i^{th} limb and the ground, so that the surface-normal forms the $z-$axis. The center of mass frame is denoted by $\mathcal{C}$; it is oriented with the world frame. The joint angles corresponding to the i^{th} limb are given by the tuple $\vec{\theta_i} = (\theta_{i,1}, \ldots, \theta_{i,m_i})^T$, with m_i components, where m_i is the number of joints in the i^{th} limb.

The relationship between joint-motions and end-effector motions is now derived. This derivation uses three key facts.

- End-effector motions arise from either reaching limb motions or body motions, and the net motion is the combination of the two.

- The motion of the body results from motions of the limbs making contact with the world; each supporting limb must produce a motion that complements the motion of the body.

[1]This thesis assumes point contact between the feet and ground, but the methods can be readily extended to other types of contact.

- The motions of the supporting limbs must be restricted by the nature of the contact – the limbs can only move along with the body in directions that forces can be applied [2].

These facts are now made concrete. Let the homogeneous transformation between end-effector frame $\mathcal{E}$ and fixed frame $\mathcal{W}$ be denoted $g_{\mathcal{W}\mathcal{E}} \in SE(3)$. In order to understand how this transformation varies, one may write it as a product of relevant intermediate transformations between parts of the Robot. Using Figure 3.2, $g_{\mathcal{W}\mathcal{E}}$ can be expanded as:

$$g_{\mathcal{W}\mathcal{E}} = g_{\mathcal{W}c_i} g_{c_i f_i} g_{f_i s_i} g_{s_i \mathcal{B}} g_{\mathcal{B}\mathcal{E}} \ . \tag{3.1}$$

The twist coordinates of the end-effector velocity with respect to $\mathcal{W}$ as seen in the end effector frame are given by

$$\widehat{V}^{\mathcal{E}}_{\mathcal{W}\mathcal{E}} = g^{-1}_{\mathcal{W}\mathcal{E}} \dot{g}_{\mathcal{W}\mathcal{E}} \ . \tag{3.2}$$

Expanding Eq. (3.2) using the expression for $g_{\mathcal{W}\mathcal{E}}$ in Eq. (3.1), applying the chain rule, and converting to twist vector form yields:

$$\begin{aligned} V^{\mathcal{E}}_{\mathcal{W}\mathcal{E}} = {} & \mathrm{Ad}_{g_{\mathcal{E}c_i}} V_{\mathcal{W}c_i} + \mathrm{Ad}_{g_{\mathcal{E}f_i}} V_{c_i f_i} + \mathrm{Ad}_{g_{\mathcal{E}s_i}} V_{f_i s_i} \\ & + \mathrm{Ad}_{g_{\mathcal{E}\mathcal{B}}} V_{s_i \mathcal{B}} + V_{\mathcal{B}\mathcal{E}}. \end{aligned} \tag{3.3}$$

The velocities $V_{\mathcal{W}c_i}$ and $V_{s_i\mathcal{B}}$ are zero since they describe motion between frames that are relatively fixed.

Recall from Section 2.1 that $J_n(\theta_n) \triangleq J^{s_n}_{s_n\mathcal{E}}(\theta_n)$, and that in general the spatial velocity given by a spatial Jacobian is transformed to a body velocity as

$$V^t_{st} = \mathrm{Ad}^{-1}_{g_{st}} J^s_{st}(\theta)\dot{\theta}.$$

Using these facts, the velocity of the of the n^{th} manipulating limb $V_{\mathcal{B}\mathcal{E}} \triangleq V^{\mathcal{E}}_{\mathcal{B}\mathcal{E}}$ can be expressed as

$$V_{\mathcal{B}\mathcal{E}} = \cancel{V^{\mathcal{E}}_{\mathcal{B}s_n}} + V^{\mathcal{E}}_{s_n\mathcal{E}} = \mathrm{Ad}^{-1}_{g_{s_n\mathcal{E}}} J_n(\vec{\theta}_n)\dot{\vec{\theta}}_n.$$

Similarly,

$$V_{f_i s_i} = -V^{s_i}_{s_i f_i} = -J_i(\vec{\theta}_i)\ \dot{\vec{\theta}}_i \ .$$

[2]Consider the act of standing *up* – to move ones body up, one must be able to push *down*.

The velocity of the foot frame with respect to the contact frame can be expanded as

$$\begin{aligned} V_{c_i f_i} &= V^{f_i}_{c_i \mathcal{B}} + V^{f_i}_{\mathcal{B} s_i} + V_{s_i f_i} \\ &= \mathrm{Ad}_{g_{f_i \mathcal{B}}} V_{c_i \mathcal{B}} + \mathrm{Ad}^{-1}_{g_{s_i f_i}} J_i(\vec{\theta}_i)\, \dot{\vec{\theta}}_i, \end{aligned} \tag{3.4}$$

where $V^{f_i}_{\mathcal{B}s_i} = 0$ for all i since the shoulder frame is rigidly attached to the body. Substituting these results into Eq. (3.3) yields

$$\begin{aligned} V_{\mathcal{W}\mathcal{E}} = \mathrm{Ad}^{-1}_{g_{f_i \mathcal{E}}} (\mathrm{Ad}^{-1}_{g_{s_i f_i}} J_i(\vec{\theta}_i)\dot{\vec{\theta}}_i + \mathrm{Ad}_{g_{f_i \mathcal{B}}} V_{c_i \mathcal{B}}) \\ - \mathrm{Ad}^{-1}_{g_{s_i \mathcal{E}}} J_i(\vec{\theta}_i)\dot{\vec{\theta}}_i + \mathrm{Ad}^{-1}_{g_{s_n \mathcal{E}}} J_n(\vec{\theta}_n)\dot{\vec{\theta}}_n. \end{aligned} \tag{3.5}$$

Combining coordinate transformations and noting that the first and third terms in (3.5) cancel each other out, one obtains:

$$V_{\mathcal{W}\mathcal{E}} = \mathrm{Ad}_{g_{\mathcal{E}\mathcal{B}}} V_{c_i \mathcal{B}} + \mathrm{Ad}^{-1}_{g_{s_n \mathcal{E}}} J(\vec{\theta}_n)\dot{\vec{\theta}}_n \ . \tag{3.6}$$

At each foothold, assume a *contact constraint*, which implies that the foot cannot move in directions along which forces can be applied and supported by the contact:

$$B^T_{c_i} V_{f_i c_i} = 0, \tag{3.7}$$

where B_{c_i} is the *wrench basis* at the i^{th} foot[3].

The velocity of the i^{th} foot-frame with respect to the i^{th} contact frame is defined by

$$\widehat{V}_{f_i c_i} = g^{-1}_{c_i f_i} \dot{g}_{c_i f_i}, \tag{3.8}$$

where the transform $g_{c_i f_i}$ can be expressed as

$$g_{c_i f_i} = g_{f_i s_i} g_{s_i \mathcal{B}} g_{\mathcal{B}\mathcal{W}} g_{\mathcal{W} c_i}.$$

Expanding out the right-hand side of the velocity (3.8), converting to twist vector form, and

[3]The columns of the wrench basis matrix form a basis for all wrenches that can be applied at a contact, as expressed in the contact frame [58].This is the same contact constraint that is applied in the context of multi-fingered hand kinematics. A wrench basis can model frictionless point contact with Coulomb friction, soft contact, and many other types of ground contact models

dropping velocities between relatively fixed frames results in:

$$\begin{aligned} V_{f_i c_i} &= \mathrm{Ad}^{-1}_{g_{s_i c_i}} V_{f_i s_i} + \mathrm{Ad}^{-1}_{g_{\mathcal{B} c_i}} V_{s_i \mathcal{B}} + \mathrm{Ad}^{-1}_{g_{\mathcal{W} c_i}} V_{\mathcal{B}\mathcal{W}} + V_{\mathcal{W} c_i} \\ &= -\mathrm{Ad}^{-1}_{g_{s_i c_i}} J(\vec{\theta}_i)\dot{\vec{\theta}}_i + \mathrm{Ad}^{-1}_{g_{\mathcal{W} c_i}} V_{\mathcal{B}\mathcal{W}}. \end{aligned} \tag{3.9}$$

Substituting Eq.s (3.4) and (3.9) into Eq. (3.7) and equating results yields:

$$\begin{aligned} B^T_{c_i} \mathrm{Ad}^{-1}_{g_{s_i c_i}} J_i(\vec{\theta}_i)\dot{\vec{\theta}}_i &= B^T_{c_i} \mathrm{Ad}^{-1}_{g_{\mathcal{W} c_i}} V_{\mathcal{B}\mathcal{W}} \\ &= -B^T_{c_i} \mathrm{Ad}^{-1}_{g_{\mathcal{W} c_i}} \mathrm{Ad}_{g_{\mathcal{W}\mathcal{B}}} V_{\mathcal{W}\mathcal{B}} \\ &= -B^T_{c_i} \mathrm{Ad}^{-1}_{g_{\mathcal{B} c_i}} V_{\mathcal{W}\mathcal{B}}\ . \end{aligned}$$

Next, define the *Stance Jacobian* matrix (which is analogous to the *hand Jacobian* [58] in multi-fingered grasping) as

$$J_{\mathcal{S}}(x_0, \vec{\theta}) = \begin{bmatrix} B^T_{c_1} \mathrm{Ad}^{-1}_{g_{S_1 c_1}} J_1(\vec{\theta}_1) & & 0 \\ & \ddots & \\ 0 & & B^T_{c_{n-1}} \mathrm{Ad}^{-1}_{g_{S_{n-1} c_{n-1}}} J_{n-1}(\vec{\theta}_{n-1}) \end{bmatrix},$$

where $\vec{\theta} = (\vec{\theta}_1, \ldots, \vec{\theta}_{n-1})$, and x_0 describes the location of the body frame relative to the world frame (which implicitly defines the contact points and normals via the terrain model). Now Eq. (3.7) becomes

$$J_{\mathcal{S}}(x_0, \vec{\theta})\dot{\vec{\theta}} = -\begin{bmatrix} B^T_{c_1} \mathrm{Ad}^{-1}_{g_{\mathcal{B} c_1}} \\ \vdots \\ B^T_{c_{n-1}} \mathrm{Ad}^{-1}_{g_{\mathcal{B} c_{n-1}}} \end{bmatrix} V_{\mathcal{W}\mathcal{B}}.$$

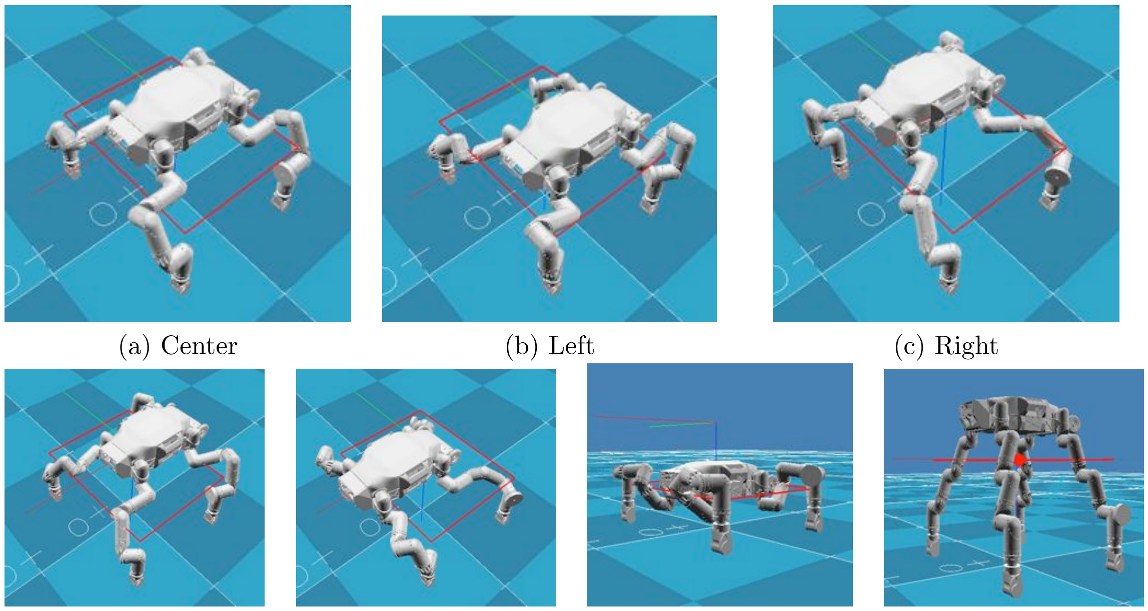
Define the *stance map* as

$$S = -\begin{bmatrix} \mathrm{Ad}^T_{g^{-1}_{\mathcal{B} c_1}} B_{c_1} & \cdots & \mathrm{Ad}^T_{g^{-1}_{\mathcal{B} c_{n-1}}} B_{c_{n-1}} \end{bmatrix}.$$

The stance map transforms foot contact forces to wrenches on the body –it is analogous to the grasp map [58] in multi-fingered robotic manipulation. Using these definitions yields a more descriptive version of Eq. (3.7):

$$J_{\mathcal{S}}(x_0, \vec{\theta})\dot{\vec{\theta}} = S^T V_{\mathcal{W}\mathcal{B}}\ . \tag{3.10}$$

Eq. (3.10) describes the relationship between the motions of leg joints and the motion of the body frame, assuming that the footholds are maintained. Reconsider the velocity of the end-effector.



(a) Center (b) Left (c) Right

(d) Back (e) Front (f) Low (g) High

Figure 3.3: Demonstration of Stance Motions using the Stance Jacobian. Motions to various extreme body positions as given in sub-captions. The motions are achieved using the methods of Chapter 4, and Algorithm 4.1. The nominal centered position is shown in (a). The red rectangle is the support region.

Applying the fact that contacts and the world frame are fixed relative to each other to Eq. (3.6) and rearranging yields the following important relationship:

$$Ad_{g_{\mathcal{B}\mathcal{E}}} V_{\mathcal{W}\mathcal{E}} = V_{c_i\mathcal{B}} + \mathrm{Ad}_{g_{\mathcal{B}s_n}} J_n(\vec{\theta}_n)\dot{\vec{\theta}}_n. \tag{3.11}$$

Let the *stance-constrained Reach Jacobian*, or simply *Reach Jacobian* be defined as follows:

$$J_{\mathcal{R}}(\Theta, x_0) = \Big[J_{\mathcal{S}}(x_0, \vec{\theta}) \quad S^T \mathrm{Ad}_{g_{\mathcal{B}s_n}} J_n(\vec{\theta}_n) \Big], \tag{3.12}$$

where $\Theta = (\vec{\theta}_1, \ldots, \vec{\theta}_n)$ is the vector of all mechanism joint angles. By rewriting Eq. (3.11) in terms of contact velocities (left multiply both sides by S^T), and recalling that abdomen and contact velocities are related by (3.10), the following new relationship is obtained:

$$J_{\mathcal{R}}(\Theta, x_0)\dot{\Theta} = S^T V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}. \tag{3.13}$$

Call (3.13) the *stance constrained Reach Constraint*, or *Reach Constraint*, as it describes all of the ways that the robot can reach for a certain goal by coordinating the motions of its legs and its manipulator limb (assuming that the footholds are maintained).

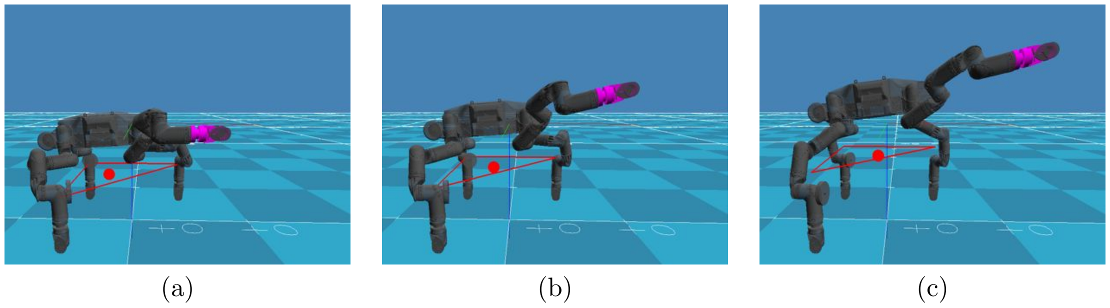


Figure 3.4: Robosimian reaching using Reach Jacobian. The red sphere specifies the x-y location of the center of mass, and the support region is shown as a red triangle.

3.4 Properties of a Stance

This section introduces some *local*, or *instantaneous*, notions of the properties of the combined stance/reach system analyzed in the last section. Most of these properties are analogous to those of multi-fingered robotic grasps, with modest extensions required to handle the presence of the reaching limb.

Definition 3. *A limbed robot stance is* ***limber*** *at a configuration* $(\vec{\theta}, x_0)$ *when for any motion of the body,* $V^{\mathcal{B}}_{\mathcal{WB}}$*, there exists* $\dot{\vec{\theta}}$ *satisfying Eq. (3.10).*

Definition 4. *A limbed robot stance is* ***locally dexterous*** *with respect to a free limb at a configuration* (Θ, x_0) *when for any motion of the end effector,* $V^{\mathcal{B}}_{\mathcal{WE}}$*, there exists* $\dot{\Theta}$ *satisfying (3.13).*

A robot is limber if it can locally move its body in all directions via leg joint motions while maintaining its footholds, and it is locally dexterous if the reaching end-effector can instantaneously move in any direction by allowed mechanism joint motions. The following propositions result directly from Definitions 3 and 4, as well as Eq.s (3.10) and (3.13).

Proposition 1. *Suppose that S is onto. A stance is limber at a configuration* (Θ, x_0) *if and only if*

$$Range(S^T) \subseteq Range(J_{\mathcal{S}}(\vec{\theta}, x_0)).$$

Proposition 2. *Suppose that S is onto. A stance is locally dexterous with respect to a limb at a configuration (Θ, x_0) if and only if*

$$Range(S^T) \subseteq Range(J_{\mathcal{R}}(\Theta, x_0)). \tag{3.14}$$

Furthermore, for a given configuration, a robot is locally dexterous if any of the following hold:

i) it is limber.

ii) $J_n(\theta_n)$ is onto the pre-image of Range(S^T) (or equivalently, onto the complement of Null(S^T)).

This fact is a consequence of Eq. (3.13). The most interesting situations occur when

- The robot is locally dexterous, but not limber, and the free limb's Jacobian matrix is not onto the compliment of Null(S^T). At these configurations, both the body and the reaching arm must work in concert to realize arbitrary end-effector motions.
- The robot is limber and the free limb Jacobian matrix is onto the pre-image of the Range(S^T). In this case, there are many ways of producing the desired end effector velocity (by combinations of body and arm motions). The specific solution might depend on other goals, and this possibility is explored in Chapter 4

Other properties which are standard in multi-fingered grasping also extend to this situation. Let FC_{c_i} denote the friction cone associated with each contact c_i, and let FC denote the totality of friction cone constraints: $FC = FC_{c_1} \times ... \times FC_{c_{n-1}}$. Assuming that the robot is standing on an immovable object or solid ground, the net wrench F on the body due to forces applied at the ground contacts is:

$$F = S\vec{f} \qquad \vec{f} \in FC,$$

where $\vec{f} = (f_1, \ldots, f_{n-1})^T$ are the contact forces at the footholds.

Definition 5. *A stance is* ***wrench-resistant*** *if it can counteract any external wrench on the abdomen F_e, i.e. there exists $f \in FC$ such that*

$$Sf = -F_e.$$

Proposition 3. *A stance is wrench-resistant if and only if S restricted to FC is full rank, i.e.*

$$S(FC) = \mathbb{R}^6 \ .$$

Note that this proposition assumes that the reaching limb is free, and does not make contact. If the reaching limb can apply wrenches, this condition is sufficient but not necessary. It can be extended to include the case of an end-effector wrench, one appending the contact-wrench basis of the manipulating limb to the stance map. This case and number of additional cases involving contact at the manipulating limb as well as friction are handled in Chapter 5.

3.5 The Center of Mass Stance Jacobian

It is well known that quasi-static stability of a given stance depends upon the location of the mechanism's center of mass relative to the *support region* generated by the stance's footholds [8, 55, 65, 66, 73]. Therefore, it is crucial to find a relationship between mechanism joint motions and center-of-mass motion. This relationship can then be used during the task planning process to maintain or enhance a suitable stability margin while realizing manipulation goals.

The center of mass of a system of N rigid bodies, $p_{\mathcal{W}\mathrm{c}} \in \mathbb{R}^3$, is given by

$$p_{\mathcal{W}\mathrm{c}} = \frac{1}{M}\sum_{i=1}^{N} \rho_i m_i,$$

where m_i and ρ_i are the mass and position of the i^{th} object, respectively, and M is the total system mass: $M = \sum_{i=1}^{N} m_i$. The system's center of mass position with respect to the world, in homogeneous coordinates, is

$$p_{\mathcal{W}\mathrm{c}} = g_{\mathcal{W}\mathrm{c}} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} ,$$

where $g_{\mathcal{W}\mathrm{c}}$ is the displacement of a reference frame located at the system's center of mass. This reference frame's position is at the center of mass, and its orientation is aligned with the world frame. Hereafter, the center of mass will be described in terms of $g_{\mathcal{W}\mathrm{c}}$, with the understanding that only the translational component of $g_{\mathcal{W}\mathrm{c}}$ is relevant to any analysis.

Let $m_{\mathcal{B}}$ be the mass of the body, located at the origin of frame $\mathcal{B}$. The mass of the j^{th} link on the i^{th} limb is denoted $m_{i,j}$, and $g_{\mathcal{W}(i,j)}$ maps positions in homogeneous coordinates from the $(i,j)^{th}$ link frame (whose origin is located at the link's center of mass) to $\mathcal{W}$

$$g_{\mathcal{W}\mathcal{C}} = \frac{m_{\mathcal{B}}}{M} g_{\mathcal{W}\mathcal{B}} + \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{n_i} m_{i,j} g_{\mathcal{W}(i,j)},$$

where M is the total robot mass. Using this notation, the velocity of the center of mass frame can be expressed as:

$$V_{\mathcal{W}\mathcal{C}}^{\mathcal{C}} = \frac{m_{\mathcal{B}}}{M} \mathrm{Ad}_{g_{c\mathcal{B}}} V_{\mathcal{W}\mathcal{B}}^{\mathcal{B}} + \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{n_i} m_{i,j} \mathrm{Ad}_{g_{\mathcal{C}(i,j)}} V_{\mathcal{W}(i,j)}^{(i,j)}, \tag{3.15}$$

where $\mathcal{C}$ denotes 'center-of-mass'. Define the *link Jacobian* as the matrix which maps the velocities of the first j joints in the i^{th} limb to the velocity of the j^{th} link in that limb, as described in the $\mathcal{B}$ reference frame.

$$J_{i,j}(\theta_{i,1}, \dots, \theta_{i,j})$$

With this notation,

$$\begin{aligned} V_{\mathcal{W}(i,j)}^{\mathcal{W}} &= V_{\mathcal{W}\mathcal{B}}^{\mathcal{W}} + \mathrm{Ad}_{g_{\mathcal{W}\mathcal{B}}} V_{\mathcal{B}(i,j)}^{\mathcal{B}} \\ &= V_{\mathcal{W}\mathcal{B}}^{\mathcal{W}} + \mathrm{Ad}_{g_{\mathcal{W}s_i}} J_{i,j}(\theta_{i,1}, \dots, \theta_{i,j}) \dot{\theta}_{i\to j}, \end{aligned}$$

where

$$\theta_{i\to j} = (\theta_{i,1}, \dots, \theta_{i,j})$$

is shorthand notation to denote the fragment of the i^{th} limb's joint velocity vector containing the first j joint velocities. Introducing the link Jacobian into (3.15), transforming (3.15) to the $\mathcal{B}$ frame, and then multiplying both sides by the transpose of the stance map (so that the stance Jacobian can be used), one obtains:

$$S^T V_{\mathcal{W}\mathcal{C}}^{\mathcal{B}} = J_{\mathcal{S}}(\theta)\dot{\theta} + \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{n_i} m_{i,j} S^T \mathrm{Ad}_{g_{\mathcal{B}s_i}} J_{i,j}(\theta_{i\to j}) \dot{\theta}_{i\to j}. \tag{3.16}$$

Recall that for a serial chain manipulator, the spatial Jacobian matrix takes the form [58]:

$$J_{st}^{s} = \begin{bmatrix} \xi_1 & \dots & \xi_n' \end{bmatrix},$$

where

$$\xi_i' = \mathrm{Ad}_{\left(e^{\hat{\xi}_1 \theta_1} \dots e^{\hat{\xi}_{i-1} \theta_{i-1}}\right)} \xi_i.$$

$$J_{\mathcal{C}}(\theta) = \begin{bmatrix} B_{c_1}^T \mathrm{Ad}_{g_{c_1 S_1}} \tilde{J}_1(\theta_1) & B_{c_1}^T \mathrm{Ad}_{g_{c_1 S_2}} \bar{J}_2(\theta_2) & \dots & B_{c_1}^T \mathrm{Ad}_{g_{c_1 S_{n-1}}} \bar{J}_{n-1}(\theta_{n-1}) & \dots & B_{c_1}^T \mathrm{Ad}_{g_{c_1 s_n}} \bar{J}_n(\theta_n) \\ B_{c_2}^T \mathrm{Ad}_{g_{c_2 S_1}} \bar{J}_1(\theta_1) & B_{c_2}^T \mathrm{Ad}_{g_{c_2 S_2}} \tilde{J}_2(\theta_2) & & & & \\ \vdots & & \ddots & & & \vdots \\ & \dots & & B_{c_M}^T \mathrm{Ad}_{g_{c_{n-1} S_{n-1}}} \tilde{J}_{n-1}(\theta_{n-1}) & \dots & B_{c_{n-1}}^T \mathrm{Ad}_{g_{c_{n-1} s_n}} \bar{J}_n(\theta_n) \end{bmatrix} \tag{3.17}$$

The vector ξ_i is the unit twist associated with the i^{th} joint at the manipulator's zero configuration (the configuration of the robot when all joint angles are zero). Define a *mass weighted Jacobian* matrix as follows:

$$\bar{J}_k = \\ \begin{bmatrix} (\sum_{j=1}^{n_k} \frac{m_{k,j}}{M})\xi_{1,k} & \dots & (\sum_{j=i}^{n_k} \frac{m_{k,j}}{M})\xi'_{i,k} & \dots & (\frac{m_{k,n_k}}{M})\xi'_{k,n_k} \end{bmatrix},$$

where n_k is the number of joints in the k^{th} limb, and $\xi_{i,j}$ is the unit twist associated with the j^{th} joint of the i^{th} limb in the robot's zero configuration. Using these definitions, equation (3.16) becomes

$$S^T V_{\mathcal{W}\mathcal{C}}^{\mathcal{B}} = J_{\mathcal{S}}(\theta)\dot{\theta} + S^T \sum_{i=1}^{n} \mathrm{Ad}_{g_{\mathcal{B}s_i}} \bar{J}_i(\theta_i)\dot{\theta}_i$$

which can be written in the following compact form

$$S^T V_{\mathcal{W}\mathcal{C}}^{\mathcal{B}} = J_{\mathcal{C}}(\theta)\dot{\theta}$$

by introducing the *Stance-Constrained Center of Mass Jacobian*, Eq. (3.17), where

$$\tilde{J}_k = \begin{bmatrix} (\sum_{j=1}^{n_k} \frac{m_{l,j}+M}{M})\xi_{k,1} & \dots & (\frac{m_{k,n_k}+M}{M})\xi'_{k,n_k} \end{bmatrix}.$$

The explicit and structured formulation of the center of mass Jacobian matrix, in terms of joint twists and general contact, is novel. Analytically, this structure is amenable to analysis for different configurations and contact models. Practically, this structure is easy to implement for planning and control efforts that require the center of mass Jacobian (it removes the need for case-by-case differentiation of the forward kinematics).

3.6 Summary

This Chapter derives the kinematics that relate motions of a legged robots joints to key features of robot motion including motion of the body, motion of a reaching limb, and motion of the center of mass. These derivations incorporate the effect of contact, and are general with respect to mechanism topology and terrain.

The particular form of the kinematic relationships derived shows a direct analogy to the kinematics of dexterous manipulation with hands, and a number of parallel facts are stated and proved.

Chapter 4

Kinematic Local Planning

4.1 Introduction and Background

This chapter uses the relationships developed in the previous chapters to develop *local* planning methods for mobile manipulation. Intuitively speaking, the goal of these local planning methods is to move the end-effector so as to satisfy task goals and constraints, while also simultaneously moving the center of mass so as to enhance, maintain, or gracefully degrade stance stability. There are many different conceivable types of problems and analytical approaches that one may take to realize these intuitively obvious goals.

4.1.1 Problem Statement

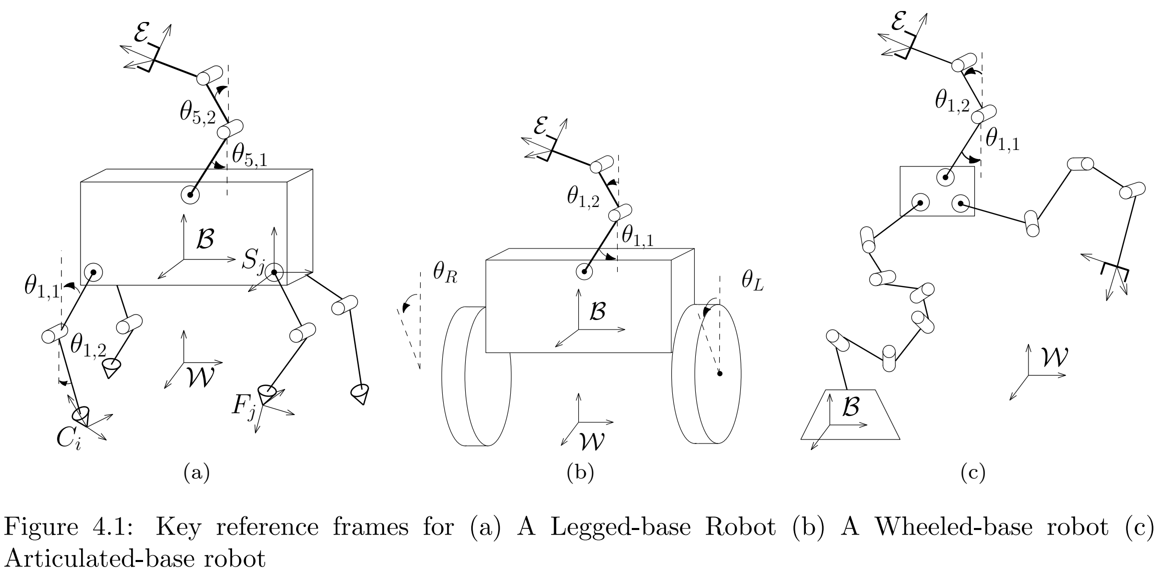


Figure 4.1: Key reference frames for (a) A Legged-base Robot (b) A Wheeled-base robot (c) Articulated-base robot

Suppose that one of the mobile robots shown in Figure 4.1 must complete a manipulation task that can be described by a tool frame location, or a sequence of such locations.

1. What arm configurations satisfy the manipulation constraints?
2. How are base and limb motions apportioned to achieve the goal?
3. How does one guard against vehicle tip-over — can the limbs be moved in such a way as to keep the system center of mass over a safe region of support? Can the robot move so as to *improve* stability with respect to gravitational forces?
4. How are natural task constraints, such as avoiding mechanism self-collision, obstacle avoidance, preferred camera gaze direction, and joint movement limits incorporated?

This is a *generalized inverse kinematic problem* where the end-effector(s) are placed at given locations, while incorporating numerous constraints as well as equilibrium with respect to gravitational forces, and optimality criteria are used to resolve ambiguities in the case of multiple possible solutions. The optimality criteria also endow the solution with desirable properties. Since the analysis in this paper is limited to quasi-static motions, the key kinematic variables governing arm motions and center of mass stability can be formulated in terms of appropriate Jacobian matrices (see Chapter 3, 4.3).

4.1.2 Related Work

Manipulation from a legged robot platform has not been explored deeply, but some work has been done for humanoids ([25,26,43]). Accounting for the position of the center of mass is very important for humanoids, and Sugihara et al. [94] detail the relationship between joint velocities and center of mass motion in relation to notions of dynamic disturbance rejection; a number of other works build on this result. Shkolnik et al. [90] provide methods for dynamic locomotion, and describe the construction of a whole body Jacobian that is applied as part of a hierarchical redundancy-resolution based controller. The present work is novel in that the concepts provided apply to a much larger class of mobile bases; moreover, for the case of legged robots, this thesis provides a formal development of the local kinematics and center-of-mass kinematics for legged robots that accounts for arbitrary contacts and mechanism topologies (i.e. number and placement of links and joints). Finally, motion planning (which addresses the problem of finding paths between *known* robot configurations) for legged robots has not been explored in a way that accounts for redundancy — for example, [43, 78] assume a known goal configuration (there is a large body of work for planning on constraint manifolds when the goal configuration is known, e.g. [4,47,99]) – this assumption does not apply in the case of high DOF humanoids without inverse kinematics in closed-form. In general, such goal configurations cannot be found using analytical inverse kinematics as the degree of redundancy (number of free joint variables) is prohibitively large. Local-optimization based approaches like the one introduced in this paper do not have this shortcoming, and can produce 'plans' without a knowledge of the goal configuration or reference trajectory.

Because systems of the type seen in Figures 1.3(a,b,c) are kinematically redundant, the solution proposed in this thesis is intellectually related to the classical methods of redundancy resolution in fixed based redundant manipulator arms [60, 91]. However, instead of using a classical Jacobian pseudo-inverse type of solution, the problem is formulated as a convex optimization problem, specifically a constrained Quadratic Programming problem. Like the task-priority method of redundancy resolution, [60], the formulation allows for multiple task priorities to be encoded as constraints or optimality criteria. However, unlike Jacobian pseudo-inverse methods, the QP formulation in this thesis readily incorporates hard constraints and multiple optimality criteria, has better performance near singularities, and in practice tends to avoid awkward solutions for large kinematic chains [11]. Its real-time performance makes the approach valuable, as it can be used at fast rates for feedback control.

This work is not the first to propose the use of Convex Optimization or Quadratic Programming techniques for solving inverse kinematics problems, for local motion planning of highly articulated mechanisms, or whole body manipulation planning. Zhang et al. [106, 107] used quadratic programming to solve kinematically redundant manipulator redundancy resolution problems. [34, 78] show the use of QPs to incorporate fast collision avoidance calculations as part of humanoid whole body motion planning. Whereas the focus of these works is a complete analysis and synthesis of local collision geometries, our focus is on analysis of a local kinematics and optimization framework that is explicit and general, and includes many goals, constraints, and gravitational stability in a unified fashion.

There has been significant work to address inverse kinematics for humanoids with a *strict hierarchy of tasks.* This work was initiated by Siciliano and Slotine [92], who generalize the Task-Priority framework [60] to include a number of strictly ordered tasks and address the problem at the joint-velocity level. A large body of work has been motivated by the problem introduced in this paper: Baerlocher and Boulic [2] extend [92] with an efficient solution architecture for kinematic trees, and Mansard et al. [52] describe a Software Framework to identify an appropriate task order. None of these works handle inequality constraints, which are fundamental to enforcing stability in the presence of contact, as well as avoiding collision.

Kanoun et al. [35] describe an efficient numerical technique and claim to introduce tasks defined by inequalities (this technique is applied in conjunction with virtual planar linkages for footstep planning in [37]) and this is improved upon by Escande et al. [17], who provide a different and more efficient solver for the same framework, and Saab et al. [74] add dynamic constraints towards use in controllers. These works do not describe how multiple goals can be handled at a given priority level, and their approach only allows motion biasing or damping at the lowest priority level, which is problematic because it means that ill-conditioned task jacobians at intermediate priority levels will lead to instabilities. Crucially, all these works handle inequalities using a slack variable, which means that at a given priority level, the solution will violate both inequality and equality portions of the task whenever the equality task is infeasible when restricted to the feasible set of the inequality task. We state and prove this fact formally in Appendix A. In contrast, the problem we formulate will always produce solutions that satisfy the inequality constraints, whenever they are independently feasible, regardless of any goals. The optimization problem formulate in this thesis is very different than the class addressed by these works, as it allows arbitrarily many tasks to be addressed simultaneously rather than in order, and the problem defined is mathematically far more general than the 'task' described in [35] at a given priority level. Additionally, none of these papers provide any explicit kinematic relationships or analysis thereof; all of them assume that velocity kinematics are obtained by differentiating task coordinates. This has the disadvantage of coordinate singularities that are guaranteed to arise on the motion group $SE(3)$ and subgroups thereof. In comparison, the kinematics in this thesis are formulated explicitly on $se(3)$ — the twist-velocity kinematics that are introduced do not suffer from coordinate singularities, and new tasks are very easily defined and do not require manually differentiating an additional task function. None of these papers address the feasibility of a combination of tasks, or provide existence or uniqueness results for a given robot configuration. Finally, the above papers do not address the kinematics of contact (some of them provide examples using humanoids, but they simply ensure that the feet maintain flat, rigid attachment with the ground – this is non-physical, as it means arbitrary forces can be sustained) in any detail; we explicitly and generally take contact into account in describing robot motion kinematically, and the effect of contact is also addressed in terms of local feasibility.

Another body of work for achieving a hierarchy of tasks considers the problem in terms of control rather than kinematics or trajectory generation. Sentis and Khatib [81] transform the robot dynamics into the coordinates of a task, and use linear control design in the task space, along the same lines as [42]. A hierarchy of tasks is established by ensuring that tasks with lower priority are addressed in the null-space of those with higher priority. In [83], the authors extend this methodology and describe the way in which contact(s) and center of mass motion is handled. They assume that contacts are rigidly fixed in 6 degrees of freedom, and that there are 6 degrees of freedom between the robot's body and each contact. These assumption are very strong, and do not apply to many circumstances (e.g. RoboSimian). This body of work does not include any experimental or theoretical verification; others (see [62]) have suggested that this class of methods is limited to work in simulation, or on robots expressly designed for torque control; it has been shown that these methods are extremely dependent on accurate robot system identification, and suffer in the presence of modeling errors (similar work on wheeled robots [84] has the same drawback). Such techniques are hopeless on robots like RoboSimian, whose joints are nearly impossible to characterize accurately [1] This body of work does not handle inequality constraints, which makes it impossible to correctly impose collision, contact, and stability constraints. Finally, these works have no discussion or analysis of feasibility of their methods, or an explicit discussion of kinematic relationships. In contrast,this thesis handles the effect of contact explicitly and in total generality, and a strict linear hierarchy between tasks is not imposed, and inequality constraints are incorporated.

There has been work to combine the control dogma of [42] and the more general hierarchical optimization principles of [92]. Righetti et al. [72] describe an approach to select contact forces and torques for humanoids with dynamically precomputed configuration trajectories (it requires knowledge of the full goal configuration, as well as a dynamic path to reach it). This work assumes that contact forces are redundant (there are internal or squeeze forces available), and the contact model is given as an implicit constraint in no detail – this thesis does not make any assumptions on contact, and its effect is described explicitly, and there is no requirement of precomputed configuration-trajectories. De lasa et al. [14] combine prioritized control and kinematics into a single framework, to track pre-computed dynamic trajectories and minimize functions of torque and acceleration. Hutter et al. [31] take a similar approach and include contact forces as a variable. None of these works include inequality constraints, which precludes them from handling contact, static equilibrium and collisions correctly. In contrast to our work, these works do not analyze local feasibility, and they do not model contact or robot kinematics explicitly or generally. Whereas these works require a strict hierarchy of tasks, wherein even the equations of motion have a defined priority, no such assumptions

[1]This robot was designed to be position controlled – its joints are very slow, stiff and have large inertia. They are highly geared, so external load has no bearing on the dynamics of the motor. Friction and other non-linear effects are very regime and load dependent, and cannot be characterized with simple models.

are made in this thesis, and tasks defined by equality and inequality as well as goals with varying weights can be specified simultaneously. The techniques of this chapter can be used both to obtain robot trajectories given only goal end-effector pose(s) and an initial configuration, and are efficient enough to be used in a feedback loop to track planned trajectories in real time.

Dynamic effects are not considered in this thesis. However, many (e.g. [44,93,94]) have obtained local controllers for linearized dynamic models of humanoids (inverted pendulums) from simple convex QPs for balancing and footstep placement. [15] extends the work of [14] for robot torque control to include inequality constraints to account for friction, joint limits, and torque limits. This work uses an approximate model of contact, and fundamental constraints on motion (like contact constraints and robot-dynamics) are not strictly enforced within their framework. All of these works require precomputed dynamic goal-trajectories of the configuration to be pre-specified, and their main application is tracking rather than generating plans or trajectories.

4.1.3 Contributions

A novel *balanced priority* local planning problem is formulated that is naturally expressed as a constrained minimization, and is very general with respect to goals and their relative weights, as well as constraints. In contrast to hierarchical least-squares approaches, this formulation, and its generalization as a inequality constrained quadratic program,

- allow. for multiple goals to addressed *simultaneously* rather than in strict order.
- is guaranteed to satisfy any inequality constraints whenever they are feasible (all prior works on hierarchical inverse-kinematics fail in this respect, and all prior work involving the operational space ideas introduced by [42] does not handle inequality constraints at all).

This novel formulation facilitates a characterization of the geometric nature and conditions on the existence and uniqueness of the solutions to this problem in terms of the robot configuration, and provide *explicit* descriptions of how a large class of geometric tasks and constraints can be included. Although the focus of this work is not on algebraic and numerical solution techniques, this class optimization problems can be solved very efficiently using generic off-the-shelf solvers, which are fast enough for use in planning trajectories as well as in a feedback loop at high frequency. The iterative use of this optimization problem extends and generalizes the classical redundancy resolution solutions of fixed base mechanisms to a wide variety of mobile bases in ways that improves performance and tunability.

A number of physical constraints like collision avoidance require setting of parameters that specify how aggressively safety should be enforced, or how conservatively (i.e. how large a safety factor is applied) stability is imposed. Linear programs that can be used to generate optimal or feasible inequality constraints are introduced, so that these parameters can be obtained automatically and very efficiently. This means that at any given robot configuration, these linear programs can generate constraints that guarantee a safe feasible set, or mathematical certify that no such constraints exist, and that the configuration is altogether unsafe.

We provide both simulated and experimental examples on physical robots, including the 28-DOF Quadruped *RoboSimian* (see Figure 1.2 a) and the 23-DOF *Surrogate* (see Figure 1.2 b) tracked-base robot.

4.1.4 Structure of the Chapter

Section 4.2 introduces, roughly in order of increasing complexity, a few basic local task planning problems and their related solutions. These solutions would be useful components of a local planning toolkit for stance-constrained manipulation. Section 4.3 derives stance reach and center-of-mass kinematics for other forms of mobility. Section 4.4 reformulates the results of this section into a generalized quadratic programing formulation, so that a large class of mobile manipulation tasks can be handled for any mobile base and mechanism topology. The application of these methods is illustrated in Section 4.8. The chapter is summarized in 4.9.

4.2 Local Planning for Legged Robots

4.2.1 Problem #1: The Minimum Norm Solution

This first solution, which ignores issues of stance equilibrium or stability, primarily establishes results which will be useful below. Given a stance for the first $(n-1)$ limbs, and a desired end-effector velocity, $V_{\mathcal{WE}}$, for the n^{th} manipulating limb, the goal is to find the minimum norm joint velocities that exactly move the end-effector at this desired velocity. This problem is exactly analogous to the classical minimum norm redundancy resolution solution, except that the different kinematic constraint between the end-effector and the robot's joints yields a solution with slightly different algebraic form. Formally, ones seeks the solution to the problem

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\dot{\Theta}^T\dot{\Theta} \\ &\text{subject to} && S^T\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} = J_{\mathcal{R}}(x_0,\Theta)\dot{\Theta}. \end{aligned} \tag{4.1}$$

The solution is

$$\dot{\Theta}^* = J^+_{\mathcal{R}} S^T \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}$$

where $J^+_{\mathcal{R}}$ is the Moore-Penrose pseudo-inverse, and the arguments of $J_{\mathcal{R}}$ are dropped for clarity. Note that the set of sub-optimal but *feasible* solutions is parametrized by

$$\dot{\Theta}^* = J^+_{\mathcal{R}} S^T \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} + (I - J^+_{\mathcal{R}} J_{\mathcal{R}})z$$

with z an arbitrary vector having the same dimension as $\dot{\Theta}$.

4.2.2 Problem #2: The Task Priority Solution

This problem is motivated by the classical Task Priority solution for redundant robot manipulators [60]. The goal is to require the manipulator limb's end-effector to track a desired velocity, and as a second priority, have the center of mass move as closely as possible in a desired direction as specified by $\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}$ — one that presumably maintains or enhances stance stability. This word problem is equivalent to:

$$\begin{aligned} &\text{Minimize} && \frac{1}{2}\dot{\Theta}^T\dot{\Theta} + \|S^T\tilde{V}^{\mathcal{B}}_{\mathfrak{C}} - J_{\mathfrak{C}}(x_0,\Theta)\dot{\Theta}\|_2^2 \\ &\text{subject to} && S^T\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} = J_{\mathcal{R}}(x_0,\Theta)\dot{\Theta}\ . \end{aligned} \tag{4.2}$$

The end-effector goal is the first priority or 'task', while maintaining quasi-static stability is a lower priority. Adapting the classical redundancy resolution solution to the slightly more complicated constraint formulae, one obtains:

$$\begin{aligned} \dot{\Theta}^* = {} & J^+_{\mathcal{R}} S^T \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} \;\; + \\ & [J_{\mathfrak{C}}(I - J^+_{\mathcal{R}} J_{\mathcal{R}})]^+ (S^T\tilde{V}^{\mathcal{B}}_{\mathfrak{C}} - J_{\mathfrak{C}} J^+_{\mathcal{R}} S^T \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}). \end{aligned} \tag{4.3}$$

It follows that for the problem to be feasible, i.e., that the center of mass velocity can be exactly tracked, the stance map must be full rank (the robot is wrench-resistant, see Definition 5), and there must exists some $x \in \mathbb{R}^{m\times n}$ such that

$$\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} = (S^T)^+ J_{\mathcal{C}} \left(J^+_{\mathcal{R}} S^T \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} + (I - J^+_{\mathcal{R}} J_{\mathcal{R}}) x \right).$$

Of course, the priorities can be reversed, and the same solution used by changing indices.

This solution could be useful in the following way. Let the desired center of mass velocity, $\tilde{V}^{\mathcal{B}}_{\mathcal{C}}$, be specified as a unit vector pointing away from the nearest face of the support polygon (into the interior of the polygon), transformed into the robot body frame. Then, solution (4.3) will track the desired end-effector velocity, if at all possible, while attempting to maintain, or even enhance, the stance's margin.

4.2.3 Problem #3: The Balanced Priority Solution

Now, a novel solution is introduced that encompasses and extends the previous ones. Moreover, one can assess the existence and uniqueness of the solution, and describe the geometric nature of resulting motions.

In many situations, it may be difficult to track the desired manipulation trajectory, while also maintaining stability, and minimizing joint motions. Therefore, in practice it makes sense to define a weighted optimization goal that blends the priorities of tracking, stance stability, and joint motions:

$$\begin{aligned} \text{minimize} \quad & \frac{\alpha}{2}\|V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} - \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}\|_2^2 + \frac{\beta}{2}\|V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} - \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{C}}\|_2^2 + \frac{\gamma}{2}\|\dot{\Theta}\|_2^2 \\ \text{subject to} \quad & J_{\mathcal{R}}(\Theta, x_0)\dot{\Theta} = S^T V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} \\ & J_{\mathcal{C}}(\Theta, x_0)\dot{\Theta} = S^T V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}}, \end{aligned} \tag{4.4}$$

where α, β, and γ are non-negative weights, whose relative magnitudes respectively correspond to the relative importance of the end-effector goal ($\|V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} - \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}\|_2^2$, where $V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}$ and $\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}}$ are actual and desired end-effector velocities respectively), desired center of mass motion ($\|V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} - \tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{C}}\|_2^2$ where $V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}}$ and $\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{C}}$ are actual and desired center-of-mass velocities respectively), and joint velocity magnitude ($\|\dot{\Theta}\|_2^2$). The solution to this problem can be tuned to vary the importance of different goals during task execution, or to approximate a particular problem from the previous section.

To better understand the problem 4.4, it is restated in new variables. Define

$$x = \begin{pmatrix} V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} \\ V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} \\ \dot{\Theta} \end{pmatrix}, \quad b = \begin{bmatrix} \alpha\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} \\ \beta\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} \\ 0 \end{bmatrix}, \quad P = \begin{bmatrix} \alpha I_{6\times 6} & 0 & 0 \\ 0 & \beta I_{6\times 6} & 0 \\ 0 & 0 & \gamma I_{N\times N} I \end{bmatrix},$$

$$F = \begin{bmatrix} S^T & 0 & -J_{\mathcal{R}}(x_0, \Theta) \\ 0 & S^T & -J_{\mathcal{C}}(x_0, \Theta) \end{bmatrix} .$$

In these variables, Equation (4.4) is equivalent to

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} x^T P x - x^T b \\ \text{subject to} & Fx = 0 \ \ . \end{array} \tag{4.5}$$

This is an equality constrained quadratic program (ECQP). Problems that can be formulated in this format have a general exact solution when F has maximal rank, and in Section 4.4, the ECQP above will be extended to an *inequality constrained* quadratic program that can also be solved efficiently, to allow us to solve a much larger class of local planning problems.

Geometrically, the solution to this system is a projection of the vector b onto the null-space of F. The nature of the projection is defined by the matrix P above; when the weights are nonzero but equal, then the projection is orthogonal. When the weights are non-zero but not equal, the projection is *oblique* — it is stretched along components of x and b with large weights, and shrunk along the components with small weights. The problem sometimes has solutions when weights are set to zero. In these cases the solution, though correct, is useless as the corresponding variable is contracted to zero.

Proposition 4. *The problem (4.4) is equivalent to the system of linear equations (4.6). When $P \succ 0$ and F is full rank, the problem (4.4) can be solved explicitly, and the solution is linear in the desired end-effector and center of mass velocities.*

Proof. Following the steps in Section 2.2, after applying the KKT conditions, one finds that optimal primal and dual variables, (x^*, ν^*) must satisfy

$$\begin{bmatrix} P & F^T \\ F & 0 \end{bmatrix} \begin{pmatrix} x^* \\ \nu^* \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix} . \tag{4.6}$$

When F has maximal rank and $P \succ 0$ (for reasons given in Section 2.2), the unique (4.6) can be obtained explicitly . Let

$$\begin{bmatrix} G_{11} & G_{12} \\ G_{12}^T & G_{22} \end{bmatrix} =$$
$$\begin{bmatrix} \frac{1}{\alpha} S^T S + \frac{1}{\beta} J_{\mathcal{R}}(x_0,\Theta) J_{\mathcal{R}}(x_0,\Theta)^T & \frac{1}{\gamma} J_{\mathcal{R}}(x_0,\Theta) J_{\mathcal{C}}(x_0,\Theta)^T \\ \frac{1}{\gamma} J_{\mathcal{C}}(x_0,\Theta) J_{\mathcal{R}}(x_0,\Theta)^T & \frac{1}{\beta} S^T S + \frac{1}{\gamma} J_{\mathcal{C}}(x_0,\Theta) J_{\mathcal{C}}(x_0,\Theta)^T \end{bmatrix}.$$

Define H to be the Schur complement of G_{11} in the 'G' matrix above,

$$H = G_{11} - G_{12} G_{22}^{-1} G_{12}^T.$$

Define

$$\nu_2^* = H^{-1}\left(\frac{1}{\beta} S^T \tilde{V}_{\mathcal{WC}}^{\mathcal{B}} - \frac{1}{\alpha} G_{12}^T G_{11}^{-1} S^T \tilde{V}_{\mathcal{W}\mathcal{E}}^{\mathcal{B}}\right)$$
$$\nu_1^* = G_{11}^{-1}\left(\frac{1}{\alpha} S^T \tilde{V}_{\mathcal{W}\mathcal{E}}^{\mathcal{B}} - \frac{1}{\alpha} G_{12} S^T \tilde{V}_{\mathcal{WC}}^{\mathcal{B}}\right).$$

(these correspond to optimal Lagrange multipliers). Then, since $x = P^{-1}(b + F^T [\nu_1, \nu_2]^*)$ it follows that

$$\begin{pmatrix} V_{\mathcal{W}\mathcal{E}}^{\mathcal{B}} \\ V_{cm} \\ \dot{\Theta} \end{pmatrix}^* = \begin{pmatrix} \frac{1}{\alpha} \tilde{V}_{\mathcal{W}\mathcal{E}}^{\mathcal{B}} + S\nu_1^* \\ \frac{1}{\beta} \tilde{V}_{\mathcal{C}}^{\mathcal{B}} + S\nu_2^* \\ -J_{\mathcal{R}}(x_0,\Theta)\nu_1^* - J_{\mathcal{C}}(x_0,\Theta)\nu_2^* \end{pmatrix}$$

□

Proposition 5. (Sufficient condition for unique optimizer) *F has full row rank if and only if either*

- *S has a trivial null-space (i.e. there are no internal forces)*
- *S has a non-trivial null-space, and there are no two internal forces f_1 and f_2 (they satisfy $Sf_1 = Sf_2 = 0$) such that $J_{\mathcal{R}}(\Theta, x_0)^T f_1 = -J_{\mathcal{C}}(\Theta, x_0)^T f_2$*

Proof.

$$F = \begin{bmatrix} S^T & 0 & -J_{\mathcal{R}}(x_0,\Theta) \\ 0 & S^T & -J_{\mathcal{C}}(x_0,\Theta) \end{bmatrix}.$$

For F to be full-rank, its rows must be linearly independent, i.e., F^T must have a trivial null-space. Suppose that $\begin{pmatrix} f_1 & f_2 \end{pmatrix}^T$ is in the null-space of F^T. Then,

$$\begin{bmatrix} S & 0 \\ 0 & S \\ -J_{\mathcal{R}}(\Theta, x_0)^T & -J_{\mathcal{C}}(\Theta, x_0)^T \end{bmatrix} \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = 0. \tag{4.7}$$

If S has a trivial null-space so that $Sf = 0 \Leftrightarrow f = 0$, then F is full rank. This proves the first case.

Suppose that S has a non-trivial null-space. Then, for Equation 4.7 to hold true, it must hold that

$$J_{\mathcal{R}}(\Theta, x_0)^T f_1 = -J_{\mathcal{C}}(\Theta, x_0)^T f_2 \ .$$

If no such $f_1,\ f_2$ exist, then the F has full row-rank. □

4.3 Other Mobile Bases

For purposes of completeness, this sections sketches the analogous governing equations for manipulating robots with wheeled bases or highly articulated torsos.

Suppose manipulator arms are attached to a differential-drive base with unit width. The base's configuration is restricted to a plane, and consists of position and heading, (x, y, ϕ). The motion of the base is described by

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{pmatrix} = \frac{1}{2} \begin{bmatrix} \cos\phi & \cos\phi \\ \sin\phi & \sin\phi \\ -1 & 1 \end{bmatrix} \begin{pmatrix} \dot{\theta}_L \\ \dot{\theta}_R \end{pmatrix},$$

where θ_L is the left wheel angle, and θ_R the right (see Fig. 1.3). The Base Jacobian is obtained by rewriting the kinematics in the body frame, and converting them to twists. One finds that the *base Jacobian* is given by

$$J_{\mathcal{B}} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{-1}{2} & \frac{1}{2} \end{bmatrix},$$

so that

$$V^{B}_{\mathcal{WB}} = J_{\mathcal{B}} \begin{pmatrix} \dot{\theta}_L \\ \dot{\theta}_R \end{pmatrix}.$$

Suppose that the robot has N arms, and let the joint angles in the i^{th} arm be denoted by θ_i. Then the Center of Mass Jacobian is simply

$$J_{\mathcal{C}}(\theta_1, \theta_2, ...\theta_N) = \begin{bmatrix} J_{\mathcal{B}} & \bar{J}_1(\theta_1) & \dots & \bar{J}_N(\theta_N) \end{bmatrix}.$$

4.3.1 Kinematics for a Serial Chain Torso

In this case, the spatial and base frame can be coincident, and the Base jacobian is simply the Base's spatial jacobian:

$$J_{\mathcal{B}}(\theta_{\mathcal{B}}) = \begin{bmatrix} \xi_1 & \xi_2^{\dagger} & \dots & \xi_{n_{\mathcal{B}}}^{\dagger} \end{bmatrix},$$

where ξ_i is the twist associated with the i^{th} joint at zero configuration and

$$\xi_i^{\dagger} = \mathrm{Ad}^{-1}_{(e^{\hat{\xi}_i \theta_i} \dots e^{\hat{\xi}_{n_{\mathcal{B}}} \theta_{n_{\mathcal{B}}}})} \xi_i \ .$$

The center of mass Jacobian in this case is

$$J_{\mathcal{C}}(\theta, x_0) = \begin{bmatrix} J_{\mathcal{B}}(\theta_{\mathcal{B}}, x_0) & \mathrm{Ad}_{g_{\mathcal{BS}_1}} \bar{J}_1(\theta_1) & \dots & \mathrm{Ad}_{g_{\mathcal{BS}_n}} \bar{J}_N(\theta_N) \end{bmatrix}.$$

4.4 A Generalized Local Planning Problem

This section generalizes the local problems of Section 4.2 to suit all of the mobile robots in Figure 1.3, to allow more general weighting of the different aspects of a task, and to accommodate a large class of natural geometric constraints that arise during mobile manipulation. In particular, the local problem is formulated as a Quadratic Program (QP) with linear inequality constraints. Problems formulated in this framework can be solved efficiently using modern software [54].

Consider the following problem in light of (4.4):

$$\begin{aligned}
&\min \frac{1}{2}\|V^{\mathcal{B}}_{\mathcal{WE}} - \tilde{V}^{\mathcal{B}}_{\mathcal{WE}}\|_{2,P_{\mathcal{E}}} + \frac{1}{2}\|V^{\mathcal{B}}_{\mathcal{W}C} - \tilde{V}^{\mathcal{B}}_{\mathcal{W}C}\|_{2,P_C} + \frac{1}{2}\|\dot{\Theta}\|_{2,P_\Theta} \\
&\qquad\qquad \text{(Wheeled/Articulated)} \qquad\qquad \text{(Legged)} \\
&\text{subject to} \quad J_R(\Theta, x_0)\dot{\Theta} = V^{\mathcal{B}}_{\mathcal{WE}} \qquad J_{\mathcal{R}}(\Theta, x_0)\dot{\Theta} = S^T V^{\mathcal{B}}_{\mathcal{WE}} \\
&\qquad\qquad\quad J_C(\Theta, x_0)\dot{\Theta} = V^{\mathcal{B}}_{\mathcal{W}C} \qquad J_{\mathcal{C}}(\Theta, x_0)\dot{\Theta} = S^T V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}},
\end{aligned} \tag{4.8}$$

where $\|x\|_{2,P} = \sqrt{x^T P x}$ defines a weighted 2-norm. The objective of (4.8) is very similar to that of (4.4) — it indicates that one wants to choose $V^{\mathcal{B}}_{\mathcal{WE}}$ to be close to a desired end-effector velocity, $\tilde{V}^{\mathcal{B}}_{\mathcal{WE}}$, and that the true center of mass velocity should be close to a specified velocity $\tilde{V}^{\mathcal{B}}_{\mathcal{W}C}$. When a weighted norm is applied to a difference in twists (as in the first and second terms of the objective in (4.8)), the weighting matrix P should be specified with respect to the same reference frame as the motion goal it is associated with. For example, the weights on the end-effector goal are intuitively specified as a diagonal matrix of non-negative weights $P^{\mathcal{H}}_{\mathcal{E}} = \text{diag}[w^{\mathcal{E}}_1, \ \ldots, \ w^{\mathcal{E}}_6]$, in a *hybrid* reference frame $\mathcal{H}$ [2]. However, to use these weights in the objective of (4.8), the weighting matrix must be transformed from frame $\mathcal{H}$ to frame $\mathcal{B}$ by applying the following formula:

$$P_{\mathcal{E}} = \text{Ad}^T_{\mathcal{HB}} P^{\mathcal{H}}_{\mathcal{E}} \text{Ad}_{\mathcal{HB}}.$$

The desired center of mass velocity $\tilde{V}^{\mathcal{B}}_{\mathcal{W}C}$ in (4.8) might be determined so that the resulting motion of the robot's center of mass remains fully within the support region (e.g. towards the center of support — a more natural way to control center of mass motion, using constraints, is given below). The desired end-effector velocity $\tilde{V}^{\mathcal{B}}_{\mathcal{WE}}$ is specified as the tangent to the desired end-effector path in SE(3) [3]. The path can be any c^2 curve.

The weights' relative magnitude corresponds to the importance of each term and component of motion in a given problem. Generally, weights in $P_{\mathcal{E}}$ are chosen to be significantly larger than the other weights, as the end effector goal is the highest priority. From Proposition 5, one has an exhaustive understanding of existence and uniqueness of solutions to the problem (4.4). Towards

[2]$\mathcal{H}$ is co-located with the end-effector reference frame but aligned with the world reference frame.

[3]A *desired velocity* for the end-effector can be determined from the transformation between the current pose and the desired pose; the velocity (twist) corresponding to the error is determined by the matrix logarithm. See [58] Chapter 2.

the same understanding for (4.8), define:

$$P = \begin{bmatrix} P_{\mathcal{E}} & 0 & 0 \\ 0 & P_{\mathcal{C}} & 0 \\ 0 & 0 & P_{\Theta} \end{bmatrix}, \quad F_1 = \begin{bmatrix} I & 0 & (J_{\mathcal{R}}(\Theta, x_0)) \\ 0 & I & (J_{\mathcal{C}}(\Theta, x_0)) \end{bmatrix}, \quad F_2 = \begin{bmatrix} S^T & 0 & (J_{\mathcal{R}}(\Theta, x_0)) \\ 0 & S^T & (J_{\mathcal{C}}(\Theta, x_0)) \end{bmatrix},$$

$x = \begin{pmatrix} V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} & V^{\mathcal{B}}_{\mathcal{W}\mathcal{C}} & \Theta \end{pmatrix}^T$, and $b = \begin{bmatrix} P_{\mathcal{E}}\tilde{V}^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} & P_{\mathcal{C}}\tilde{V}_{cm} & 0 \end{bmatrix}^T$ so that (4.8) can be rewritten as:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}x^T P x - x^T b \\ &\text{subject to} && F_i x = 0 \ . \end{aligned} \tag{4.9}$$

Where $i = 1$ corresponds to wheeled and articulated bases while $i = 2$ applies to legged bases

The system of equations that governs the solution to (4.9) is

$$\begin{bmatrix} P & F_i^T \\ F_i & 0 \end{bmatrix} \begin{pmatrix} x^* \\ \nu^* \end{pmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} \quad i \ = 1, \ 2. \tag{4.10}$$

It is shown in Section 2.2 that the above equation has a unique solution if $P \succ 0$ and F_i is full rank. Existence and uniqueness are stated explicitly for the problem (4.8) as a proposition:

Proposition 6. *The constrained minimization (4.8) has a solution whenever $P \succeq 0$ and the equation (4.10) has a solution. (4.8) has a unique solution whenever the coefficient matrix on the left-hand-side of (4.10) is invertible. This is true*

- *for wheeled an articulated robots whenever $P \succ 0$ (because F_1 above is always full rank)*
- *for legged robots whenever $P \succ 0$ and F_2 is full rank, conditions for which are specified in Proposition 5*

Practically, this means that singularities do not have as large an impact as they do for existing iterative IK (inverse kinematics) solvers. Intuitively, this makes sense, since velocities are not being *enforced*, but instead *encouraged*. Geometrically, the solution is an oblique projection of possible joint velocities onto affine subspaces defined by the kinematics.

4.5 Fully Integrated Planning

This instantaneous solution framework is now expanded and rewritten in order to easily handle a broad variety of constraints and goals, and to make the problem size as compact as possible in the interest of efficiency [4]. This is done by substituting the equality constraints into the objective explicitly, and by adding linear inequality constraints to accommodate hard limits.

Let F_i^g be a reference frame (attached anywhere to the mechanism, not necessarily an end-effector) with an associated motion goal [5] and let $\tilde{V}_{\mathcal{B}F_i^g}^{\mathcal{B}}$ be the corresponding desired instantaneous velocity of this frame. Let $J_{F_i^g}$ be the corresponding Jacobian matrix [6], such that $J_{F_i^g}\dot{\Theta} = S^T V_{\mathcal{B}F_i^g}^{\mathcal{B}}$ for a legged base or $J_{F_i^g}\dot{\Theta} = V_{\mathcal{B}F_i^g}^{\mathcal{B}}$ for a wheeled or serial chain base. The reference frames and motion are illustrated in Figure 4.2 (a). In order to write the motion goals efficiently in the objective function, the error norm is squared and expanded as

$$\begin{aligned}
&\|V_{\mathcal{B}F_i^g}^{\mathcal{B}} - \tilde{V}_{\mathcal{B}F_i^g}^{\mathcal{B}}\|_{2,P_{F_i^g}}^2 \\
&= \dot{\Theta}^T L^T P_{F_i^g} L \dot{\Theta} - 2\dot{\Theta}^T L^T \tilde{V}_{\mathcal{B}F_i^g}^{\mathcal{B}} + (\tilde{V}_{\mathcal{B}F_i^g}^{\mathcal{B}})^T \tilde{V}_{\mathcal{B}F_i^g}^{\mathcal{B}}
\end{aligned}$$

where $L_{F_i^g} = S^+ J_{F_i^g}$ [7] for a legged base, and $L_{F_i^g} = J_{F_i^g}$ for a wheeled or articulated base.

In order to efficiently represent *hard* constraints, notice that the motion of a frame F_i^r can be restricted in the 'direction' of $\tilde{V}_{\mathcal{B}F_i^r}^{\mathcal{B}}$ by enforcing

$$(\tilde{V}_{\mathcal{B}F_i^r}^{\mathcal{B}})^T V_{\mathcal{B}F_i^r}^{\mathcal{B}} = (\tilde{V}_{\mathcal{B}F_i^r}^{\mathcal{B}})^T L_{F_i^r} \dot{\Theta} \geq \alpha_i,$$

where $\alpha_i \geq 0$ is a minimum directional velocity (in the direction of $\tilde{V}_{\mathcal{B}F_i^r}^{\mathcal{B}}$) to be specified by the user. The larger the magnitude of α_i, the more aggressively the robot tries to move in the direction of $\tilde{V}_{\mathcal{B}F_i^r}^{\mathcal{B}}$). This situation is illustrated by the link collision in Figure 4.2 (b).

Suppose that there are n motion goals, and p hard constraints. Define

[4] The worst case complexity of solving a QP with linear constraints is shown to be $O(n^3 L)$, where n is the size of the decision variable and L is the program input size [20].

[5] Examples of motion goals include motion of the end-effector to a desired location, motion of the center of mass towards the center of support, motion of the abdomen along a trajectory, limiting the rotation of the head/neck of the robot, etc.

[6] Compute $J_{F_i^g}$ using sections 3.3 and 4.3; if the link in question is on a free limb, then include the motion of the base, otherwise do not.

[7] For a legged robot, S^T will be injective whenever the sum of the ranks of the base legs' wrench bases is greater than 6. Therefore the kinematic constraints can be 'inverted', changing an equation of contact velocities to one of twists. This inverse is 'exact' whenever goals are feasible, and approximate (in a least-squares sense) otherwise.

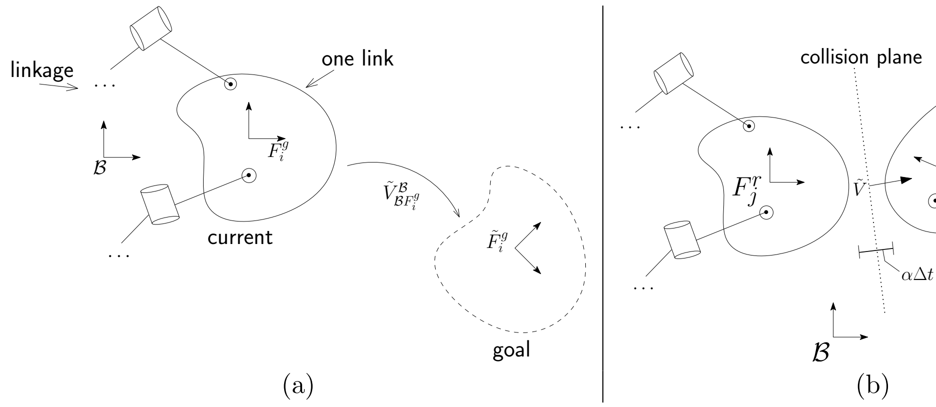


Figure 4.2: (a) Link i with current and goal positions and orientations, and the velocity that generates the motion towards the goal. (b) Collision repulsion/avoidance for colliding links i and j. The desired velocity is defined by the normal to the collision plane as shown.

$$P = \sum_{i=1}^{n} L_{F_i^g}^T P_{F_i^g} L_{F_i^g} + P_\Theta \ , \qquad \beta = -2 \sum_{i=1}^{n} L_{F_i^g}^T \tilde{V}^{\mathcal{B}}_{\mathcal{B}F_i^g} \ ,$$

$$A = \begin{bmatrix} — & (\tilde{V}_1^r)^T L_{F_1^r} & — \\ — & \vdots & — \\ — & (\tilde{V}_p^r)^T L_{F_p^r} & — \end{bmatrix} , \quad \vec{\alpha} = \begin{pmatrix} \alpha_1 \\ \vdots \alpha_p \end{pmatrix}$$

Problem (4.8) can now be extended and written as a much more general constrained minimization problem:

$$\begin{aligned} &\text{minimize} \quad \dot{\Theta}^T P \dot{\Theta} + \dot{\Theta}^T \beta \\ &\text{subject to} \quad A\dot{\Theta} \preceq \vec{\alpha} \ , \end{aligned} \tag{4.11}$$

where the inequality constraint holds element wise. With this more general formulation, a vast number of manipulation goals, sub-goals and constraints can be naturally included. Some of these include:

- **Pointing**: The $z-$axis of a given link frame F_i can be pointed in particular direction by adding the link's velocity and the twist in the desired direction to the objective. One neglects rotation in the pointing direction by letting

$$P_{F_i} = \mathrm{Ad}_{g_{F_i\mathcal{B}}}^T \mathrm{diag}[w_1^{F_i}, w_2^{F_i}, \dots w_5^{F_i}, 0] \mathrm{Ad}_{g_{F_i\mathcal{B}}}$$

(the instantaneous rotation about the frame-fixed $z-$axis is ignored). This could be used, for example, to encourage a camera direction ('gaze') as shown in Figure 4.5 in the next section.

- **Tracking**: A link frame F_i can be moved along a desired trajectory by adding it to the objective, along with the tangent to the trajectory at the current configuration

- **Collision Repulsion**: If the robot is in a configuration at which it makes contact with obstacles or itself, the links in contact can be forced to move away from the collision state by defining a repulsive velocity as the normal to the collision plane [50], and adding a hard inequality constraint forcing the link to move in the repulsive direction by making the corresponding α_i in equation (4.5) a positive number. For self-collisions, one or both collision links can be made to move away from collision. Concretely, suppose that link a and link b, with reference frames F_a^r and F_b^r respectively, are close to colliding. Suppose that the unit velocity $\tilde{V}$ is along the collision normal from a to b. If one enforces

$$\tilde{V}^T V^{\mathcal{B}}_{\mathcal{B}F_i^r} > \alpha \ , \quad \tilde{V}^T V^{\mathcal{B}}_{\mathcal{B}F_j^r} < -\alpha \ ,$$

for some $\alpha \geq 0$, the two links will de-collide locally. This feature is illustrated in Fig. 4.2 (b)

- **Hard Static Equilibrium Constraints**: The robot's center of mass can be kept within the robot's support region using linear inequality constraints. Let v_i be the twist corresponding to pure translation towards the i^{th} side of the support region. Let the distance to the i^{th} side be d_i. If one enforces the constraint

$$v_i^T V^{\mathcal{B}}_{\mathcal{B}\mathcal{C}} = v_i^T L_{\mathcal{C}}(\Theta, x_0)\dot{\Theta} \leq d_i$$

for every side of the support region, and if the robot follows the prescribed velocity for much less than 1 second, the center of mass will not leave the support region.

- **Frame Boundaries**: A reference frame's origin (or the position vector difference between reference frames) can be kept in any polyhedral region in space using inequality constraints on frame velocity; these are constructed in the same way as the hard static equilibrium constraints. This could be used, for example, to keep the robot within some workspace boundaries, or to enforce hard constraints on gaze.

- **Joint Position, Velocity and Acceleration Limits**: These are straightforward to implement as inequality constraints on joint velocities. Concretely, suppose a joint lies at angle Θ_i, and $\tilde{\Theta}_i > \Theta_i$ is known to be a limit that must be avoided, then enforce

$$\dot{\Theta}_i \leq \frac{\tilde{\Theta}_i - \Theta_i}{\Delta t},$$

where Δt is a time-step. Similarly, if the maximum joint velocity is $\dot{\Theta}_{\max}$, this can be imposed as $-\dot{\Theta}_{\max} \leq \dot{\Theta} \leq \dot{\Theta}_{\max}$.

- **Joint Velocity and Acceleration Limits**:

- **Configuration Biasing**: The robot can be biased towards a known nominal configuration by adding a body velocity bias (as a motion goal) to a nominal body pose, and a joint angle bias that penalizes motions away from nominal joint angles. Suppose that $\tilde{\Theta}$ is a preferred configuration, and that the current configuration is Θ. Define $\Phi = \frac{1}{\Delta t}(\Theta - \tilde{\Theta})$. Then, the robot is biased towards this configuration by adding $\|\dot{\Theta} - \Phi\|$ to the objective, to penalize velocities that move away from the preferred configuration. This approach is useful for maintaining the internal posture of a legged robot's stance.

Examples demonstrating all of these features are given in Section 4.8.

4.6 Feasibility Certificates and Optimal Constraints

The ability to certify feasibility of a solution to (4.11) [8] or lack thereof is crucial for ensuring that partial plans that end in unsafe configurations are not executed. The problem (4.11) is infeasible if and only if the constraints cannot be met. Practically, checking feasibility is an important part of computationally solving constrained optimization problems. These methods fall under the title of Phase I or Feasibility methods [6], and are used as a preliminary step to find a strictly feasible point (or to certify/prove that the problem is infeasible) before applying a numerical optimization scheme. In the context of local planning for a mobile robot, they help us understand how best to specify constraints, and in the case of infeasibility, exactly which constraints make local planning impossible.

Constraint feasibility can be checked very rapidly by solving the following linear program (the parameters are the same as those in (4.11)):

[8] A constrained optimization is feasible if the set of points that satisfy the constraints is not empty. In the context of planning using optimization, examples of infeasibility might include configurations which cannot move without inducing collision, or goals that cannot be achieved without falling.

$$\begin{aligned} \text{minimize} \quad & -t \\ \text{subject to} \quad & A\dot{\Theta} - \vec{\alpha} \leq t \\ & t \leq 1 \ . \end{aligned} \tag{4.12}$$

The optimal solution value is 1 if the constraints are feasible and 0 otherwise.

In order to choose $\vec{\alpha}$ in a clever way, one might ensure that feasibility is satisfied using (4.12) for the minimum values of $\vec{\alpha}$, and thereafter select an 'optimal' $\vec{\alpha}$ in the sense of the following problem:

$$\begin{aligned} \text{minimize} \quad & \vec{1}^T \vec{\alpha} \\ \text{subject to} \quad & A\dot{\Theta} \leq \alpha \\ & -1 \leq \alpha \leq 1 \end{aligned} \tag{4.13}$$

The resulting α defines the constraints that most aggressively avoid the limits placed on the system. In other words, the α that solves (4.13) provides the safest possible operation locally when the constraints include things like equilibrium, collision repulsion, and singularities.

4.7 Iterative Algorithm

This section describes a method that integrates the ideas presented in this thesis so far (this is the algorithm used in the experiments of Section 4.8). Define a robot configuration data structure, $\mathcal{C}$, that contains the homogeneous transformations to every joint and link frame as well as the center of mass frames, as well as the instantaneous twists of every joint in the robot as seen in the base frame. Assume n frames are commanded to follow trajectories, and up to p inequality constraints; when there are fewer than p constraints for an iteration, the unused rows and elements of A and $\vec{\alpha}$ are chosen to be trivially satisfied (i.e. they are chosen so that the inequality they define is always true, for example $A = 0$ and $\alpha = 1$). In addition, for checking feasibility, $\vec{\alpha}$ is set to the minimum reasonable value (e.g. a very small positive number for collision avoidance, or exactly the distance to a support region boundary). We also assume the existence of the following functions:

GoalDist(*$\mathcal{C}$*)

Returns the distance to the goal (e.g. distance between end-effector current and desired poses.

SupportRegionVector(*$\mathcal{C}$*,i)

Returns direction from the center of mass to the i^{th} face of the quasi-static support region in the base frame, for $i \in 1 \ldots s$ where s is the number of faces of the support region.

COMDist($\mathcal{C}$,i)

Computes the distance from the center of mass to the i^{th} face of the support region.

CheckFeasibility(A,$\vec{\alpha}$)

Returns the solution to the LP (4.12) .

SetAlpha($\mathcal{C}$, A)

Returns a sensible choice of $\vec{\alpha}$ using (4.13) or otherwise.

SolveQP($P, \beta, A, \vec{\alpha}$)

Returns the solution to (4.11). A key part of any algorithm that works using the ideas presented here is the QP solver. CVXGEN [54] was used to generate a fast, custom, primal-dual interior point method solver (a small, stand-alone C code) which solves the QP.

ComputeStepSize($\mathcal{C}$,$\dot{\Theta}$)

Computes a step size Δt by searching $[0, 1]$ and ensuring that stepping by $\Delta t \times \dot{\Theta}$ does not result in violation of constraints. This function is unnecessary in most cases (the step size can be set to 1).

Update($\mathcal{C}$, $\dot{\Theta}$, Δt)

Updates the configuration after moving by $\Delta t \times \dot{\Theta}$.

Algorithm 4.1 QP-based path-planning and goal configuration search.

```
Initialize $\mathcal{C}$  while $GoalDist(\mathcal{C}) > \epsilon$ do
    for $i = 1 \dots n$ do
        Compute desired twists $\tilde{V}^{\mathcal{B}}_{\mathcal{B}F_i}$  Select weighting matrices $P_i$
    end
    for $i = 1 \dots s$ do
        $\tilde{V}^r_i$ = SupportRegionVector($\mathcal{C}$,i)
        $\alpha_i$ = COMDist(c,i)
    end
    if In Collision then
        for $i = 1 \dots$ # of Collisions do
            Set $\tilde{V}^r_{\mathcal{B}(i+s)}$ to be collision normal
        end
    end
    Construct $P, \beta, A, \alpha$ as given in (4.11)  if CheckFeasibility(A,$\alpha$) then
        SetAlpha($\mathcal{C}$,A)  $\dot{\Theta}$ = SolveQP($P, \beta, A, \vec{\alpha}$)  $\Delta t$ = ComputeStepSize($\mathcal{C}$,$\dot{\Theta}$)  Update($\mathcal{C}$, $\dot{\Theta}$, $\Delta t$))
    else
        Return Failure
    end
end
```

The trajectories produced by this algorithm can be interpreted as 'selective' numerical integration of the robot's kinematics, where at each step, the local derivative is 'selected' to be safe and to achieve various goals, and move by a small amount along the chosen derivative direction. After moving, 'reselect' and repeat.

4.8 Examples

This section demonstrates and validates the proposed methods of this thesis on two real-world robots. The *Surrogate* robot (shown in Figure 1.2 (b)) is a highly redundant 21 DOF robot torso mounted on 2-DOF differential drive mobile base (Figures 4.4 to 4.7). Because the torso and arms are heavy and mounted near the edge of the base, the robot can easily tip over; arm and torso motions must be chosen carefully to ensure that the robot does not fall. The *Robosimian* robot (shown in Figure 1.2 (a)) is a quadruped whose four 7-DOF limbs are kinematically redundant. Each of Robosimian's limbs can be used for walking or for manipulation, as a hand with retractable fingers is mounted on the end of each limb. In order to perform manipulation tasks while standing, the motion of the supporting limbs and manipulating limb must be coordinated to maintain contact, and avoid falling down. It should be noted that the Robosimian legs (and Surrogate arms) are heavy, and so their movements significantly change the robot's center of mass. Simulations and experiments were performed with each of these systems.

The simulations and experiments use an iterative solution of the local quadratic program in Section 4.4, where the velocities resulting from the solution move the robot end effector closer to the desired goal while also moving the torso and non-manipulating arm to maintain balance. At the end of each iteration, the velocities are integrated (multiplied by the time constant) to form small joint position displacements. In our investigations, these displacements are limited to 0.1 radians per iteration. The self-collision constraint is incorporated into all of the simulations and experiments. Collisions between robotic bodies are computed using Bullet Collision Detection [12]. If collisions are detected after applying a joint position update, the previous QP is run again with additional velocity constraints enforcing repulsive motion between the bodies found in contact. Bullet provides a pairwise list of bodies in collision, and approximate collision locations.

Figure 4.3 shows snapshots from a simulation in which the left Surrogate arm must reach far forward to a hand goal pose. It should be noted that the right arm automatically extends backward in order to maintain the overall center of mass within the support region of the base.

Figure 4.4 demonstrates our proposed approach in other ways. The simulation snapshots shown

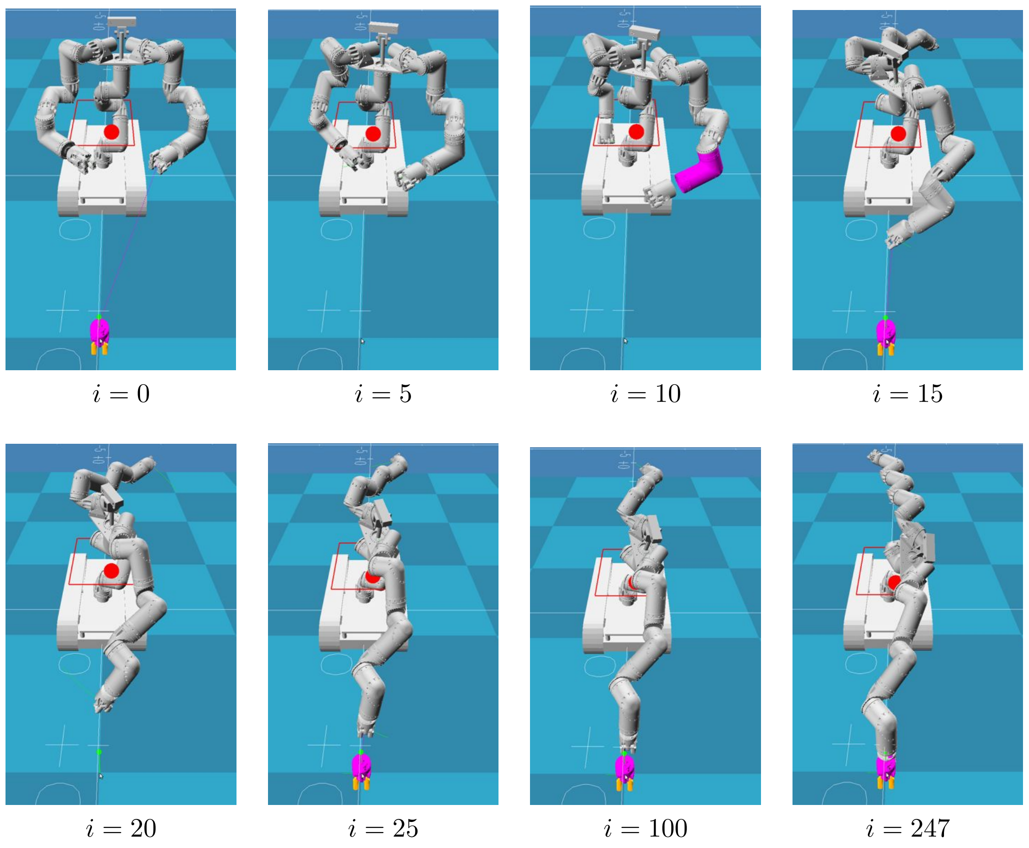


Figure 4.3: Snapshots of a simulation in which Surrogate uses the local QP solution to extend its left limb to a point 1 m in front and 0.2 m below its base. The desired end effector location is shown by the highlighted hand at iteration $i = 0$. Iterations 5 and 10 show the free right limb being constrained from contacting the robotic torso. The robot center of mass (red ball) is within the support polygon (red rectangle) for all iterations.

in Figures 4.4(a-c,e,f) all start in the same posture as Figure 4.4(d). Figures 4.4 show Surrogate reaching to different goals. Figure 4.4(e) shows the result analogous to 4.4(a) if the center-of-mass balance constraints are not applied, while 4.4(f) shows the result analogous to 4.4(b) when self-collision constraints are not incorporated. Figure 4.5 shows snapshots from a simulation another example in which the robot's gaze is also specified as a constraint to the planning algorithm. Fig. 4.7 shows snapshots fromm a SURROGATE effort to turn a valve 90°.

Computational Burden: The average run time for a single iteration (computing all Jacobians, constraints, and collisions, and solving the QP) was 348 microseconds μs for the simulations in the previous figures. On average just solving the QP took 267 μs, or 78% of the computation time. All computations were restricted to a single processing core of a 2.4 GHz i7. In the reported cases (a)-(f) in Figure 4.4 the maximum average iteration time was for case (a), which ends in a near singular configuration, at 584 μs, while the minimum was case (b), at 274 μs. The iterations were stopped when the end effector displacement error was less than 0.001 meters and 0.001 radians. Figure 4.4

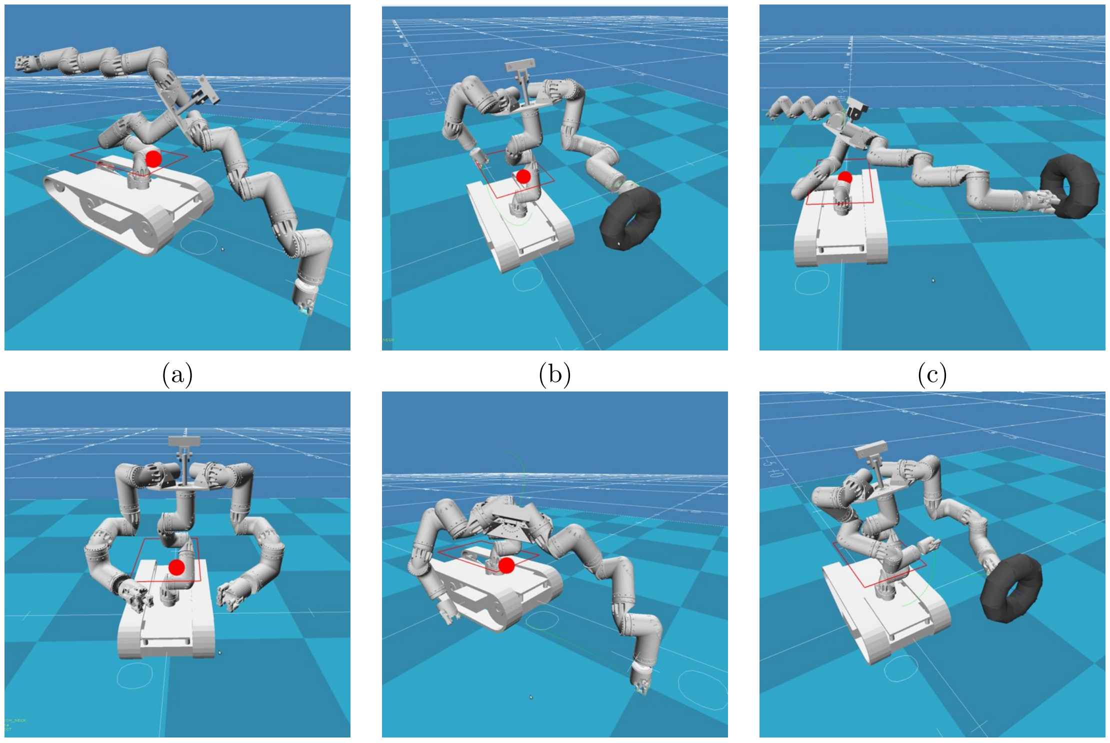

(a) (b) (c) (d) (e) (f)

Figure 4.4: Inverse Kinematics (IK) computed to specified end goals. The robot differential drive base is fixed. The red sphere is the robot center of mass (COM), and the red rectangle is the support polygon projected to the height of the COM. (a) Reaching to a point 1 meter in front of the robot, and 0.2 m below the drive plane. (b) Reaching to a torus 0.5 m in front of the robot. (c) Reaching to a torus 1.25 m to the side of the robot. (d) Starting pose for all IK searches. (e) IK computed to *(a)* without using balance constraints. (f) IK computed to *(b)* stopping on detected collisions

case (a) required 247 iterations to complete at a total time of 0.14 seconds, and case (b) took 17 iterations at a total time of 0.004 seconds.

Comparison against other approaches: The Surrogate robot has 7 degrees of freedom (DOF) in each limb and in its torso. The serial chain from the robot base to the primary end effector is 14 DOF, with an extra 7 DOF on the free arm which can be used for balancing. This leaves 8 redundant DOF in the main serial chain, with an extra 7 DOF in the free limb. The limbs and torso on the Surrogate robot do not have a kinematic wrist, which makes deriving analytic inverse kinematics difficult. As a comparison, IKfast was used to compute and inverse kinematic solution for the limb and the torso. Each IKfast call for the limb or torso requires fixing one joint, and solving for the inverse kinematic (IK) solution of the remaining 6 joints in the limb or torso (resulting in up to 8 configurations). Each IKfast call for a Surrogate limb takes approximately 1000 microseconds. As the redundant space in the main serial chain is so high (8 DOF), searching over this space and using analytic IK to solve for joint angles was shown to be computationally intractable.

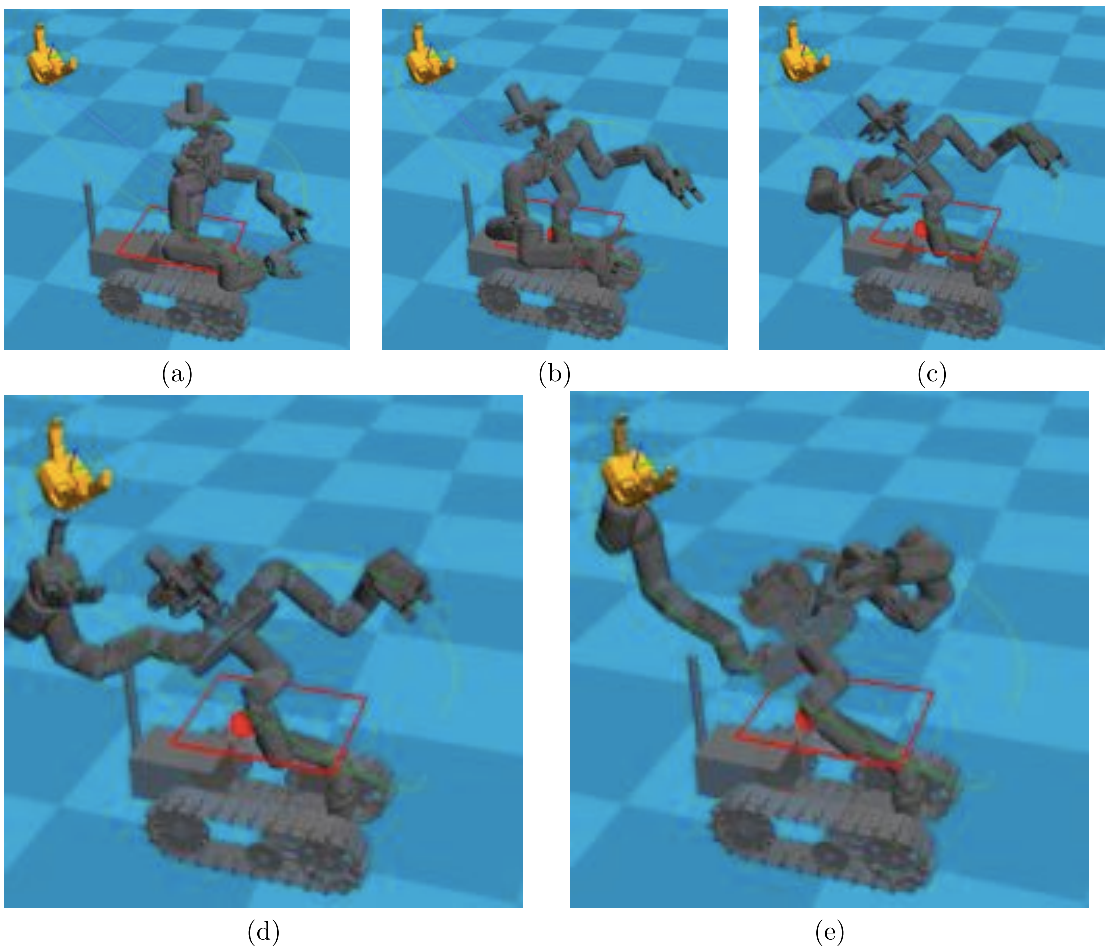


Figure 4.5: An example of including the robot's gaze as a goal. The robot reaches out to the yellow hand pose, avoiding collision and maintaining stability, and 'looks' at the goal by pointing the cameras at the goal.

Figure 4.6 shows snapshots from an experiment in which RoboSimian turns a valve using vision guidance and the planning tools described in the previous sections (including self-collision avoidance). The time taken for Algorithm 4.1 to plan the entire sequence was 6 milliseconds. Note that the entire mechanism adjusts throughout the sequence to maintain gravitational stability and to accommodate for the turning motion within its reachable workspace. Such a manipulation task for RoboSimian using analytical inverse kinematics coupled with a search over additional degrees of freedom is intractable. The entire experiment and an additional valve turning experiment can be found in the supplemental video.

Figure 4.8 is an example demonstrating the effect of joint biasing. The robot is made to move its body between five poses, with the last pose the same as the first. The effect of biasing is clear when one compares the trajectories of the joints (see Figures 4.9-4.12). It is clear that configuration biasing (weighing joint motions towards a desired position, in this case the initial configuration)

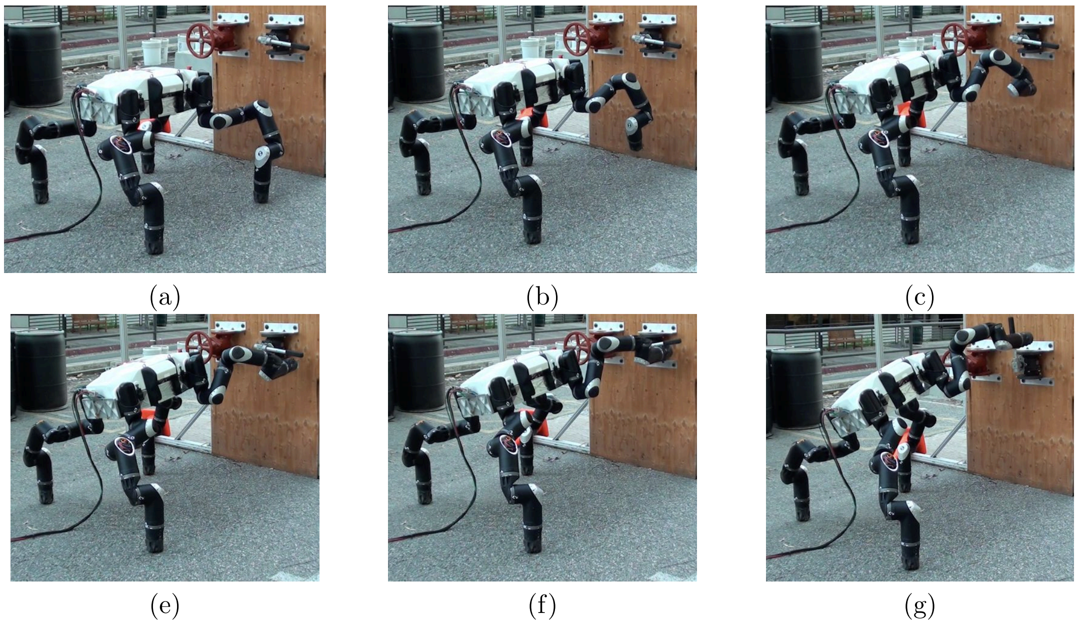


Figure 4.6: RoboSimian Turning a valve using QP Inverse Kinematics: The robot's motion is computed by iteratively solving the QP, and then executed on the robot in real time. (a) Starting pose. (b) Lifting the limb while remaining within support region. (c) Approaching the valve. (d) Contacting the valve. (e) Begin turning the valve. (f) The valve is fully open.

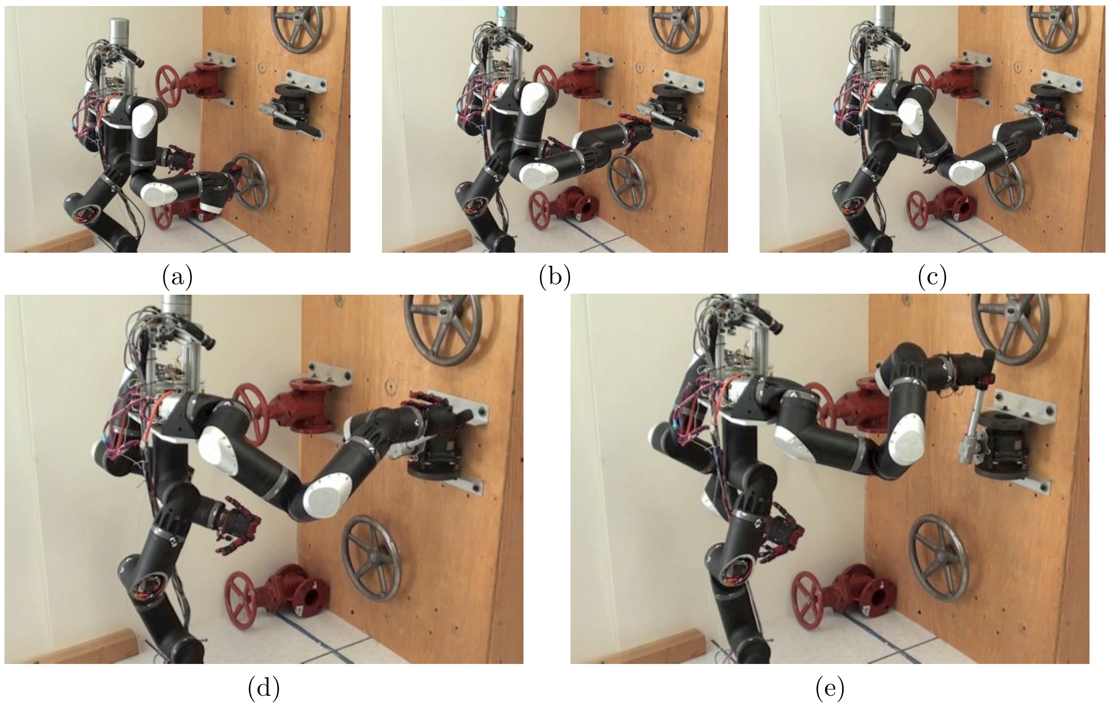


Figure 4.7: Turning a valve using QP Inverse Kinematics: The Robot's motion is computed by iteratively solving the QP, and then executed on the robot in real time. (a) Starting pose. (b) Pre-contact with the valve. (c) Contact with the valve. (d) Halfway through turning the valve. (e) The valve is fully open. The robotic torso significantly extended to achieve required end effector goals, and the free limb has contracted to maintain balance stability.

results in the robot being closer to that configuration when the body is moved to different poses before returning to the initial pose.

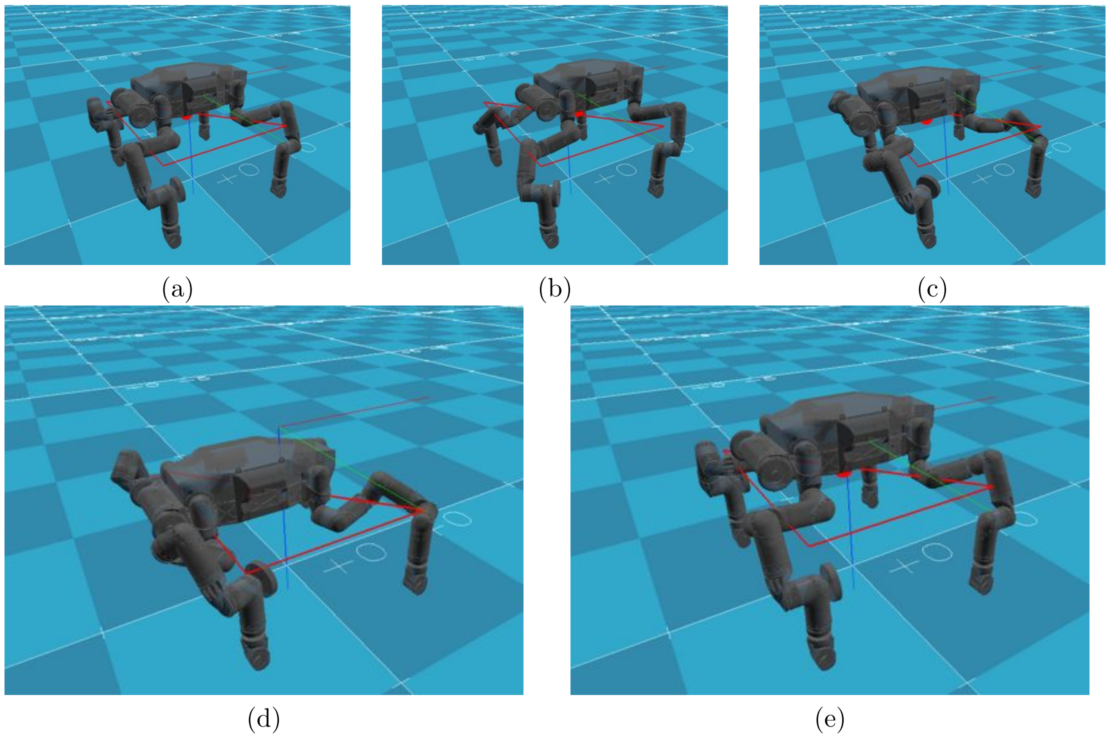


Figure 4.8: Sequence of Robot Stance Motions to demonstrate the effect of biasing to the nominal configuration shown in (a).

RoboSimian is made to perform the same reaching motion with and without joint-limits, in order to demonstrate their effect on the solution. The robot is commanded to raise its front right hand to a particular location, with (Figure 4.13 (c)) and without very restrictive limits (Figure 4.13 (b)) on Joints 3 (≥ -1 radians) and 4 (≤ -1 radians) of the front right limb.

The limits are strictly observed, using inequality constraints. Figure 4.14 shows a comparison of the enforced and unenforced joint trajectories on joints 3 and 4 for which limits are imposed. The motion of the remaining joints in the front right limb in the two cases is shown in Figure 4.15.

In order to compare experimental and planned motions, a simple reaching motion is planned, and then executed on the robot. The robot is made to raise its front right hand as shown in Figure 4.16. Error is measured with respect the 'desired' hand pose, shown in yellow.

This motion was obtained by solving a quadratic program of the form (4.11) within Algorithm 4.1. where the inequality constraints included balance (the center of mass is not allowed to leave the support region, with a 5 cm safety pad) and joint-velocity (magnitude less than 0.75 radians per second) and acceleration limits (magnitude less than 1.5 radians per second per second).

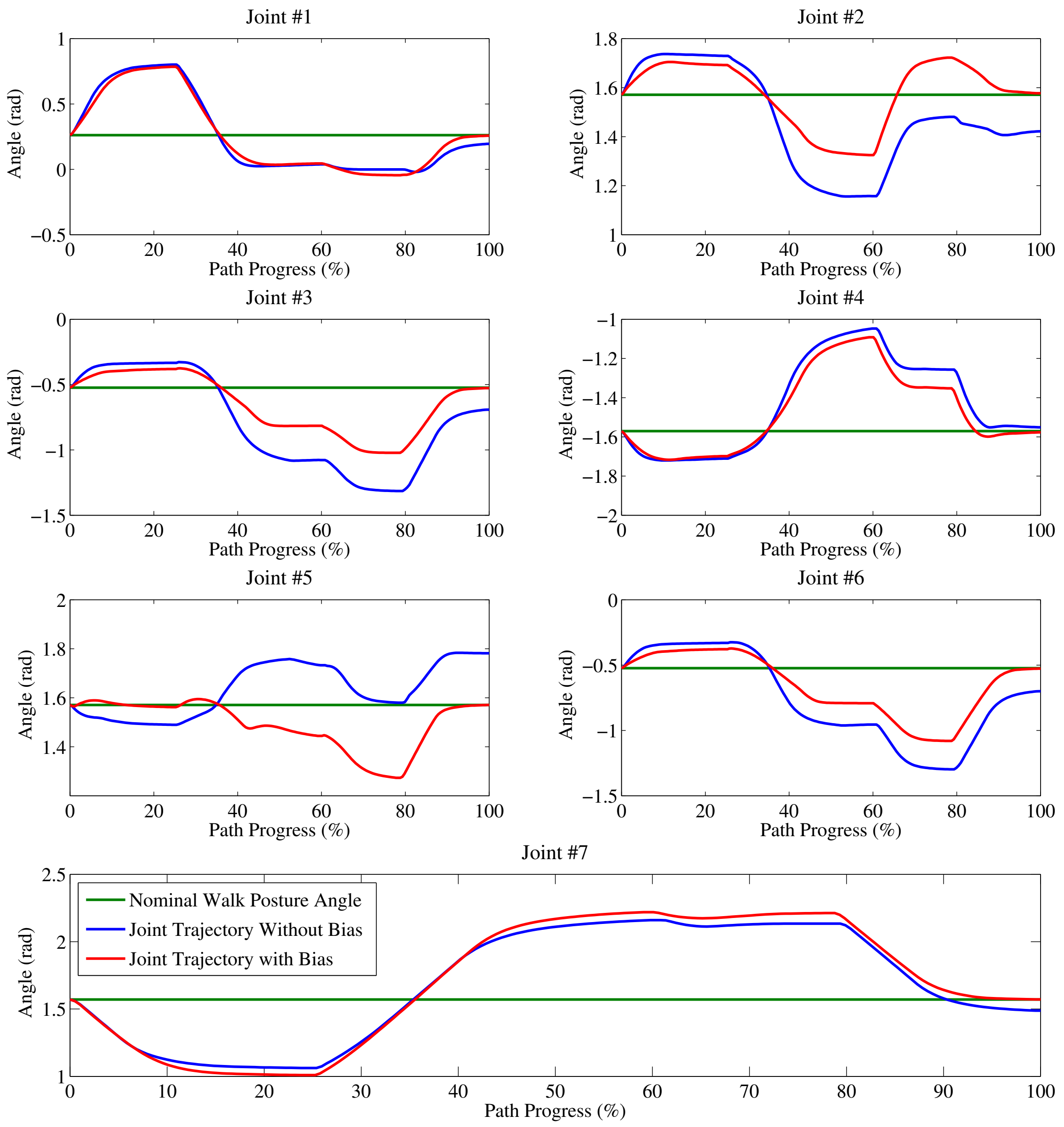


Figure 4.9: Comparison of Front-Right Limb Joint Trajectories with and without biasing to nominal posture as the robot moves between the configurations shown in 4.8. Angles are plotted against percentage of path completion for ease of comparison.

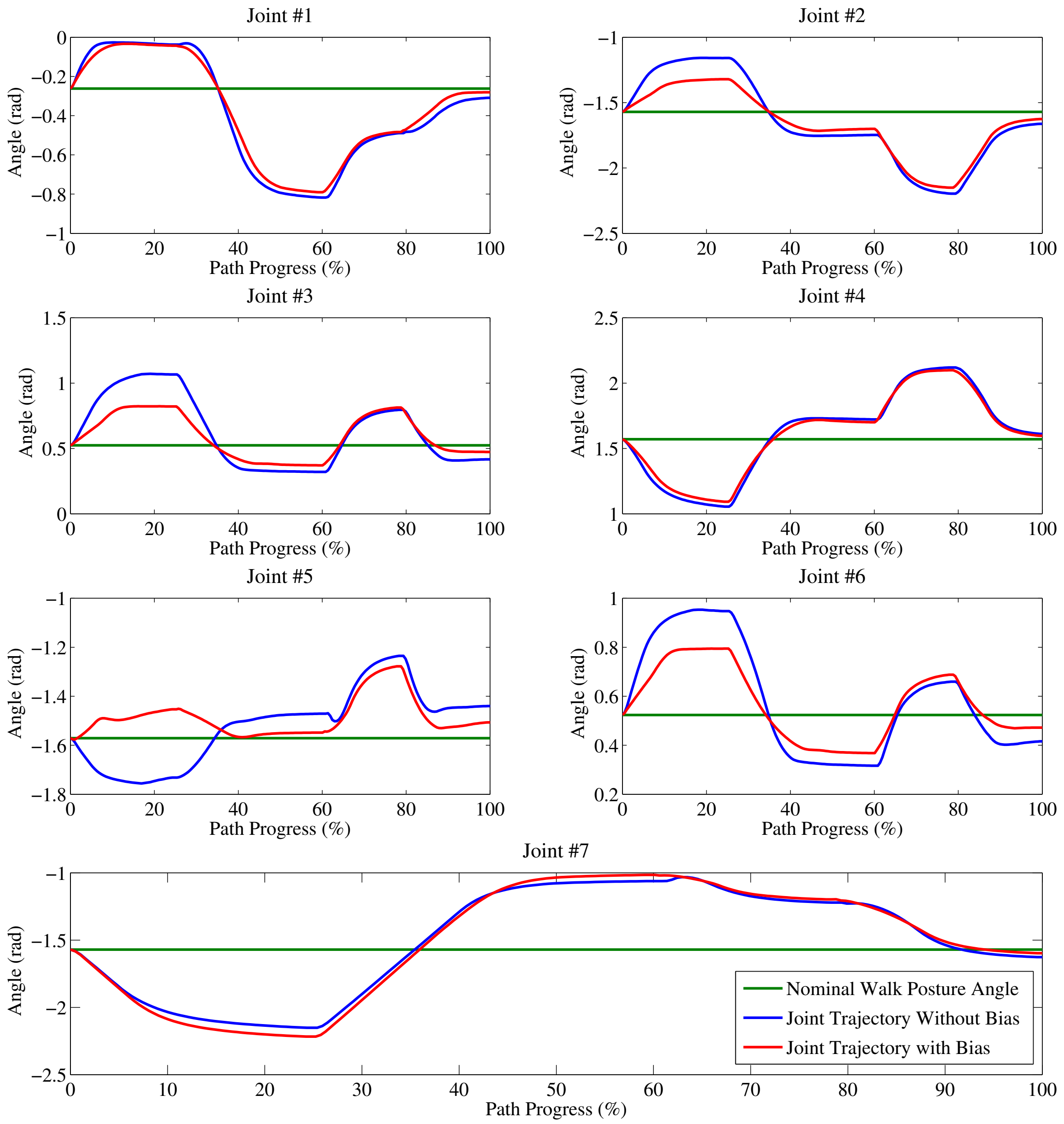


Figure 4.10: Comparison of Back-Right Limb Joint Trajectories with and without biasing to nominal posture as the robot moves between the configurations shown in 4.8. Angles are plotted against percentage of path completion for ease of comparison.

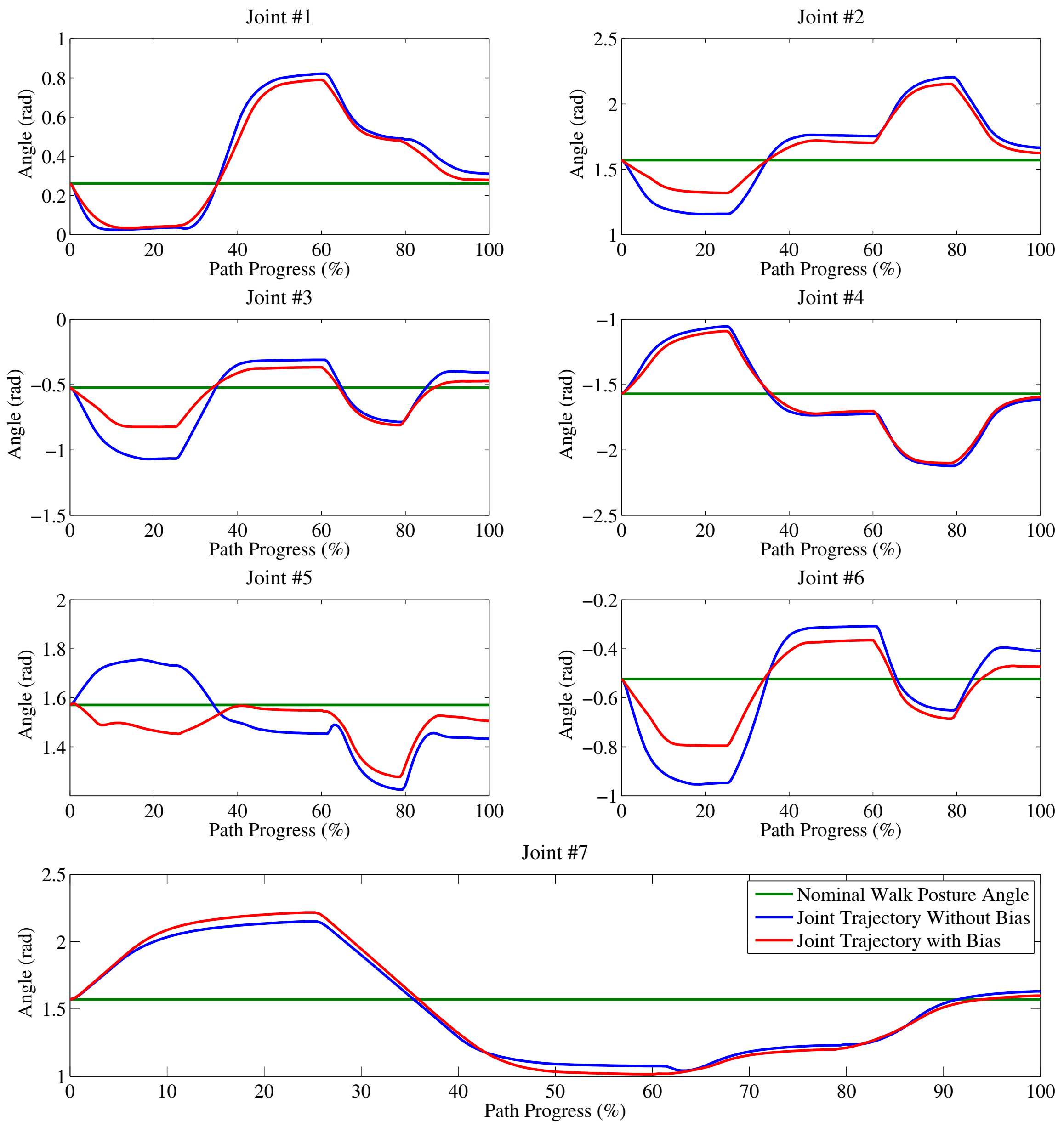


Figure 4.11: Comparison of Back-Left Limb Joint Trajectories with and without biasing to nominal posture as the robot moves between the configurations shown in 4.8. Angles are plotted against percentage of path completion for ease of comparison.

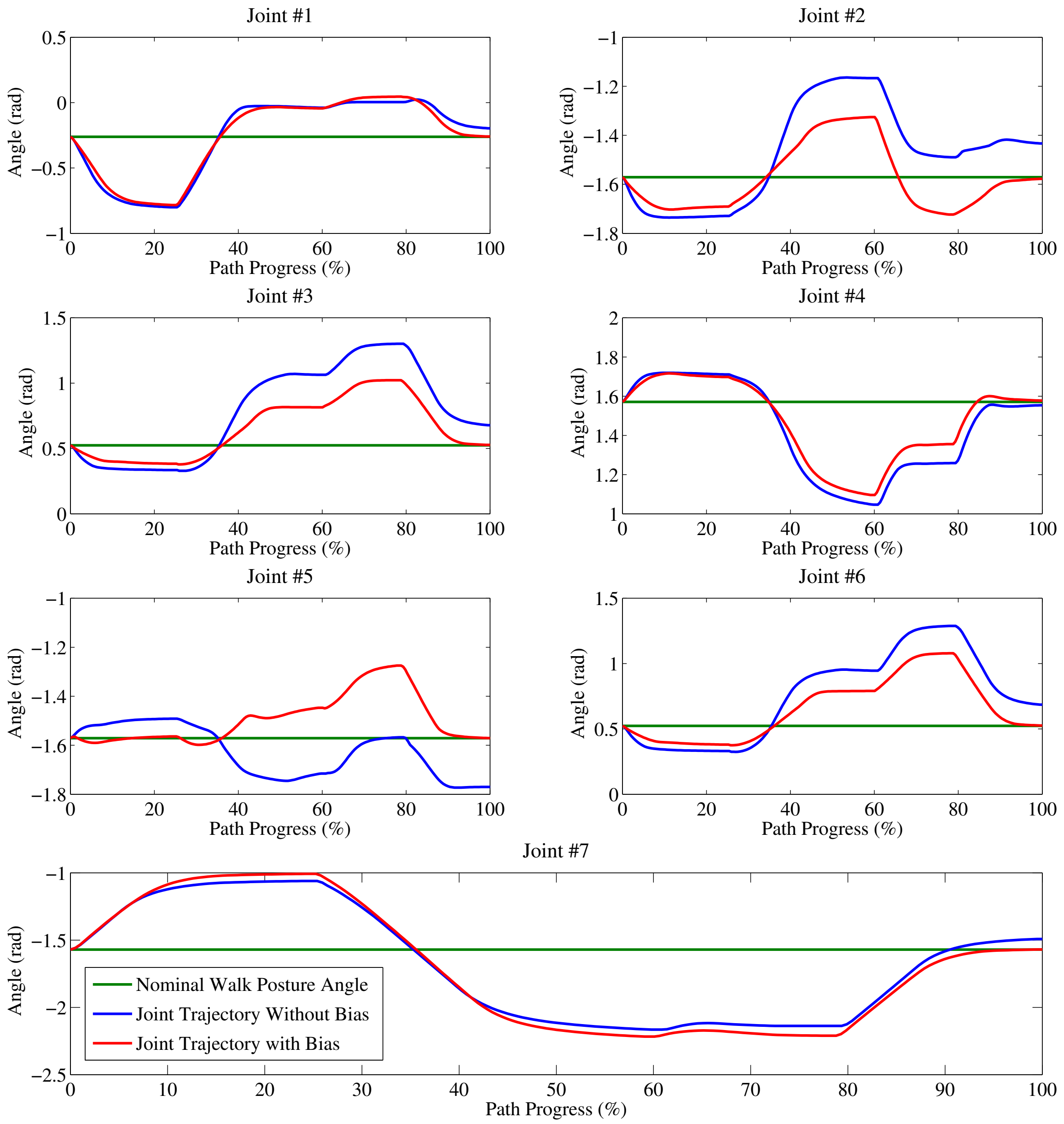
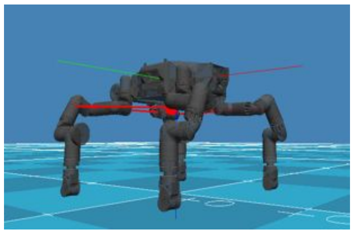
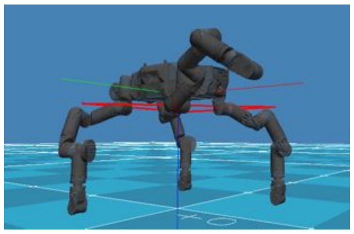


Figure 4.12: Comparison of Front-Left Limb Joint Trajectories with and without biasing to nominal posture as the robot moves between the configurations shown in 4.8. Angles are plotted against percentage of path completion for ease of comparison.


(a)


(b)

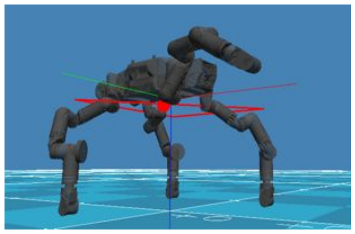

(c)

Figure 4.13: (a) Initial Configuration. (b) Final solution without limits. (c) Final solution with joint limits. The final hand pose is the same, but the robot configurations are slightly different – the joint-limited final configuration raises the body more than in the unrestricted case in order to compensate for the limited joint range.

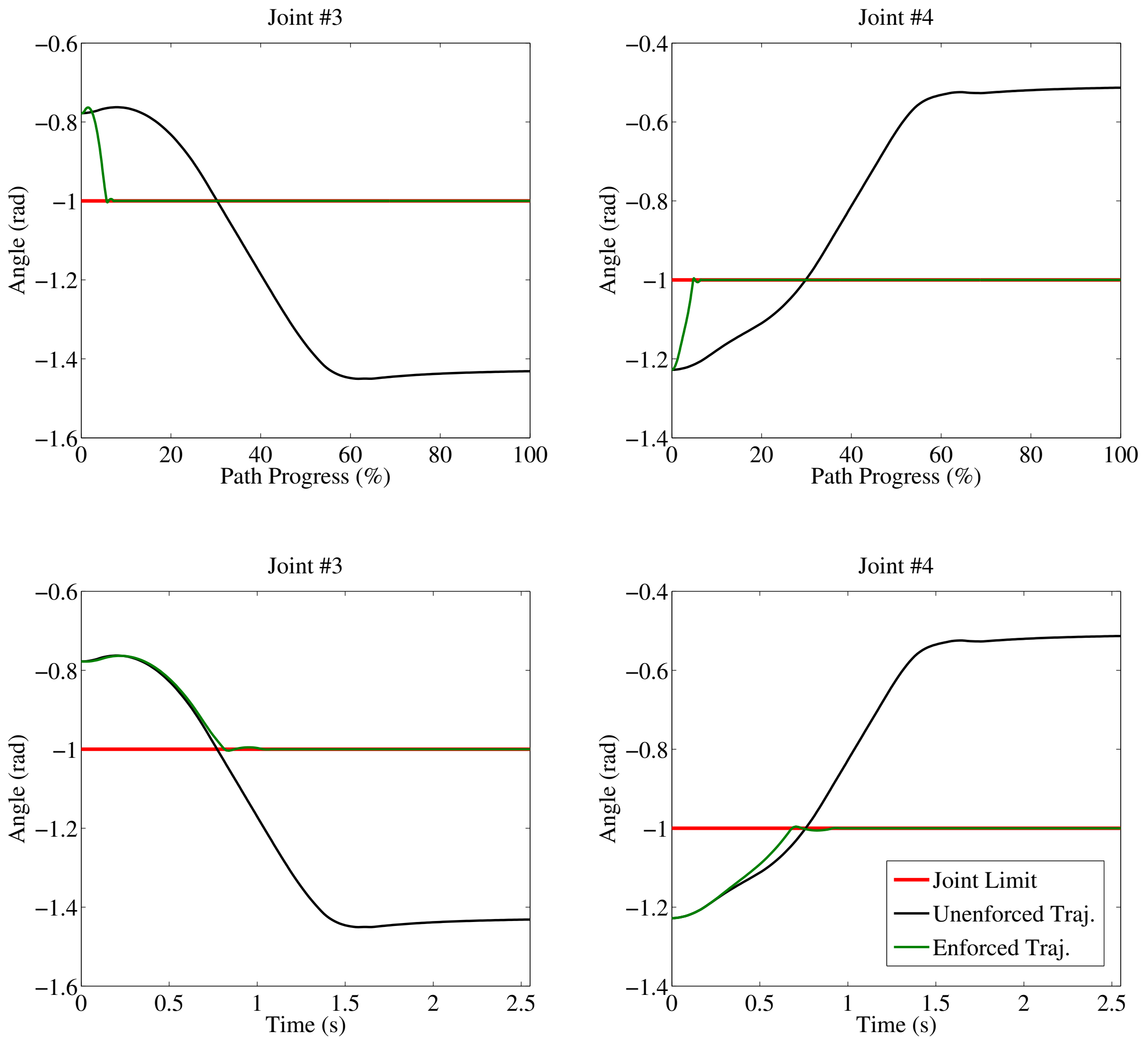


Figure 4.14: Comparison of front right limb joint trajectories with and without joint limits. Angles are plotted against percentage of path completion as well as against time. The trajectory with enforced joint limits takes longer than the one no joint limits; however, they are plotted on the same time axes to show that the trajectories are the same until joints 3 and 4 approach their respective joint limits.

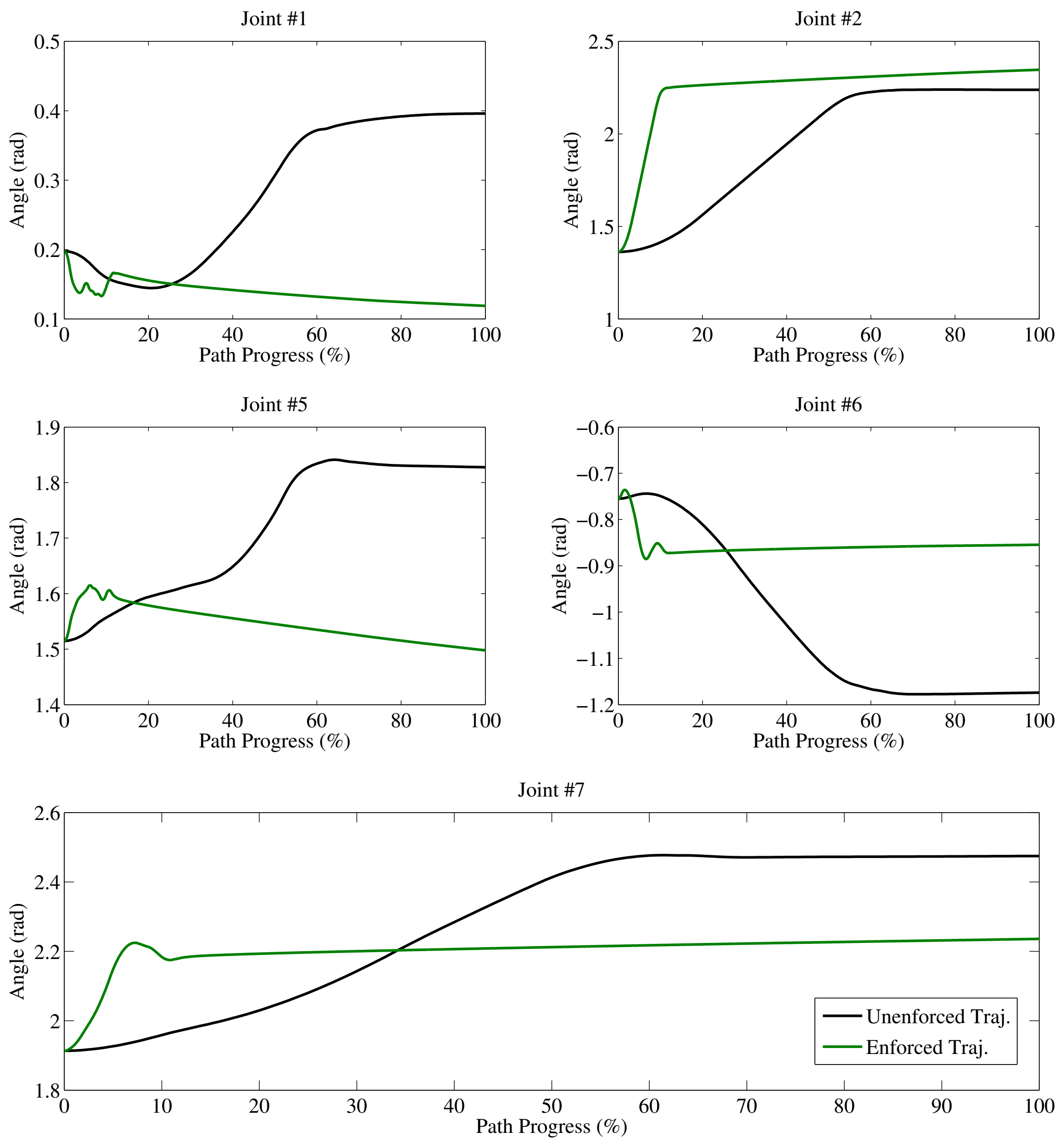


Figure 4.15: Comparison of front right limb joint trajectories with and without joint limits. Angles are plotted against percentage of path completion for ease of comparison.

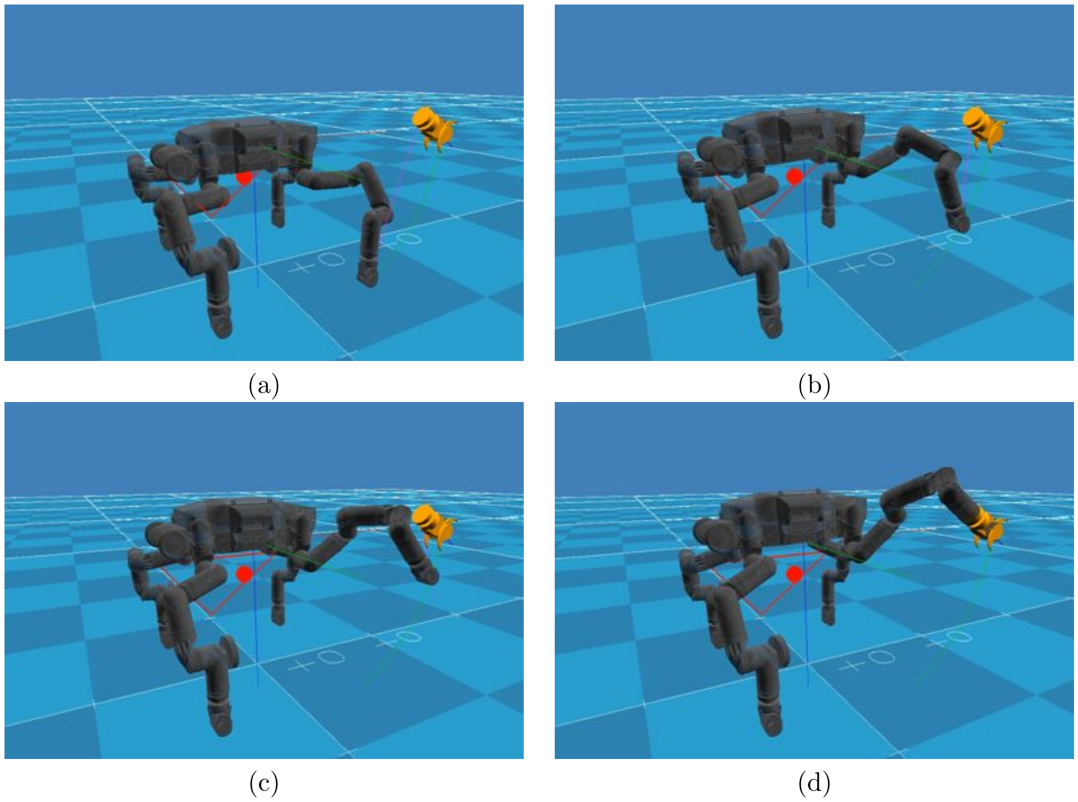


Figure 4.16: Robosimian raises its Front Right Limb. This motion is used to show a comparison between actual and planned trajectories, as well as speed of computation. The desired hand end-effector position and orientation are shown in yellow.

The goals in the objective included the end-effector goal, a body-motion goal (penalty on moving the body), and a regularizer. The weight on the motion of the end-effector was exactly

$$P_{\mathcal{E}} = J_{\mathcal{R}}(\Theta, x)^T Ad^T_{g_{\mathcal{B}\mathcal{E}}} \; I_{6\times 6} \; Ad_{g_{\mathcal{B}\mathcal{E}}} J_{\mathcal{R}}(\Theta, x),$$

and the body was encouraged to keep still, with a penalty on body velocity determined by

$$P_{\mathcal{B}} = J_S(\theta, x)^T \begin{bmatrix} 0.005 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.005 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.005 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} J_S(\theta, x),$$

and the regularizer was given by

$$P_\theta = 0.001 \;\times\; I_{28\times 28}.$$

In terms of these weights, $P = P_\theta + P_\mathcal{B} + P_\mathcal{E}$. Solving the QP took on average 550 microseconds, while the full solution required a total of 140 iterations and 0.0771 seconds. The time-step was 0.015 seconds, and the full motion took 2.115 seconds.

The trajectory is executed using trapezoidal velocity profiles that are fitted to the generated waypoints and then tracked by joint-level position controllers. Planned and experimental joint angles are shown in Figures 4.17 - 4.20, while planned and actual joint velocities are shown in Figures 4.21 - 4.24. Tracking is good but not perfect – the experimental trajectory does deviate from the planned trajectory a little bit (see figures 4.25 and 4.26).

Figure 4.27 shows experimental and actual distances from the center of mass to each of the edges of the support region. Figure 4.28 shows computation time and error convergence as a function of the iterations of Algorithm 1 in the manuscript.

This thesis includes a video supplement showing many other demonstrations of the theories developed in this chapter as well as Chapter 3 (see Appendix B). The video includes a number of simulated and experimental examples performed using the RoboSimian and Surrogate robots. It is available at `http://resolver.caltech.edu/CaltechAUTHORS:20160316-202027429`.

4.9 Summary

This Chapter provides a simple and general framework to apply the kinematics of the previous chapter to planning and controlling the motions of a redundant mobile robot. A local planning problem that extends and generalizes the classical techniques of redundancy resolution is formulated, and called the 'Balanced Priority Solution'. This problem trades-off motion towards a manipulation goal, margin of stability, and magnitude of motion. The solution is interpreted geometrically, and its existence and uniqueness are characterized.

The local problem is extended to include inequality constraints that can be used to incorporate a large variety of safety-critical constraints. Linear programs are formulated to generate these inequality constraints, and to certify a configurations safety or lack thereof.

All of the local problems that are formulated are used in an iterative algorithm, in order to generate plans or trajectories for a given task. A number of examples, both experimental and simulated, are provided on the RoboSimian and Surrogate Robots.

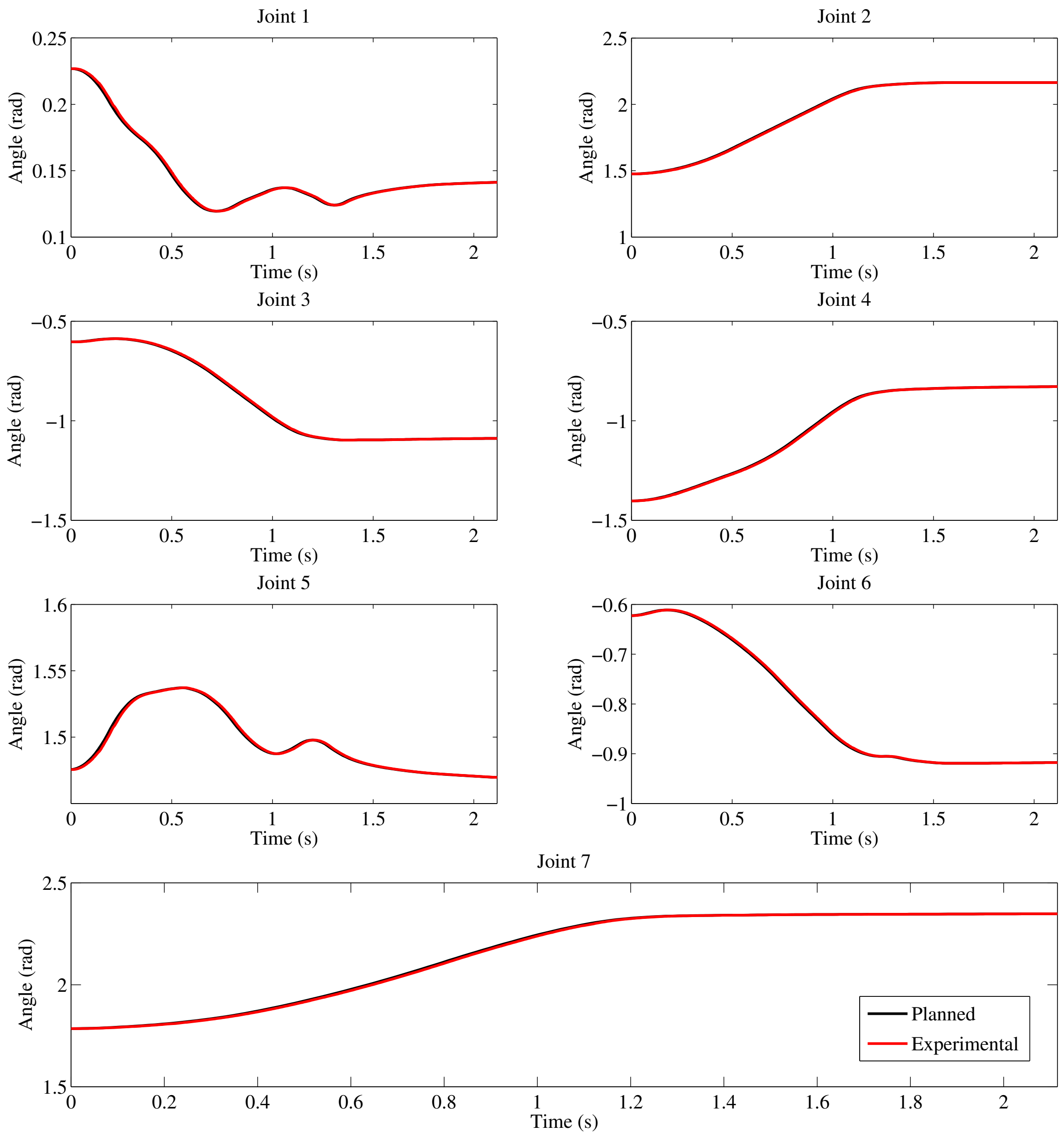


Figure 4.17: Planned and experimentally measured joint angles in the front right limb, corresponding to the example in Figure 4.16.

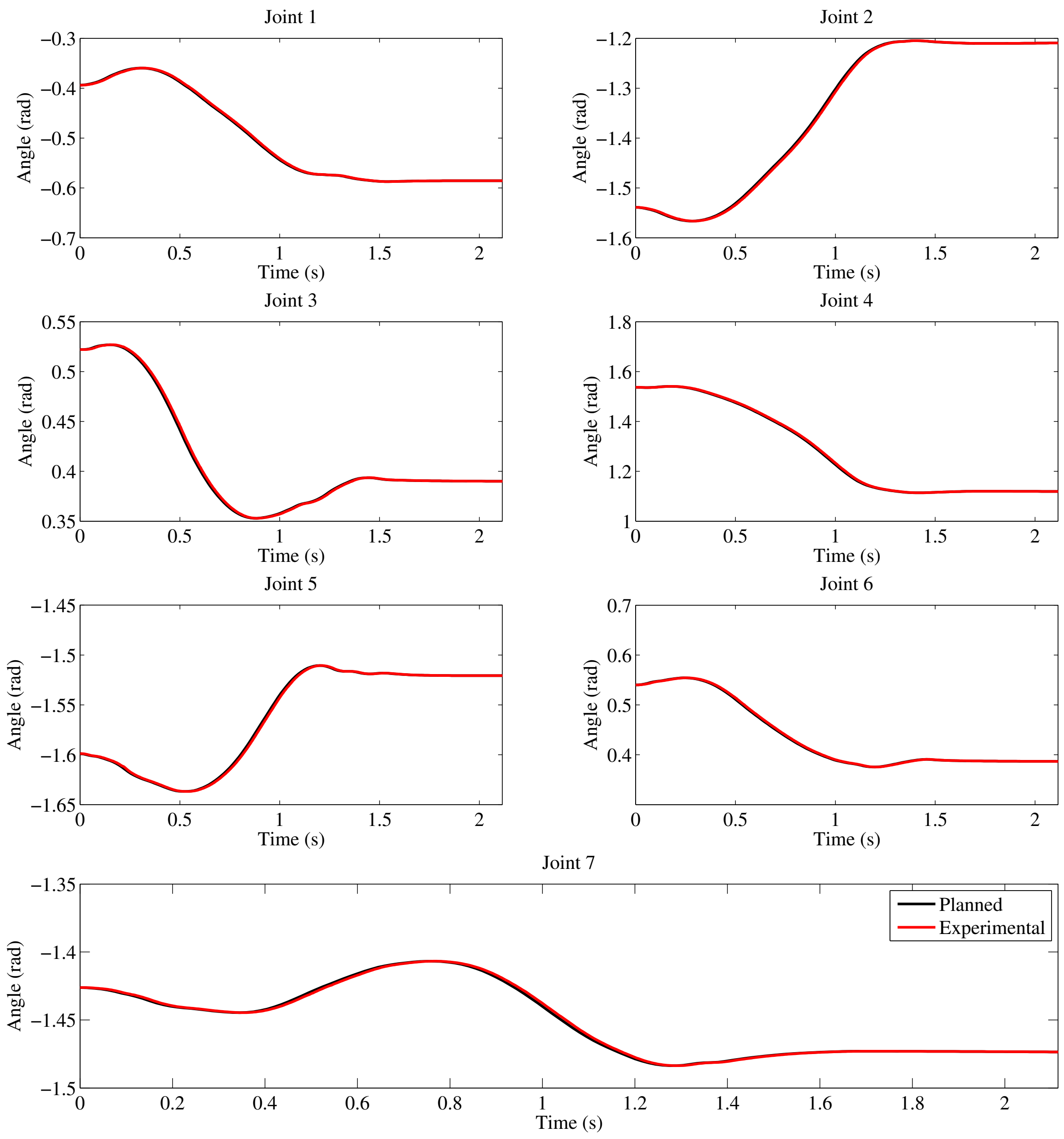


Figure 4.18: Planned and experimentally measured joint angles in the back right limb, corresponding to the example in Figure 4.16.

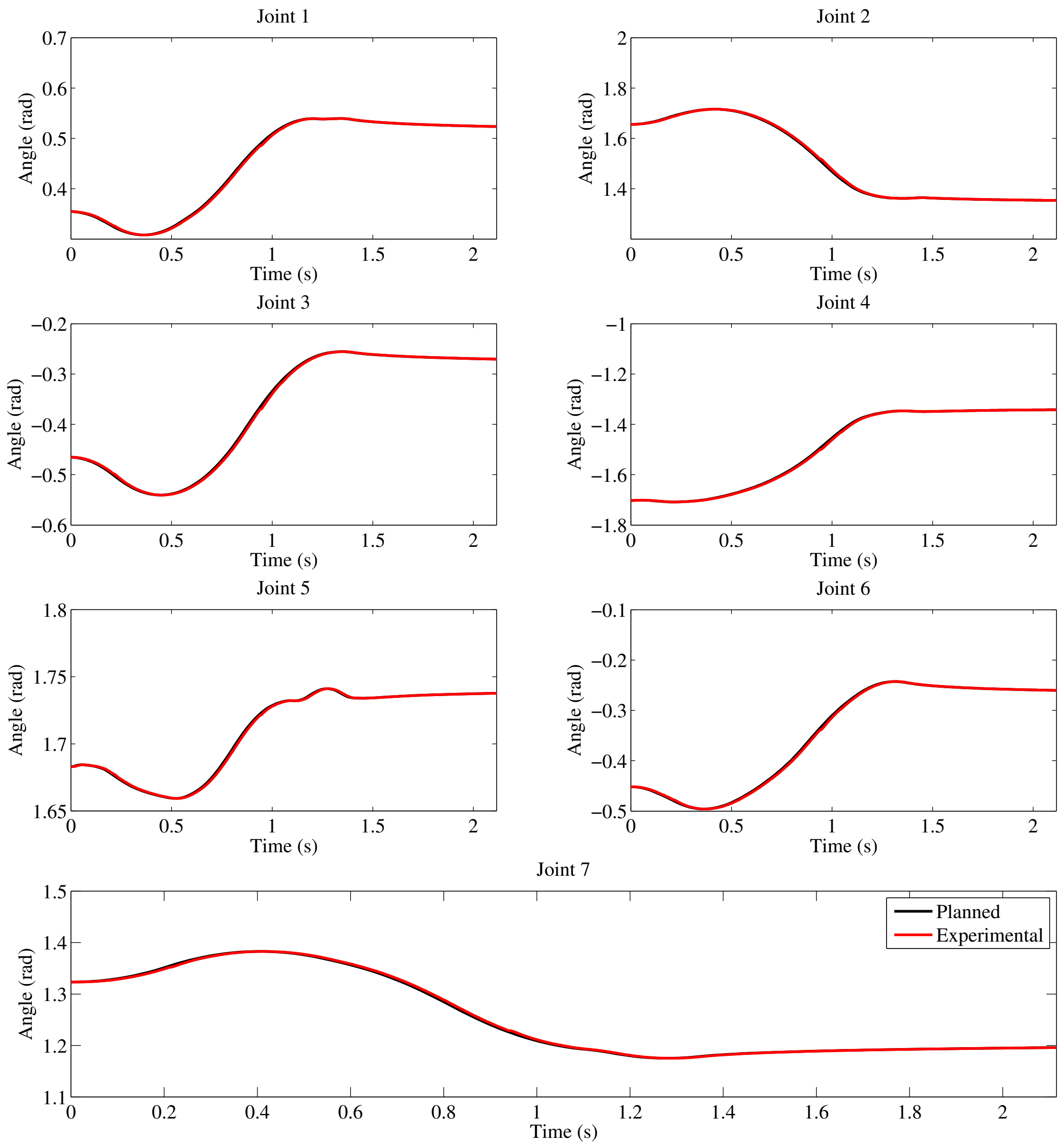


Figure 4.19: Planned and experimentally measured joint A=angles in the back left limb, corresponding to the example in Figure 4.16.

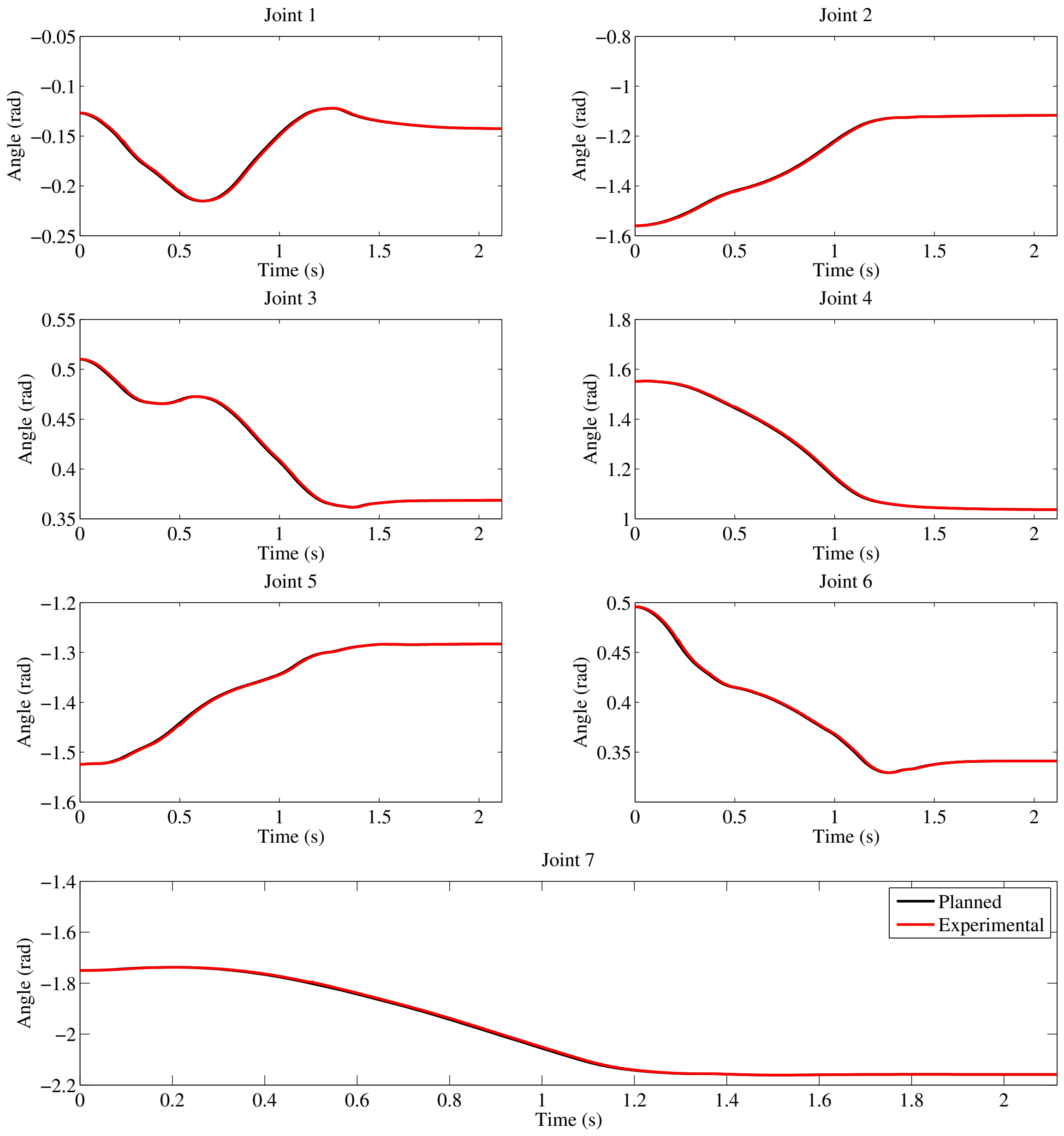


Figure 4.20: Planned and experimentally measured joint angles in the front left limb, corresponding to the example in Figure 4.16.

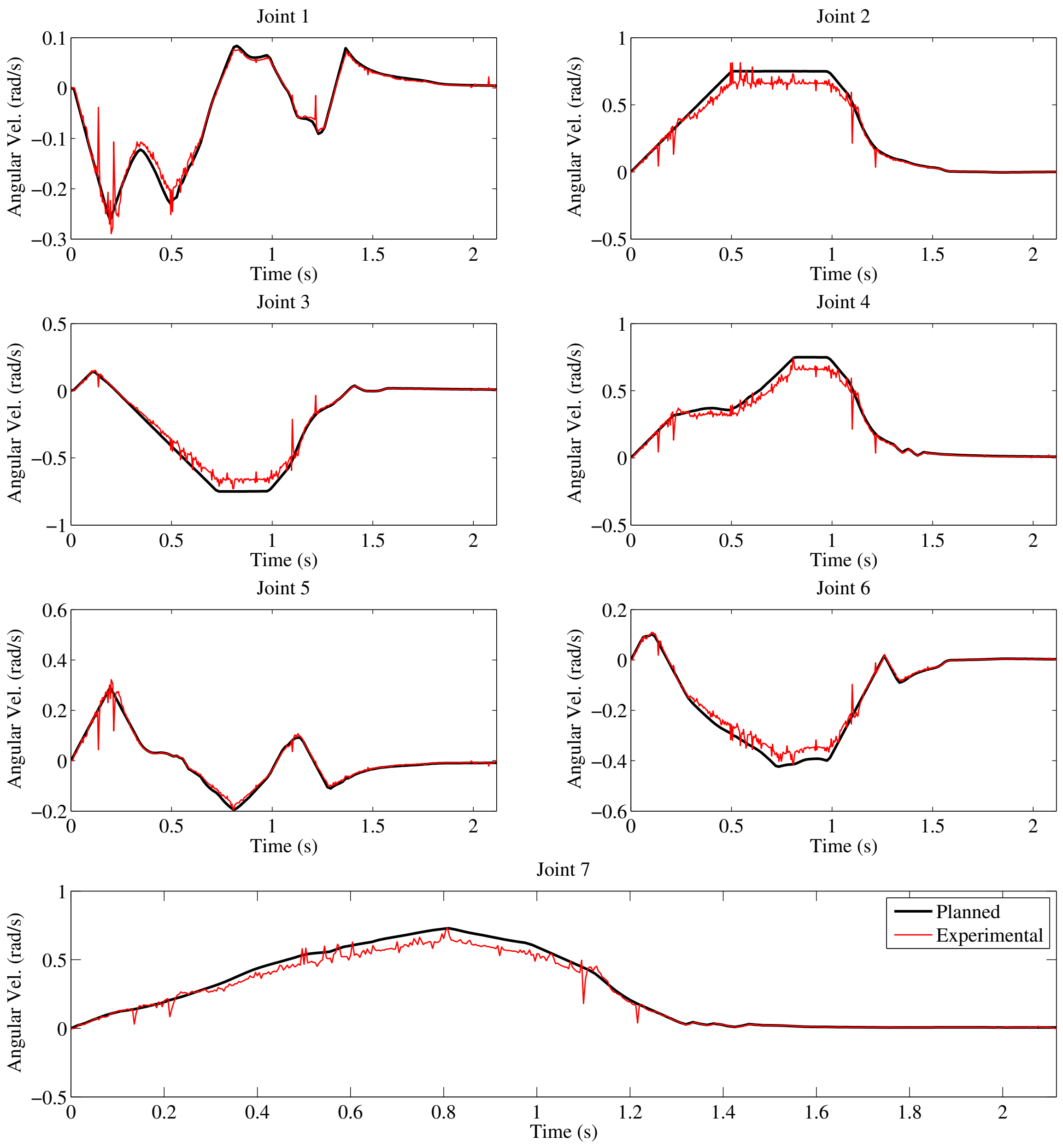


Figure 4.21: Planned and experimentally measured joint velocities in the front right limb, corresponding to the example in Figure 4.16.

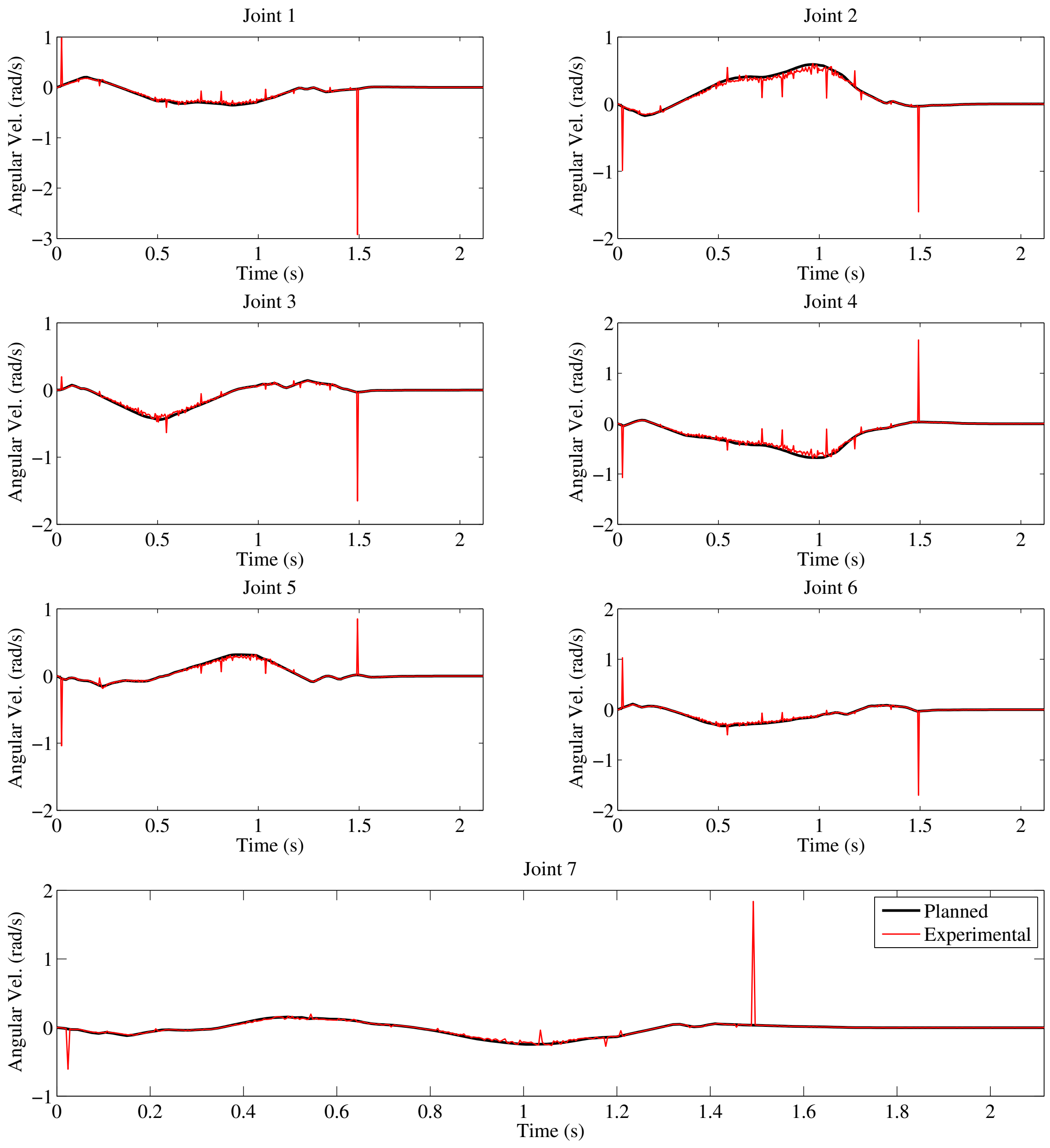


Figure 4.22: Planned and experimentally measured joint velocities in the back right limb, corresponding to the example in Figure 4.16.

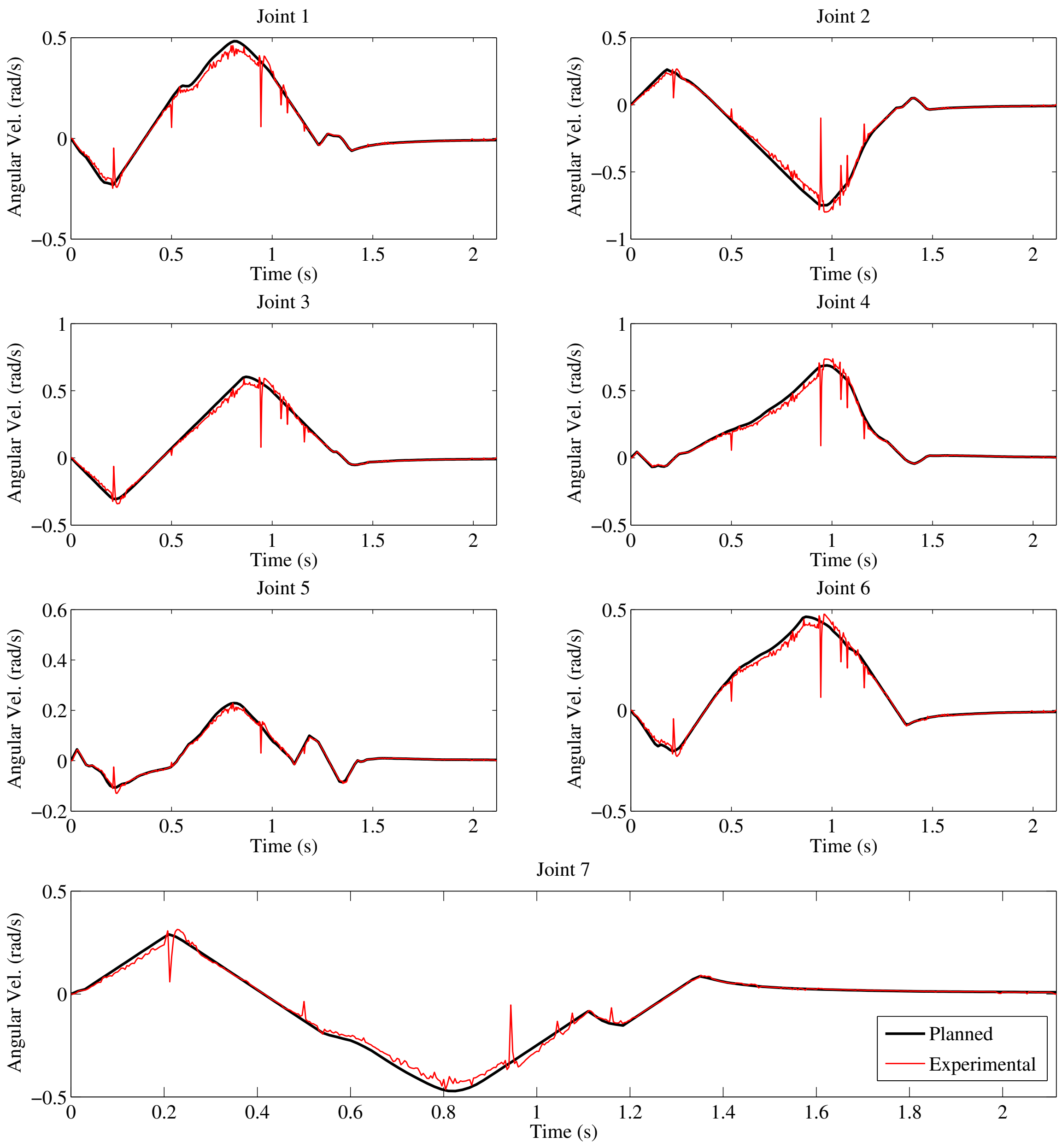


Figure 4.23: Planned and experimentally measured joint velocities in the back left limb, corresponding to the example in Figure 4.16.

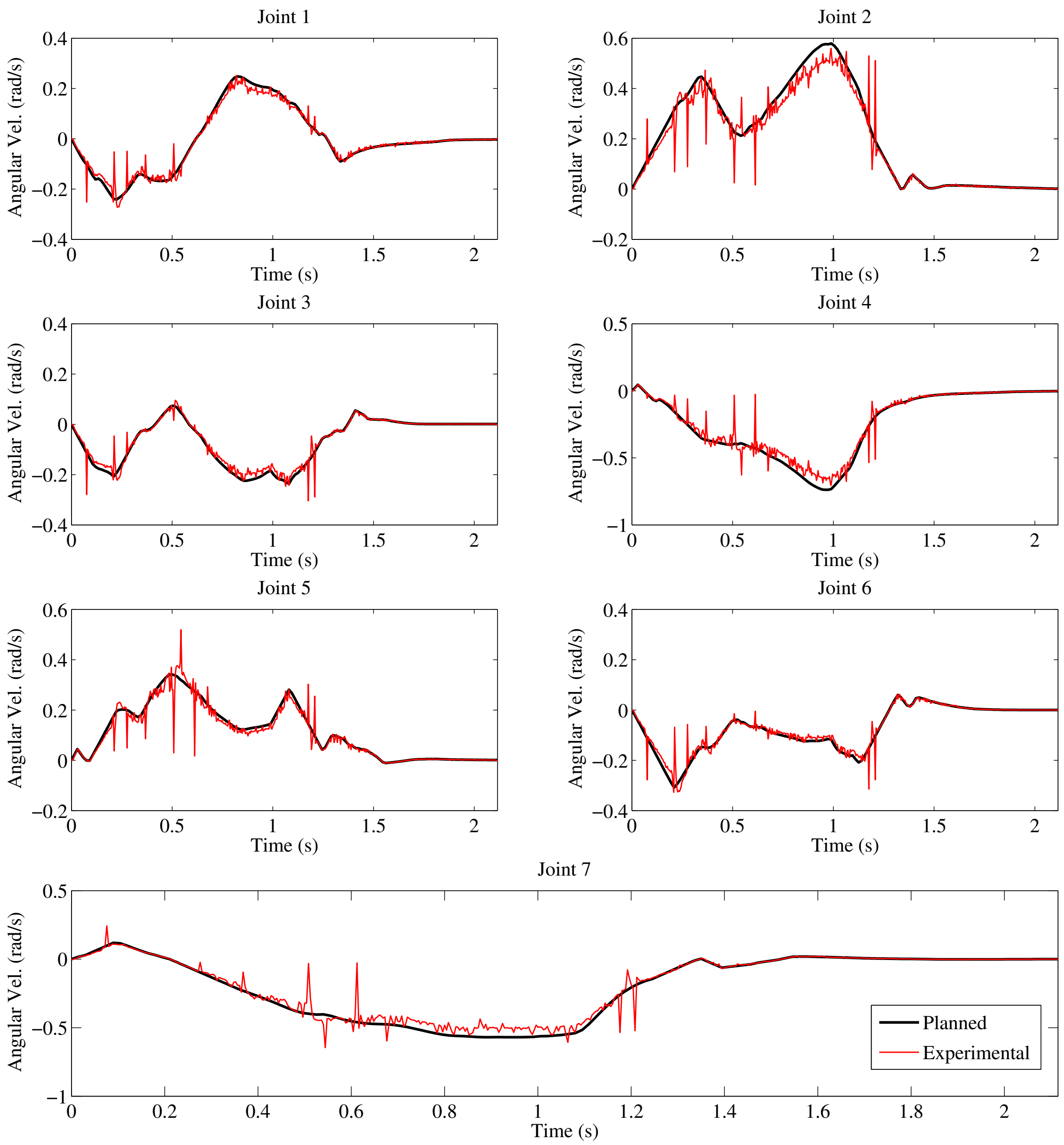


Figure 4.24: Planned and experimentally measured joint velocities in the front left limb, corresponding to the example in Figure 4.16.

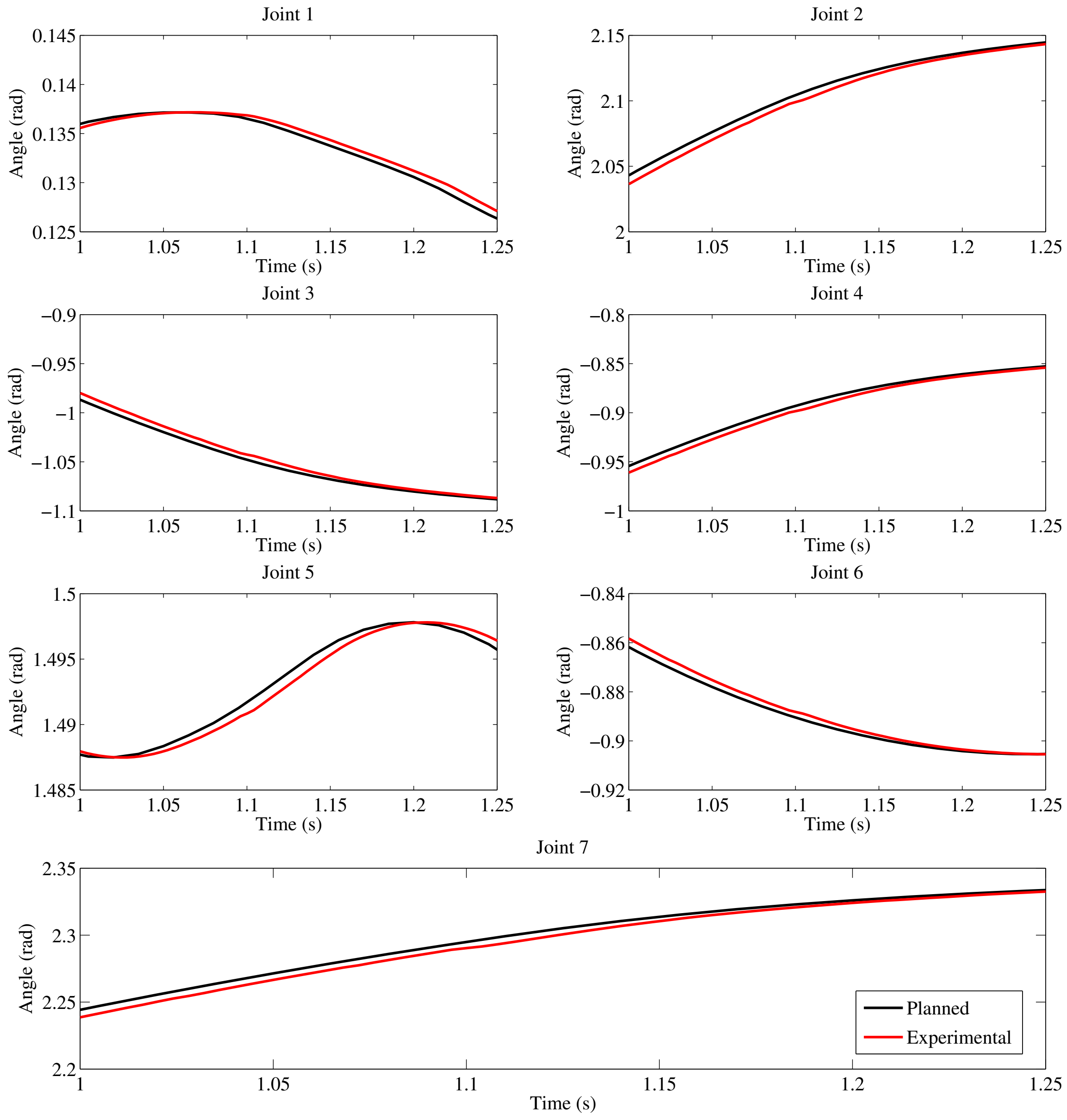


Figure 4.25: A one quarter-second close-up of the joint positions in the front right limb, to demonstrate the difference between executed and actual trajectories for the motion associated with Figure 4.16. 4.16.

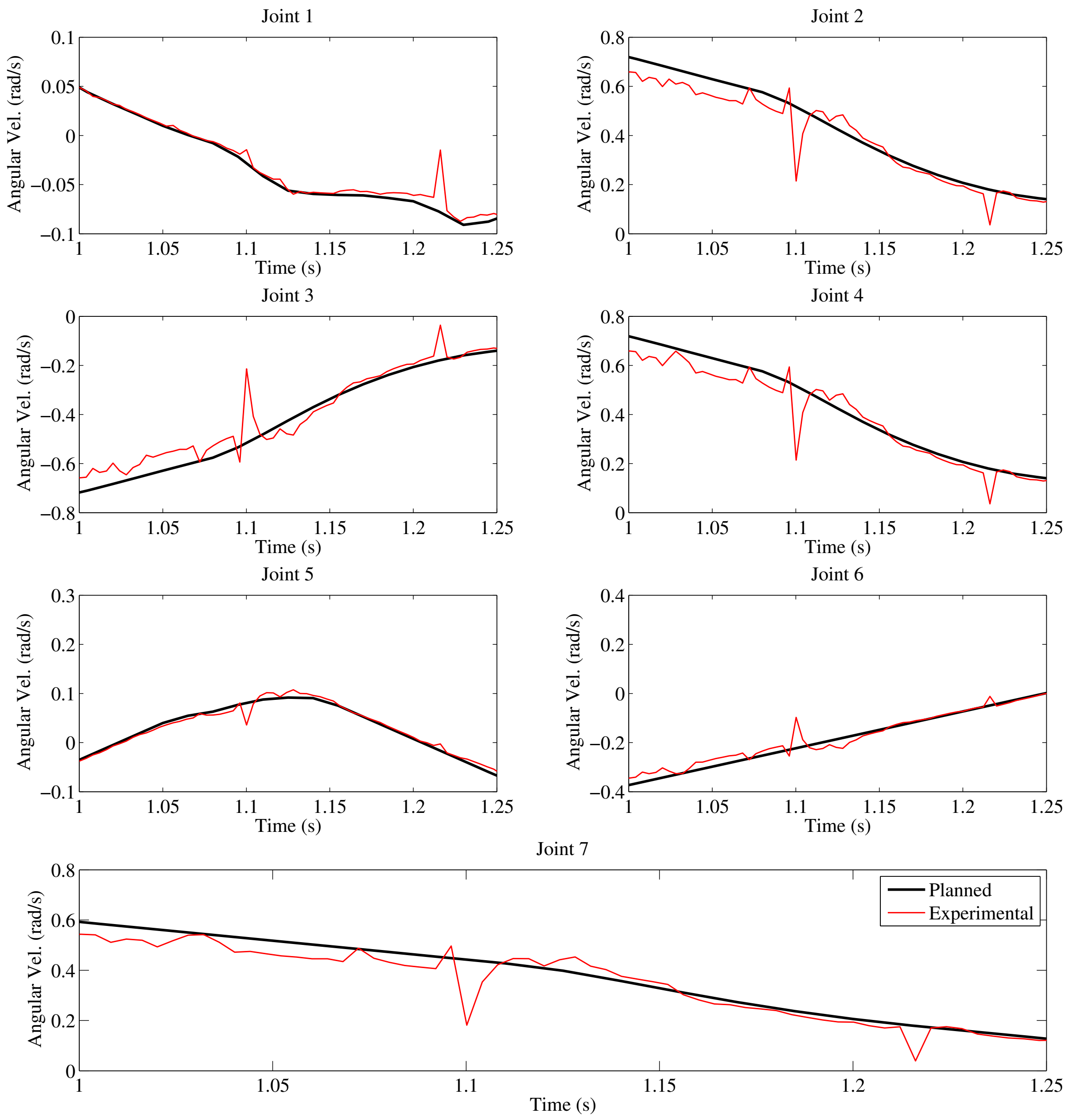


Figure 4.26: A quarter-second close-up of the joint velocities in the front right limb, to demonstrate the difference between executed and actual trajectories for the motion associated with Figure 4.16.

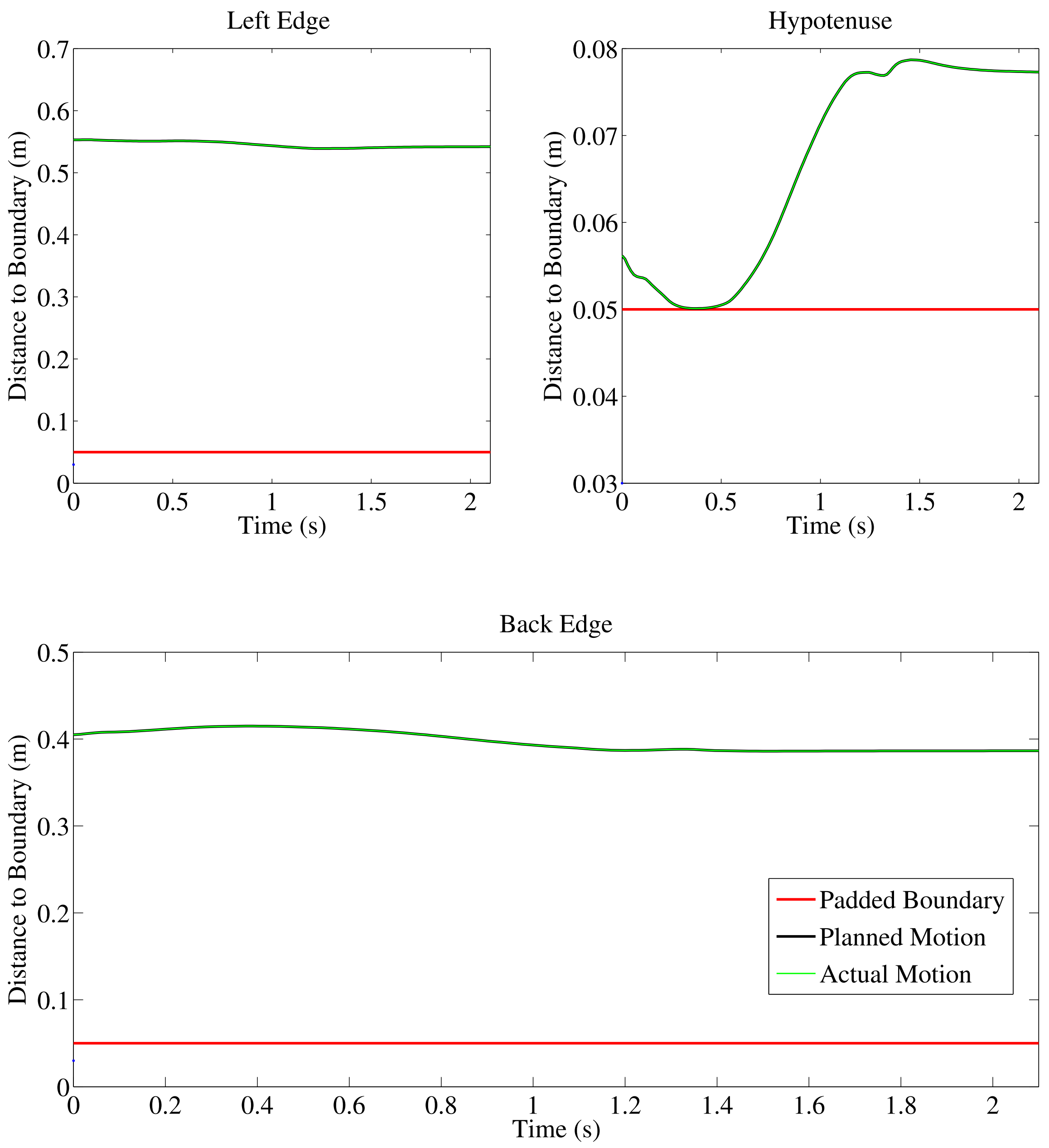


Figure 4.27: Balance Data for both planned and experimental motions associated with the reach in Figure 4.16

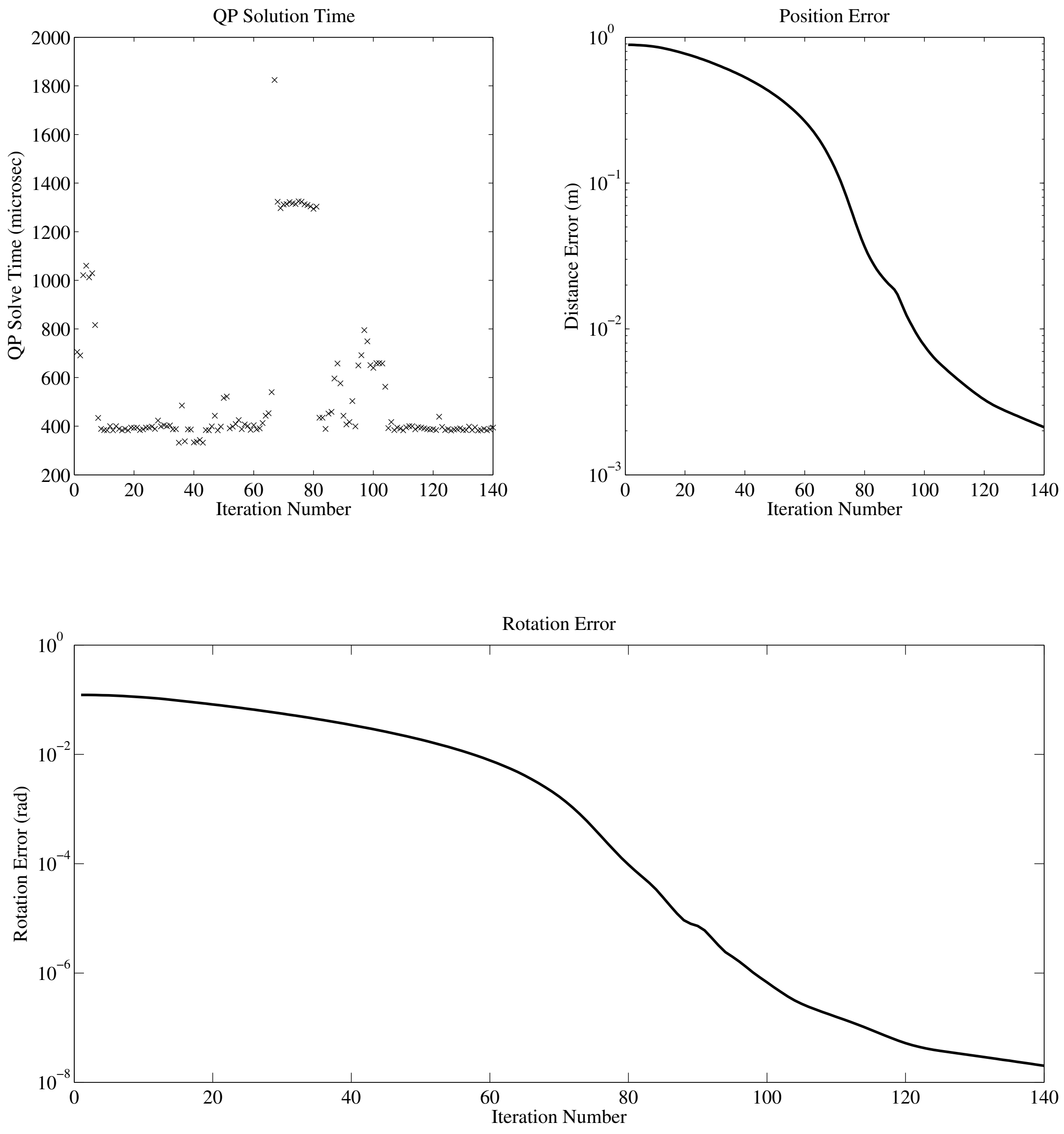


Figure 4.28: Computation time and Error as a function of iteration number, for generating the motion shown in Figure 4.16. The 'error' in rotation is given by $e_{\text{rot}} = 1 - \langle q_{\text{current}}, q_{\text{desired}} \rangle^2$, where q_{current} is the quaternion associated with the current orientation, and q_{desired} is the quaternion associated with the orientation of the desired configuration. There is no metric on rotations; however, e_{rot} is positive definite and represents the 'error angle' between current and desired poses.

Chapter 5

Combined Force and Velocity Kinematics and Local Planning for Multi-limbed Robots

5.1 Introduction

This chapter considers how a multi-limbed robot can carry out manipulation tasks involving simultaneous control of compatible position and force goals while maintaining a stable quasi-static stance in the presence of gravity. A particular motivation for this work comes from experiences with using the RoboSimian robot (see Fig. 5.1) to compete in the DARPA Robotics Challenge (DRC). Each of RoboSimian's four limbs incorporate seven actuated revolute joints and each limb has at its distal end a hand with three retractable fingers. Each limb can either be used as a leg to enable locomotion, or as a limb to carry out manipulation tasks, aided by the multi-fingered hands.

Some of the DRC tasks, such as turning a water valve handle mounted on a wall (see Fig. 5.1), require the vehicle to simultaneously execute a complex manipulation task involving both position and force control of the end-effector on one limb, while also maintaining stance stability throughout the task execution using the other limbs. In the valve turning example, the manipulating limb must generate a significant torque about the valve's rotational axis, control the limb and palm (in the example shown, no hand is mounted) to apply an inward-normal force to maintain frictional contact, and compatibly track the motion of the grasped valve handle as it rotates. All of the actuators of the entire body may need to be coordinated, not just those of the manipulating limb, to generate the needed forces. Additionally the end-effector forces generated during the task must not destabilize the robot's stance.

This chapter introduces the kinematics and methods to implement complex tasks of this sort, involving both end-effector task and motion goals, as well as stance stability. Moreover, the control algorithm is constructed through a novel formulation whose locally optimal solution can be reduced to a very efficient convex program. This approach allows implementation of real-time solutions on robots having the complexity of RoboSimian. The underlying formalism of this approach also allows the development of new definitions and new criteria for the feasibility of quasi-static stances under the influence of manipulation forces.

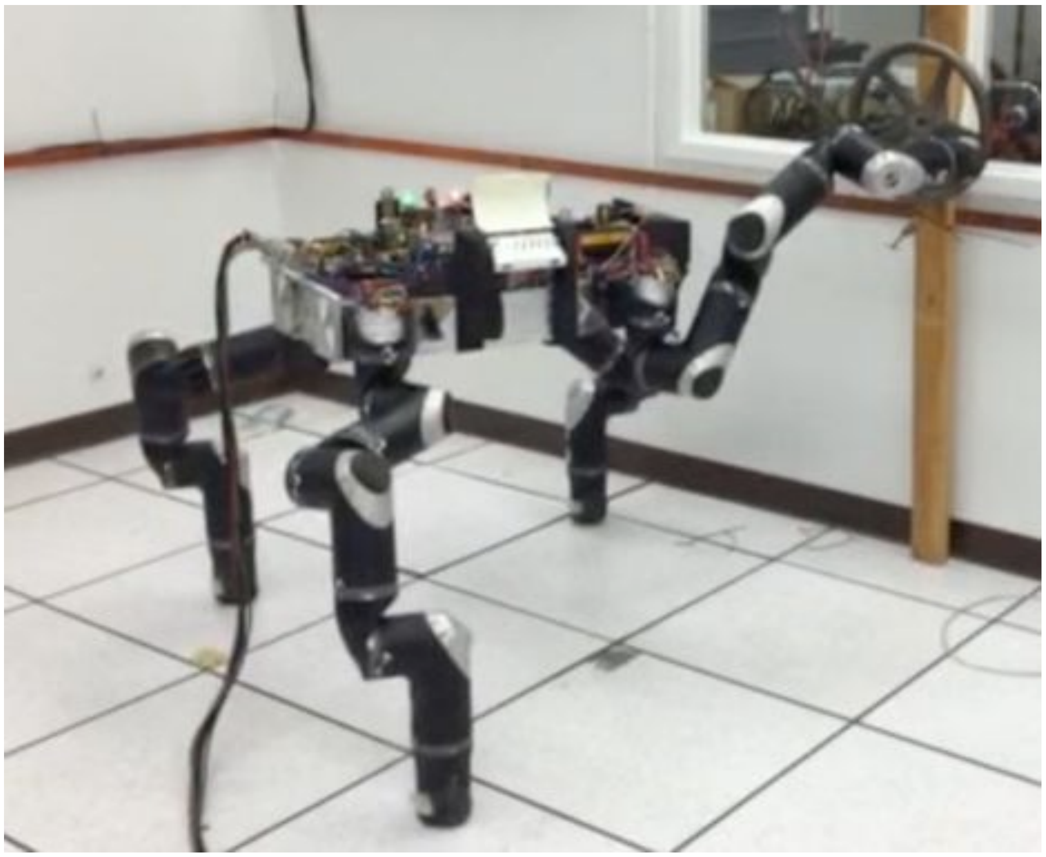


Figure 5.1: Photograph of the Four-Limbed RoboSimian Robot used by the JPL/Caltech team in the DARPA Robotics Challenge. This snapshot shows RoboSimian turning a circular valve on the vertical wall in the background using one of its limbs, while maintaining quasi-static stance stability using its other three limbs. Because the robot did not have hands mounted at the time, it needed to maintain a normal inward force, while moving around the axis of the valve.

5.1.1 Relation to Prior Work

The problem addressed in this chapter is closely related to several important and historical problems in the field of robotics. There is a vast prior literature on simultaneous control of end-effector force and position during manipulation tasks using a fixed-based robot manipulator [98]. Numerous frameworks for this problem have been introduced, such as hybrid position-force control [71], impedance control [29], stiffness control [76, 77], and operational space control [40]. Each of these approaches in turn has a rich literature concerning the design, analysis, and control of manipulation tasks within the respective framework. The method put forth in this chapter can be seen as a form of stiffness or admittance control for multi-limbed robots that also incorporates stance stability criteria into the formulation. This chapter includes a novel Quadratic Programming (QP) approach to formulate the problem. It has been shown in previous work [87, 88] that such a framework can very efficiently solve the complex kinematic coordination problems that arise in multi-limbed robot coordination tasks.

The approach taken in this chapter can also be considered as a form of *whole body manipulation* [105] and *mobile manipulation* [9,41,101] for the particular class of quasi-statically stable legged robots. As opposed to fixed based manipulators, where stability is not an issue, or wheeled-based manipulators where stability analysis is straightforward, manipulation by multi-limbed walking robots must also incorporate stance stability into the overall manipulation planning process. New, explicit results are provided on quasi-static stability for the class of problems defined above.

While this chapter considers quasistically stable robots (with $n \geq 3$ legs on the ground at all times), humanoid robots must also contend with manipulation tasks while maintaining postural stability, and previous studies have analyzed and developed techniques for bipedal robots to carry out manipulation [19, 24, 32]. Theoretical advances in Convex Programming, and the associated introduction of efficient numerical optimization codes, allow the proposal of new approaches which have not only serious computational speed advantages, but also allow added flexibility and generality in specifying the task objectives. The present work is not the first to propose the use of Convex Optimization or Quadratic Programming techniques for local motion planning of highly articulated mechanisms. For a review of related work, see [88] and Chapter 4 of this thesis. Dynamic effects are not considered in this chapter. However, many (e.g. [44, 93]) have obtained controllers for full dynamic models of humanoids with contact from simple convex QPs, for balancing and walking.

5.1.2 Structure of the Chapter

Section 5.2 summarizes the motion-force task compatibility and stiffness-compliance modeling assumptions that underlie the proposed method. Section 5.3 introduces a optimization-based framework for task control in the context of a fixed base serial chain manipulator mechanism. Section 5.4 extends the framework of Sections 5.2 and the kinematics introduced in Chapter 3 to the multi-limbed robot case, while incorporating quasi-static stance constraints. A simple planar example is presented in section 5.5. The chapter concludes in 5.6.

5.2 Task Compatibility and Compliance

The section describes a procedure to encode *simultaneously feasible* force and velocity goals. This formulation adopts the invariant form [80] of the classical Raibert/Craig hybrid position-force control [71] framework, and combines it with an assumed compliance model of the environment.

It is assumed that a set of constraint or task wrenches to be controlled by application of end-effector forces are given. These task wrenches are assumed to lie in a wrench space, W_{task}, having integer dimension, N_W:

$$W_{task} = \text{span}\begin{bmatrix} W_1 & \cdots & W_{N_W} \end{bmatrix} \triangleq B_W \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_{N_W} \end{bmatrix}, \tag{5.1}$$

for all $\omega_i \in \mathbb{R}$, $i = 1, \ldots, N_w$. The vector W_i is the i^{th} basis element for the wrenches associated with an l-dimensional task space ($l = 3$ for planar tasks, and $l = 6$ for spatial tasks). It is assumed that $N_w \leq l$. The matrix B_W is termed the *wrench basis matrix* for the task, as it defines the space of wrenches to be controlled.

The velocity (or infinitesimal displacement) task goals must reside in a *compatible* N_V-dimensional space of velocities [58]:

$$V_{task} = \text{span}\begin{bmatrix} \xi_1 & \cdots & \xi_{N_V} \end{bmatrix} = D_V \begin{bmatrix} v_1 \\ \vdots \\ v_{N_V} \end{bmatrix}, \tag{5.2}$$

where $N_V + N_W = l$. The matrix D_V is said to be the *twist basis* for the manipulation task. Each twist basis vector, ξ_j must have the following compatibility relationship with each element of the wrench basis [80]:

$$\xi_j \cdot W_i = 0$$

for all $i = 1, \ldots, N_W$ and $j = 1, \ldots, N_V$. When the twist and wrench basis elements are represented in screw coordinates, the associated screws must be *reciprocal* to each other. Whenever this compatibility is violated, either the goals are *antagonistic* (for some $i, j,\ \ W_i \cdot V_j < 0$) or *infeasible* (for some i, j, $W_i \cdot V_j > 0$) [64].

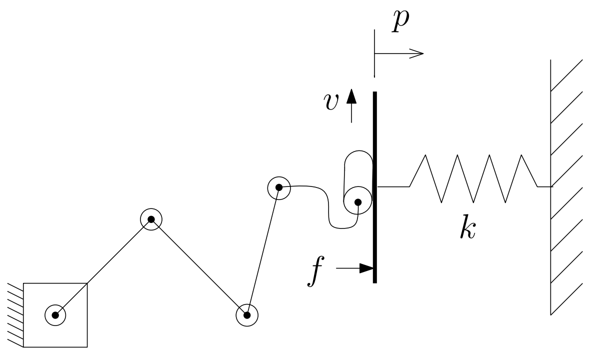


Figure 5.2: A Simple Model for Compliance

In general, a wrench basis B maps a contact wrench (f) to a full wrenches F as $F = Bf$, and a twist basis D maps a contact velocity v to a full twist V as $V = Dv$.

Next, a specific relationship between forces and motions is adopted when the end-effector is in contact with its environment. Initially a simple compliance model is assumed that relates task wrenches at the end-effector to velocities (or infinitesimal displacements) of the end-effector along directions defined by the wrench basis.

For simplicity, first suppose that the robot interacts with a rigid surface that in turn is attached to a spring, as shown in Fig. 5.2. The robot contacts the surface at position p in the world frame, and the robot applies a force f along the normal to the rigid surface. Assuming the spring is linear, the force applied by the robot and position of the contact satisfy Hooke's law,

$$f = kp,$$

where p represents displacement from some initial contact position given by $p(0) = 0$. To establish and maintain a constant task force f_d, the manipulator must move the contact to $p = \frac{1}{k} f_d$.

In practice, one may want to apply an arbitrary wrench (i.e., a combination of forces and torques, and not a simple Cartesian force) at the contact. In order to extend this very simple linear compliance to handle this more general case, recall that Poinsot's theorem states that every wrench can be interpreted as a 'screw' force – the combination of linear force along an axis and a torque about the same axis. A'screw-spring' relationship is hypothesized between the movement of the end-effector contact along a screw axis to the wrench produced by this movement (as illustrated by Fig. 5.3). Let ξ (a unit twist) describe a particular screw displacement. Then the contact wrench, F, as a function of end-effector position p along this screw is given by

$$F = (kp)\xi \ .$$

Let the initial wrench at the end-effector be given by F_0, corresponding to position $p_0 = 0$, and

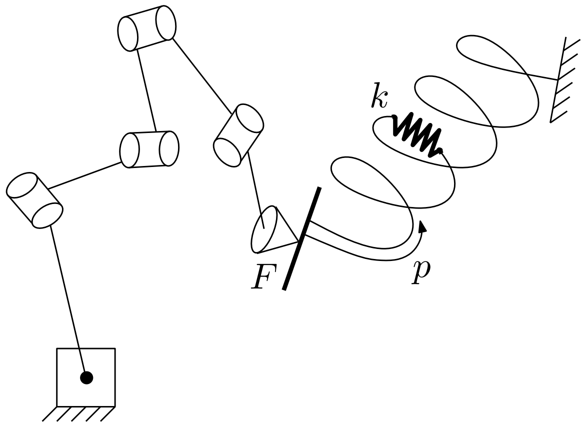


Figure 5.3: Extending the Model to 3D

suppose the desired wrench is F_d. The change in the wrench to be applied by the robot is $\Delta F = F_0 - F_d$. With this assumption, $p_d\xi = -\frac{1}{k}\Delta F$; the desired wrench can be realized by moving the contact point along $p_d\xi$.

More generally, suppose one can compute

$$\Delta F = F_{\text{meas}} - F_d,$$

where F_{meas} is the measured wrench, during execution. Then, the desired wrench can be achieved by moving the end-effector with velocity

$$V = -\frac{\mu}{k}\Delta F,$$

where μ is an arbitrary proportional gain. For convenience, set $\mu = 1$ henceforth.

To extend this principle to the case of multiple contact wrenches, suppose that a manipulator is required to achieve a task wrench W_{task} lying in the span of $B_W = \begin{bmatrix} W_1 \ldots W_{N_W} \end{bmatrix}$. Since each of the W_i are associated with a different contact, they may each be endowed with a different stiffness

$k_i > 0$. Define a *stiffness matrix* to have the form

$$K = \begin{bmatrix} k_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & k_{N_w} \end{bmatrix}. \tag{5.3}$$

In general, the only requirement on a stiffness matrix K is that it is positive semi-definite. However, the wrench basis can always be selected so that it is diagonal and positive as in (5.3).

Now, suppose that the end-effector moves along a screw ξ. Let

$$p_i = (\xi^T W_i)p$$

represent the displacement along the direction of W_i. Then, the net contact wrench is

$$F = \sum_{i=1}^{N_W} k_i p_i W_i = pKB_W B_W^T \xi.$$

Again, letting $\Delta F = F_{\text{meas}} - W_{\text{task}}$, the desired wrench can be achieved by moving with a velocity V satisfying

$$B_W K B_W^T V = -\Delta F. \tag{5.4}$$

Eq. (5.4) is a general linear compliance relationship that allows for the understanding of how motions result in forces; going forward, this relationship is used in planning motion locally to achieve position and force goals. The relationship (5.4) is well known; for example, it is along the same lines as the compliance models in [79] and [76].

There are clear shortcomings with this model for the environment.

1. The model assumes a linear elastic response of the environment. However, non-linear stiffness or compliance models are readily incorporated via linearization of the elastic model [1, 79] to construct a configuration dependent stiffness matrix.

2. In most situations, the stiffness of the contact is not a priori known accurately. In fact, very poor assumptions about the environment's compliance can lead to catastrophic results. To ensure safety, the assumed contact stiffness should be assumed larger than the true contact stiffness. The price to pay for this conservative modeling approach is the potentially increased time needed to reach a desired contact wrench e.g., if the true stiffness is k and ones assumes it to be $\tilde{k}$, if one does not measure force, a force equal to $\frac{k}{\tilde{k}} f$ is applied. If an accurate

measurement, f_{cur}, of the force is available, the desired force can be regulated with a simple motion control law (e.g. law $\dot{p} = \frac{1}{k}(f_d - f_{cur})$ results in exponential asymptotic convergence to the desired force.)

The approach taken in this chapter is motivated by the following practical considerations in working with the RoboSimian vehicle, and many similar robots. To realize a very high strength to weight ratio, the motors in RoboSimian's legs are highly geared (using harmonic drives), preventing accurate control of joint torques. However, very fine control over joint displacements are possible, suggesting that a control design approach which models the relationship between joint motions and end-effector forces. Moreover, the distal tips of each RoboSimian limb are covered with a thick compliant rubbery pad, which can be accurately modeled using the simple methods of this section. In the language of impedance control, one must design an *admittance* for such systems.

5.3 Simultaneous Force and Position Control: Fixed Base Manipulator

To gain intuition for the problem to be solved with a multi-limbed robot, the simpler case of a fixed base manipulator is first considered, and the corresponding simplified problem. For a fixed base serial chain manipulator with the ability to measure end-effector wrenches, the goal is to implement a local planning and control system which solves the following word problem:

How does one move the Robot's joints so that

1. *The end effector moves in the direction of a pose goal along the subspace of allowed motions ?*
2. *The manipulator controls the applied wrenches along the allowed wrench subspace, based on a compliance model ?*
3. *The resulting manipulator motion is minimal when there are redundant degrees of freedom ?*

The solution to this intuitive word problem can be formalized as the following constrained minimization:

$$\begin{aligned} \min_{\dot\theta} \quad & \frac{\alpha}{2}\|v - v_d\|_2^2 + \frac{\beta}{2}\|\Delta f - (f_d - f_{\text{meas}})\|_2^2 + \frac{\gamma}{2}\|\dot\theta_2^2\| \\ \text{subject to} \quad & D_V^T V = v \\ & B_W^T V = K^{-1}\Delta f \\ & J(\theta)\dot\theta = V \end{aligned}$$

where α, β, and γ are positive weighting coefficients, B_W is a wrench basis that spans the space of wrenches that are to be applied by the end-effector, and D_V is a twist-basis whose range includes the compatible subspace of end-effector velocities. f_d and f_{meas} are the desired and measured wrenches in the basis B_W's coordinates. The quantity $(f_d - f_{meas})$ is the end-effector wrench error, expressed in the wrench basis, which should be controlled to zero by motions of the manipulator. The coefficients α and β trade off the relative importance of errors in the velocity and wrench goals, while γ weights the importance of using a minimal motion to solve the given problem. The first two constraints ensure the compatibility relations described in the last section, and it is required that B_W and D_V, when expressed in a common frame, satisfy the relationship $B_W^T D_V = 0$. The last constraint describes the manipulator's kinematic relationship between joint motions and end-effector velocities.

To simplify the form of the problem and its solution, define

$$\eta \triangleq \alpha D_V v_d + \beta B_W K^{-1}(f_d - f_{\text{meas}}),$$

which can be interpreted as the weighted instantaneous end-effector displacement which moves towards the task goals. With this definition, the constrained minimization problem can be converted into an unconstrained minimization problem:

$$\min_{\dot\theta} \quad \frac{1}{2}\dot\theta^T(P + \gamma I)\dot\theta - \eta^T(J\dot\theta), \tag{5.5}$$

where

$$P = J(\theta)^T(\alpha B_W^T B_W + \beta D_V^T D_V)J(\theta)\ .$$

It can be shown that a solution to (5.5) exists whenever α, β, γ are positive. When they are all positive, the solution at each instant is given by

$$\dot{\theta}^* = (P + \gamma I)^{-1} J^T \eta.$$

The resulting motion $J(\theta)\dot{\theta}^*$ can be interpreted as a weighted or oblique projection of η onto the range of $J(\theta)$. The extent to which the force and motion goals are locally satisfied by $\dot{\theta}^*$ is governed by the ratio of α to β. Moving according to $\dot{\theta}^*$ is a method to locally step towards achieving the task goal. In practice, this solution could be used iteratively for planning and control, or to bias samples in a randomized planner, and thereby make global search attempts tractable.

5.4 Simultaneous force and Position Control for Legged Robots

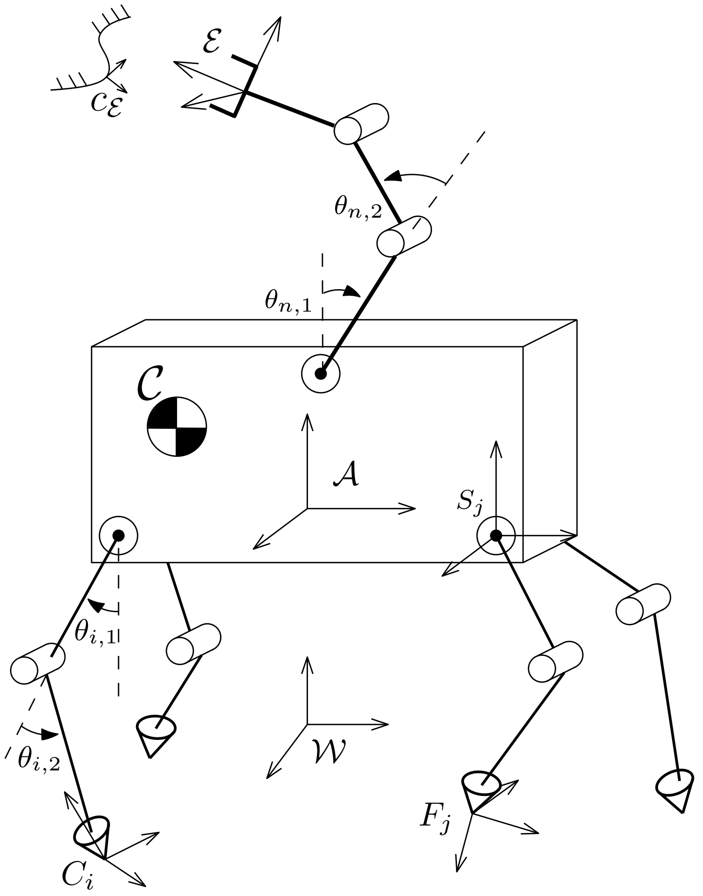


Figure 5.4: Key Reference Frames for Stance and Reach when the reaching limb is in contact.

In this section, the ideas of Section 5.3 and 3.3 are extended to the case of multi-limbed robots which use one or more limbs to carry out simultaneous velocity and force manipulation tasks, while also maintaining a static equilibrium. The present analysis is restricted to the case of one manipulating limb, with the remaining limbs providing support. The manipulating limb contacts the world, with the goal of imparting wrenches to its contact as defined by the wrench basis B_W. Now the manipulation task wrenches can be expressed in the abdomen frame as

$$F_{\mathcal{E}}^{\mathcal{A}} = \mathrm{Ad}_{g_{\mathcal{E}\mathcal{A}}}^{T} B_W f_{\mathcal{E}},$$

where $f_{\mathcal{E}}$ is the manipulating contact force expressed in the coordinates of B_W. Define a twist basis D_V that compliments B_W and satisfies the local compatibility condition. In [87], it was shown that an end-effector motion can arise from some combination of abdomen movement and free-limb movement. However, for free-limb forces, wrenches applied by the end-effector must be resisted by the supporting limbs in order to maintain static equilibrium.

To accommodate a manipulating limb in contact, it must be accounted for in the stance map,

$$S = -\Big[\overbrace{\mathrm{Ad}_{g_{\mathcal{A}c_1}^{-1}}^{T} B_{c_1} \quad \dots \quad \mathrm{Ad}_{g_{\mathcal{A}c_M}^{-1}}^{T} B_{c_{M-1}}}^{\text{ground contacts}} \qquad \underbrace{\mathrm{Ad}_{g_{\mathcal{A}\mathcal{E}}}^{T} B_W}_{\text{manipulation contact}}\Big] \tag{5.6}$$

as well as in the vector of contact forces, $f = \begin{bmatrix} f_1 & \dots f_{M-1} & f_{\mathcal{E}} \end{bmatrix}^T$. The Stance Jacobian too must include the block diagonal term associated with the manipulating limb (the index M in (5.6) is replaced by $\mathcal{E}$).

Definition 6. *A multi-legged robot is in static equilibrium if there exists f satisfying*

$$Sf = Ad_{g_{\mathcal{C}\mathcal{A}}} \mathcal{G}$$
$$f \in FC,$$

where f is a vector of contact frame forces including those of the manipulating limb, $\mathcal{G}$ is the gravity wrench acting on the origin of the center of mass frame $\mathcal{C}$, and FC is the friction cone [1].

This definition can be used to obtain the set $\mathcal{F}_{\mathcal{E}}$ of end-effector forces that can possibly be produced by the mechanism in its current configuration

$$\mathcal{F}_{\mathcal{E}} = \{x - \mathcal{G} \mid x \in \mathcal{R}(S)\},$$

[1] The friction cone is the set of forces that satisfy the chosen contact-friction model. For background and examples, see [58].

where $\mathcal{R}(A)$ is the range of matrix A. In general, however, the set of feasible manipulating forces $F_{\mathcal{E}}$ depends not only on the wrench basis B_W but also on the nature of contact of the supporting legs, defined by the friction cone. The following provides some general conditions under which a stance can produce any manipulating force:

Proposition 7. *A legged robot can sustain any end-effector manipulating force, $f_{\mathcal{E}}$, if and only if S is surjective, and there exists f_G satisfying $Sf_g = \mathcal{G}$, and there exists an internal force $f_{int} \in \mathcal{N}(S)$* [2] *such that $f_{int} \in int(FC)$.*

Proof [3]. $\Leftarrow$ Let $f_{\mathcal{E}} \in \mathrm{FC}_{\mathcal{E}}$. Since S is surjective, there exists $f' \in \mathrm{FC}$ such that $Sf' + B_W f_{\mathcal{E}} = 0$. Now, since $f_{\mathrm{int}} \in \mathrm{FC}$ and since FC is a cone, for any $a \geq 0$, $af_{\mathrm{int}} \in \mathrm{FC}$. For a large enough, $af_n + f' \in \mathrm{FC}$ since the friction cone is closed and since $f_n \in \mathrm{int}(FC)$. Then, $S(f_g + \begin{bmatrix} f' + af_n & f_{\mathcal{E}} \end{bmatrix}^T) = G$.

$\Rightarrow$ Suppose that $Sf + B_W f_{\mathcal{E}} = G$. Pick $f_1 \in \mathrm{int(FC)}$ so that $Sf_1 = F \neq 0$ and select f_2 satisfying $Sf_2 = -F$. Then, set $f_{\mathrm{int}} = f_1 + f_2$. Now it holds that $Sf_{\mathrm{int}} = 0$ and $f_{\mathrm{int}} \in \mathrm{int(FC)}$.

Now that it is clear how forces at the contacts result in forces at the abdomen, one can ask how do motions of the abdomen result in forces on the abdomen? It makes little sense to define a stiffness that relates abdomen motions to wrenches on the abdomen directly as was done for the end effector. Instead, one must obtain a compliance relationship between contact forces at the feet or end-effectors and abdomen motions.

Let the stiffness matrix for the i^{th} limb stance limb that makes contact be K^i (see equation (5.3) in section 5.2) , defined with respect to its contact frame. *Contact frame* velocities ν_i and *contact frame* forces f_i are related by

$$K^i \nu_i = f_i.$$

More generally, if the manipulating limb makes contact, then one must include the effect of the manipulating limb's motion in the subspace defined by the basis B_W. Contact velocities and forces (in the coordinates of B_W) are related by

$$f_{\mathcal{E}} = K^{\mathcal{E}} \nu_{\mathcal{E}}.$$

In [87], it is shown that instantaneous contact velocities are related to abdomen velocities by

$$S^T V^{\mathcal{A}}_{\mathcal{WA}} = \nu,$$

[2] $\mathcal{N}(A)$ = Null Space of matrix A.

[3] This proposition is similar in nature to Proposition 5.2 in [58] and the proof is along the same lines

where $\nu = \begin{bmatrix} \nu_1 & \dots & \nu_{M-1} & \nu_{\mathcal{E}} \end{bmatrix}^T$. Therefore, the following compliance relationship between abdomen wrenches and velocities is obtained:

$$K_{\mathcal{S}} V^{\mathcal{A}}_{\mathcal{W}\mathcal{A}} = \Delta F_{\mathcal{A}}, \tag{5.7}$$

where $K_{\mathcal{S}}$ is called the *stance stiffness matrix*:

$$K_{\mathcal{S}} = S\tilde{K}S^T, \tag{5.8}$$

where

$$\tilde{K} = \begin{bmatrix} K^1 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & K^M & 0 \\ 0 & 0 & 0 & K^{\mathcal{E}} \end{bmatrix}.$$

When the manipulating limb is not in contact, $\tilde{K}$ will not include the last diagonal block representing the manipulating contact stiffness. Note that in general, $K_{\mathcal{S}}$ is invertible, since the stiffness matrices K_i and $K_{\mathcal{E}}$ are positive definite, by convention (see section 5.2), and since S will generally be onto [4].

Now the issue of achieving a manipulation goal that requires moving from one pose to another while achieving and maintaining a wrench is addressed. Compared to the fixed base manipulator of Section 5.3, one must account for gravity and for the motion of the supporting legs. One must also include the compliance model from Section 5.2 to relate forces and velocities.

Consider the following word problem: given a combined pose and wrench goal for the manipulating limb, move the manipulating and supporting limbs' joints so as to

- Ensure that the end-effector moves towards the pose goal, within the motion subspace defined by D_V.
- Control the end-effector wrench to lie with the wrench subspace defined by B_W, towards achieving a desired wrench.
- Ensure that static stance equilibrium is achieved by regulating the net wrench at the abdomen due to gravity.
- Move minimally when the robot is redundant.

[4]when it is not, the robot is not *force resistant* [87], and is in a poor configuration for manipulation

$$\begin{aligned}
\min_{\dot{\Theta}} \quad & \frac{\alpha}{2}\|v_{\mathcal{E}} - v_{\mathcal{E},d}\|_2^2 + \frac{\beta}{2}\|\Delta F_{\mathcal{A}} - F_{meas}\|_2^2 \\
& + \frac{\alpha}{2}\|\Delta f_{\mathcal{E}} - (f_{\mathcal{E},d} - f_{\mathcal{E},\text{meas}})\|_2^2 + \frac{\gamma}{2}\|\dot{\Theta}\|_2^2 \\
\text{subject to} \quad & S^T V_{\mathcal{W}\mathcal{E}}^{\mathcal{A}} = J_{\mathcal{R}}\dot{\Theta} \\
& D_V^T \text{Ad}_{g_{\mathcal{E}\mathcal{A}}} V_{\mathcal{W}\mathcal{E}}^{\mathcal{A}} = v_{\mathcal{E}} \\
& B_W^T \text{Ad}_{g_{\mathcal{E}\mathcal{A}}} V_{\mathcal{W}\mathcal{E}}^{\mathcal{A}} = K_{\mathcal{E}}^{-1}\Delta f_{\mathcal{E}} \\
& S^T V_{\mathcal{W}\mathcal{A}}^{\mathcal{A}} = J_{\mathcal{S}}\dot{\theta} \\
& K_{\mathcal{S}} V_{\mathcal{W}\mathcal{A}}^{\mathcal{A}} = \Delta F_{\mathcal{A}} \ ,
\end{aligned} \tag{5.9}$$

where $F_{\mathcal{A}}$ is the net wrench on the abdomen, F_{meas} is the measured wrench at the abdomen,

$$F_{meas} = Sf_{meas} - \mathcal{G}$$

with f_{meas} being the vector of measured contact wrenches at the supporting feet and $\mathcal{G}$ the gravity wrench as seen in the abdomen frame. $v_{\mathcal{E},d}$ is the motion goal for the end-effector expressed in the twist basis D_V and $\Delta f_{\mathcal{E}}$ is the desired change in force written in the wrench basis B_W, as seen in the abdomen frame. $f_{\mathcal{E},\text{meas}}$ and $f_{\mathcal{E},d}$ are the measured and desired end effector forces, respectively, in the task wrench basis' coordinates. The matrices $K_{\mathcal{E}}$ and $K_{\mathcal{S}}$ are the end-effector and stance stiffness matrices, respectively. All of the full wrenches above and henceforth are expressed in the abdomen frame.

Conceptually, the solution to (5.9) provides a proportional control law that will drive error in position and force to zero, locally. To see this more clearly, the problem above can be simplified by making the relationship between variables explicit. Let

$$\eta_{\mathcal{E}} = \alpha D_V v_{\mathcal{E},d} + \alpha B_W K_{\mathcal{E}}^{-1}(f_{\mathcal{E},d} - f_{\mathcal{E},\text{meas}}) \ ,$$

and let

$$\eta_{\mathcal{A}} = K_{\mathcal{S}}^{-1} F_{\text{Meas}}.$$

Then, the problem (5.9) can be written as

$$\begin{aligned} \min_{\dot\Theta} \quad & \frac{\alpha}{2}\|V^{\mathcal{A}}_{\mathcal{W}\mathcal{E}} - \eta_{\mathcal{E}}\|_2^2 + \frac{\beta}{2}\|V^{\mathcal{A}}_{\mathcal{W}\mathcal{A}} - \eta_{A}\|_2^2 + \frac{\gamma}{2}\|\dot\Theta\|_2^2 \\ \text{subject to} \quad & S^T V^{\mathcal{A}}_{\mathcal{W}\mathcal{E}} = J_{\mathcal{R}}\dot\Theta \\ & S^T V^{\mathcal{A}}_{\mathcal{W}\mathcal{A}} = J_{\mathcal{S}}\dot\theta \ . \end{aligned} \tag{5.10}$$

Define

$$x = \begin{pmatrix} V^{\mathcal{B}}_{\mathcal{W}\mathcal{E}} \\ V^{\mathcal{A}}_{\mathcal{W}\mathcal{A}} \\ \dot\Theta \end{pmatrix}, \quad b = \begin{bmatrix} \alpha\eta_{\mathcal{E}} \\ \beta\eta_{\mathcal{A}} \\ 0 \end{bmatrix}, \quad P = \begin{bmatrix} \alpha I_n & 0 & 0 \\ 0 & \beta I_n & 0 \\ 0 & 0 & \gamma I_m \end{bmatrix},$$

$$F = \begin{bmatrix} S^T & 0 & -J_{\mathcal{R}}(x_0, \Theta) \\ 0 & S^T & -J_{\mathcal{C}}(x_0, \Theta) \end{bmatrix}.$$

In these variables, the Problem (5.10) is equivalent to

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T P x - x^T b \\ \text{subject to} \quad & Fx = 0 \ . \end{aligned} \tag{5.11}$$

The KKT conditions for this problem are equivalent to the following linear system:

$$\begin{bmatrix} P & F^T \\ F & 0 \end{bmatrix} \begin{pmatrix} x^* \\ \nu^* \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{5.12}$$

Proposition 8. *The problem (5.10) has a solution whenever α, β and γ are non-negative and the linear system (5.12) has a solution (x^*, ν^*). It has a unique solution when $P \succ 0$ and F is full rank — in this case, the solution can be obtained explicitly.*

The proof of this proposition follows readily from writing down the KKT conditions for (5.10), and considering the rank of the resulting 'KKT Matrix', that relates the optimal values of $\dot\Theta$ and its dual to the problem data (see section and 2.2 [6]). It is along the same lines as the proof of Proposition 4.

Note that the problems (5.9) and (5.10) do not include friction cone constraints at the contacts. The local analysis (Prop. 8) in the presence of the friction cone constraints is the same when forces are interior to the friction cone. When the forces are on the boundary, local feasibility depends on the particular friction model in use. The issue of feasibility with respect to the friction cone and actuator limits is addressed in the following section.

5.5 Example

.

This section will apply some of the concepts from previous sections to illustrate the kinds of tasks which are suitable for the proposed method, and the solutions the produces. In particular, the optimization problem (5.10) is applied to the redundant planar mobile-manipulator shown in Fig. 5.5. This manipulator is capable of walking quasistatically, and manipulating while standing. Gravity, $\mathcal{G}$, acts downward. This example is an extension of the one in [87].

Suppose this manipulator is tasked with painting a compliant ceiling. Although most ceilings are very stiff, we can lump the stiffness of the roller and ceiling into one. The task requires pushing upward, and moving side to side. Because the associated twist and wrench are reciprocal, this task will satisfy local compatibility throughout execution.

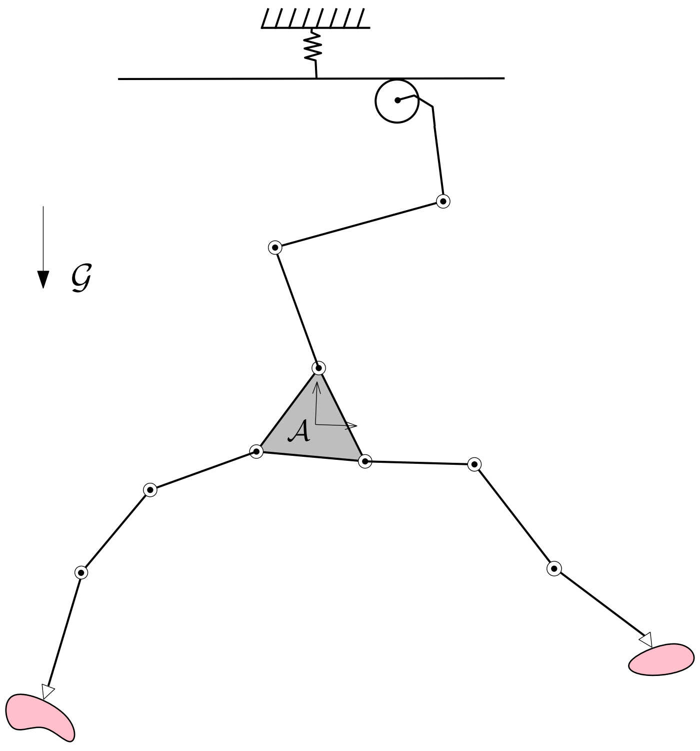


Figure 5.5: Redundant Planar Walker

Suppose that the end-effector begins at $x = -1.5$ and must move to $x = 1$, and in the process achieve a force of 1 N upward. Assume that the ceiling's stiffness is $k_c = 4Nm^{-1}$ (the stiffness is intentionally low for the purposes of illustration). Iteratively applying the solution to (5.10) with $\alpha = \beta = 10$ and $\gamma = 0.1$ results in the motion shown in Fig. 5.6. The resulting motion is smooth and uses all of the body's degrees of freedom to achieve the task, while remaining balanced. The initial mechanism position is shown in blue, while the final one is red.

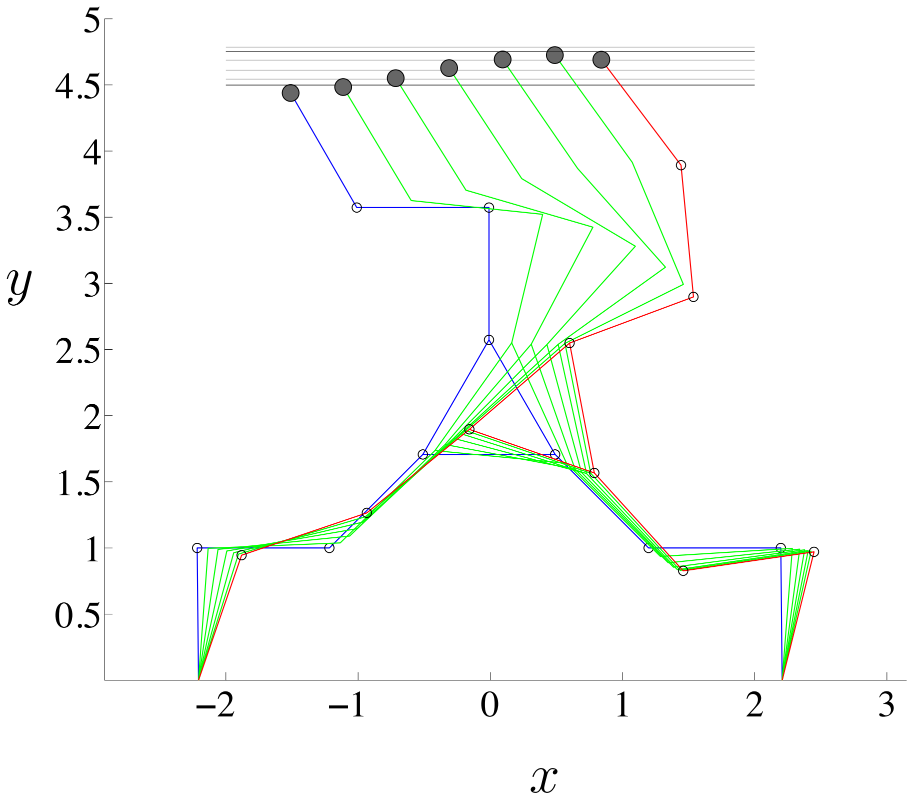


Figure 5.6: Simulated Solution

5.6 Summary

This chapter studied the kinematics and feasibility of combined force and position control for multi-limbed robots. A classical framework was used for the combination of forces and positions, based on which a novel compliance model is provided to relate forces to robot motion. With these assumptions, first the local planning problem for combined force/position tasks with a fixed-base manipulator is considered. Then, necessary kinematic extensions for multi-limbed robots are made, and the corresponding local planning problem is posed as a constrained optimization.

To gain intuition, stance and wrench feasibility were defined and analyzed, and conditions under which they are possible or infeasible were obtained. Finally, a simple example with a planar walker was shown.

Chapter 6

Conclusion

6.1 Summary of Thesis Contributions

The main contributions of this thesis are the formal development of the kinematics of mobile robots with general mechanism topologies, and the formulation and use of local planning and optimization problems for a general class of manipulation tasks. By formulating the kinematics explicitly, the feasibility of local motions is easily analyzed, and the properties of locally optimal motions can be evaluated as a function of the robot's configuration.

Stance and Reach Kinematics provide a formal mathematical framework that allows the definition of *limberness*, *local dexterity*, and *wrench resistance* as well as analysis of conditions when these properties are satisfied. The kinematics of a legged robot's stance and these local properties provide an explicit analogy between a legged robot's stance and manipulation of and object within a dexterous robotic hand. Although this similarity has been noted by many, this thesis is the first work to make it concrete. The kinematics of center-of-mass motion is derived, and the *Stance Constrained Center-of-mass Jacobian* that relates joint motions to motion of the center-of-mass is introduced. This relationship is later used to ensure that a mobile robot maintains balance in the presence of gravity during a manipulation task.

The kinematics of stance adjustment, reaching, and center-of-mass motion are also derived for robots with wheeled and articulated bases. This allows the definition the general *local planning problem* associated with a general mobile manipulation task for almost any mobile robot. This local planning problem is easily translated into a convex quadratic program which can be solved efficiently. Conditions on existence and uniqueness of solutions to this problem are provided, and in the case of legged robots, the conditions can be interpreted physically.

The local planning problem is generalized by including *inequality constraints* that allow a vast number of practical constraints to be included with ease. Efficient linear programs are provided to generate and certify feasible constraints. These optimization problems are then used within an iterative algorithm to generate motion plans that achieve manipulation tasks under a large number of critical physical constraints (e.g. collision avoidance, static equilibrium, joint limits, etc.). The local problem can be solved quickly enough to be used within a feedback loop at a high rate.

The formulation of kinematics for legged robots enables the analysis of combined force and motion control, and extension of classical linear stiffness contact-models to legged robots. This thesis defines the *stance stiffness* of a legged robot, which relates forces on the abdomen to motions of the abdomen. This compliance relationship can be used to reformulate the goal of maintaining static equilibrium by selecting local motions to regulate forces on the abdomen directly (instead of using a geometric notion of stability). The local planning problem for a mobile manipulation task that includes both force and motion goals (such as painting a wall or opening a door) is formulated and analyzed. A novel definition for static equilibrium is provided.

A number of examples are provided both in simulation and on the physical RoboSimian and Surrogate robots as NASA-JPL. These examples demonstrate the classes of goals and constraints that can be handled by the local planning problems introduced in this thesis, including joint limits, static equilibrium, configuration biasing, and gaze constraints. A detailed example, including weights and parameters, and comparisons of experimental and planned motions, is shown.

6.2 Opportunities for Future Work

Using the explicit form and structure of stance kinematics, analysis and characterizations of robot *configurations* that satisfy static equilibrium conditions in the presence of torque limits may be possible. As shown in the thesis, checking whether a configuration satisfies static equilibrium is a convex problem. However, the problem of checking a *region* of robot configuration space (including the robot's internal configuration) is likely not convex. Local convexity and kinematic structure may be exploited to provide efficient techniques find configurations satisfying static equilibrium. Convex relaxations may also be successful along these lines. These effort could also be extended to include different contact models including rolling contact, and bounded dynamic effects.

The presented local planning techniques, like all iterative methods using local optimization, is susceptible to local minima. This occurs when the objective cannot be decreased without violating the constraints locally, for example, due to clutter (collisions). To overcome local minima, a global outer-loop that generates intermediate goals must be used. The question of how one does this to achieve a complete planning algorithm (that is guaranteed to find a solution if exists, and returns failure otherwise) is complex, and is a subject of ongoing work. A methodology that may be fruitful is to consider the *logarithm* of error between current and desired end-effector poses – the rate of change of this logarithm has a great deal of structure, and can be written using only the logarithm itself, and the rigid body velocity of the end-effector. If one can find conditions under which setting the end-effector rigid body velocity instant-by-instant using a local optimization scheme results in the logarithm converging to zero (e.g. using a Lyapunov function or showing that the dynamics are dissipative), then this shows that the technique is complete. Thereafter, additional effort could be applied to handling inequality constraints, singularities, perturbations, multiple goals, etc. An introduction to this kind of approach is available in [10].

This thesis only considers motions that are quasi-static, and the models developed are purely kinematic – dynamic effects are ignored. Though dynamic constraints can be added, this thesis omits them because the dynamics and stability of articulated mechanisms in contact is very poorly understood – this is an area that requires a great deal of new theory. In order to perform stable dynamic motions with robots, a working definition of a robot's stability in the presence of gravity is required. Initial efforts could be focused to identify conditions on robot configurations that guarantee local neighborhoods of stability including small dynamic effects. This effort would likely require use and analysis of the robot's gravitational potential and kinetic energy functions, and may leverage tools and results from geometric mechanics.

Appendices

Appendix A

The Problem with using a Slack Variable for Hierarchical Quadratic Programming

This section describes a critical flaw associated with 'hierarchical quadratic programming' techniques for inverse kinematics. The most general optimization problem these approaches consider at a given priority level is introduced in [35], and is of the form

$$\begin{aligned} \min \quad & \frac{1}{2}\|Ax-b\|_2^2 + \frac{1}{2}\|w\|_2^2 + \rho^2\frac{1}{2}\|x\|_2^2 \\ & \text{subject to} \quad Cx - w \preceq \ d \ , \end{aligned} \tag{A.1}$$

where the configuration is given by x, the goal is given by A, b (these constraints ensure that higher priority tasks aren't violated), and the (projected) constraints are given by C, d, with w a slack variable.

Fact 1. *If the system of equations*

$$Ax = b \qquad Cx \preceq d$$

is not jointly feasible, then the solution produced by solving (A.1), for any $\rho \geq 0$, will violate both $Ax = b$ and $Cx \preceq d$. This is true even if there exists x satisfying $Cx \preceq d$.

This fact is intuitively clear – since the problem they solve trades off the residual of equality and inequality portions of the task, if the combination of the two is infeasible, then both will be violated by the optimal solution.

The implication of this fact is that all existing Inverse Kinematics frameworks that (claim to) incorporate inequality constraints at each priority level are incapable of enforcing safety critical constraints *even when they are feasible.*

Proof. We prove this fact directly by explicit calculation of the solution to (A.1) as well using counterexamples. We assume a basic knowledge of Constrained Optimization. The Constrained Lagrangian associated with (A.1) is given by

$$\mathcal{L}(x, w, \lambda) = \frac{1}{2}(Ax - b)^T(Ax - b) + \frac{1}{2}w^T w + \frac{1}{2}\rho^2 x^T x + \lambda^T(Cx - d).$$

where λ is a dual variable or lagrange multiplier. Derivatives with respect to x, w must be zero, meaning:

$$x = (A^T A + \rho^2 I)^{-1}(A^T b - C^T \lambda) \quad \text{and} \quad \lambda = w.$$

The dual function is given by

$$g(\lambda) = -\frac{1}{2}\lambda^T(C(A^T A + \rho^2 I)^{-1}C^T + I)\lambda + \lambda^T(C(A^T A + \rho^2 I)^{-1}A^T b - d).$$

Strong duality holds. The dual problem is

$$\begin{aligned} \text{minimize} \quad & g(\lambda) \\ \text{subject to} \quad & \lambda \succeq 0 \ . \end{aligned}$$

The optimality conditions on λ^* are given by

$$\lambda^* \succeq 0 \qquad \nabla g(\lambda^*) \geq 0 \qquad \lambda_i^*(\nabla g(\lambda^*))_i = 0 \qquad i = 1, \ldots, n.$$

The solution to this problem is therefore

$$\lambda_i = \max(v_i, 0),$$

where

$$v = (C(A^T A + \rho^2 I)^{-1}C^T + I)^{-1}(C(A^T A + \rho^2 I)A^T b - d). \tag{A.2}$$

Since $w = \lambda$, w is identically zero only when v is entirely non-positive. w_i is non-zero if-and-only-if $Cx* \npreceq d$ (it is impossible for $Cx^* \preceq d$ to hold with w to be non-zero, as this would contradict minimality of the optimal objective value, which includes $\|w\|_2$). Since there is no restriction on A, b, C, d or ρ, it cannot be guaranteed that w will be zero, and that there exists x satisfying $Cx \preceq d$.

In particular, whenever $Ax = b$ and $Cx \preceq d$ are *not jointly feasible*, the quantity

$$y = C(A^T A + \rho^2 I)A^T b - d$$

must have a positive element (regardless of whether $Cx \preceq d$ is independently feasible). Since $(C(A^T A + \rho^2 I)^{-1} C^T + I)^{-1}$ is positive definite, v will also have a positive element in every position that y has a positive element. This means that the w will be non-zero, and the inequality constraint will be violated with $Cx^* \npreceq d$. This concludes a proof. We provide two (counter) examples below:

For the particular parameters

$$A = I_{2\times 2} \quad b = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \quad C = I_{2\times 2} \quad d = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Clearly the feasible set for $Cx \preceq d$ is non-empty. The optimal solution is

$$x^* = \begin{bmatrix} 3/2 \\ 3/2 \end{bmatrix} \quad w^* = \begin{bmatrix} 1/2 \\ 0 \\ 0 \\ 1/2 \end{bmatrix}.$$

It is easy to check that $Cx^* \npreceq d$. Neither the desired equality nor the inequality is satisfied.

A less trivial counterexample:

$$A = \begin{bmatrix} -1.4 & -4.8 & -0.7 & 2.6 & 1.7 & 4.9 & -0.9 & -4.1 & 0 \\ 2.4 & -1.7 & 3.9 & -1.2 & -0.6 & 0.1 & 2.5 & -3.9 & -3.5 \\ -1.1 & -0.8 & -1.1 & -2.8 & 3.3 & 3.8 & 3.3 & -3.6 & -4.5 \\ 1.8 & -2.3 & 2.7 & 2.9 & 2.7 & 0.9 & 2.9 & 1.8 & 3.5 \\ 2.0 & -3.0 & -1.0 & 4.5 & -3.3 & -3.5 & -1.8 & 0 & 0.6 \\ -0.6 & 3.2 & 3.1 & -1.7 & 3.6 & -3.0 & 0.3 & -3.1 & 4.3 \end{bmatrix} \qquad b = -\begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad d = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Clearly the feasible set for $Cx \preceq d$ is non-empty. The solution to this instance is given by

$$x^* = \begin{bmatrix} 0.8549 \\ 2.0479 \\ 1.8029 \\ -0.1405 \\ 1.0279 \\ 1.8424 \\ -4.5994 \\ 2.9850 \\ -2.1935 \end{bmatrix} \qquad w^* = \begin{bmatrix} 0 \\ 1.0479 \\ 0.8029 \\ 0 \\ 0.0279 \end{bmatrix}$$

Three of the five inequality constraints are violated, and the equality goal is not achieved.

$\square$

This fact has the following consequence: all prior hierarchical least-squares/QP approaches produced to date are unsafe — the inequality constraint at a given priority level may be violated even when it is feasible. This means that constraints like collisions, static equilibrium, and joint limits cannot be guaranteed to hold even when they are feasible.

Appendix B

Video Supplement

This thesis includes a video supplement, showing several demonstrations of the local planning techniques from Chapter 4 and 3 using the RoboSimian and Surrogate robots. The video includes both simulated and experimental examples. The video is available through the Caltech Library thesis repository, and on YouTube at `https://youtu.be/Jd4KL0NujwM`.

Bibliography

[1] J. Angeles. On the nature of the cartesian stiffness matrix. 3(5):163–170, 2010.

[2] Paolo Baerlocher and Ronan Boulic. An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *The visual computer*, 20(6):402–417, 2004.

[3] Dmitry Berenson, James Kuffner, and Howie Choset. An optimization approach to planning for mobile manipulation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1187–1192. IEEE, 2008.

[4] Dmitry Berenson, Siddhartha S Srinivasa, Dave Ferguson, and James J Kuffner. Manipulation planning on constraint manifolds. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 625–632. IEEE, 2009.

[5] Dimitri P Bertsekas. *Convex optimization theory.* Athena Scientific Belmont, MA, 2009.

[6] Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[7] Stephen P Boyd and Ben Wegbreit. Fast computation of optimal contact forces. *Robotics, IEEE Transactions on*, 23(6):1117–1132, 2007.

[8] Timothy Bretl and Sanjay Lall. Testing static equilibrium for legged robots. *Robotics, IEEE Transactions on*, 24(4):794–807, 2008.

[9] Oliver Brock, Oussama Khatib, and Sriram Viji. Task-consistent obstacle avoidance and motion behavior for mobile manipulation. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 388–393, 2002.

[10] Francesco Bullo and RM Murray. Proportional derivative (pd) control on the euclidean group. In *European Control Conference*, volume 2, pages 1091–1097, 1995.

[11] Samuel R. Buss. Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods. 2009.

[12] Erwin Courmans. Bullet physics engine. `http://bulletphysics.org/Bullet/BulletFull/`.

[13] George B Dantzig. Programming in a linear structure. *Washington, DC*, 1948.

[14] Martin De Lasa and Aaron Hertzmann. Prioritized optimization for task-space control. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5755–5762. IEEE, 2009.

[15] Martin De Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. In *ACM Transactions on Graphics (TOG)*, volume 29, page 131. ACM, 2010.

[16] Steven Dubowsky and Evangelos Papadopoulos. The kinematics, dynamics, and control of free-flying and free-floating space robotic systems. *Robotics and Automation, IEEE Transactions on*, 9(5):531–543, 1993.

[17] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, page 0278364914521306, 2014.

[18] Andrew A Frank. Automatic control systems for legged locomotion machines, 1968.

[19] M. Gienger, M. Toussaint, and C. Goerick. Task maps in humanoid robot manipulation. In *IEEE/RSJ Int. Conf. Intell. Robots and Systems*, pages 2758–2764, Sept. 2008.

[20] Donald Goldfarb and Shucheng Liu. An o (n 3 l) primal interior point algorithm for convex quadratic programming. *Mathematical Programming*, 49(1-3):325–340, 1990.

[21] Li Han and Jeffrey C Trinkle. Dextrous manipulation by rolling and finger gaiting. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 1, pages 730–735. IEEE, 1998.

[22] Li Han and Jeffrey C Trinkle. The instantaneous kinematics of manipulation. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 3, pages 1944–1949. IEEE, 1998.

[23] Li Han, Jeffrey C Trinkle, and Zexiang X Li. Grasp analysis as linear matrix inequality problems. *Robotics and Automation, IEEE Transactions on*, 16(6):663–674, 2000.

[24] K. Harada, S. Kajita, F. Kanehiro, K. Fujiwara, K. Kaneko, K. Yokoi, and H. Hirukawa. Real-time planning of humanoid robot's gait for force-controlled manipulation. *IEEE/ASME Trans. on Mechatronics*, 12(1):53–62, 2007.

[25] Kensuke Harada, Shuuji Kajita, Fumio Kanehiro, Kiyoshi Fujiwara, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. Real-time planning of humanoid robot's gait for force-controlled manipulation. *Mechatronics, IEEE/ASME Transactions on*, 12(1):53–62, 2007.

[26] Kensuke Harada, Shuuji Kajita, Kenji Kaneko, and Hirohisa Hirukawa. Dynamics and balance of a humanoid robot during manipulation tasks. *Robotics, IEEE Transactions on*, 22(3):568–575, 2006.

[27] Kris Hauser, Timothy Bretl, Jean-Claude Latombe, and Brian Wilcox. Motion planning for a six-legged lunar robot. In *Algorithmic Foundation of Robotics VII*, pages 301–316. Springer, 2008.

[28] Paul Hebert, Max Bajracharya, Jeremy Ma, Nicolas Hudson, Alper Aydemir, Jason Reid, Charles Bergh, James Borders, Matthew Frost, Michael Hagman, et al. Mobile manipulation and mobility as manipulationdesign and algorithms of robosimian. *Journal of Field Robotics*, 32(2):255–274, 2015.

[29] N. Hogan. Impedance control: An approach to manipulation. *ASME J. Dyn. Syst. Measurment Control*, 107:1–7, Mar. 1985.

[30] Robert Holmberg and Oussama Khatib. Development and control of a holonomic mobile robot for mobile manipulation tasks. *The International Journal of Robotics Research*, 19(11):1066–1074, 2000.

[31] Marco Hutter, Hannes Sommer, Christian Gehring, Mark Hoepflinger, Michael Bloesch, and Roland Siegwart. Quadrupedal locomotion using hierarchical operational space control. *The International Journal of Robotics Research*, page 0278364913519834, 2014.

[32] K. Inoue, H. Yoshida, T. Arai, and Y. Mae. Mobile manipulation of humanoids: Real-time control based on manipulabitliy and stability. In *IEEE Int. Conf. Robotics and Automation*, pages 2217–2222, San Francisco, May 2000.

[33] Aaron M Johnson, G Clark Haynes, and Daniel E Koditschek. Standing self-manipulation for a legged robot. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 272–279. IEEE, 2012.

[34] F. Kanehiro, F. Lamiraux, O. Kanoun, E. Yoshida, and J.-P. Laumond. A local collsion avoidance method for non-strictly convex polyhedra. In *Proc. Robotics: Science and Systems IV*, 2008.

[35] Oussama Kanoun, Florent Lamiraux, and P-B Wieber. Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. *Robotics, IEEE Transactions on*, 27(4):785–792, 2011.

[36] Oussama Kanoun, Florent Lamiraux, Pierre-Brice Wieber, Fumio Kanehiro, Eiichi Yoshida, and Jean-Paul Laumond. Prioritizing linear equality and inequality systems: application to local motion planning for redundant robots. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2939–2944. IEEE, 2009.

[37] Oussama Kanoun, Jean-Paul Laumond, and Eiichi Yoshida. Planning foot placements for a humanoid robot: A problem of inverse kinematics. *The International Journal of Robotics Research*, 2010.

[38] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[39] Jeffrey Kerr and Bernard Roth. Analysis of multifingered hands. *The International Journal of Robotics Research*, 4(4):3–17, 1986.

[40] O. Khatib. A unified approach for motion and force control of robot manipulators. *IEEE Trans. Robotics and Automation*, 3(1):43–53, Feb. 1987.

[41] O. Khatib. Mobile manipulation: The robotic assistant. *Robotics and Autonomous Systems*, 26(2-3):175–183, 1999.

[42] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43–53, 1987.

[43] James Kuffner, Koichi Nishiwaki, Satoshi Kagami, Masayuki Inaba, and Hirochika Inoue. Motion planning for humanoid robots. In *Robotics Research*, pages 365–374. Springer, 2005.

[44] Scott Kuindersma, Frank Permenter, and Russ Tedrake. An efficiently solvable quadratic program for stabilizing dynamic locomotion. *arXiv preprint arXiv:1311.1839*, 2013.

[45] Vijay Kumar and Kenneth J. Waldron. Sub-optimal algorithms for force distribution in multifingered grippers. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 252–257. IEEE, 1987.

[46] Steven M LaValle. Rapidly-exploring random trees a ew tool for path planning. 1998.

[47] Steven M LaValle, Jeffery H Yakey, and Lydia E Kavraki. A probabilistic roadmap approach for systems with closed kinematic chains. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 3, pages 1671–1676. IEEE, 1999.

[48] David G Luenberger. *Optimization by vector space methods*. John Wiley & Sons, 1997.

[49] Ricky Lynch. Analysis of the dynamics and control of a two degree of freedom robotic manipulator mounted on a moving base. Technical report, DTIC Document, 1985.

[50] Anthony A Maciejewski and Charles A Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The international journal of robotics research*, 4(3):109–117, 1985.

[51] Dinesh Manocha and John F Canny. Efficient inverse kinematics for general 6r manipulators. *Robotics and Automation, IEEE Transactions on*, 10(5):648–657, 1994.

[52] Nicolas Mansard, Olivier Stasse, Paul Evrard, and Abderrahmane Kheddar. A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–6. IEEE, 2009.

[53] Richard Mason, Elon Rimon, and Joel Burdick. Stable poses of 3-dimensional objects. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 1, pages 391–398. IEEE, 1997.

[54] Jacob Mattingley and Stephen Boyd. Cvxgen: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, 2012.

[55] Robert B McGhee and Andrew A Frank. On the stability properties of quadruped creeping gaits. *Mathematical Biosciences*, 3:331–351, 1968.

[56] Robert B McGhee and Geoffrey I Iswandhi. Adaptive locomotion of a multilegged robot over rough terrain. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(4):176–182, 1979.

[57] David J Montana. The kinematics of contact and grasp. *The International Journal of Robotics Research*, 7(3):17–32, 1988.

[58] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation.* CRC press, 1994.

[59] Richard M Murray and S Shankar Sastry. Grasping and manipulation using multifingered robot hands. In *Robotics: proceedings of symposia in applied mathematics*, volume 41, pages 91–128, 1990.

[60] Y. Nakamura, H. Hanafusa, and T. Yoshikawa. Task-priority based redundancy control of robot manipulators. *Int. J. Robotics Research*, 6(2):2–15, 1987.

[61] Yoshihiko Nakamura, Hideo Hanafusa, and Tsuneo Yoshikawa. Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, 6(2):3–15, 1987.

[62] Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008.

[63] Yurii Nesterov, Arkadii Nemirovskii, and Yinyu Ye. *Interior-point polynomial algorithms in convex programming*, volume 13. SIAM, 1994.

[64] M.S. Ohwovoriole and B. Roth. An extension of screw theory. *ASME J. Mechanical Design*, 103(4):725–735, 1981.

[65] Yizhar Or and Elon Rimon. Robust multiple-contact postures in a two-dimensional gravitational field. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 5, pages 4783–4788. IEEE, 2004.

[66] Yizhar Or and Elon Rimon. Computation of multiple-contact frictional equilibrium postures in three-dimensional gravitational environments. *Department of Mechanical Engineering*, 2006.

[67] Evangelos Papadopoulos and Steven Dubowsky. On the nature of control algorithms for free-floating space manipulators. *Robotics and Automation, IEEE Transactions on*, 7(6):750–758, 1991.

[68] Evangelos Papadopoulos and Steven Dubowsky. Dynamic singularities in free-floating space manipulators. In *Space Robotics: Dynamics and Control*, pages 77–100. Springer, 1993.

[69] M Raghavan and B Roth. Solving polynomial systems for the kinematic analysis and synthesis of mechanisms and robot manipulators. *Journal of Vibration and Acoustics*, 117(B):71–79, 1995.

[70] Madhusudan Raghavan and Bernard Roth. Inverse kinematics of the general 6r manipulator and related linkages. *Journal of Mechanical Design*, 115(3):502–508, 1993.

[71] M.H. Raibert and J.J. Craig. Hybrid position/force control of manipulators. *ASME J. Dyn. Syst. Measurment Control*, 103:126–133, 1981.

[72] Ludovic Righetti, Jonas Buchli, Michael Mistry, Mrinal Kalakrishnan, and Stefan Schaal. Optimal distribution of contact forces with inverse-dynamics control. *The International Journal of Robotics Research*, 32(3):280–298, 2013.

[73] Elon Rimon, Richard Mason, Joel W Burdick, and Yizhar Or. A general stance stability test based on stratified morse theory with application to quasi-static locomotion planning. *Robotics, IEEE Transactions on*, 24(3):626–641, 2008.

[74] Layale Saab, Nicolas Mansard, François Keith, J-Y Fourquet, and Philippe Soueres. Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1091–1096. IEEE, 2011.

[75] J Kenneth Salisbury and B Roth. Kinematic and force analysis of articulated mechanical hands. *Journal of Mechanical Design*, 105(1):35–41, 1983.

[76] J.K. Salisbury. Active stiffness control of a manipulator in cartesian coordinates. In *IEEE Conf. Decision Control*, pages 95–100, Dec. 1980.

[77] J.K. Salisbury and J.J. Craig. Articulated hands: Force control and kinematic issues. *Int. J. Robotics Research*, 1(1):4–17, Mar. 1982.

[78] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *Int. J. Robotics Research*, page 0278364914528132, 2014.

[79] J.M. Selig. The spatial stiffness matrix from simple stretched springs. In *IEEE Int. Conf. Robotics and Automation*, San Francisco, 2000.

[80] J.M. Selig and P.R. McAree. A simple approach to invariant hybrid control. In *IEEE Int. Conf. Robotics and Automation*, Minneapolis, MN, 1996.

[81] Luis Sentis and Oussama Khatib. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics*, 2(04):505–518, 2005.

[82] Luis Sentis and Oussama Khatib. A whole-body control framework for humanoids operating in human environments. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 2641–2648. IEEE, 2006.

[83] Luis Sentis, Jaeheung Park, and Oussama Khatib. Compliant control of multicontact and center-of-mass behaviors in humanoid robots. *Robotics, IEEE Transactions on*, 26(3):483–501, 2010.

[84] Luis Sentis, Josh Petersen, and Roland Philippsen. Implementation and stability analysis of prioritized whole-body compliant controllers on a wheeled humanoid robot in uneven terrains. *Autonomous Robots*, 35(4):301–319, 2013.

[85] Homayoun Seraji. An on-line approach to coordinated mobility and manipulation. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 28–35. IEEE, 1993.

[86] Homayoun Seraji. A unified approach to motion control of mobile manipulators. *The International Journal of Robotics Research*, 17(2):107–118, 1998.

[87] K. Shankar and J.W. Burdick. Kinematics and methods for combined quasi-static stance/reach planning in multi-limbed robots. In *IEEE Int. Conf. Robotics and Automation*, Hong Kong, 2014.

[88] K. Shankar, J.W. Burdick, and N. Hudson. A quadratic programming approach to quasi-static whole-body manipulation. In *Workshop Algorithmic Foundations Robotics*, Istanbul, 2014.

[89] Krishna Shankar and Joel W Burdick. Kinematics and methods for combined quasi-static stance/reach planning in multi-limbed robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2014.

[90] Alexander Shkolnik and Russ Tedrake. Inverse kinematics for a point-foot quadruped robot with dynamic redundancy resolution. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4331–4336. IEEE, 2007.

[91] B. Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *J. Intelligent and Robotic Systems*, 3:201–212, 1990.

[92] Bruno Siciliano and Jean-Jacques E Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *Advanced Robotics, 1991.'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 1211–1216. IEEE, 1991.

[93] Benjamin J Stephens and Christopher G Atkeson. Dynamic balance force control for compliant humanoid robots. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1248–1255. IEEE, 2010.

[94] Tomomichi Sugihara and Yoshihiko Nakamura. Whole-body cooperative balancing of humanoid robot using cog jacobian. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2575–2580. IEEE, 2002.

[95] Yoji Umetani and Kazuya Yoshida. Resolved motion rate control of space manipulators with generalized jacobian matrix. *Robotics and Automation, IEEE Transactions on*, 5(3):303–314, 1989.

[96] Kenneth J. Waldron. Force and motion management in legged locomotion. *Robotics and Automation, IEEE Journal of*, 2(4):214–220, 1986.

[97] Daniel E Whitney. Resolved motion rate control of manipulators and human prostheses. *Man-Machine Systems, IEEE Transactions on*, 10(2):47–53, 1969.

[98] D.E. Whitney. Historical-perspective and state-of-the-art in robot force control. *Int. J. Robotics Research*, 6(1):3–14, Spring 1987.

[99] Jeffery Howard Yakey, Steven M LaValle, and Lydia E Kavraki. Randomized path planning for linkages with closed kinematic chains. *Robotics and Automation, IEEE Transactions on*, 17(6):951–958, 2001.

[100] Y. Yamamoto and X. Yun. Coordinating locomotion and manipulation of a mobile manipulator. In *Proc. 31st IEEE Conf. on Decision and Control*, pages 2643–2648, 1992.

[101] Y. Yamamoto and X. Yun. Coordinating locomotion and manipulation of a mobile manipulator. In *IEEE Conf. Decision and Control*, pages 2643–2648, 1992.

[102] Yoshio Yamamoto. *Control and coordination of locomotion and manipulation of a wheeled mobile manipulators*. PhD thesis, Citeseer, 1994.

[103] Yusaku Yamamoto and Xiaoping Yun. Effect of the dynamic interaction on coordinated control of mobile manipulators. *Robotics and Automation, IEEE Transactions on*, 12(5):816–824, 1996.

[104] Yuandong Yang and Oliver Brock. Elastic roadmapsmotion generation for autonomous mobile manipulation. *Autonomous Robots*, 28(1):113–130, 2010.

[105] E. Yoshida, M. Poirier, J.-P. Laumond, O. Kanoun, F. Lamiraux, R. Alami, and K. Yokoi. Whole-body motion planning for pivoting based manipulation by humanoids. In *IEEE Int. Conf. Robots and Automation*, pages 3181–3186, May 2008.

[106] Y. Zhang, S.S. Ge, and T.H. Lee. A unified quadratic-programming-based dynamical system approach to joint torque optimizaiton of physically constrained redundant manipulators. *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(5):2126–2132, 2004.

[107] Y. Zhang and Z. Zhang. *Repetitive Motion Planning and Control of Redundant Robot Manipulators*. Springer, 2013.