ENSEMBLE CONTROL OF ROBOTIC SYSTEMS

BY

AARON TRENT BECKER

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2012

Urbana, Illinois

Doctoral Committee:

Assistant Professor Timothy Bretl, Chair
Professor Seth Hutchinson
Associate Professor Daniel Liberzon
Assistant Professor Alejandro Domínguez-García

Abstract

In this dissertation, we apply the framework of ensemble control theory to derive an approximate steering algorithm for two classical robotic systems—the nonholonomic unicycle and the plate-ball manipulator—in the presence of model perturbation that scales all inputs by an unknown but bounded constant.

The basic idea is to maintain the set of all possible configurations and to select inputs that reduce the size of this set and drive it toward some goal configuration. The key insight is that the evolution of this set can be described by a family of control systems that depend continuously on the unknown constant. Ensemble control theory provides conditions under which it is possible to steer this entire family to a neighborhood of the goal configuration with a single open-loop input trajectory. For both the nonholonomic unicycle and the plate-ball manipulator, we show how to construct this trajectory using piecewise-constant inputs. We also validate our approach with hardware experiments, where the nonholonomic unicycle is a differential-drive robot with unknown wheel size, and the plate-ball manipulator is a planar motion stage that uses magnetic actuation to orient a sphere of unknown radius.

We conclude by showing how the same framework can be applied to feedback control of multi-robot systems under the constraint that every robot receives exactly the same control input. We focus on the nonholonomic unicycle, instantiated in experiment by a collection of differential-drive robots. Assuming that each robot has a unique wheel size, we derive a globally asymptotically stabilizing feedback control policy. We show that this policy is robust to standard models of noise and scales to an arbitrary number of robots. These results suggest that our approach may have possible future application to control of micro- and nano-scale robotic systems, which are often subject to similar constraints.

To my wonderful wife L.A.B.

Acknowledgments

It has been a pleasure to study and work with robots—many people cannot follow their passion and I am grateful to God for these years to do so.

Thanks to my advisor and zen master Timothy Bretl who has pushed me further than I thought was possible and mentored me through the research progress. It has been a privilege to learn under you.

Thank you to my doctoral committee and their commitment to the high standards of the University of Illinois at Urbana-Champaign. Daniel Liberzon has been a wonderful teacher throughout my graduate career, and I am thankful for his and Alejandro Domínguez-García's encouragement. Seth Hutchinson has been a source of valuable advice from the first day I visited this university. It was he who challenged me to make a robotic sphere manipulator—four prototypes later I am thankful for his vision.

To Dan Block and his Santa Claus workshop: thank you for your mentoring and for hiring me as an elf through my master's program. Dan encouraged me to pursue a PhD degree and his wonderful robots are featured prominently throughout this dissertation.

Thanks to my research group the RMSLab and the robotics community of the University of Illinois at Urbana-Champaign. You have been a pleasure to work with. I enjoyed our "Robot Movie Nights". I thank especially Colin Das and his unflagging optimism, Carlos Orduño and his commitment to professionalism, Hugo Leon who can build anything, Miles Johnson who holds the group together, Bob Sandheinrich—together we wrote my first robotics paper and produced a video that made the best of youtube but never mentioned our names—Bob, here is some overdue recognition. Thanks to Cem Onyuksel—together we implemented feedback control of many differential-drive robots featured in this dissertation. Thanks also to Zoe McCarthy who directed me to Fourier transforms, Syed Bilal Mehdi who provided a sounding board for research ideas, Or Dantsker who can procure anything, and

Aadeel Acktar, Dennis Matthews, and Navid Aghasadeghi.

Thank you to the Human Dynamics and Controls Lab led by Elizabeth Hsiao-Wecksler and especially David Lee and Alex Shorter. You broadened my view of robotics, taught me much about medical robotics, and made me welcome in your lab.

Finally, thanks to my wonderful wife Laney who gives me a reason to come home from the lab and has made our home a happy sanctuary. Thank you to our three children: Logan, Lincoln, and our coming baby. You have brought such joy to my life! May you continue to love to read, to climb, and to sing. Logan, you memorized this abstract after I substituted it for bedtime stories—thank you for being cheerful. To Lincoln I dedicate the many videos with "balls!" developed for this dissertation.

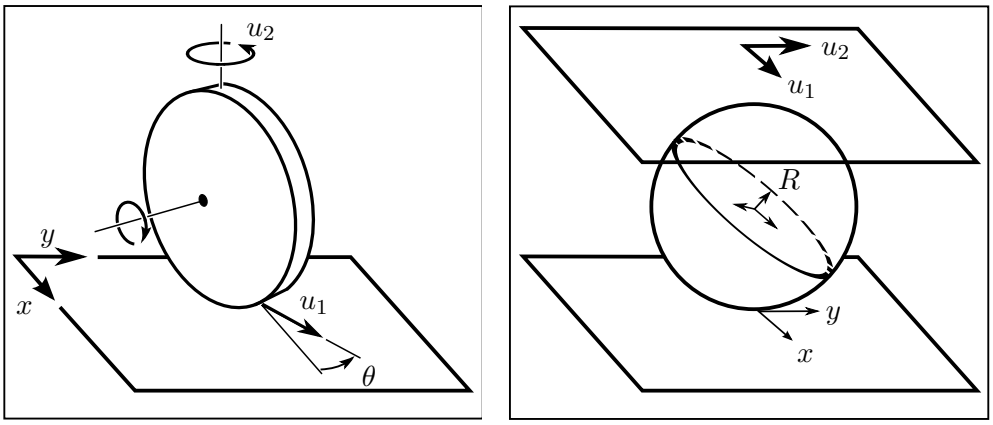
Table of Contents

List of Symbols

$\equiv$	identically equal
$\approx$	approximately equal
$A^\top$	matrix transpose of matrix A
ϵ	uncertain parameter for an ensemble
δ	spread of ϵ in an ensemble is $[1-\delta, 1+\delta], \delta \in [0,1)$ (plate-ball system: $\epsilon^{-1} \in [1-\delta, 1+\delta], \delta \in [0,1)$)
$\underline{\boldsymbol{\delta}}(\cdot)$	Dirac-delta function
$\delta_\#$	noise parameter
r	nominal radius of spheres in an ensemble
$\frac{r}{\epsilon}$	radius of spheres in an ensemble
q	configuration of an agent
$\mathcal{Q}$	configuration space
$\mathcal{U}$	inputs space
u_1	linear velocity input command (plate-ball system: velocity of ball center along x-axis)
u_2	angular velocity input command (plate-ball system: velocity of ball center along y-axis)
R	a rotation matrix, $R_x(\theta)$ is a rotation of θ about the x-axis
$\mathbb{R}$	real numbers
$\mathbb{R}^m$	m-dimensioned real numbers
$\mathbb{R}_+$	positive real numbers
SO(3)	the special orthogonal group in three dimensions, otherwise known as the rotation group SO(3)

Chapter 1

Ensemble Control



Nonholonomic Unicycle
J.P. Laumond, 1987

Plate-Ball Manipulator
Brockett and Dai, 1993

Figure 1.1: Two classical robotic systems. On the left, a nonholonomic unicycle, a canonical model for robot motion planning. On the right, a plate-ball manipulator, a canonical model for robotic manipulation. This dissertation answers the question, "Are these systems still controllable under model perturbation that scales all the inputs by an unknown but bounded constant?"

Figure 1.1 depicts two classical robotic systems: the nonholonomic unicycle and the plate-ball manipulator. The nonholonomic unicycle is a canonical model for mobile robotics (Laumound, 1987 [1]), and is used to describe robots including differential-drive vehicles, tanks and cars. It has two inputs, forward speed u_1 and turning rate u_2. The plate-ball manipulator is a canonical model for robotic manipulation by rolling (Brockett, 1993 [2]). In the classical version of this system, the ball is held between two parallel plates and manipulated by maneuvering the upper plate while holding the lower plate fixed. The ball can be brought to any position and orientation though translations of the upper plate. Inputs are speed of the ball center along the x-axis u_1 and speed along the y-axis u_2.

Figure 1.2: In this dissertation we apply the framework of ensemble control theory to control the three robotic systems shown above.

For decades it has been known that both these systems are controllable. There are many ways to solve the open-loop steering problem, sometimes called the motion planning problem, for these two systems. This dissertation addresses the question, "Are these two systems still controllable under model perturbation that scales all the inputs by an unknown but bounded constant?" This type of model perturbation arises if we do not know the radius of the wheel and if we do not know the radius of the ball.

Rather than steer a single system with an unknown parameter, we chose to model the system as a set of control systems that depend continuously on the unknown parameter. This set we call an *ensemble*. This dissertation proves that these ensembles are still controllable, after defining precisely what it means to be "controllable" in this context. We then give a steering algorithm based on piecewise-constant inputs and Taylor series approximations. Next, we show that by introducing feedback we can use a similar approach to control multiple nonholonomic unicycles when each unicycle receives exactly the same input command. We use the three hardware platforms shown in Fig. 1.2 to validate our approach.

Fig. 1.3 uses a nonholonomic unicycle to illustrate the main contribution of this thesis. In this figure, a unicycle with an unknown wheel radius is commanded to translate sideways (a parallel-parking maneuver). A controller designed for the nominal wheel radius results in a wide spread of the set of possible ending positions. Our controller brings all possible ending positions arbitrarily close to the desired goal position.

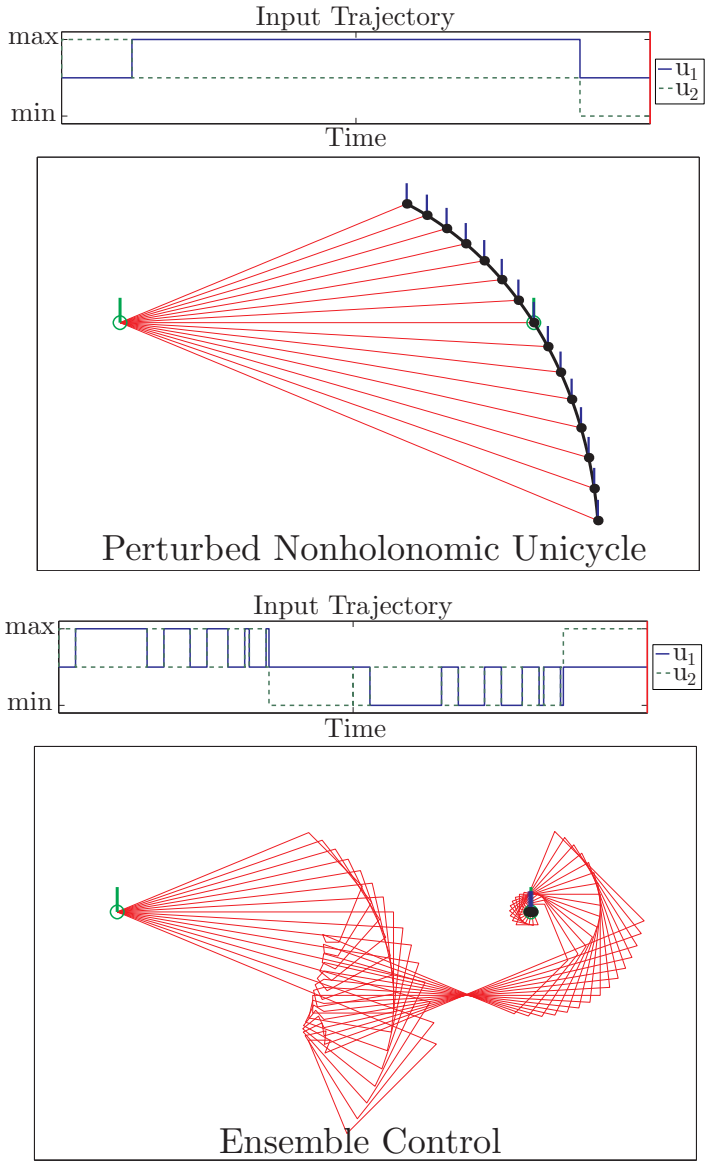


Figure 1.3: Piecewise constant input trajectories applied to a unicycle with unknown wheel radius. Top: a nominal controller results in a wide range of possible ending positions of the unicycle as a function of wheel size. Bottom: our controller brings all possible ending positions arbitrarily close to the desired goal position. Goal and start positions are drawn in green. The input trajectory for each controller is shown above the plots. Representative unicycles are drawn using black discs and a blue orientation marker. Red lines show state trajectories.
(Video online: http://www.youtube.com/watch?v=_v4KUBfzbv0)

1.1 Framework of Ensemble Control Theory

In this section we state formally what it means to be ensemble controllable. These definitions closely follow [3–14] and will be used throughout the dissertation for the systems shown in Fig. 1.2.

Definition 1. *Consider the family of control systems*

$$\dot{q}(t,\epsilon) = f\left(q(t,\epsilon), u(t), t, \epsilon\right),$$

where $q \in \mathcal{Q} \subset \mathbb{R}^n$, $u \in \mathcal{U} \subset \mathbb{R}^m$, $\epsilon \in [1-\delta, 1+\delta]$ *for* $0 \leq \delta < 1$, *and* f *is a smooth function. This family is* ensemble controllable *on the function space* $L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$ *if for all* $\mu > 0$ *and continuous* $q_{start}, q_{goal} \in L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$ *there exists* $T > 0$ *and piecewise-continuous* $u\colon [0,T] \to \mathcal{U}$ *such that* $q(0,\epsilon) = q_{start}(\epsilon)$ *and* $\|q(T,\epsilon) - q_{goal}(\epsilon)\| \leq \mu$ *for all* $\epsilon \in [1-\delta, 1+\delta]$.

Note that we allow q_{start} and q_{goal} to be arbitrary functions of ϵ. As pointed out by [13], the reader should interpret being ensemble controllable as being approximately controllable on $L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$.

To illustrate these concepts the following section focuses on a specific example, the nonholonomic unicycle. This system, shown on the left in Fig. 1.1, will be studied in Chapter 2 and revisited in Chapter 4.

1.1.1 A Case Study: An Ensemble of Unicycles

Consider Fig 1.4, a single unicycle that rolls without slipping. We describe its configuration by $q = (x, y, \theta)$ and its configuration space by $\mathcal{Q} = \mathbb{R}^2 \times \mathbb{S}^1$. The control inputs are the forward speed u_1 and the turning rate u_2. Corresponding to these inputs, we define vector fields $g_1, g_2\colon \mathcal{Q} \to T_q\mathcal{Q}$ by

$$g_1(q) = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \qquad\qquad g_2(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

and write the kinematics of the unicycle in the standard form

$$\dot{q}(t) = g_1(q(t))u_1(t) + g_2(q(t))u_2(t). \tag{1.1}$$

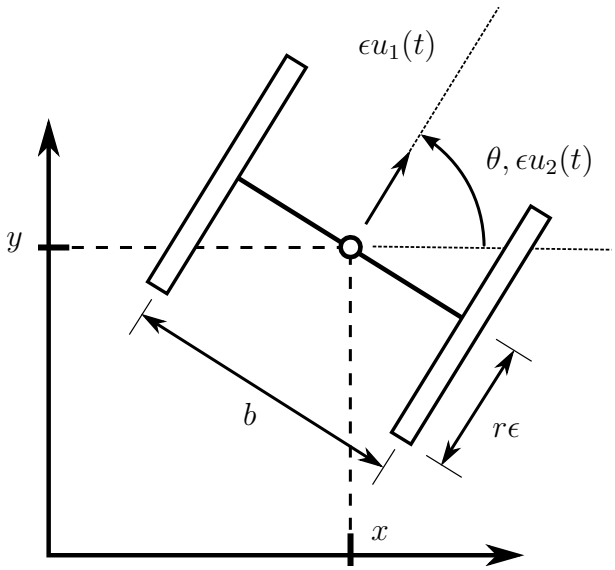


Figure 1.4: A differential-drive robot is a nonholonomic unicycle. This system has parameter perturbation that scales the wheel radius by an unknown, bounded constant ϵ. The state is (x, y, θ) with inputs $u_1(t)$ and $u_2(t)$.

Given $q_{\text{start}}, q_{\text{goal}} \in \mathcal{Q}$ and $\mu > 0$, the approximate steering problem is to find open-loop inputs

$$(u_1(t), u_2(t)) : [0, T] \to \mathcal{U}$$

that result in $q(0) = q_{\text{start}}$ and $\|q(T) - q_{\text{goal}}\| \leq \mu$ for free final time T, where $\|\cdot\|$ is a suitable norm on $\mathcal{Q}$. If such inputs always exist then we say that (1.1) is *approximately controllable*—and indeed they do, since g_1, g_2, and the Lie bracket $[g_1, g_2]$ span the tangent space $T_q\mathcal{Q}$ everywhere.

In Chapter 2 we will solve this same approximate steering problem, but under model perturbation that scales both the forward speed u_1 and the turning rate u_2 by some unknown, bounded constant. The resulting kinematics have the form

$$\dot{q}(t) = \epsilon \left(g_1(q(t))u_1(t) + g_2(q(t))u_2(t) \right), \tag{1.2}$$

where $\epsilon \in [1 - \delta, 1 + \delta]$ for some $0 \leq \delta < 1$. Rather than try to steer one unicycle governed by (1.2)—where ϵ is unknown—our approach is to steer an uncountably infinite collection of unicycles parameterized by ϵ, each one

governed by

$$\dot{q}(t,\epsilon) = \epsilon\big(g_1\left(q(t,\epsilon)\right)u_1(t) + g_2\left(q(t,\epsilon)\right)u_2(t)\big). \tag{1.3}$$

Following the terminology introduced in Section 1.1, we call this fictitious collection of unicycles an *ensemble* and call (1.3) an *ensemble control system*. The idea is that if we can find open-loop inputs $u_1(t)$ and $u_2(t)$ that result in $q(0,\epsilon) = q_{\text{start}}$ and $\|q(T,\epsilon) - q_{\text{goal}}\| \leq \mu$ for all $\epsilon \in [1-\delta, 1+\delta]$, then we can certainly guarantee that the actual unicycle, which corresponds to one particular value ϵ^* of ϵ, will satisfy $\|q(T,\epsilon^*) - q_{\text{goal}}\| \leq \mu$. If such inputs always exist then we say that (1.3) is *ensemble controllable*, interpreted as being approximately controllable on the function space $L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$. In Chapter 2.2 we will in fact show that (1.3) is not ensemble controllable, but will proceed to derive a reduced subsystem that is. Our proof will depend on being able to approximate arbitrary elements of the tangent space to $L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$, capturing the essence of classical tests like the Lie algebra rank condition. Solving the approximate steering problem with respect to the subsystem will produce inputs that reach an arbitrarily small neighborhood of any Cartesian position, but not of any heading.

We will apply our work to a differential-drive robot with unknown but bounded wheel radius, showing that (1.2) is an appropriate model and validating our approach to approximate steering with hardware experiments.

1.1.2 A Brief History of Ensemble Control Theory

Ensemble control, as presented in [3–10], extends the theory of nonlinear controllability from finite-dimensional systems, for example of the form (1.1), to a particular class of infinite-dimensional systems characterized by a dispersion parameter, for example of the form (1.3) where this parameter is ϵ. The fact that standard controllability theorems rely on checking a rank condition is a clue that such an extension might be necessary. Chow's theorem, for instance, implies that (1.1) is small-time locally controllable—hence, approximately controllable—because the Lie algebra generated by g_1 and g_2 has rank 3 everywhere, equal to the dimension of $T_q\mathcal{Q}$ (e.g., see [15]). Both $L_2\left([1-\delta, 1+\delta], \mathcal{Q}\right)$ and its tangent space have infinite dimension, so we will never accumulate enough vector fields to satisfy this same rank condition for

(1.3). However, we note that the rank condition is only used to guarantee that it is possible to approximate motion in any direction we like. For (1.1), we see that g_1, g_2, and

$$[g_1, g_2] = \frac{\partial g_2}{\partial q} g_1 - \frac{\partial g_1}{\partial q} g_2$$

are linearly independent and span the tangent space $T_q\mathcal{Q}$ at every configuration $q \in \mathcal{Q}$, so any element of $T_q\mathcal{Q}$ can be approximated by rapid switching between inputs. For (1.3), it is possible to arrive at a similar result. In particular, we will take the same basic approach as in [9], using repeated bracketing to get higher-order powers of ϵ and then using polynomial approximation to construct arbitrary vector flows. Systems like ours are ignored by [9] after noting that

$$\dot{q}(t, \epsilon) = \epsilon \sum_{i=1}^{m} g_i \left(q(t, \epsilon)\right) u_i(t)$$

is not ensemble controllable if $g_1, \ldots, g_m$ generate a nilpotent Lie algebra. We will indeed show that (1.3) is not ensemble controllable in Section 2.2, but will then proceed to derive a reduced subsystem that is ensemble controllable.

The origins of this approach are within the physics community. In this context, an "ensemble" is a very large collection of identical or nearly identical molecules, atoms, or elementary particles, and the goal of "ensemble control" is to manipulate the average properties of such an ensemble. Early work in this area was done, for example, by Simon van der Meer, who won the 1984 Nobel prize in physics for controlling the density at which circulating protons are packed in an accelerator using applied magnetic fields [16]. The more recent work of Brockett, Khaneja, and Li has found primary application so far to quantum systems, for example manipulating nuclear spins in nuclear magnetic resonance (NMR) spectroscopy [3–10]. Robotics researchers are also beginning to adopt the term "ensemble," for example in the context of multi-robot formations [17] and artificial muscle actuators [18], but the formal methodology of ensemble control has yet to be applied. Other approaches to dealing with infinite-dimensional systems—such as taking advantage of differential flatness—have been developed in parallel, as in [19]. The main tool used in this other work is functional analysis, which has recently started to inform ongoing work in ensemble control [13].

1.2 Alternative Frameworks to Ensemble Control Theory

We note that robots often have both proprioceptive (e.g., odometry) and exteroceptive (e.g., sonar, laser, vision) sensors. With state estimates that come from these sensors, it is easy to build a feedback controller that guarantees exact asymptotic convergence to any given Cartesian position under the same type of model perturbation that we consider [20, Chap. 11.6.2]. It is just as easy to build a robust feedback controller that regulates posture and not just Cartesian position [21]. Methods like these extend to a broader class of model perturbation (e.g., scaling forward speed and turning rate by different amounts) and to other types of uncertainty. There is also an enormous literature on odometry calibration for wheeled mobile robots to reduce model perturbation [22, 23], from offline approaches like "UMBmark" [24] to approaches that are online [25] and even simultaneous with localization [26]. These citations represent only a fraction of prior work on calibration and robust feedback control, all of which is more effective than what we propose when sensors are available.

Instead, we use the differential-drive robot as a hardware platform because the application of ensemble control theory to this system is easy to understand and leads to results that readers may find surprising. For example, our approximate steering algorithm—derived for an infinite-dimensional family of control systems and not just for a single unicycle—ultimately requires solving only one set of linear equations, which can be precomputed in closed form. Similarly, the formulation of these linear equations relies on series expansions that make explicit the trade-off between the cost and complexity of the resulting input trajectory and the extent to which this input trajectory is robust to model perturbation. Finally, the fact that inputs executed in open-loop will bring a real mobile robot to a neighborhood of the same Cartesian position regardless of wheel radius is something that we did not initially think possible (see the online video http://www.youtube.com/watch?v=8yYD_KMwfaM).

A similar case can be made for plate-ball manipulators. When sensor measurements of the ball's position and orientation are available, it is possible to use feedback control to overcome the parametric uncertainty we consider. Several practical stabilizing controllers were presented in [27–30]. Oriolo and Vendittelli presented an iterative feedback controller for stabilizing the plate-

ball system with guaranteed convergence even when the ball radius is set to an incorrect value [31, 32]. Our method neither uses nor takes advantage of sensor feedback, yet open-loop inputs bring a real sphere to a desired position and orientation regardless of the sphere radius (see the online video http://www.youtube.com/watch?v=nPGz0Nd3QzE).

1.2.1 Robust Control

Robust control provides a framework for the design of feedback policies that compensate for model perturbation. Consider the dynamic system

$$\begin{aligned} \dot{x} &= f(x, u, \epsilon) \\ y &= h(x, u, \epsilon), \end{aligned}$$

where x is the state, u is the input, y is the measurement, and ϵ is an unknown but bounded parameter. We are free to define an equivalent system

$$\begin{aligned} \dot{x} &= Ax + Bu + w_1 \\ y &= Cx + Du + w_2, \end{aligned}$$

for example by linearization, that pushes all model perturbation and nonlinearity into the mapping

$$(x, u, \epsilon) \mapsto (w_1, w_2)\,.$$

The idea is then to replace this one unknown mapping by a set of known linear mappings that capture all possible input/output behavior. This approach has been a topic of study for over fifty years—a modern reference is the book [33]. Although our own work has much the same flavor, robust control theory is primarily focused on the problem of closed-loop stabilization with feedback, whereas we focus on the problem of open-loop steering in the absence of sensor measurements. We emphasize again that robust feedback control is, in general, much more effective than what we propose when sensors are available, as they typically are for mobile robots. We also note previous work on robust feedforward control [34], on robust control using series expansions similar to what we will describe in Section 2.3 and 3.4.2 [35],

and on the relationship between ensemble control and robust control [36], this last work developed independently from a different perspective.

1.2.2 Motion Planning under Uncertainty

There is a vast literature on motion planning under uncertainty in robotics, excellent reviews of which may be found in texts such as [22, 37, 38] and examples of which range from early work on preimage backchaining [39] to very recent work on needle-steering using the stochastic motion roadmap [40]. As one example, we have drawn particular inspiration from work on sensorless manipulation [41]. In this work, like our own, the basic idea is to explicitly maintain the set of all possible robot configurations and to select a sequence of actions that reduces the size of this set and drives it toward some goal configuration. Carefully selected primitive operations can make this easier. For example, sensorless manipulation strategies often use a sequential composition of primitive operations, "squeezing" a part either virtually with a programmable force field or simply between two flat, parallel plates [42, 43]. Sensorless manipulation strategies also may take advantage of limit cycle behavior, for example engineering fixed points and basins of attraction so that parts only exit a feeder when they reach the correct orientation [44, 45]. These two strategies have been applied to a much wider array of mechanisms such as vibratory bowls and tables [46, 47] or assembly lines [42, 48, 49], and have also been extended to situations with stochastic uncertainty [50, 51] and closed-loop feedback [52, 53]. Our interest in this particular collection of work also stems from our belief that ensemble control theory may provide new insight into sensorless manipulation of many objects at once.

1.3 Applications of Ensemble Control to Robotics

In this dissertation, we apply the framework of ensemble control theory to derive an approximate steering algorithm for two classical robotic systems—the nonholonomic unicycle and the plate-ball manipulator—in the presence of model perturbation that scales all inputs by an unknown but bounded constant. Chapter 2 focuses on the nonholonomic unicycle and Chapter 3 focuses on the plate-ball manipulator.

In Chapter 4 we show how the same framework can be applied to feedback control of multi-robot systems under the constraint that every robot receives exactly the same control input. We focus on the nonholonomic unicycle, instantiated in experiment by a collection of differential-drive robots shown in Fig. 1.2c. Assuming that each robot has a unique wheel size, we derive a globally asymptotically stabilizing feedback control policy. We show that this policy is robust to standard models of noise and scales to an arbitrary number of robots. These results suggest that our approach may have possible future application to control of micro- and nano-scale robotic systems, which are often subject to similar constraints.

We conclude in Chapter 5 with a discussion about directions for future research.

Chapter 2

Approximate Steering of a Unicycle under Bounded Model Perturbation

In this chapter we apply the framework of ensemble control theory [3–14] to derive an approximate steering algorithm for a nonholonomic unicycle in the presence of model perturbation that scales both the forward speed and the turning rate by an unknown but bounded constant. The basic idea, similar to early work on sensorless manipulation [41], is to maintain the set of all possible configurations of the unicycle and to select inputs that reduce the size of this set and drive it toward some goal configuration. The key insight is that the evolution of this set can be described by a family of control systems that depend continuously on the unknown constant. Ensemble control theory provides conditions under which it is possible to steer this entire family to a neighborhood of the goal configuration with a single open-loop input trajectory. These conditions mimic classical tests of nonlinear controllability like the Lie algebra rank condition [54] but involve approximations by repeated Lie bracketing that are reminiscent of seminal work on steering nonholonomic systems by Lafferriere and Sussman [55].

2.1 Problem Statement

Consider a single unicycle that rolls without slipping. We describe its configuration by $q = (x, y, \theta)$ and its configuration space by $\mathcal{Q} = \mathbb{R}^2 \times \mathbb{S}^1$. The control inputs are the forward speed u_1 and the turning rate u_2. We restrict $(u_1, u_2) \in \mathcal{U}$ for some constraint set $\mathcal{U} \subset \mathbb{R}^2$, where we assume that $\mathcal{U}$ is symmetric with respect to the origin and that the affine hull of $\mathcal{U}$ is $\mathbb{R}^2$.

The material in this chapter appeared as [56] (© 2012 IEEE) and in a preliminary conference version [57].

Corresponding to these inputs, we define vector fields $g_1, g_2 \colon \mathcal{Q} \to T_q\mathcal{Q}$ by

$$g_1(q) = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \qquad\qquad g_2(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

and write the kinematics of the unicycle in the standard form

$$\dot{q}(t) = g_1(q(t))u_1(t) + g_2(q(t))u_2(t). \tag{2.1}$$

Given $q_{\text{start}}, q_{\text{goal}} \in \mathcal{Q}$ and $\mu > 0$, the approximate steering problem is to find open-loop inputs

$$(u_1(t), u_2(t)) : [0, T] \to \mathcal{U}$$

that result in $q(0) = q_{\text{start}}$ and $\|q(T) - q_{\text{goal}}\| \leq \mu$ for free final time T, where $\|\cdot\|$ is a suitable norm on $\mathcal{Q}$. If such inputs always exist then we say that (2.1) is *approximately controllable*—and indeed they do, since g_1, g_2, and the Lie bracket $[g_1, g_2]$ span the tangent space $T_q\mathcal{Q}$ everywhere.

We will solve this same approximate steering problem, but under model perturbation that scales both the forward speed u_1 and the turning rate u_2 by some unknown, bounded constant. The resulting kinematics have the form

$$\dot{q}(t) = \epsilon \left(g_1(q(t))u_1(t) + g_2(q(t))u_2(t)\right), \tag{2.2}$$

where $\epsilon \in [1 - \delta, 1 + \delta]$ for some $0 \leq \delta < 1$. Rather than try to steer one unicycle governed by (2.2)—where ϵ is unknown—our approach is to steer an uncountably infinite collection of unicycles parameterized by ϵ, each one governed by

$$\dot{q}(t, \epsilon) = \epsilon\big(g_1\left(q(t, \epsilon)\right) u_1(t) + g_2\left(q(t, \epsilon)\right) u_2(t)\big). \tag{2.3}$$

The remainder of this chapter is organized as follows. We begin in Section 2.2, by showing that (2.3) is not ensemble controllable but that a reduced subsystem is.

Based on this result, we derive an approximate steering algorithm in Section 2.3 that brings the unicycle to within an arbitrarily small neighborhood of any given Cartesian position, regardless of ϵ. Finally, in Section 2.4, we validate our approach in experiments with a differential-drive robot that has

unknown but bounded wheel radius.

2.2 Analysis of Controllability

In this section, we will establish controllability results for the system (2.3). Our method of approach will closely follow the one taken in [3–10].

2.2.1 Finding a Controllable Subsystem

We begin with a proof by construction of the following negative result, which was originally suggested by [5].

Theorem 2. *If $\delta > 0$, then the system* (2.3) *is not ensemble controllable.*

Proof: Notice that for any u_1 and u_2, we have

$$\dot{\theta}(t, \epsilon) = \epsilon u_2(t).$$

As a consequence, if we define an auxiliary state $\gamma(t)$ such that $\gamma(0) = 0$ and

$$\dot{\gamma}(t) = u_2(t),$$

then it is clear that

$$\theta(t, \epsilon) = \theta(0, \epsilon) + \epsilon\gamma(t)$$

for all $\epsilon \in [1 - \delta, 1 + \delta]$, where we assume without loss of generality that we are working in the coordinates of a single local chart on S^1. For any $\Delta\theta > 0$, choose

$$q_{\text{start}}(\epsilon) = (x_{\text{start}}(\epsilon), y_{\text{start}}(\epsilon), \theta_{\text{start}}(\epsilon))$$

and

$$q_{\text{goal}}(\epsilon) = (x_{\text{goal}}(\epsilon), y_{\text{goal}}(\epsilon), \theta_{\text{goal}}(\epsilon))$$

satisfying

$$\theta_{\text{goal}}(\epsilon) - \theta_{\text{start}}(\epsilon) = \Delta\theta$$

for all $\epsilon \in [1-\delta, 1+\delta]$, and let $\mu = \delta\Delta\theta/2$. We have

$$\begin{aligned}\|q(T,\epsilon) - q_{\text{goal}}(\epsilon)\| &\geq \|\theta(T,\epsilon) - \theta_{\text{goal}}(\epsilon)\| \\ &= \|\theta(0,\epsilon) + \epsilon\gamma(T) - \theta_{\text{goal}}(\epsilon)\| \\ &= \|\epsilon\gamma(T) - (\theta_{\text{goal}}(\epsilon) - \theta_{\text{start}}(\epsilon))\| \\ &= \|\epsilon\gamma(T) - \Delta\theta\|\,.\end{aligned}$$

Since we have assumed $\delta > 0$, then for any $\gamma(T)$ there exists some $\epsilon \in [1-\delta, 1+\delta]$ at which

$$\|\epsilon\gamma(T) - \Delta\theta\| > \mu,$$

and so (2.3) is not ensemble controllable by definition.

■

This result suggests the construction of a subsystem that, as we will show in the following section, is ensemble controllable. We write the configuration of this subsystem as

$$p(t,\epsilon) = (x(t,\epsilon), y(t,\epsilon), \gamma(t))\,,$$

where $\gamma(t)$ is the auxiliary state we introduced in the proof of Theorem 2.

We have just shown that the evolution of this subsystem is governed by the alternate kinematic model

$$\dot{p}(t,\epsilon) = \epsilon h_1\left(p(t,\epsilon),\epsilon\right) u_1(t) + h_2\left(p(t,\epsilon),\epsilon\right) u_2(t), \tag{2.4}$$

where

$$\begin{aligned} h_1\left(p(t,\epsilon),\epsilon\right) &= \begin{bmatrix} \cos\left(\theta(0,\epsilon) + \epsilon\gamma(t)\right) \\ \sin\left(\theta(0,\epsilon) + \epsilon\gamma(t)\right) \\ 0 \end{bmatrix} \\ h_2\left(p(t,\epsilon),\epsilon\right) &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \end{aligned} \tag{2.5}$$

and $\theta(0,\epsilon)$ is the initial heading given by q_{start}, as before. For convenience,

we will abbreviate

$$c(t,\epsilon) = \cos\left(\theta(0,\epsilon) + \epsilon\gamma(t)\right)$$
$$s(t,\epsilon) = \sin\left(\theta(0,\epsilon) + \epsilon\gamma(t)\right)$$

so that

$$h_1\left(p(t,\epsilon),\epsilon\right) = \begin{bmatrix} c(t,\epsilon) \\ s(t,\epsilon) \\ 0 \end{bmatrix}. \tag{2.6}$$

Since there is no longer any functional dependence of $p_3(t,\epsilon)$ on ϵ, it is clear that we have removed the feature of (2.3) that allowed us to conclude a lack of controllability. We will see that the resulting subsystem (2.4) is, in fact, controllable.

Before proceeding, notice that the vector field h_1 in (2.5) may be expressed

$$h_1\left(p(t,\epsilon),\epsilon\right) = R(\epsilon)\begin{bmatrix} \cos\left(\epsilon\gamma(t)\right) \\ \sin\left(\epsilon\gamma(t)\right) \\ 0 \end{bmatrix}$$

where

$$R(\epsilon) = \begin{bmatrix} \cos\theta(0,\epsilon) & -\sin\theta(0,\epsilon) & 0 \\ \sin\theta(0,\epsilon) & \cos\theta(0,\epsilon) & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

so if we apply the transformation

$$p'(t,\epsilon) = R(\epsilon)^T p(t,\epsilon),$$

then without loss of generality it is always possible to assume that $\theta(0,\epsilon) = 0$ for all ϵ.

2.2.2 Controllability By Polynomial Approximation

We will now prove that the reduced subsystem derived in the previous section is ensemble controllable. We will do this by using repeated bracketing to get higher-order powers of ϵ, and then by using polynomial approximation to construct arbitrary vector flows. This approach is similar to what appears in [9], and involves computations that are reminiscent of [55].

Theorem 3. *The system* (2.4) *is ensemble controllable.*

Proof: Taking Lie brackets, we have

$$\begin{aligned}
[\epsilon h_1, h_2] &= \epsilon \left(\frac{\partial h_2}{\partial p} h_1 - \frac{\partial h_1}{\partial p} h_2 \right) \\
&= 0 - \epsilon \begin{bmatrix} 0 & 0 & -\epsilon s \\ 0 & 0 & \epsilon c \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= \epsilon^2 \begin{bmatrix} s \\ -c \\ 0 \end{bmatrix}
\end{aligned}$$

and

$$\begin{aligned}
[[\epsilon h_1, h_2], h_2] &= 0 - \epsilon^2 \begin{bmatrix} 0 & 0 & \epsilon c \\ 0 & 0 & \epsilon s \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
&= -\epsilon^3 \begin{bmatrix} c \\ s \\ 0 \end{bmatrix} \\
&= -\epsilon^3 h_1.
\end{aligned}$$

Let us define

$$h_3 = \begin{bmatrix} -s \\ c \\ 0 \end{bmatrix},$$

so that $[\epsilon h_1, h_2] = -\epsilon^2 h_3$. Repeating this process, we can produce control vector fields of the form $\epsilon^{2i+1} h_1$ and $\epsilon^{2i+2} h_3$ for any $i \geq 0$. Since we have assumed that $\mathcal{U}$ is symmetric and that the affine hull of $\mathcal{U}$ is $\mathbb{R}^2$, then with piecewise-constant inputs (i.e., a sufficient number of "back-and-forth" maneuvers) we can produce flows of the form

$$\exp\left(a_0 \epsilon h_1\right) \cdots \exp\left(a_{k-1} \epsilon^{2k-1} h_1\right) = \exp\left(\sum_{i=0}^{k-1} a_i \epsilon^{2i+1} h_1 \right)$$

and

$$\exp\left(b_0\epsilon^2 h_3\right)\cdots\exp\left(b_{k-1}\epsilon^{2k}h_3\right)=\exp\left(\sum_{i=0}^{k-1}b_i\epsilon^{2i+2}h_3\right)$$

for freely chosen coefficients $a, b \in \mathbb{R}^k$. Let

$$p_{\text{start}}(\epsilon)=(x_{\text{start}}(\epsilon), y_{\text{start}}(\epsilon), \gamma_{\text{start}})$$

and

$$p_{\text{goal}}(\epsilon)=(x_{\text{goal}}(\epsilon), y_{\text{goal}}(\epsilon), \gamma_{\text{goal}})$$

for any given continuous real-valued functions

$$x_{\text{start}}, y_{\text{start}}, x_{\text{goal}}, y_{\text{goal}} \in L_2\left([1-\delta, 1+\delta], \mathbb{R}\right)$$

and for any given $\gamma_{\text{start}}, \gamma_{\text{goal}} \in S^1$. Define

$$c=\gamma_{\text{goal}}-\gamma_{\text{start}}$$

and take

$$\begin{bmatrix}\alpha(\epsilon)\\ \beta(\epsilon)\end{bmatrix}=\begin{bmatrix}\cos c & \sin c\\ -\sin c & \cos c\end{bmatrix}\begin{bmatrix}x_{\text{goal}}(\epsilon)-x_{\text{start}}(\epsilon)\\ y_{\text{goal}}(\epsilon)-y_{\text{start}}(\epsilon)\end{bmatrix}$$

for all $\epsilon \in [1-\delta, 1+\delta]$, where continuity of α and β follows from continuity of $x_{\text{start}}, x_{\text{goal}}, y_{\text{start}}, y_{\text{goal}}$. We can represent the desired change in configuration by the flow

$$\exp\left(\beta(\epsilon)h_3\right)\exp\left(\alpha(\epsilon)h_1\right)\exp\left(ch_2\right).$$

The Stone-Weierstrass theorem [58] tells us that given $\eta > 0$ and a continuous real function

$$\nu(\epsilon)\colon [1-\delta, 1+\delta] \to \mathbb{R},$$

there exists a polynomial function $\rho(\epsilon)$ such that

$$|\rho(\epsilon)-\nu(\epsilon)| < \eta$$

for all $\epsilon \in [\underline{\epsilon}, \bar{\epsilon}]$. An immediate corollary is that continuous real functions on the domain $[\underline{\epsilon}, \bar{\epsilon}] = [1-\delta, 1+\delta]$ for some $0 \le \delta < 1$ can be uniformly approximated either by an odd polynomial or by an even polynomial. (This result would not be true on an arbitrary domain, which is why we restrict

$\delta < 1$.) As a consequence, we can choose $a, b \in \mathbb{R}^k$ so that

$$\alpha(\epsilon) \approx \sum_{i=0}^{k-1} a_i \epsilon^{2i+1}$$
$$\beta(\epsilon) \approx \sum_{i=0}^{k-1} b_i \epsilon^{2i+2}$$

for $\epsilon \in [1-\delta, 1+\delta]$, with error vanishing in k. The time complexity of the resulting motion increases with k and with the number of switches required to approximate flows along each vector field $\epsilon^{2i+1} h_1$ and $\epsilon^{2i+2} h_3$, but remains finite for any given $\mu > 0$. Our result follows. ■

2.3 Approximate Steering Algorithm

In the previous section, we showed that the subsystem (2.4) is ensemble controllable. Based on this result, we will now derive an approximate steering algorithm for this subsystem. Although the boundary conditions given to this algorithm could in general be arbitrary continuous functions $p_{\text{start}}(\epsilon)$ and $p_{\text{goal}}(\epsilon)$, for our application of interest—where (2.4) captures the range of possible outcomes for a single unicycle—these functions are always constant and have the form

$$p_{\text{start}}(\epsilon) = (x_{\text{start}}, y_{\text{start}}, \gamma_{\text{start}})$$
$$p_{\text{goal}}(\epsilon) = (x_{\text{goal}}, y_{\text{goal}}, \gamma_{\text{goal}}),$$

where we may as well assume that $\gamma_{\text{start}} = \gamma_{\text{goal}} = \gamma$. If we apply the transformation

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \cos\gamma & \sin\gamma \\ -\sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} x_{\text{goal}} - x_{\text{start}} \\ y_{\text{goal}} - y_{\text{start}} \end{bmatrix},$$

then without loss of generality we may further assume that

$$p_{\text{start}}(\epsilon) = (0, 0, 0)$$
$$p_{\text{goal}}(\epsilon) = (\Delta x, \Delta y, 0).$$

Finally, we assume that $(1,0) \in \mathcal{U}$ and that $(v,1) \in \mathcal{U}$ for some $v \geq 0$, hence also that $(-1,0), (-v,-1) \in \mathcal{U}$. We make this assumption primarily for convenience. Scaling either input would require only scaling the corresponding time for which it is applied, and taking the reflection $-\Delta y$ would directly address the case where $(v,-1) \in \mathcal{U}$. However, the fact that it is possible to "go straight" is important for the simplicity of our algorithm. If $(1,0) \notin \mathcal{U}$, then we will assume that the corresponding control vector field is approximated by rapid switching, which is possible because the affine hull of $\mathcal{U}$ is $\mathbb{R}^2$. In any case, our model applies unchanged to both a differential-drive robot ($v = 0$) and a car-like robot ($v \neq 0$).

2.3.1 One Motion Primitive with Piecewise-Constant Inputs

Consider the following input for $\psi \geq 0$ and $a', b' \in \mathbb{R}$:

$$u(t) = \begin{cases} (v,1) & 0 \leq t < \psi \\ (\operatorname{sgn} a', 0) & \psi \leq t < \psi + |a'| \\ (-v,-1) & \psi + |a'| \leq t < 3\psi + |a'| \\ (\operatorname{sgn} b', 0) & 3\psi + |a'| \leq t < 3\psi + |a'| + |b'| \\ (v,1) & 3\psi + |a'| + |b'| \leq t < 4\psi + |a'| + |b'| \,. \end{cases}$$

We call this input a *motion primitive*. If $\gamma(0) = 0$, then the result of applying this motion primitive is to achieve

$$p(\Delta t, \epsilon) - p(0,\epsilon) = \begin{bmatrix} (a'+b')\epsilon \cos(\epsilon\psi) \\ (a'-b')\epsilon \sin(\epsilon\psi) \\ 0 \end{bmatrix}$$

in time

$$\Delta t = 4\psi + |a'| + |b'| \,.$$

With the input transformation

$$a' = \frac{a+b}{2} \qquad\qquad b' = \frac{a-b}{2}$$

for freely chosen $a, b \in \mathbb{R}$, we can write this expression as

$$p(\Delta t, \epsilon) - p(0, \epsilon) = \begin{bmatrix} a\epsilon \cos(\epsilon\psi) \\ b\epsilon \sin(\epsilon\psi) \\ 0 \end{bmatrix}$$

We will denote this motion primitive by the triple (a, b, ψ) and use it as the basis for our approximate steering algorithm.

2.3.2 Composition of Two Motion Primitives

Because our motion primitives leave γ invariant, we are free to concatenate them. For example, consider the sequential application of two primitives (a_1, b_1, ψ_1) and (a_2, b_2, ψ_2). If $\gamma(0) = 0$, then the result is to achieve

$$p(\Delta t, \epsilon) - p(0, \epsilon) = \begin{bmatrix} a_1\epsilon \cos(\epsilon\psi_1) + a_2\epsilon \cos(\epsilon\psi_2) \\ b_1\epsilon \sin(\epsilon\psi_1) + b_2\epsilon \sin(\epsilon\psi_2) \\ 0 \end{bmatrix}$$

in time

$$\Delta t = (4\psi_1 + |a_1'| + |b_1'|) + (4\psi_2 + |a_2'| + |b_2'|),$$

where

$$a_i' = \frac{a_i + b_i}{2} \qquad b_i' = \frac{a_i - b_i}{2}$$

for $i \in \{1, 2\}$. In fact, we can compose these two primitives in a slightly different way that achieves the same result in less time. Assume that $\psi_2 > \psi_1$.

Consider the following input:

$$u(t) = \begin{cases} (v, 1) & 0 \leq t < \psi_1 \\ (\operatorname{sgn} a_1', 0) & \cdots \leq t < \cdots + |a_1'| \\ (v, 1) & \cdots \leq t < \cdots + (\psi_2 - \psi_1) \\ (\operatorname{sgn} a_2', 0) & \cdots \leq t < \cdots + |a_2'| \\ (-v, -1) & \cdots \leq t < \cdots + (\psi_2 + \psi_1) \\ (\operatorname{sgn} b_1', 0) & \cdots \leq t < \cdots + |b_1'| \\ (-v, -1) & \cdots \leq t < \cdots + (\psi_2 - \psi_1) \\ (\operatorname{sgn} b_2', 0) & \cdots \leq t < \cdots + |b_2'| \\ (v, 1) & \cdots \leq t < \cdots + \psi_2. \end{cases}$$

It is easy to verify that $p(\Delta t, \epsilon) - p(0, \epsilon)$ remains the same but that

$$\Delta t = (|a_1'| + |b_1'|) + (4\psi_2 + |a_2'| + |b_2'|),$$

which is lower than before by $4\psi_1$. Figure 2.1 shows an example, for which $\psi_1 = \pi/4$, $\psi_2 = \pi/2$, and $v = 1/2$.

2.3.3 Composition of Many Motion Primitives

We generalize our result of the previous section as follows. Given $\phi > 0$, consider a sequence of $k + 1$ motion primitives

$$(a_{j+1}, b_j, \psi_j = j\phi)$$

for $j \in \{0, \ldots, k\}$, where we restrict $a_{k+1} = b_0 = 0$. We have indexed these primitives so that they are defined by the choice of $a, b \in \mathbb{R}^k$, where $a = (a_1, \ldots, a_k)$ and $b = (b_1, \ldots, b_k)$ as usual. We compose these primitives as in Section 2.3.2, noting that because $\psi_0 = 0$, the resulting inputs begin

with translation and not with rotation. In particular, we have

$$u(t) = \begin{cases} (\operatorname{sgn} a_1', 0) & 0 \le t < |a_1'| \\ (v, 1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} a_1', 0) & \cdots \le t < \cdots + |a_2'| \\ \quad \vdots & \\ (v, 1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} a_k', 0) & \cdots \le t < \cdots + |a_k'| \\ (-v, -1) & \cdots \le t < \cdots + k\phi \\ (\operatorname{sgn} b_1', 0) & \cdots \le t < \cdots + |b_1'| \\ (-v, -1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} b_2', 0) & \cdots \le t < \cdots + |b_2'| \\ \quad \vdots & \\ (-v, -1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} b_k', 0) & \cdots \le t < \cdots + |b_k'| \\ (v, 1) & \cdots \le t < \cdots + (k-1)\phi. \end{cases} \tag{2.7}$$

where

$$a' = \frac{1}{2}\left(\begin{bmatrix} a \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ b \end{bmatrix}\right) \qquad b' = \frac{1}{2}\left(\begin{bmatrix} a \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ b \end{bmatrix}\right).$$

As before, it is easy to verify that

$$p(\Delta t, \epsilon) - p(0, \epsilon) = \begin{bmatrix} \sum_{j=1}^{k} a_j \epsilon \cos\left(\epsilon(j-1)\phi\right) \\ \sum_{j=1}^{k} b_j \epsilon \sin\left(\epsilon j \phi\right) \\ 0 \end{bmatrix},$$

where

$$\Delta t = 4(k-1)\phi + \sum_{i=1}^{k} \left(|a_i'| + |b_i'|\right). \tag{2.8}$$

As in Section 2.2, our problem has been reduced to function approximation. Given $\mu > 0$ and $(\Delta x, \Delta y) \in \mathbb{R}^2$, we need to find $\phi > 0$ and $a, b \in \mathbb{R}^k$ for

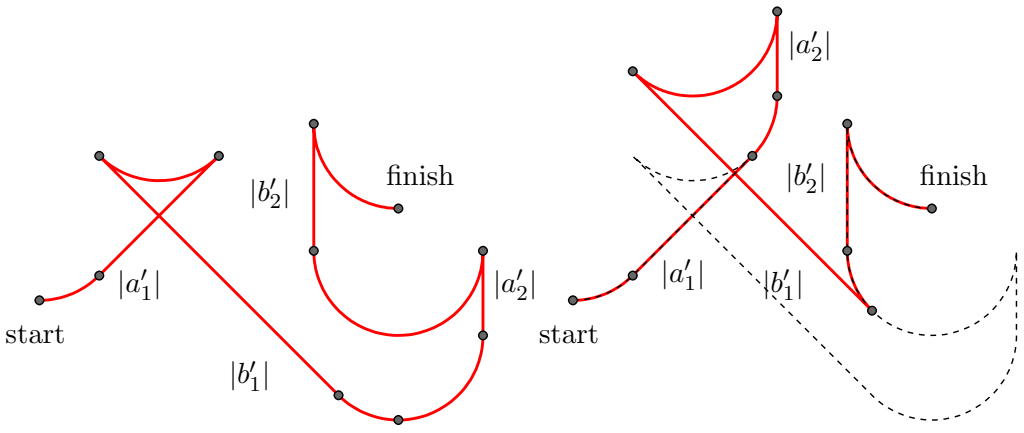


Figure 2.1: Two different ways to compose motion primitives $(a_1, b_1, \psi_1 = \pi/4)$ and $(a_2, b_2, \psi_2 = \pi/2)$, where $v = 1/2$. In (a), they are concatenated. In (b), they are interwoven, with the same result but lower execution time. The concatenation is shown in (b) as a dotted line for comparison. In this figure, we are only showing the nominal path, corresponding to $\epsilon = 1$.

sufficiently large k so that

$$\left| \Delta x - \sum_{j=1}^{k} a_j \epsilon \cos\left(\epsilon(j-1)\phi\right) \right| \leq \mu$$

and

$$\left| \Delta y - \sum_{j=1}^{k} b_j \epsilon \sin\left(\epsilon j \phi\right) \right| \leq \mu$$

for all $\epsilon \in [1-\delta, 1+\delta]$. The resulting input (2.7) would then be a solution to the approximate steering problem for (2.4).

2.3.4 Achieving Error of a Particular Order

In this section, we construct motion-plans composed of piecewise-constant inputs to achieve tolerances of a particular order in ϵ. An example path is shown in Fig. 2.2

We can express the result

$$\Delta p_1(\epsilon) = p_1(\Delta t, \epsilon) - p_1(0, \epsilon)$$

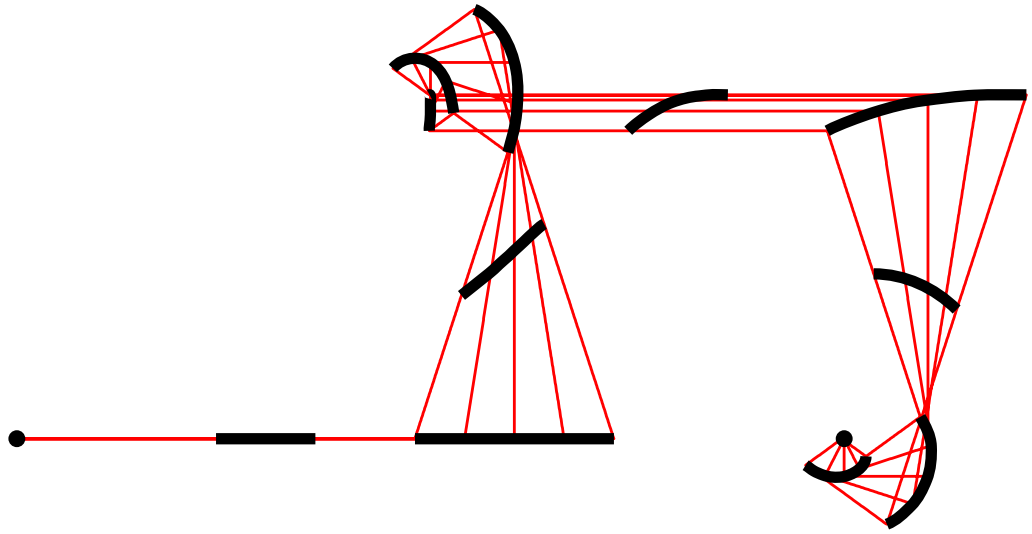


Figure 2.2: An ensemble with $\epsilon \in [0.8, 1.2]$ moving a unit distance in the x direction achieving 4th-order error in $|\epsilon - 1|$, which corresponds to a maximum error bound of $0.2^3 = 0.008$. Thin red lines show the path followed for particular values of ϵ. The actual robot follows only one of these paths. Thick black lines show the entire ensemble at instants of time. (Video online: http://www.youtube.com/watch?v=FppnS5xcRow)

$$\Delta p_2(\epsilon) = p_2(\Delta t, \epsilon) - p_2(0, \epsilon)$$

of applying (2.7) as Taylor series about $\epsilon = 1$:

$$\begin{aligned}
\Delta p_1(\epsilon) &= \Delta p_1(1) + \left(\frac{\partial \Delta p_1}{\partial \epsilon}\bigg|_{\epsilon=1} \right) (\epsilon - 1) + \cdots \\
\Delta p_2(\epsilon) &= \Delta p_2(1) + \left(\frac{\partial \Delta p_2}{\partial \epsilon}\bigg|_{\epsilon=1} \right) (\epsilon - 1) + \cdots .
\end{aligned}$$

Each series has the form

$$\begin{aligned}
\Delta p_1(\epsilon) &= \sum_{i=1}^{k} r_i \, (\epsilon - 1)^{i-1} + O\left(|\epsilon - 1|^k \right) \\
\Delta p_2(\epsilon) &= \sum_{i=1}^{k} s_i \, (\epsilon - 1)^{i-1} + O\left(|\epsilon - 1|^k \right),
\end{aligned}$$

where we collect $r = (r_1, \ldots, r_k)$ and $s = (s_1, \ldots, s_k)$ so that $r, s \in \mathbb{R}^k$.

Explicit formulas for r and s are given by

$$\begin{aligned} r &= Aa \\ s &= Bb, \end{aligned} \tag{2.9}$$

where the matrices $A, B \in \mathbb{R}^{k \times k}$ have elements

$$\begin{aligned} A_{ij} &= \frac{1}{(i-1)!} \left(\frac{\partial^{i-1} \left(\epsilon \cos\left(\epsilon (j-1) \phi \right) \right)}{\partial \epsilon^{i-1}} \bigg|_{\epsilon=1} \right) \\ B_{ij} &= \frac{1}{(i-1)!} \left(\frac{\partial^{i-1} \left(\epsilon \sin\left(\epsilon j \phi \right) \right)}{\partial \epsilon^{i-1}} \bigg|_{\epsilon=1} \right) \end{aligned}$$

for all $i, j \in \{1, \ldots, k\}$. Note that A and B do not depend on ϵ. To approximate $\Delta x = 1$ and $\Delta y = 1$ with error that is of order k in $|\epsilon - 1|$, we require only a solution a, b to the system of linear equations (2.9) that results in

$$\begin{aligned} r &= \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T \\ s &= \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T. \end{aligned}$$

Both A and B are square matrices, so assuming both are non-singular (and well-conditioned)—which will hold for almost all choices of the angle ϕ—then (2.9) has a unique solution. An immediate consequence is that achievable error decreases exponentially in the number $k+1$ of primitives, a result that was shown empirically in [57]. This relationship is highlighted in Fig. 2.3.

By linearity, if the parameters a and b achieve

$$(\Delta p_1, \Delta p_2) = (1, 1),$$

then the scaled parameters $a\Delta x$ and $b\Delta y$ achieve

$$(\Delta p_1, \Delta p_2) = (\Delta x, \Delta y)$$

for arbitrary Δx and Δy. In other words, scaling a single, precomputed maneuver gets you everywhere for free. Subsequently, we need only compute

$$a' = \frac{1}{2} \left(\begin{bmatrix} a \\ 0 \end{bmatrix} \Delta x + \begin{bmatrix} 0 \\ b \end{bmatrix} \Delta y \right)$$

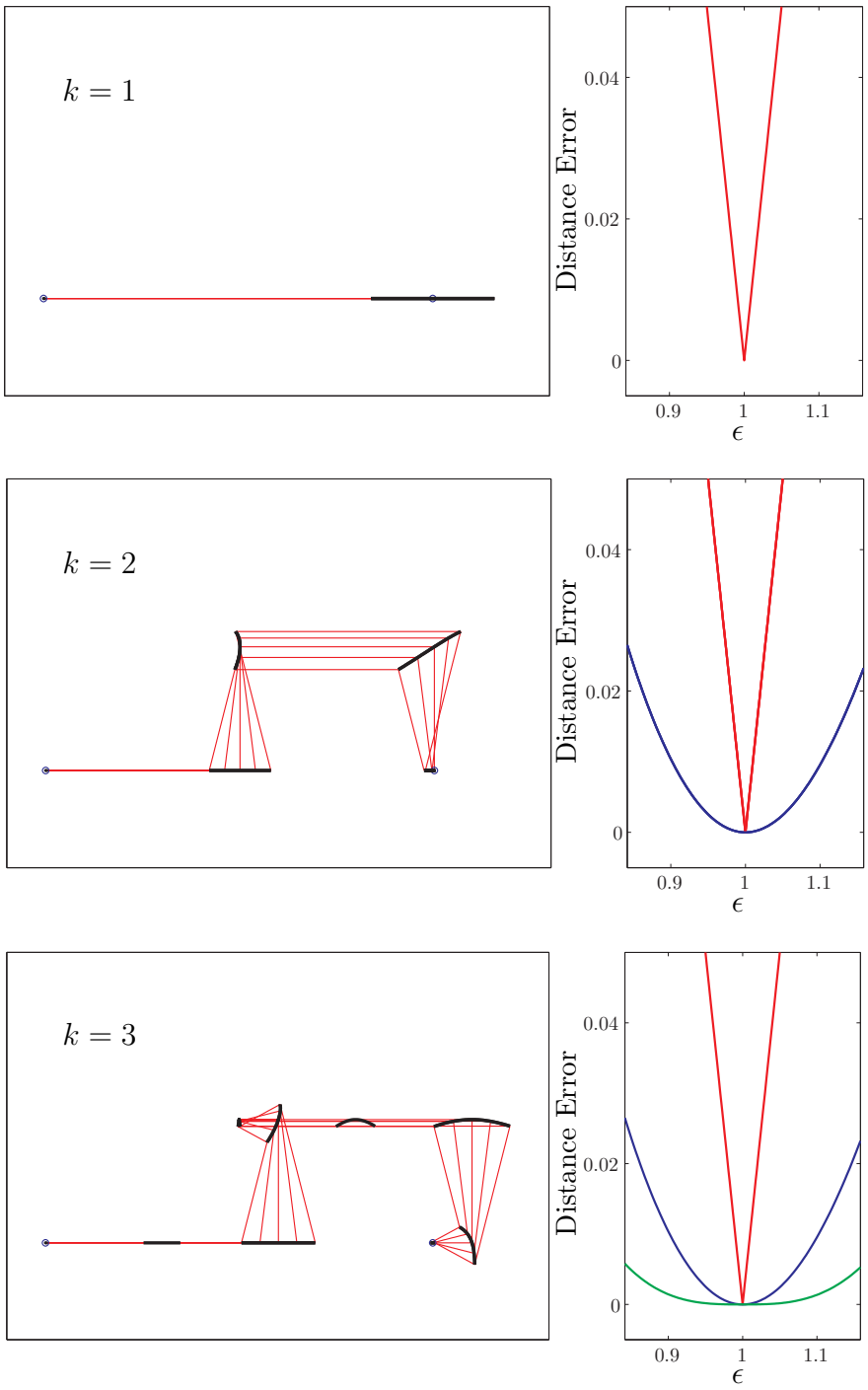


Figure 2.3: An ensemble with $\epsilon \in [0.8, 1.2]$ moving a unit distance in the x direction achieving different orders of error in $|\epsilon - 1|$. Thin red lines show the path followed for particular values of ϵ. The actual robot follows only one of these paths. Thick black lines show the entire ensemble at instants of time. The right column shows the resulting error as a function of ϵ.

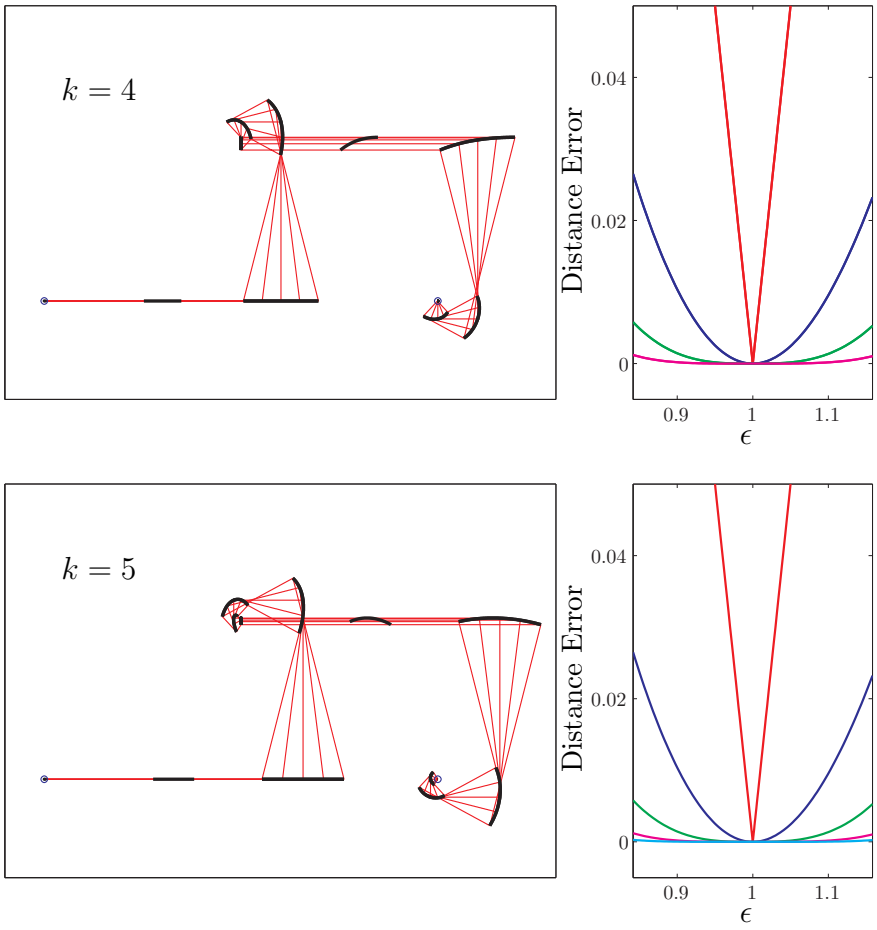


Figure 2.3: (Continued.)

$$b' = \frac{1}{2}\left(\begin{bmatrix} a \\ 0 \end{bmatrix} \Delta x - \begin{bmatrix} 0 \\ b \end{bmatrix} \Delta y\right).$$

One advantage of this strategy over the one used in [57] is that we no longer have to sample ϵ in order to compute the parameters a and b. Doing so had previously introduced approximation error that was difficult to quantify. Now, the series expansion gives us an explicit bound on this error. In particular, to achieve a tolerance $\mu > 0$, we simply choose any integer $k > 0$ that satisfies $\delta^{k-1} < \mu$.

We can also quantify the trade-off between the time cost Δt and the resulting uncertainty. The total elapsed time to reach $(\Delta x, \Delta y)$ with kth-order error in $|\epsilon - 1|$ is given by (2.8). Direct computation verifies that $|a_i'|$ and $|b_i'|$ decay rapidly with i, so the term $4(k-1)\phi$ dominates the elapsed time.

As a consequence, the time cost is $O(k)$. Note that if switching is required to generate $(u_1, u_2) = (1, 0)$—i.e., if $(1, 0) \notin \mathcal{U}$—this will only scale the cost by a constant factor.

Finally, we consider the total distance traveled in the workspace. If $\epsilon = 1$, this distance is given by

$$d(\Delta x, \Delta y) = 4(k-1)\phi v + \sum_{i=1}^{k} \left(|a_i'| + |b_i'| \right).$$

We may compute an upper bound on d by solving the following convex optimization problem, which is linear in the decision variables Δx and Δy:

$$\begin{aligned} \text{minimize} \quad & d(\Delta x, \Delta y) \\ \text{subject to} \quad & |\Delta x| \leq 1 \\ & |\Delta y| \leq 1. \end{aligned}$$

Call the solution to this problem $d_{\min}$. Recall that $\epsilon \in [1-\delta, 1+\delta]$ and $0 \leq \delta < 1$, so the distance traveled for any ϵ is at most $2d_{\min}$. As a corollary, we know that $(x(t, \epsilon), y(t, \epsilon))$ remains always inside a ball of radius $2d_{\min}$ during the application of our steering algorithm. This interesting result indicates that it might be possible to prove some form of small-time local controllability as the basis for extending our work from steering to motion planning (e.g., as in [1, 59]), although it is not obvious yet how to proceed.

Figures 2.4-2.5 provide pseudo-code that implements our approximate steering algorithm. We emphasize that this algorithm produces an open-loop input trajectory that neither requires nor takes advantage of sensor feedback.

2.3.5 Results in Simulation

Figure 2.2 shows the results of applying our approximate steering algorithm to an ensemble control system of the form (2.4) for which $\delta = 0.2$ and $(\Delta x, \Delta y) = (1, 0)$. In this example, we chose $k = 4$, so that maximum error is expected to be $O(\delta^4)$.

Equivalently, we expect that

$$0.008 = \delta^3 > \max_{\epsilon \in [1-\delta, 1+\delta]} \{\Delta p_1(\epsilon) - \Delta x\}$$

ComputePrimitive(ϕ, k)

Given an angle ϕ and a non-negative integer k, do the following:

- Compute the elements of $A, B \in \mathbb{R}^{k \times k}$ according to

$$A_{ij} = \frac{1}{(i-1)!} \left(\frac{\partial^{i-1} \left(\epsilon \cos\left(\epsilon (j-1) \phi \right) \right)}{\partial \epsilon^{i-1}} \bigg|_{\epsilon=1} \right)$$
$$B_{ij} = \frac{1}{(i-1)!} \left(\frac{\partial^{i-1} \left(\epsilon \sin\left(\epsilon j \phi \right) \right)}{\partial \epsilon^{i-1}} \bigg|_{\epsilon=1} \right)$$

  for all $i, j \in \{1, \ldots, k\}$.

- Compute $a, b \in \mathbb{R}^k$ according to

$$a = A^{-1} r$$
$$b = B^{-1} s,$$

  where

$$r = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T$$
$$s = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T.$$

Return (a, b).

Figure 2.4: The algorithm that we use to precompute a motion primitive.

$$0.008 = \delta^3 > \max_{\epsilon \in [1-\delta, 1+\delta]} \left\{ \Delta p_2(\epsilon) - \Delta y \right\}.$$

We chose $\phi = \pi/2$.

As a consequence, the matrices A and B have a very simple form:

$$A = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 1 & -\pi/2 & -1 & 3\pi/2 \\ 0 & -\pi/2 & \pi^2/2 & 3\pi/2 \\ 0 & \pi^3/48 & \pi^2/16 & -9\pi^3/16 \end{bmatrix}$$

and

$$B = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 1 & -\pi & -1 & 2\pi \\ -\pi^2/8 & -\pi & 9\pi^2/8 & 2\pi \\ -\pi^2/8 & \pi^3/6 & 9\pi^2/8 & -4\pi^3/3 \end{bmatrix}.$$

ComputeInput$(\phi, k, x_{\text{start}}, y_{\text{start}}, x_{\text{goal}}, y_{\text{goal}}, \gamma)$

Given an angle ϕ, a non-negative integer k, a start configuration $(x_{\text{start}}, y_{\text{start}}, \gamma)$, and a goal configuration $(x_{\text{goal}}, y_{\text{goal}}, \gamma)$, do the following:

- Compute the motion primitive

$$(a, b) = \text{ComputePrimitive}(\phi, k).$$

- Compute the desired change in Cartesian position

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \cos\gamma & \sin\gamma \\ -\sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} x_{\text{goal}} - x_{\text{start}} \\ y_{\text{goal}} - y_{\text{start}} \end{bmatrix}.$$

- Compute the parameters

$$a' = \frac{1}{2}\left(\begin{bmatrix} a \\ 0 \end{bmatrix} \Delta x + \begin{bmatrix} 0 \\ b \end{bmatrix} \Delta y \right)$$
$$b' = \frac{1}{2}\left(\begin{bmatrix} a \\ 0 \end{bmatrix} \Delta x - \begin{bmatrix} 0 \\ b \end{bmatrix} \Delta y \right).$$

- Compute the piecewise-constant input

$$u(t) = \begin{cases} (\operatorname{sgn} a'_1, 0) & 0 \le t < |a'_1| \\ (v, 1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} a'_1, 0) & \cdots \le t < \cdots + |a'_2| \\ \quad \vdots & \\ (v, 1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} a'_k, 0) & \cdots \le t < \cdots + |a'_k| \\ (-v, -1) & \cdots \le t < \cdots + k\phi \\ (\operatorname{sgn} b'_1, 0) & \cdots \le t < \cdots + |b'_1| \\ (-v, -1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} b'_2, 0) & \cdots \le t < \cdots + |b'_2| \\ \quad \vdots & \\ (-v, -1) & \cdots \le t < \cdots + \phi \\ (\operatorname{sgn} b'_k, 0) & \cdots \le t < \cdots + |b'_k| \\ (v, 1) & \cdots \le t < \cdots + (k-1)\phi. \end{cases}$$

Return u.

Figure 2.5: Our approximate steering algorithm. It acts only to scale the primitive generated by the subroutine ComputePrimitive, which need only be called once for given ϕ and k. We recommend choosing $\phi = \pi/2$ and the smallest integer k such that $\delta^{k-1} < \mu$ for a given tolerance $\mu > 0$.

The linear equations (2.9) can be solved exactly to produce

$$a = \begin{bmatrix} 1 + (2/\pi^2) \\ 3\,(8 + 3\pi^2)\,/\,(4\pi^3) \\ 2/\pi^2 \\ (24 + \pi^2)\,/\,(12\pi^3) \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} (9/8) + (1/\pi^2) \\ (6 + 4\pi^2)\,/\,(3\pi^3) \\ (1/8) + (1/\pi^2) \\ (6 + \pi^2)\,/\,(6\pi^3) \end{bmatrix}.$$

We also compute exactly the bound on total distance traveled, in this case achieved when $(\Delta x, \Delta y) = (-1, -1)$:

$$d_{\min} = \frac{9}{4} + \frac{6 + \pi(8 + 3\pi)}{2\pi^3} \approx 3.23.$$

We verify in simulation that the maximum error is 0.003 and that the distance traveled is 2.41, both satisfying our predicted bounds. Figure 2.6 shows the same precomputed motion primitive scaled to reach a variety of goal configurations.

2.4 Hardware Experiments

In this section we apply our approximate steering algorithm to a differential-drive robot with unknown but bounded wheel radius. First, we describe the robot that we used. Then, we show that (2.2) is an appropriate model of this robot. Finally, we show the results of hardware experiments.

2.4.1 Experimental Setup

Figure 2.7 shows the robot we used in our experiments. It is a differential-drive robot with a caster wheel in front for stability. It moves on a flat tile floor and uses only dead-reckoning for navigation. In particular, the robot runs a feedback control loop to read the wheel encoders, update a dead-reckoning position estimate, and regulate the speed of each motor.

Although we use no other sensors for feedback control, global position data is available from an off-board vision system for later analysis. This vision system records pose information at 27 Hz with a position accuracy of 2 cm and an orientation accuracy of 1°.

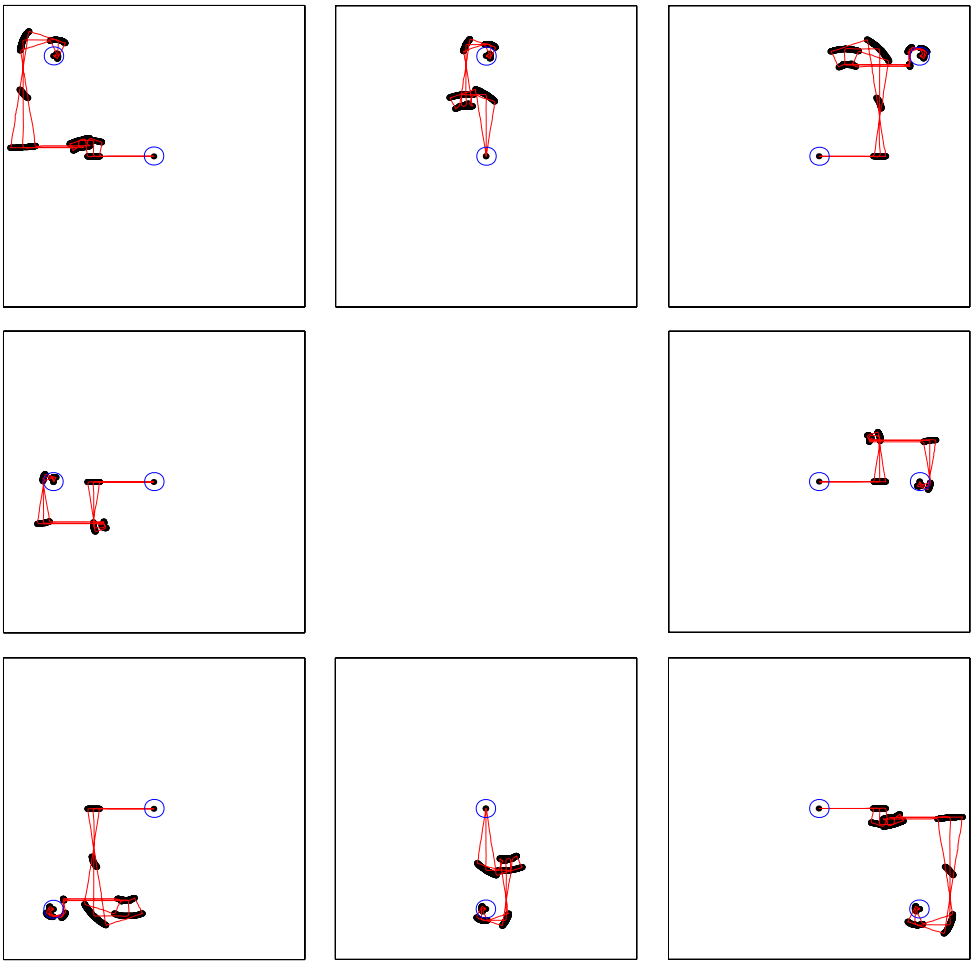


Figure 2.6: Example trajectories for $k = 4$ and $\phi = \pi/2$. All of them are scaled versions of the maneuver shown in Fig. 2.2. Thin red lines are particular values of ϵ, thick black lines are the entire ensemble at instants of time. (Video online: http://www.youtube.com/watch?v=DcDCg0Ne0vo)

Before conducting our experiments, we applied a standard calibration procedure to find the effective wheelbase and wheel radius in order to reduce systematic dead-reckoning error [24,61]. The calibration was done with wheels of diameter 12.7 cm. However, these wheels are interchangeable—in our experiments, we used four sets that varied between 10.16-15.24 cm in diameter, as shown in Fig. 2.8. We did not recalibrate for these other wheels, and assumed that the wheel diameter was unknown but bounded in the set $[10.2, 15.2]$, or in other words the set $[0.8, 1.2]$ relative to the nominal diameter 12.7 cm.

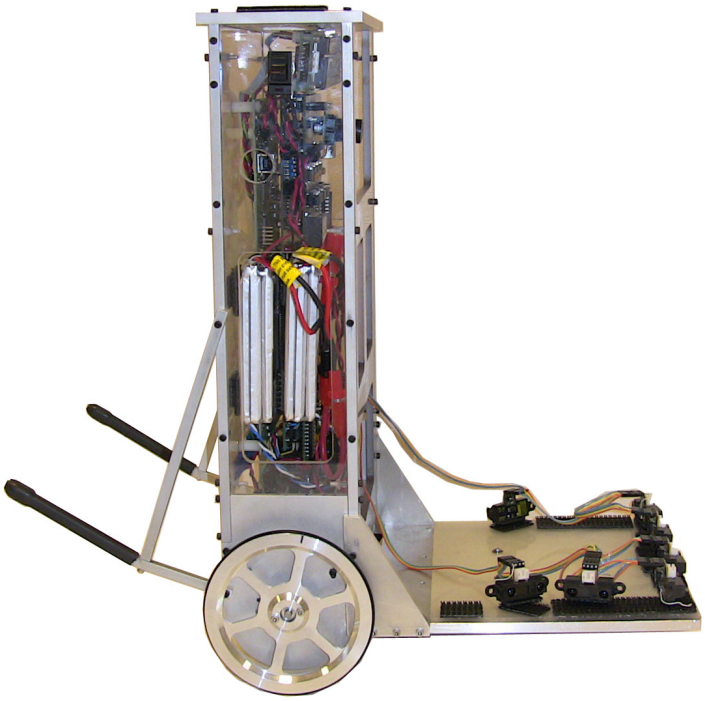


Figure 2.7: The differential-drive robot used for experimental validation. Robot courtesy of College of Engineering Control Systems Laboratory [60].

2.4.2 Application of the Model to a Differential-Drive Robot

We will show that

$$\dot{q}(t) = \epsilon \left(g_1(q(t))u_1(t) + g_2(q(t))u_2(t) \right)$$

is a valid kinematic model of our robot, where

$$g_1(q) = \begin{bmatrix} \cos q_3 \\ \sin q_3 \\ 0 \end{bmatrix} \qquad \text{and} \qquad g_2(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

It suffices to show that the forward speed v and turning rate ω of a differential-drive robot with unknown but bounded wheel radius are given by $v = \epsilon u_1$ and $\omega = \epsilon u_2$, respectively, for control inputs $u_1, u_2 \in \mathbb{R}$. Recall that for wheel radius r and wheel separation l, the forward speed and turning rate of a differential-drive robot are given by

$$v = \frac{r(\omega_R + \omega_L)}{2} \qquad \text{and} \qquad \omega = \frac{r(\omega_R - \omega_L)}{l},$$

Figure 2.8: Four wheel sizes used for experimental validation. These wheels are 10.16, 10.48, 12.7 and 15.24 cm in diameter.

where ω_R and ω_L are the angular velocities of the right and left wheels, respectively. Assume that the wheel radius, a positive constant, is unknown but bounded according to $r \in [r_{\min}, r_{\max}]$. If we define

$$\bar{r} = \frac{r_{\max} + r_{\min}}{2} \qquad \text{and} \qquad \delta = \frac{r_{\max} - r_{\min}}{2\bar{r}}$$

then we can write $r = \epsilon\bar{r}$ for some $\epsilon \in [1-\delta, 1+\delta]$, so that

$$v = \epsilon\left(\frac{\bar{r}(\omega_R + \omega_L)}{2}\right) \quad \text{and} \quad \omega = \epsilon\left(\frac{\bar{r}(\omega_R - \omega_L)}{l}\right).$$

This expression simplifies if we select wheel angular velocities

$$\omega_R = \frac{2u_1 + bu_2}{2\bar{r}} \quad \text{and} \quad \omega_L = \frac{2u_1 - bu_2}{2\bar{r}}$$

for any given $u_1, u_2 \in \mathbb{R}$, so that

$$v = \epsilon u_1 \quad \text{and} \quad \omega = \epsilon u_2,$$

and we have our result.

2.4.3 Experimental Results

Figures 2.9 and 2.10 show the results of our experiments, which successfully validated our approach. The start configuration was $(0, 0, 0)$. The goal configuration was $(4.25 \text{ m}, 2.25 \text{ m}, 0)$. The value of k was chosen to achieve an error tolerance of 2 cm. We applied the algorithm described in Section 2.3 to generate a single input trajectory that was applied in open-loop. Five runs

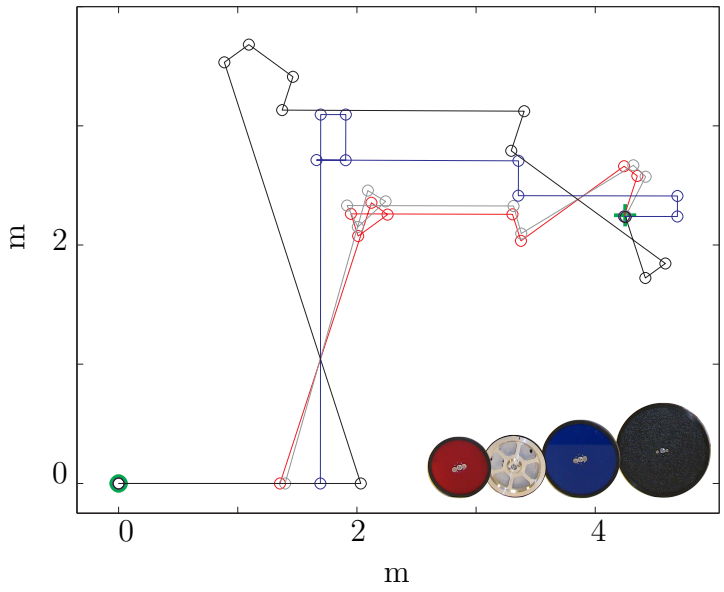


Figure 2.9: Expected paths for four wheel sizes on a move to [4.25,2.25] m. Red, grey, blue and black plots correspond to 10.16, 10.48, 12.7 and 15.24 cm wheels, respectively.

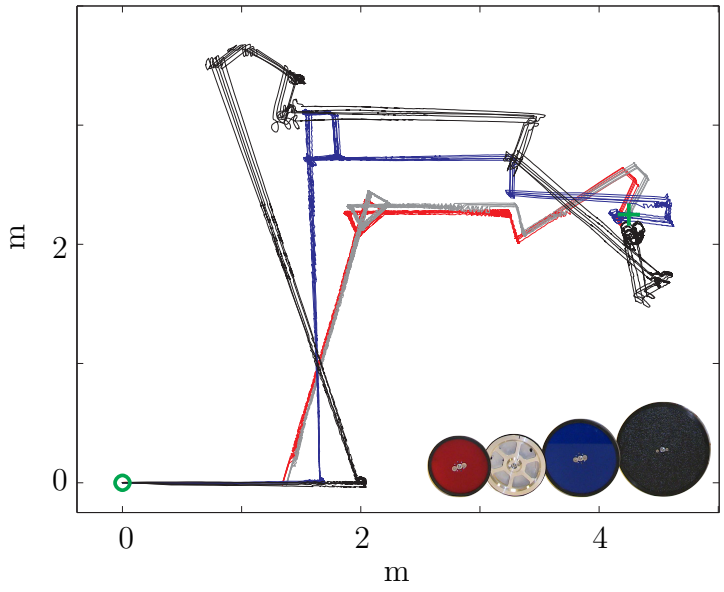


Figure 2.10: Ground truth data gathered from the camera system. Five runs for each wheel set are shown. Loops at the corners are artifacts from the camera system. Red, grey, blue and black plots correspond to 10.16, 10.48, 12.7 and 15.24 cm wheels, respectively.
(Video online: http://www.youtube.com/watch?v=8yYD_KMwfaM)

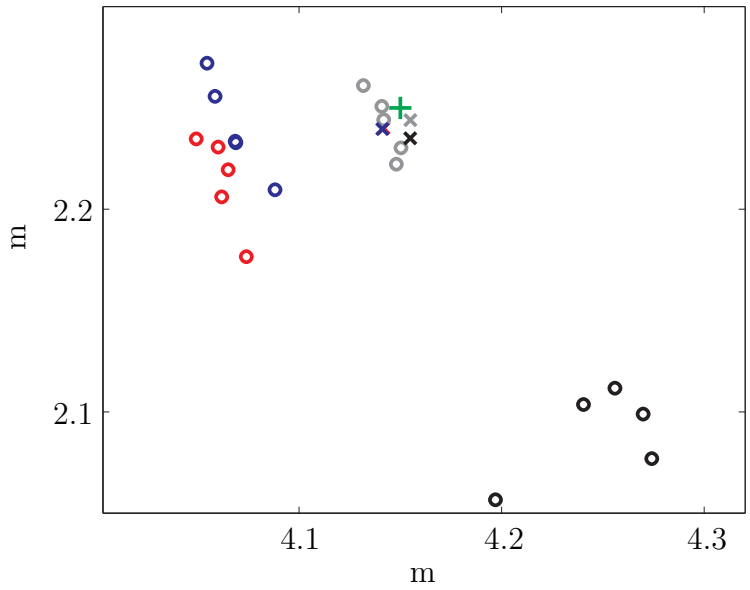


Figure 2.11: Ending position for each run. Green '+' for goal position, 'x' for expected ending position under zero odometry drift, 'o' for actual ending positions. Red, grey, blue and black plots correspond to 10.16, 10.48, 12.7 and 15.24 cm wheels. (Note the zoomed scale.)

were recorded for each wheel size. All of the resulting trajectories reached a small neighborhood of the goal position, as shown in Fig. 2.11 and reported in aggregate in Table 2.1 and Fig. 2.12. The size of this neighborhood is slightly larger than the predicted tolerance of 2 cm. This error is due largely to drift as a result of wheel slip, gear backlash, surface irregularities, wheel flex, and other disturbances. Another contributing factor is that different wheels are made of different materials. The 10.48 cm wheels are aluminum with rubber o-rings stretched over the rim, while the other wheels are ABS plastic with a molded rubber traction ring on the rim. The edge of each plastic wheel has a rectangular cross-section, making the effective wheel base slightly larger than for the aluminum wheels. The vision system also adds to observed error (although we emphasize that this vision system was used only for data collection and not for closed-loop feedback in our experiments). In particular, ground truth position information was calculated from fiducial markers on the top of the robot. These markers were level and centered over the wheelbase for the 10.48 cm wheels, but tilted by 10° for the largest wheels.

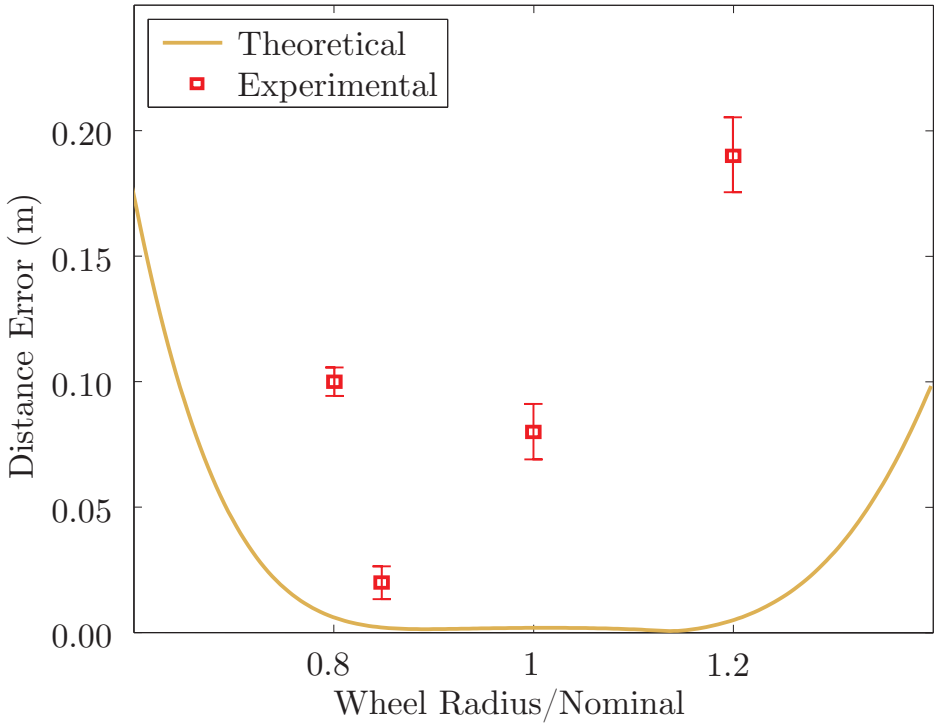


Figure 2.12: Distance error for four wheel sizes on a move to [4.25,2.25]. This test was repeated 5 times per wheel size. The yellow line is the theoretical error. The red line shows the mean errors for each wheel size tested ± one standard deviation.

Table 2.1: In-group Error Measurements

wheel diam(cm)	distance mean(m)	distance var (m^2)	θ mean(rad)	θ var$(rad)^2$
10.16	0.10	3.2e-5	-0.002	1.9e-6
10.48	0.02	4.1e-5	-0.002	6.1e-7
12.70	0.08	1.2e-4	-0.008	2.5e-5
15.24	0.19	2.2e-4	-0.017	1.9e-4

2.5 Conclusion

In this chapter we applied the framework of ensemble control theory to derive an approximate steering algorithm that brings a nonholonomic unicycle to within an arbitrarily small neighborhood of any given Cartesian position despite model perturbation that scales both the forward speed and the turning rate by an unknown but bounded constant. This algorithm has trivial computational complexity, requiring only the solution of linear equations. We validated our approach using a differential-drive robot with unknown but bounded wheel radius and showed the results with hardware experiments.

In Chapter 3, we apply a similar approach to a different robotic system, the plate-ball system. In this case the full state of the plate-ball system is controllable. Later, in Chapter 4 we revisit robots with unicycle kinematics. In that chapter, instead of steering one robot with an unknown parameter, we are interested in steering many such robots using a common control input.

Chapter 3

Approximate Steering of Plate-Ball System under Bounded Model Perturbation

In this chapter we revisit the classical plate-ball system and prove this system remains controllable under model perturbation that scales the ball radius by an unknown but bounded constant. We present an algorithm for approximate steering and validate the algorithm with hardware experiments. To perform these experiments we introduce a new version of the plate-ball system based on magnetic actuation shown in Fig. 3.1. This system is easy to implement and, with our steering algorithm, enables simultaneous manipulation of multiple balls with different radii.

3.1 Introduction

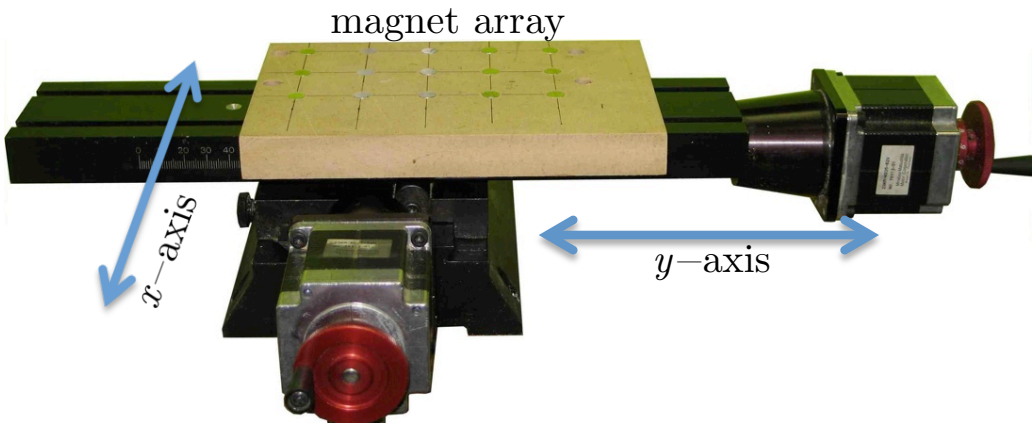
The plate-ball system is a canonical example of manipulation by rolling contacts [2]. In the classical version of this system, the ball is held between two parallel plates and manipulated by maneuvering the upper plate while holding the lower plate fixed. The ball can be brought to any position and orientation though translations of the upper plate.

We consider a variant of the plate-ball system in which the ball radius is an unknown but bounded constant. This variant has been considered previously by Oriolo et al. [32], who proposed a method of iterative feedback control. We are interested in the case where no sensor feedback is available.

Our main contribution is to prove this system is still controllable and present an algorithm for approximate (open-loop) steering. To do so, we apply the framework of ensemble control theory [3, 4, 9, 10, 56] to derive an approximate steering algorithm. The basic idea, similar to early work on sensorless manipulation [41], is to maintain the set of all possible configurations of the sphere and to select inputs that reduce the size of this set and drive it

Material from this chapter appeared in a preliminary conference version as [62].

(a) ensemble plate-ball system



(b) underlying mechanism

Figure 3.1: The ensemble plate-ball system consists of an array of n metal spheres separated from an array of n magnets by a stationary sheet of cardboard. The magnet array is attached to an xy CNC table, and the spheres roll without slipping above their respective magnets.
(Video online: http://www.youtube.com/watch?v=nPGz0Nd3QzE)

toward some goal configuration. The key insight is that the evolution of this set can be described by a family of control systems that depend continuously on the unknown constant. Ensemble control theory provides conditions under which it is possible to steer this entire family to a neighborhood of the goal configuration with a single open-loop input trajectory. These conditions mimic classical tests of nonlinear controllability like the Lie algebra rank condition [54] but involve approximations by repeated Lie bracketing that are reminiscent of seminal work on steering nonholonomic systems by Lafferriere and Sussman [55].

Our second contribution is a new experimental version of the plate-ball system based on magnetic actuation, shown in Fig. 3.1. This platform has several advantages over traditional plate-ball systems. The magnetic actuation makes the mechanism easy to implement, allows an unobstructed view of the ball on the plane, and enables simultaneous manipulation of multiple balls. If these balls have unique radii, by using our algorithm this system can steer each ball to independent arbitrary orientations and simultaneously translate the balls to a desired position.

This result hints at a new approach to robust manipulation of so-called "toleranced parts" [63, 64], an ongoing problem in automated assembly and industrial parts handling. A considerable amount of work remains to be done, however, before ideas like this one find their way into practice.

3.1.1 Manipulation by Rolling

Manipulation of spherical objects by rolling has been investigated in depth by members of the math, control, and robotic manipulation community. This research can be traced to Brockett and Dai who analyzed an approximation of the problem and determined the optimal controller for this approximation [2]. Jurdjevic analyzed the optimal shortest-length paths, showing that the optimal solution curve minimizes the integral of the geodesic curvature [65]. Two optimal shortest-length paths (calculated using the psuedospectral optimization solver GPOPS [66–70]) are shown in Fig. 3.2.

Li provided a symbolic algorithm for steering a plate-ball system [71], while Marigo gave a numeric algorithm [72]. Several practical stabilizing controllers were presented in [27–30], and Oriolo and Vendittelli presented an iterative

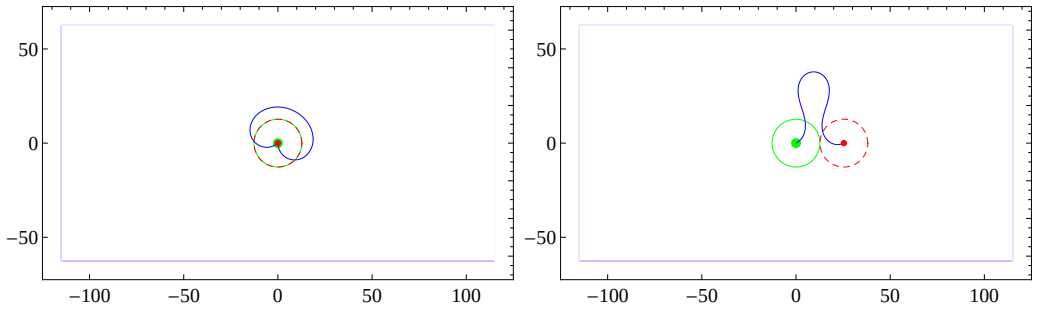


Figure 3.2: Optimal shortest paths for a single sphere rolling on the plane to a desired position and orientation. This optimal control problem was first investigated by Jurdjevic [65]. Green shows the initial ball position, dashed red the final position, and the path of the contact point between the ball and plate is in blue. The left path creates a net rotation of π about the z-axis with no change in position while the right path translates the ball 1 diameter in the positive x direction with no change in orientation.

feedback controller for stabilizing the plate-ball system [31, 32]. Oriolo and Vendittelli's method is noteworthy because it guarantees asymptotic stability with exponential convergence. This policy iterates between driving the orientation error to zero in finite time and controlling the position error to zero while cycling the orientation variables. This feedback control policy is robust to a perturbed radius value. A MATLAB implementation of their policy is available online [73], and can be used to generate the results shown in Fig. 3.3. When sensors are available, robust feedback control is, in general, more effective than what we propose. However, these control techniques do not generalize to simultaneously control multiple balls with the same plate-ball system.

Svinin and Hosoe extended the problem to plate-ball systems with limited contact area [74, 75]. This enables the manipulation of objects that are only partially spherical, such as a hemisphere.

Several robotic plate-ball systems have been implemented (see [72, 76]). Our approach using multiple balls is inherently underactuated, and in that respect is similar to Choudhury and Lynch's work that showed a single degree-of-freedom manipulator was sufficient for orienting a sphere. They designed a hardware system consisting of an elliptical bowl mounted on top of a linear motor with the bowl's primary axis oriented 45 degrees from the linear motor orientation [77]. By applying acceleration profiles to the linear motor, their

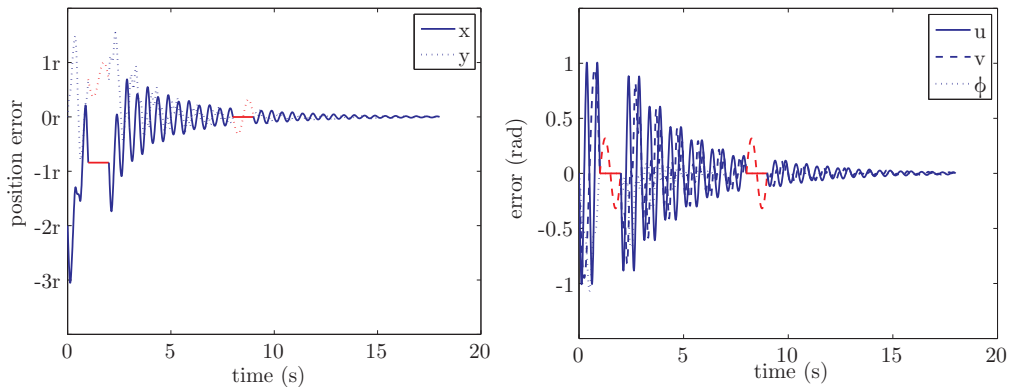


Figure 3.3: Oriolo and Vendittelli presented an iterative approach for stabilizing the plate-ball system. Our implementation of their algorithm is shown above for moving a ball 1 diameter in the positive x direction with no net change in orientation. This policy iterates between driving the orientation error to zero in finite time (red lines) and controlling the position error to zero while cycling the orientation variables (blue lines).

approach steered the orientation to a desired coordinate in SO(3).

Lastly, the control algorithms demonstrated in this work might find application at a much smaller scale using microspheres. Studies by Ding et al. on rolling friction of microspheres [78] demonstrate that even on the micro-scale spheres can roll with little slip. A study by Agayan et al. of the slipping friction of optically and magnetically manipulated microspheres on glass-water interface demonstrated techniques for manipulation that may benefit from our methodology [79].

The remainder of this chapter proceeds as follows. We begin with our problem statement (Section 3.2). We then prove our problem is controllable (Section 3.3). We design an approximate steering algorithm (Section 3.4), and validate the algorithm with a new hardware platform (Section 3.5). We discuss broader implications in our concluding remarks (Section 3.6).

3.2 Problem Statement

Consider the system shown in Fig. 3.4, a sphere of radius r that rolls without slipping on a horizontal plane. Following Jurdjevic [65], we describe its configuration q by $\left([x, y]^\top, R\right)$ consisting of the position $[x, y]^\top$ and the rotation matrix R. The configuration space $\mathcal{Q}$ is $\mathbb{R}^2 \times SO(3)$. The control inputs are

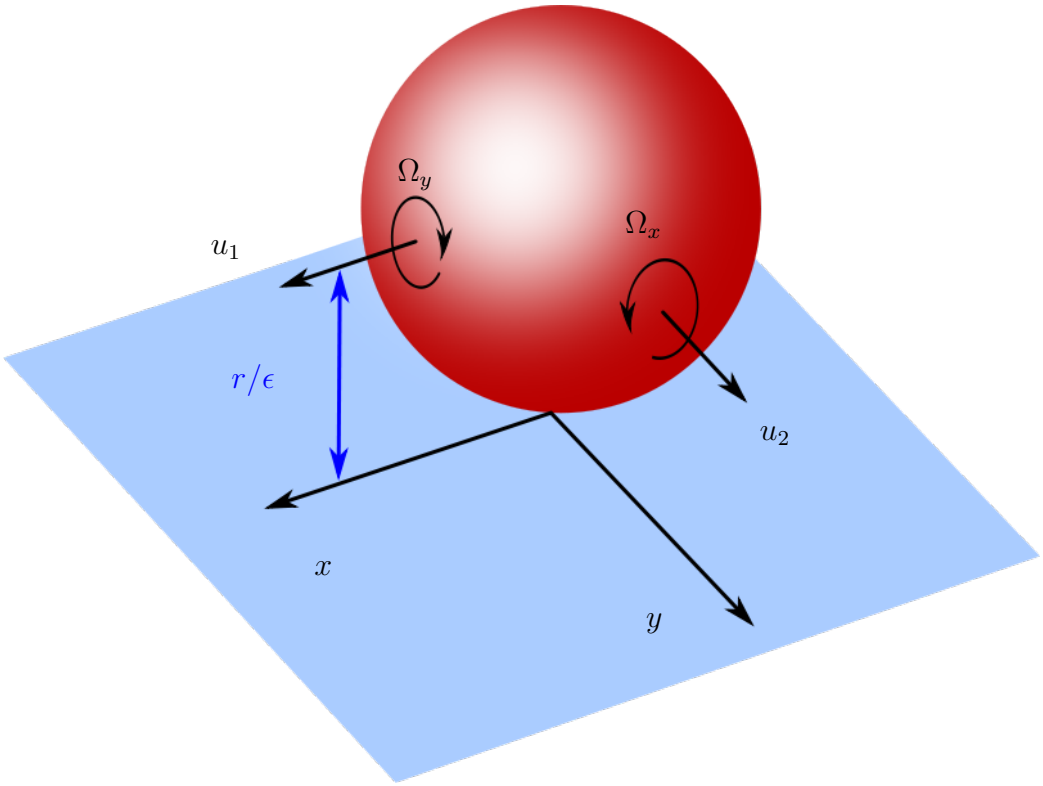


Figure 3.4: A sphere with an unknown radius r/ϵ rolling on the horizontal plane. This chapter models such a sphere as an ensemble control system, proves that it is controllable, and derives a motion planning algorithm that steers the sphere to within an arbitrarily small neighborhood of any desired configuration in $\mathbb{R}^2 \times SO(3)$.

u_1 and u_2, the velocities of the sphere center in the $+x$ and $+y$ directions. We define the infinitesimal generators of rotation

$$\Omega_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}, \quad \Omega_y = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \Omega_z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

and write the kinematics of the sphere by the differential system

$$\begin{aligned} \dot{x}(t) &= u_1(t) \\ \dot{y}(t) &= u_2(t) \\ \dot{R}(t) &= R(t)\frac{1}{r}\Big(\Omega_x u_1(t) + \Omega_y u_2(t)\Big). \end{aligned} \tag{3.1}$$

Given $q_{\text{start}}, q_{\text{goal}} \in \mathbb{R}^2 \times SO(3)$, a gain $K \in [0,1]$ that weights the relative importance of rotation and position error, and an error bound $\mu > 0$, the approximate steering problem is to find open-loop inputs

$$(u_1(t), u_2(t)) : [0, T] \to \mathcal{U}$$

that result in $\big([x(0), y(0)]^\top, R(0)\big) = q_{\text{start}}$ and

$$K \left\| \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} - \begin{bmatrix} x_{goal} \\ y_{goal} \end{bmatrix} \right\|_2 + (1-K)\,\text{angle}\Big(R(T), R_{\text{goal}}\Big) \leq \mu$$

for free final time T. Here

$$\text{angle}\Big(R_a, R_b\Big) = \arccos\left(\frac{\text{trace}\left(R_a^\top R_b\right) - 1}{2}\right)$$

is the minimum rotation angle between two rotation matrices. If such inputs always exist then we say that (3.1) is *approximately controllable*—and indeed they do, as shown in [65].

3.2.1 Ensemble of Spheres

We will solve this same approximate steering problem, but for a sphere with radius r/ϵ, where ϵ is an unknown but bounded constant. The resulting

kinematics have the form

$$\begin{aligned} \dot{x}(t) &= u_1(t) \\ \dot{y}(t) &= u_2(t) \\ \dot{R}(t) &= R(t)\frac{\epsilon}{r}\begin{bmatrix} 0 & 0 & -u_2 \\ 0 & 0 & u_1 \\ u_2 & -u_1 & 0 \end{bmatrix} \end{aligned} \tag{3.2}$$

where $\epsilon^{-1} \in [1-\delta, 1+\delta]$ for some $0 \leq \delta < 1$. Rather than try to steer one sphere governed by (3.2)—where ϵ is unknown—our approach is to steer an uncountably infinite collection of spheres parameterized by ϵ, each one governed by

$$\begin{aligned} \dot{x}(t) &= u_1(t) \\ \dot{y}(t) &= u_2(t) \\ \dot{R}(t,\epsilon) &= R(t,\epsilon)\frac{\epsilon}{r}\Big(\Omega_x u_1(t) + \Omega_y u_2(t)\Big). \end{aligned} \tag{3.3}$$

We call this fictitious collection of spheres an *ensemble* and call (3.3) an *ensemble control system*. The idea is that if we can find open-loop inputs $u_1(t)$ and $u_2(t)$ that result in $\big([x(0), y(0)]^\top, R(0,\epsilon)\big) = q_{\text{start}}$ and

$$K \left\| \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} - \begin{bmatrix} x_{goal} \\ y_{goal} \end{bmatrix} \right\|_2 + (1-K)\,\text{angle}\Big(R(T,\epsilon), R_{\text{goal}}\Big) \leq \mu$$

for all $\epsilon^{-1} \in [1-\delta, 1+\delta]$, then we can certainly guarantee that the actual sphere, which corresponds to one particular value ϵ^* of ϵ, will satisfy

$$K \left\| \begin{bmatrix} x(T) \\ y(T) \end{bmatrix} - \begin{bmatrix} x_{goal} \\ y_{goal} \end{bmatrix} \right\|_2 + (1-K)\,\text{angle}\Big(R(T,\epsilon^*), R_{\text{goal}}\Big) \leq \mu$$

If such inputs always exist then we say that (3.3) is *ensemble controllable*, defined as being approximately controllable on the function space

$$L_2\left([1-\delta, 1+\delta], \mathbb{R}^2 \times SO(3)\right).$$

3.3 Analysis of Controllability

We will now prove that the system (3.3) is ensemble controllable. We will do this by using repeated bracketing to get higher-order powers of ϵ, and then by using polynomial approximation to construct arbitrary vector flows. This approach is similar to that in Chapter 2.2.

Theorem 4. *The system* (3.3) *is ensemble controllable.*

Proof: We will first show that we can generate arbitrary changes in the ensemble orientation with no net change in x, y position. Any rotation A in SO(3) can be parameterized by the rotations ψ, ϕ, θ about the world z-axis, x-axis and then z-axis.

$$A = R_z(\theta)R_x(\phi)R_z(\psi)$$

To approximate any rotation A with an ensemble, it is then sufficient to construct the three approximate rotations

$$R_z\big(\theta(\epsilon)\big) \approx R_z(\theta), \quad R_x\big(\phi(\epsilon)\big) \approx R_x(\phi), \text{ and } R_z\big(\psi(\epsilon)\big) \approx R_z(\psi).$$

We will proceed by showing how to construct $R_z\big(\theta(\epsilon)\big)$. For small time dt, by rolling clockwise on the horizontal plane in a square pattern with sides of length $\sqrt{dt}$, we can approximate a counter-clockwise rotation about the z-axis

$$\exp\{\epsilon\sqrt{dt}\Omega_y\}\exp\{-\epsilon\sqrt{dt}\Omega_x\}\exp\{-\epsilon\sqrt{dt}\Omega_y\}\exp\{\epsilon\sqrt{dt}\Omega_x\} \\ = \exp\{-\epsilon^2 dt\Omega_z\} + O\left(\epsilon^3\right).$$

This Lie bracket movement generates the previously restricted motion about the z-axis, but the final x and y positions are unchanged. We take further Lie brackets to find new control vector fields:

$$\begin{aligned}[\epsilon\Omega_y, \epsilon\Omega_x] &= \begin{bmatrix} 0 & \epsilon^2 & 0 \\ -\epsilon^2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\ &= -\epsilon^2\Omega_z \\ [\epsilon\Omega_y\,[\epsilon\Omega_y, \epsilon\Omega_x]] &= -\epsilon^3\Omega_x\end{aligned}$$

$$
\begin{aligned}
[\epsilon\Omega_y\,[\epsilon\Omega_y\,[\epsilon\Omega_y, \epsilon\Omega_x]]] &= \epsilon^4\Omega_z \\
&\vdots \\
&= -1^k\epsilon^{2k}\Omega_z \\
&= -1^k\epsilon^{2k-1}\Omega_x.
\end{aligned}
$$

By successive Lie brackets, we can produce control vector fields of the form $\{\epsilon^{2k}\Omega_z\}$ and $\{\epsilon^{2k-1}\Omega_x\}$ for any $k \geq 1$ that leave the Cartesian position of the sphere unchanged. As in Chapter 2.2.2, with piecewise-constant inputs we can produce flows of the form

$$
\exp\left\{\sum_{k=1}^{m} a_k\epsilon^{2k}\Omega_z\right\}
$$

for freely chosen constants $a \in \mathbb{R}^m$. The Stone-Weierstrass theorem [58] tells us that given $\eta > 0$ and a continuous real function

$$
\nu(\epsilon)\colon [1-\delta, 1+\delta] \to \mathbb{R},
$$

there exists a polynomial function $\rho(\epsilon)$ such that

$$
|\rho(\epsilon) - \nu(\epsilon)| < \eta
$$

for all $\epsilon \in [\underline{\epsilon}, \overline{\epsilon}]$. An immediate corollary is that continuous real functions on the domain $[\overline{\epsilon}^{-1}, \underline{\epsilon}^{-1}] = [1-\delta, 1+\delta]$ for some $0 \leq \delta < 1$ can be uniformly approximated either by an odd polynomial or by an even polynomial. (This result would not be true on an arbitrary domain, which is why we restrict $\delta < 1$.) As a consequence, we can choose $a \in \mathbb{R}^m$ so that

$$
\theta(\epsilon) \approx \sum_{k=1}^{m} a_i\epsilon^{2k}
$$

for $\epsilon^{-1} \in [1-\delta, 1+\delta]$, with error vanishing in k. The time complexity of the resulting motion increases with k and with the number of switches required to approximate flows along each vector field $\epsilon^{2k}\Omega_z$ and $\epsilon^{2k-1}\Omega_x$, but remains finite for any given $\mu > 0$.

We further note that every Lie bracket movement is a roll in a closed pattern on the horizontal plane and thus leaves the final sphere x and y

position unchanged. To steer the ensemble from

$$\left([x_{\text{start}}, y_{\text{start}}]^\top, R_{\text{start}}\right) \text{ to } \left([x_{\text{goal}}, y_{\text{goal}}]^\top, R_{\text{goal}}\right),$$

we can rotate first in x and then in y to $(x_{\text{goal}}, y_{\text{goal}})$, resulting in the net rotation

$$R_1(\epsilon) = R_y\left(\frac{y_{\text{goal}} - y_{\text{start}}}{\epsilon}\right) R_x\left(\frac{x_{\text{goal}} - x_{\text{start}}}{\epsilon}\right) R_{\text{start}}.$$

We then approximate the ensemble rotation $A(\epsilon) = R_{\text{goal}} R_1^\top(\epsilon)$ through Lie bracket movement. Our result follows. ■

3.4 Approximate Steering Algorithm

In this section we provide two methods for approximate open-loop steering of a rolling sphere with an unknown radius. An overview of each is shown in Tables 3.1 and 3.2. The first method consists of concatenating Taylor series approximations of in-plane rotations. Since each rotation is approximate, using fewer rotations reduces the cumulative orientation error. It is well known that three orthogonal rotations span SO(3). We show that two straight line rolls on the horizontal plane also span SO(3) [80].

We conclude the section with a second method for approximate open-loop steering of a rolling sphere using an optimized n-segment path.

3.4.1 Piecewise-Constant Rolling Primitive

Consider the motion primitive given by Pryor in [81,82] of the following form for a non-negative integer k, and freely chosen $\phi, \beta_k \in \mathbb{R}$:

$$(u_1, u_2) = \begin{cases} (0,-1) & 0 \quad \leq t < \phi k \\ (1,0) & \cdots \leq t < \cdots + \beta_k/2 \\ (0,1) & \cdots \leq t < \cdots + 2\phi k \\ (1,0) & \cdots \leq t < \cdots + \beta_k/2 \\ (0,-1) & \cdots \leq t < \cdots + \phi k \end{cases} \tag{3.4}$$

Table 3.1: Orientation Method 1: Taylor Series Approximation

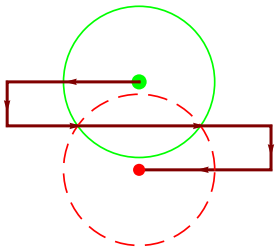
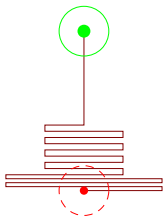
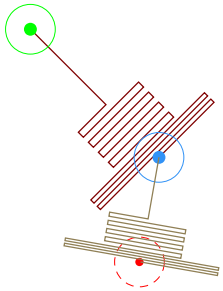
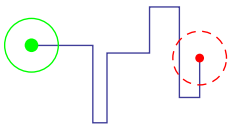
Path Name	Example Path	Rotation
Piecewise-constant rolling primitive (Section 3.4.1)		$\approx R_y\left(\beta_k \epsilon \cos\left(\phi k \epsilon\right)\right)$
(nth-order) Taylor series approximate rotation (using n piecewise-constant rolling primitives) (Section 3.4.2)		$\approx R_y\left(\sum_{k=0}^{n-1} \beta_k \epsilon \cos\left(\phi k \epsilon\right)\right)$
Two-step rotation using two Taylor series approximate rotations to approximate $R_z(\phi_3)R_x(\phi_2)R_z(\phi_1)$ (Section 3.4.3)		$\approx R_p\big(r(\phi_2-\pi), \phi_3\big)$ $\cdot R_p\big(r\pi, (\phi_3-\phi_1)/2\big)$

Table 3.2: Orientation Method 2: n-Segment Optimized Path

Path Name	Example Path	Rotation
n-Segment Optimized Path (Section 3.4.4)		$R(\Theta, \epsilon)$ $= \prod_{i=1}^{n} R_x\left(\frac{\epsilon}{r}\theta_{2i-1}\right) R_y\left(\frac{\epsilon}{r}\theta_{2i}\right)$

Such a primitive is shown in Fig. 3.5 and requires time $t = 4\phi k + \beta_k$. To first order, the result is to achieve

$$\begin{aligned}\Delta x(\epsilon) &= \beta_k \\ \Delta y(\epsilon) &= 0, \\ \Delta R(\epsilon) &= R_y\left(\beta_k \epsilon \cos\left(\phi k \epsilon\right)\right) + O(\beta_k^2)\end{aligned}$$

The approximation degrades for large values of β_k, but by repeating the primitive (3.4) j times with parameter $\beta_k' = \beta_k / j$ the approximation improves. Pryor proved in [82] that we can build a primitive with arbitrary accuracy by increasing j.

3.4.2 Taylor Series Approximate Rotation

Because the primitives (3.4) all rotate in the same direction, we are free to concatenate them. The result after applying n primitives with $k = 0, \ldots, n-1$ is

$$\begin{aligned}\Delta x(\epsilon) &= \sum_{k=0}^{n-1} \beta_k \\ \Delta y(\epsilon) &= 0 \\ \Delta R(\epsilon) &\approx \prod_{k=0}^{n-1} R_y\left(\beta_k \epsilon \cos\left(\phi k \epsilon\right)\right) \\ &\approx R_y\left(\sum_{k=0}^{n-1} \beta_k \epsilon \cos\left(\phi k \epsilon\right)\right) \\ &\approx R_y\left(\Delta\theta\left(\epsilon\right)\right),\end{aligned}$$

where we define

$$\Delta\theta\left(\epsilon\right) = \sum_{k=0}^{n-1} \beta_k \epsilon \cos\left(\phi k \epsilon\right). \tag{3.5}$$

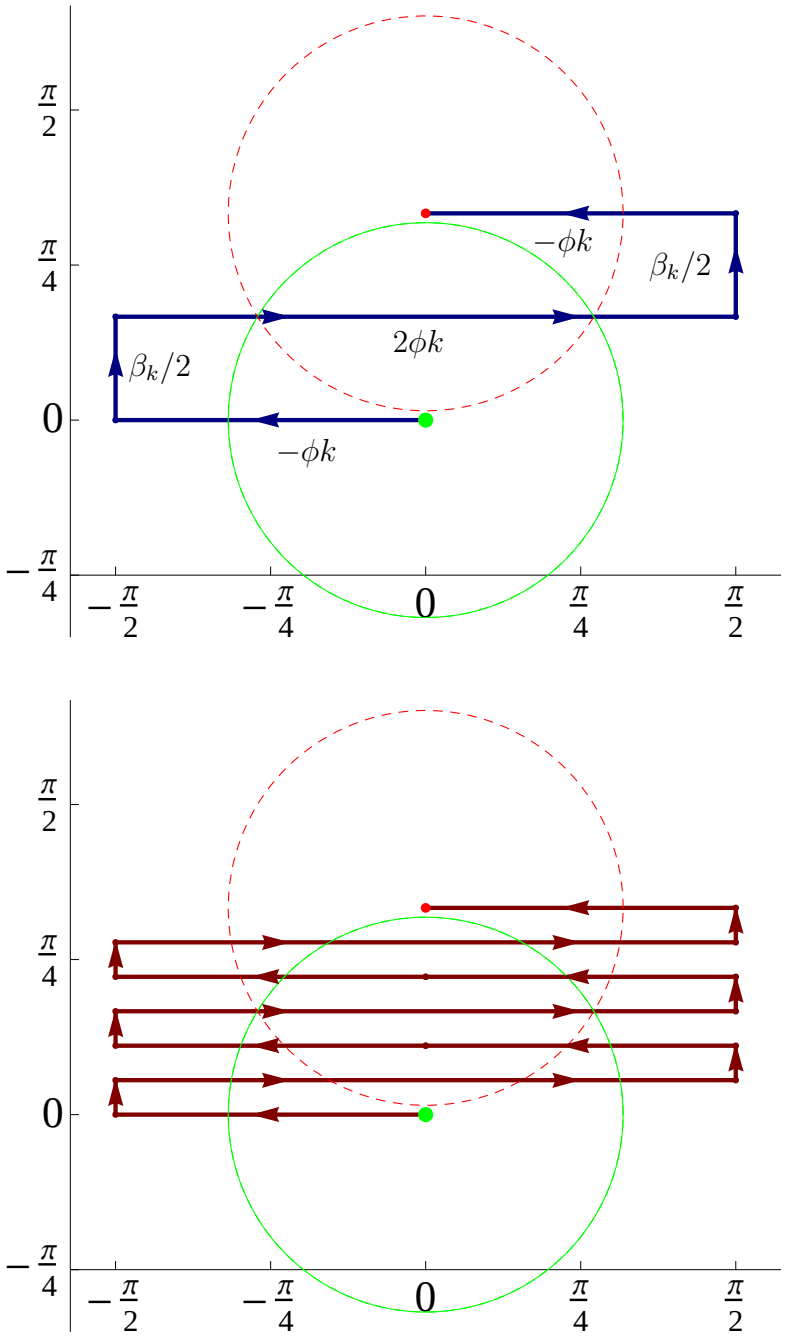


Figure 3.5: (Top) A sample primitive of the form (3.4), with $\beta_k = \pi/3$ and $\phi k = \pi/2$. A sphere following this path generates to first order the rotation $R_y\left(\beta_k \epsilon \cos(\phi k \epsilon)\right)$. (Bottom) By subdividing the rotation we can improve arbitrarily this approximation, at the cost of a longer path. An outline of a unit radius sphere is shown in green at the start of the path and dashed in red at the end.

Achieving Error of a Particular Order:

We may express the angle turned about the y-axis after applying n primitives, $\Delta\theta\,(\epsilon)$, as a Taylor's series in ϵ about $\epsilon = 1$:

$$\Delta\theta(\epsilon) = \Delta\theta(0) + \left.\frac{\partial \Delta\theta}{\partial \epsilon}\right|_{\epsilon=1} (\epsilon - 1) + \cdots .$$

This series has the form

$$\Delta\theta(\epsilon) = \sum_{j=1}^{n} s_j (\epsilon - 1)^{j-1} + O\left(|\epsilon - 1|^n\right).$$

Define

$$s = \begin{bmatrix} s_1 \\ \vdots \\ s_n \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_n \end{bmatrix},$$

so we can write

$$s = S\beta \tag{3.6}$$

where the matrix $S \in \mathbb{R}^{n \times n}$ has elements

$$S_{ij} = \frac{1}{(i-1)!} \left(\left.\frac{\partial^{i-1} \left(\epsilon \cos\left(\epsilon j \phi\right)\right)}{\partial \epsilon^{i-1}}\right|_{\epsilon=1} \right).$$

To achieve $\Delta\theta = \alpha$ with error that is of order n in ϵ, we require only that $s = [\alpha, 0, \ldots, 0]^\top$.

The achievable error decreases exponentially in the number n of primitives used. As in Chapter 2, we need to solve a system of n linear equations to achieve nth-order error. As a consequence, exactly n primitives are required to achieve nth-order error, for any n. There is the implicit assumption here that S is nonsingular (and, in practice, well conditioned), but this assumption will hold for "almost all" choices of ϕ. By linearity, it is clear that the scaled parameters $\beta\Delta\theta$ will reach arbitrary $\Delta\theta$. We can use this method to generate paths that satisfy arbitrary accuracy bounds, but these paths may be very long. Fig 3.6 depicts paths generated to turn π about the x-axis with a threshold of $\pi/6$.

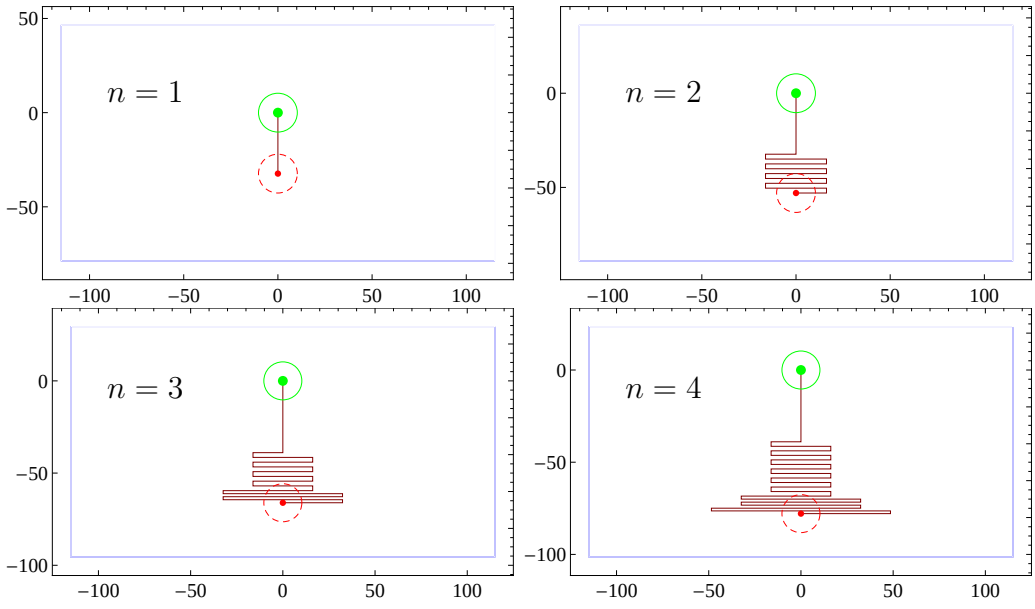


Figure 3.6: Rolling paths generated using Taylor series to approximate the rotation $R_x(\pi)$ with a threshold of $\pi/6$ and a nominal ball radius of 10.3mm to (left to right, top to bottom) 1, 2, 3 and 4-th order. Path lengths are {32.4, 311.9, 584.0, 919.4} mm. The start point is shown in green and the stop point in red. The 230 × 125 mm workspace for our manipulator is represented by a blue rectangle.

This 2D path can then be rotated an angle ψ about the z-axis by

$$\begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}$$

to approximate any in-plane rotation.

3.4.3 Arbitrary Orientation in SO(3) with Two Straight Rolls

Any rotation A can be generated by the three rotations $R_z(\theta)R_x(\phi)R_z(\psi)$ [80]. This is illustrated in Fig 3.7 and proved below. We use the shorthand notation $s_\alpha = \sin(\alpha)$, $c_\alpha = \cos(\alpha)$.

$$\begin{aligned} A &= R_z(\theta)R_x(\phi)R_z(\psi) \\ &= \begin{bmatrix} c_\theta c_\psi - c_\phi s_\theta s_\psi & -c_\phi c_\psi s_\theta - c_\theta s_\psi & s_\theta s_\phi \\ c_\psi s_\theta + c_\theta c_\phi s_\psi & c_\theta c_\phi c_\psi - s_\theta s_\psi & -c_\theta s_\phi \\ s_\phi s_\psi & c_\psi s_\phi & c_\phi \end{bmatrix}. \end{aligned}$$

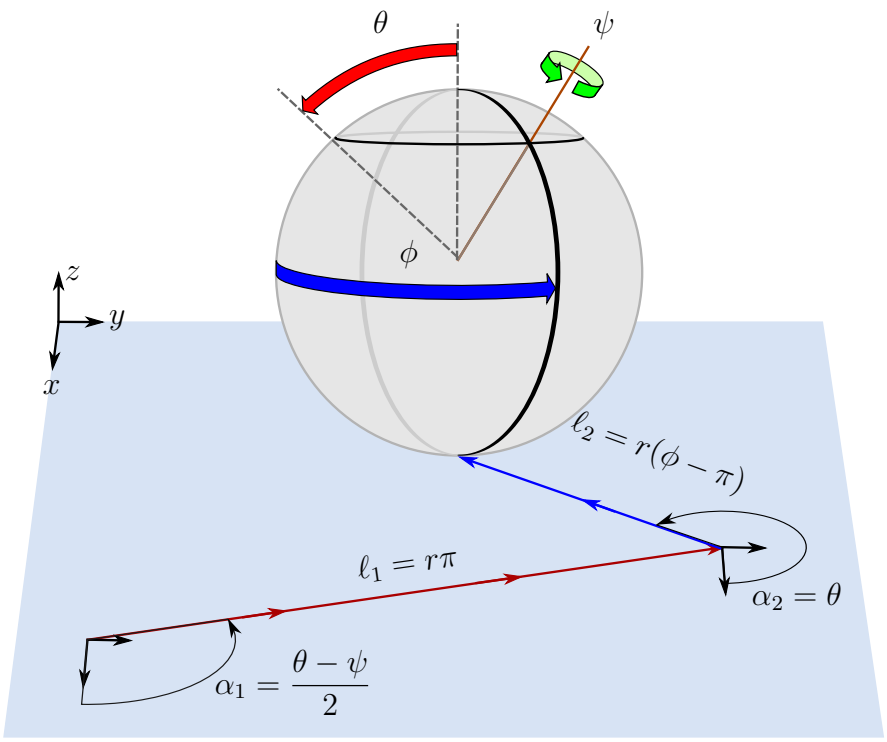


Figure 3.7: Any orientation in SO(3) can be generated by two straight-line rolls on the horizontal plane. Let $R_p(\ell, \alpha)$ represent a roll along a line of length ℓ making the angle α with the x-axis. Then, for a ball with radius r, $R_z(\theta)R_x(\phi)R_z(\psi) = R_p\left(r(\phi - \pi), \theta\right) R_p\left(r\pi, (\theta - \psi)/2\right)$. (Video online: http://demonstrations.wolfram.com/ReOrientASphereWithTwoStraightRolls/)

Our system is constrained to rolling on the horizontal plane. For a sphere of radius r, any roll along a line of length ℓ making the angle α with the x-axis results in the rotation

$$R_p(\ell, \alpha) = \begin{bmatrix} c_\alpha^2 + c_{\ell/r} s_\alpha^2 & (1 - c_{\ell/r}) c_\alpha s_\alpha & s_{\ell/r} s_\alpha \\ (1 - c_{\ell/r}) c_\alpha s_\alpha & c_{\ell/r} c_\alpha^2 + s_\alpha^2 & -c_\alpha s_{\ell/r} \\ -s_{\ell/r} s_\alpha & c_\alpha s_{\ell/r} & c_{\ell/r} \end{bmatrix}.$$

The following two straight-line rolls duplicate $R_z(\theta)R_x(\phi)R_z(\psi)$:

$$R_p\big(r(\phi - \pi), \theta\big) R_p\big(r\pi, (\theta - \psi)/2\big) = \begin{bmatrix} c_\theta c_\psi - c_\phi s_\theta s_\psi & -c_\phi c_\psi s_\theta - c_\theta s_\psi & s_\theta s_\phi \\ c_\psi s_\theta + c_\theta c_\phi s_\psi & c_\theta c_\phi c_\psi - s_\theta s_\psi & -c_\theta s_\phi \\ s_\phi s_\psi & c_\psi s_\phi & c_\phi \end{bmatrix}$$
$$= A$$

The first rotation is of length $\ell_1 = r(\pi - \phi)$ with $|\phi| \leq \pi$ and the second rotation is of length $\ell_2 = r\pi$, giving a total path length of $\pi r \leq \ell_1 + \ell_2 \leq 2\pi r$.

We can now decompose any desired rotation in SO(3) into two straight line rotations with magnitude ℓ_1/r and ℓ_2/r radians. Each of these rotations can be implemented with the Taylor series approximate rotations of Section 3.4.2. A drawback is that this method does not provide a way to control the change in x and y. In the following section we use optimization to generate shorter paths that steer toward desired $(\Delta x, \Delta y, \Delta R)$ values.

3.4.4 Approximate Steering with n-Segment Optimized Path

We want to steer an ensemble of spheres with radii r/ϵ to a desired rotation R_{goal} and position $(x_{\text{goal}}, y_{\text{goal}})$ by alternating rotations along the positive x and y-axes. The composite rotation $R(\Theta, \epsilon)$ after n such rotations is

$$\begin{aligned} R(\Theta, \epsilon) &= \prod_{i=1}^{n} R_x\left(\frac{\epsilon}{r}\theta_{2i-1}\right) R_y\left(\frac{\epsilon}{r}\theta_{2i}\right) \\ \Delta x(\Theta) &= \sum_{i=1}^{n} \theta_{2i-1} \\ \Delta y(\Theta) &= \sum_{i=1}^{n} \theta_{2i} \\ \text{for } \Theta &= [\theta_1, \ldots, \theta_{2n}], \quad \theta_i \geq 0. \end{aligned} \tag{3.7}$$

We assume without loss of generality that $[x_{\text{start}}, y_{\text{start}}, R_{\text{start}}] = [0, 0, \mathbb{I}(3)]$. The problem is to find Θ such that

$$K \left\| \begin{bmatrix} \Delta x(\Theta) \\ \Delta y(\Theta) \end{bmatrix} - \begin{bmatrix} x_{\text{goal}} \\ y_{\text{goal}} \end{bmatrix} \right\|_2 + \int_{1-\delta}^{1+\delta} \text{angle}\left(R\left(\Theta, \epsilon\right)\right), R_{\text{goal}}) \, d\epsilon \tag{3.8}$$

is minimized. To simplify our optimization routine we replace the integral in (3.8) with a summation over a finite set of ϵ, and use gradient descent on Θ from multiple random seed values Θ_{init} to search for local minima in

$$K \left\| \begin{bmatrix} \Delta x(\Theta) \\ \Delta y(\Theta) \end{bmatrix} - \begin{bmatrix} x_{\text{goal}} \\ y_{\text{goal}} \end{bmatrix} \right\|_2 + \sum_{k=0}^{N-1} \text{angle}\left(R\left(\Theta, 1 - \delta + \frac{2\delta k}{N-1}\right), R_{\text{goal}}\right) \tag{3.9}$$

For the paths generated in Fig. 3.8, we used a path with 10 segments, 500 random seed values, and $N = 15$.

3.5 Hardware Experiments

To validate our control algorithms we built a ball-plate system based on magnetic actuation. In this section we describe the system design, the methodology for state estimation, and characterize the system repeatability.

3.5.1 System Design

Our ensemble plate-ball system is shown in Fig. 3.1, page 41. A grid array of n magnets attached to a CNC xy-table can reorient n spheres by rolling when a nonmoving substrate is placed between the spheres and magnet array. A table-top CNC mill table (Sherline 8541 metric, 800 steps per mm) is used as an xy planar manipulator to slide a tray in the horizontal plane. The tray is manufactured of MDF and is bolted to the mill. Fifteen holes are drilled into the surface. Each holds a 5×8 mm diameter cylindrical neodymium rare-earth magnet flush to the surface of the tray. The xy table provides a 230×125 mm workspace. By securing a stationary, 0.15 mm thick cardboard sheet on top of the tray, we have a manipulator that can roll variable-sized ferro-magnetic spheres in the horizontal plane with minimal slip. Six steel ball-bearings with diameters $\{16, 18, 19, 22, 24, 25.4\}$ mm were used to test manipulation algorithms in this chapter.

Our prototype was inspired by the kinetic sculptures of Bruce Shapiro and Jean-Pierre Hébert, which draw paths in sand using a rolling steel ball actuated by a hidden servo-controlled magnet [83, 84].

3.5.2 Measuring System Position and Orientation

To validate our control policies we must accurately measure the position and orientation of spheres as they are manipulated by the plate-ball system. Tracking the orientation of a sphere has inspired several approaches including using colored circles painted at the vertices of Platonic solids [85], painting a fiducial pattern on a sphere and then comparing camera images

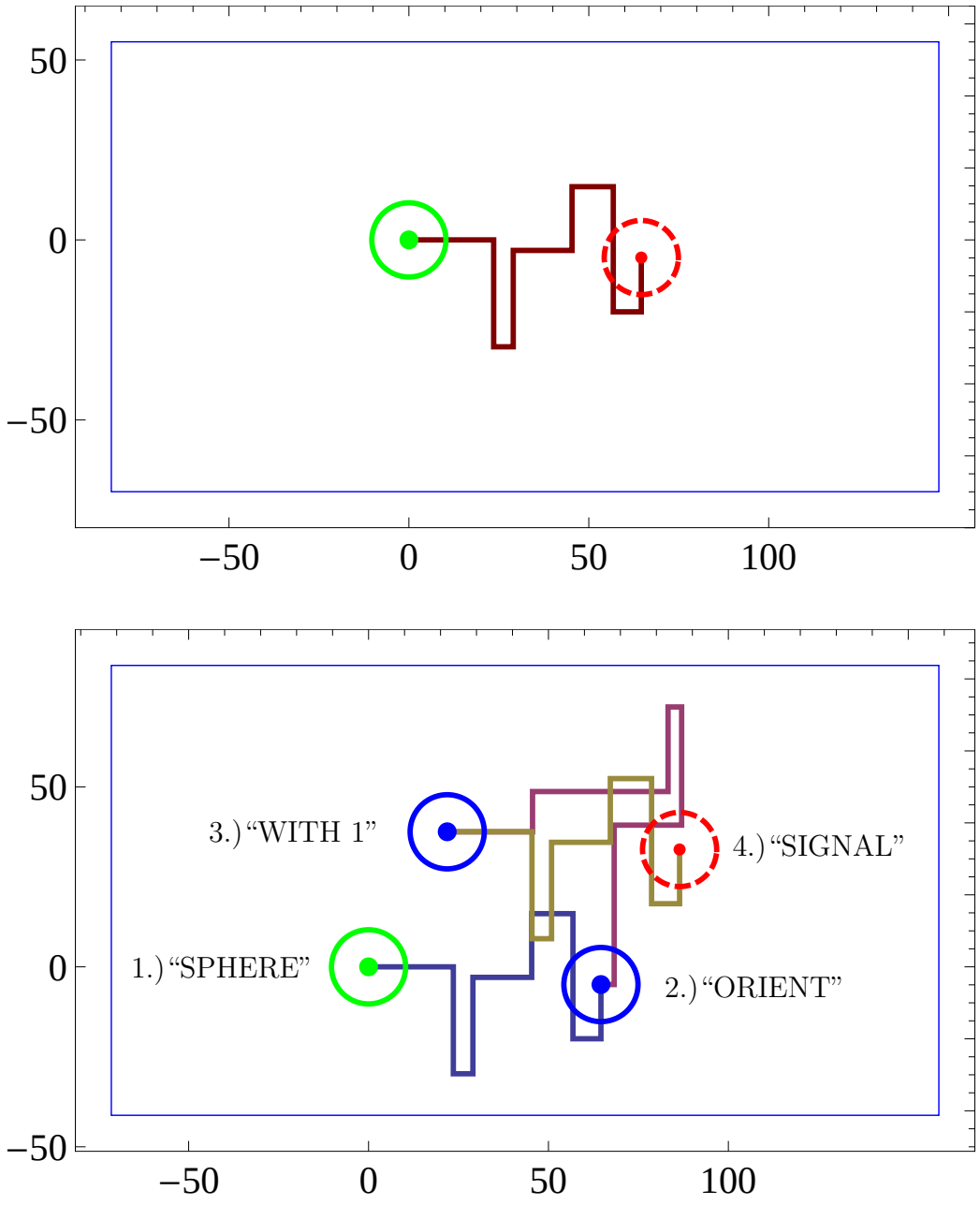


Figure 3.8: Top: a 189 mm long path generated by optimization that rotates spheres approximately $R_x(\pi)$. Bottom: a 577 mm long three-stage path that rotates spheres approximately $R_x(\pi), R_z(\pi/2), R_x(\pi)$. Six spheres were labelled so that different words appeared at the end of each stage. Snapshots from an experiment are shown in Fig. 3.9. Each path is composed of 10 segments and designed for ball radii of 10.3 mm±25%. The start point is shown in green, intermediate points in blue, and the stop point in red. The position error gain K was 0 for these paths.

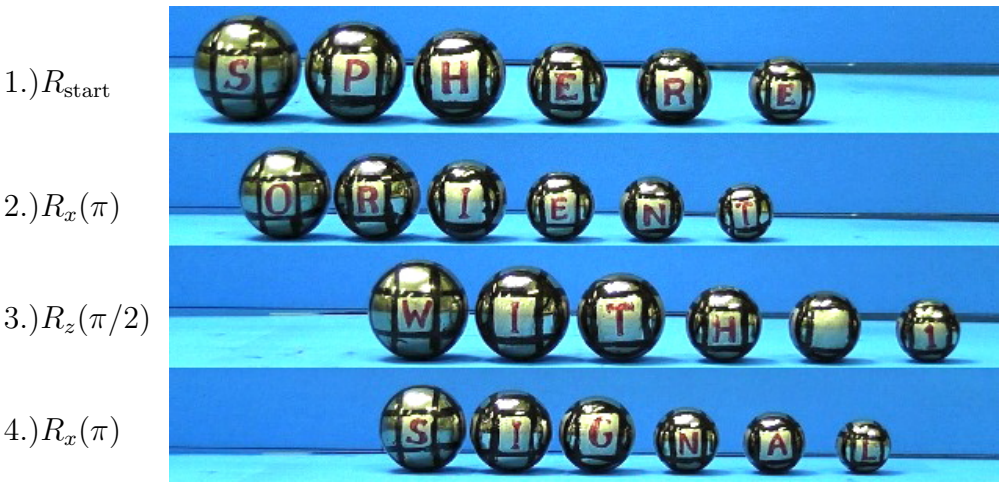


Figure 3.9: Snapshots of six different-sized spheres while rolling along the three-stage path in Fig 3.8. This path consists of a roll that approximates $R_x(\pi)$, followed by $R_z(\pi/2)$, then $R_x(\pi)$. Four letters were printed on appropriate sides of each sphere to illustrate the movement.
(Video online: http://www.youtube.com/watch?v=nPGz0Nd3QzE)

to a library of generated rotated images [86], applying a 2D Gray-coding to the sphere [87], and estimating sphere orientation using many dots precisely applied to the ball [88]. Each of these approaches requires accurate marker application, which increases in difficulty as the sphere shrinks in size. An alternate approach applies fiducial stickers to the sphere, learns the positions during a training phase, and then tracks these fiducials [89]. For our initial approach, we adapted this method to use hand-drawn fiducials because stickers would affect the effective sphere diameter. We then estimated the sphere position by tracking 5 points in each image using Horn's minimum least squares method [90], producing at frame k the rotation matrix R_k. We used '$\top$' shaped fiducials. This method is only useful if the fiducial remains in view for all measurements. Using a single fixed-position camera restricts ΔR to rotations about the optical axis—an overhead camera can only measure rotations about the z-axis.

To enable full state estimation, we followed the approach of Lynch [85]. He showed that if markers are placed at the vertices of a dodecahedron and we can ensure one face is always visible, three marker colors is sufficient to determine the orientation. At least one face is always visible if the camera has a visible angle of $\geq 70.56°$. A 25.4 mm sphere requires the camera to

be at least 36 mm away from the sphere. In practice, markers far from the sphere center in the camera image become distorted and the lighting less reliable, so the camera must be further away. In our setup the camera is 0.5 m from the sphere. Our markers are circular because this shape is least affected by distortion as the sphere rolls. We selected the three colors {`red,green,blue`} because they are easily discernible in HSV color space.

Implementation

We used steel ball-bearings 25.4 mm in diameter. Spheres can be difficult to illuminate uniformly. To ease the load on our vision processing, we painted the balls flat white. We washed the bearings with soap and water, spray-painted two coats of primer (Rust-Oleum grey primer 7582) and then over-coated with 3 thin coats of flat white (Rust-Oleum flat white 7590). In our initial tests the markers were painted on with paint-pens (Elmer's Painters Red, Krylon Brights 60670323 and 07031664A). Unfortunately this created raised bumps on the surface of the sphere that caused the ball to roll around rather than over the painted areas. These deviations are clearly seen in Fig. 3.19, page 74. We switched to using colored pens (Pentel Color Pen FinePoint #S360 102 Red, 111 Light Green, 136 Baby Blue) for subsequent tests. We illuminated the sphere by arranging a tube fluorescent light fixture at each of the four sides of the plate-ball table.

To accurately mark the vertices of a dodecahedron, we constructed a paper template of an icosahedron that encloses the sphere. The centers of each face of this icosahedron form a dodecahedron that is circumscribed by the sphere. We then drill out these centers in the template, fold the template around the sphere, and mark the vertices as shown in Fig. 3.10. Given a sphere of radius r, the edges of the icosahedron whose faces are each tangent to the sphere are of length

$$\frac{4\sqrt{3}}{3+\sqrt{5}}r \approx 1.323r.$$

Our template, constructed for a sphere 25.4 mm in diameter, has edges 16.8 mm long. The template, marking procedure and a camera frame after vision processing are shown in Fig. 3.10.

For vision processing we used a Basler 602fc color digital camera which captures 656×490 pixels at 100 fps. Pseudo code for measuring the trans-

lation and rotation is given in Alg. 3.1.

Algorithm 3.1 Measure Ball Translation (x_b, y_b) and Rotation Matrix R

grab a frame
segment out the ball and measure ball center location and radius:
    $(x_b, y_b), r$
adaptive threshold over the ball to find markers
measure position of each marker:
    $\left(x_i, y_i, r + \sqrt{r^2 - (x_b - x_i)^2 - (y_b - y_i)^2}\right)$
assign marker color based on shortest distance of hue in HSV colorspace
create *edgeList* (markers closer than a threshold distance in $\mathbb{R}^3$)
if depth-first-search on *edgeList* finds 5 markers in a non-repeating loop **then** (this is a face)
    record colors of face vertices in CW order
    search through 60 possible color patterns to find matching face
    R = HORN'SMETHOD(5 face markers, (x_b, y_b, r))
else
    for all measured markers **do**
        compare marker positions to identified markers in previous frame
        record marker if colors match and distance is less than a threshold
    end for
    if 2 or more markers identified **then**
        R = HORN'SMETHOD(recorded markers, (x_b, y_b, r))
    else
        procedure failed
    end if
end if
record R, (x_b, y_b)

3.5.3 Characterizing System Repeatability

Our control algorithm assumes the spheres roll without slipping. To test the accuracy of this simplification, we ran a series of repeatability tests with different substrate materials.

The substrate separates the metal spheres from the tray holding the magnets. The non-moving substrate is clamped along the perimeter of the workspace. A good substrate should be thin, because magnetic strength decreases proportional to the inverse cube of the distance from the dipole. The substrate should have low friction on the bottom surface that the tray and magnets slide along, but high friction on the top surface to prevent the

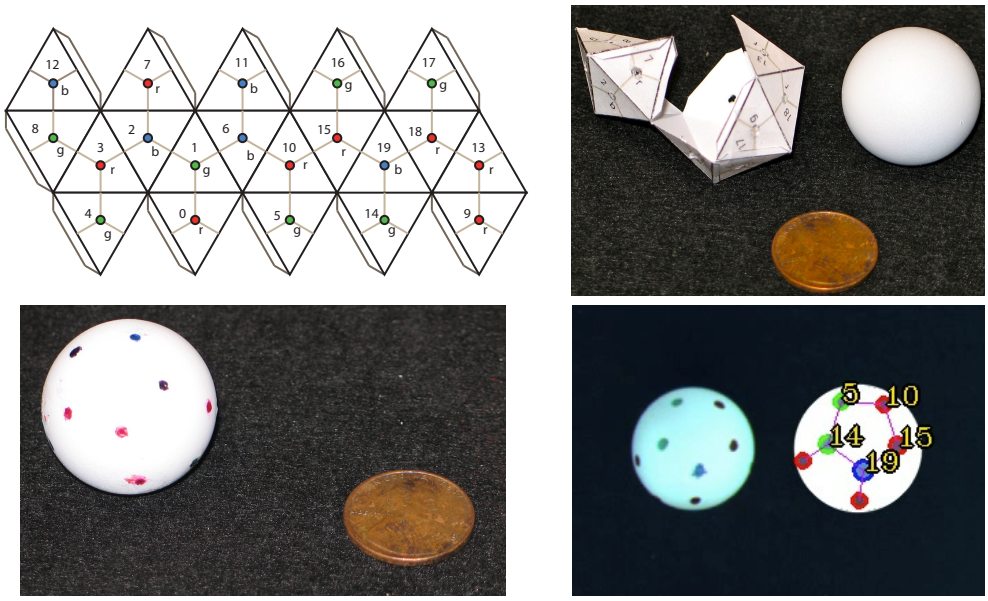


Figure 3.10: The paper template (top) is used to mark 20 equally-spaced vertices such that each face has a unique pattern (bottom left). A vision system identifies this pattern to determine the ball orientation and measures the ball position (bottom right).
(Video online: http://www.youtube.com/watch?v=-9go60KxtRI)

spheres from slipping. Moreover, the top surface should be flat and non-deformable so that rolling is uniform. We tested 5 surfaces: 0.26 mm acrylic, 0.35 mm cardboard, 2 and 3 mm craft foam, and 1.5 mm thick stiff felt. The craft foam had high friction with the tray, so we placed the acrylic sheet beneath the foam for all tests. We ran two tests for each substrate material.

Straight Line: rolling $2\pi r$ along the x-axis, then $-2\pi r$ to return to the origin.

Square Box: rolling in a square pattern with $-2\pi r$ sides.

Each test was run with three 25.4 mm diameter steel (grade 200, 1 ± 0.002 in, sphericity within 0.002 in) spheres, and the patterns were repeated 10 times, with the orientation recorded at the beginning and at the end of each pattern. These tests used slightly smaller permanent magnets (3×8 mm diameter) than those used in the final prototype. Plots of the resulting orientation error are shown in Fig 3.11. These plots show a steady increase in the average orientation error. The best substrate material for all trials was 0.35 mm cardboard, followed by 0.26 mm acrylic, 2 mm foam, 3 mm foam,

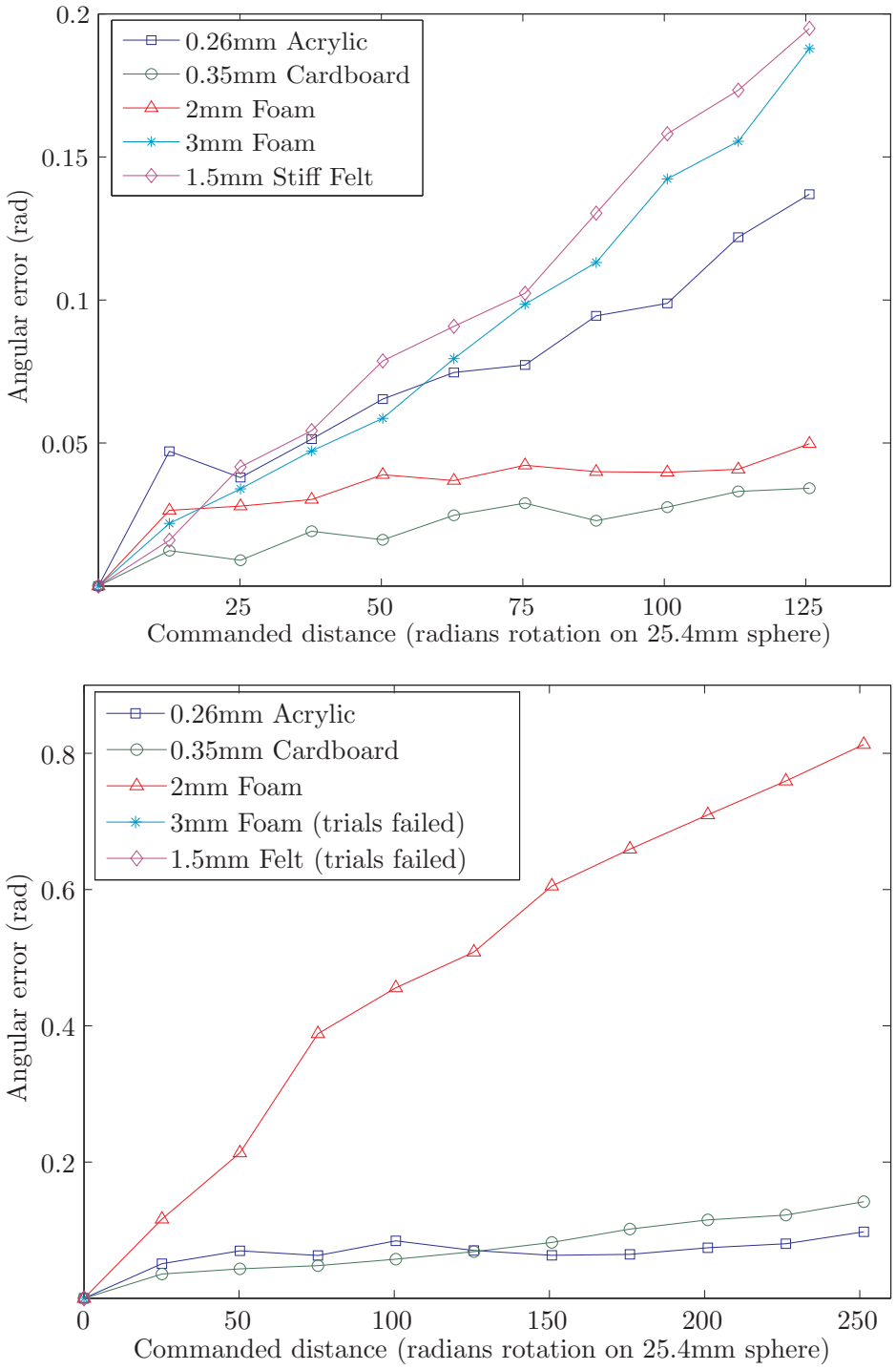


Figure 3.11: Repeatability test for rolling 25.4 mm spheres. Every marker is the average of 3 trials and each test repeated a given pattern 10 times. (Top) pattern 1: back and forth 2π in x, (Bottom) pattern 2: a square box, each side a 2π rotation. The best surfaces had a drift of about 0.09 radians per 250 radians of commanded movement (0.04% error).

and 1.5 mm felt. In the longer trials using the square box pattern, both the 3 mm craft foam and 1.5 mm felt failed when the spheres detached from their magnets and rolled freely atop the surface. The best surface, 0.15 mm cardboard, is thin, horizontally rigid, smooth, and a compromise for friction between the tray and between the spheres. This surface had a drift of about 0.09 radians per 250 radians of commanded movement, a 0.04% error.

3.5.4 Results

In this section we compare experimental results on our hardware platform. We first compare rotation sequences generated by a Taylor's series approximation of various orders and paths discovered by a n-segment optimization. We then demonstrate how multiple paths can be concatenated to move an ensemble through a sequence of orientations. Next, we compare approximating an arbitrary rotation—in this case rotating 2 radians about the $[3, 2, 1]$ axis by concatenating two sequences generated by Taylor series approximation and a sequence discovered using n-segment optimization. Our final test describes using n-segment optimization to generate movements in $\mathbb{R}^2 \times SO(3)$.

1. **Testing Ensemble Rotation Primitives:** We compared motion primitives designed to rotate the ensemble approximately π about the world y-axis. All tests were performed with 6 spheres ranging in size $\pm 25\%$ of nominal diameter. We compare primitives based on 1,2,3 and 4^{th}-order Taylor series with a 10-segment optimized path designed with position error gain $K = 0$. These movements are primitives because they can be concatenated to approximate any desired rotation. Note first that due to symmetry the path used to generate the rotation $R_y(a)$ can be rotated to generate a rotation of a about any axis parallel to the plane. Secondly, the Taylor series approximations can be linearly scaled in the x-axis direction to approximate any rotation β about the world y-axis. The Taylor series paths and the 10-segment optimized path are shown in Figs. 3.6 and 3.8. These paths were tested by measuring the starting and ending orientation for 10 iterations of the Taylor series paths and 25 iterations of the optimized path. The errors recorded for these tests are shown in Fig 3.12 for all six sphere sizes.

   For all spheres, the 10-segment optimized path performed best. Re-

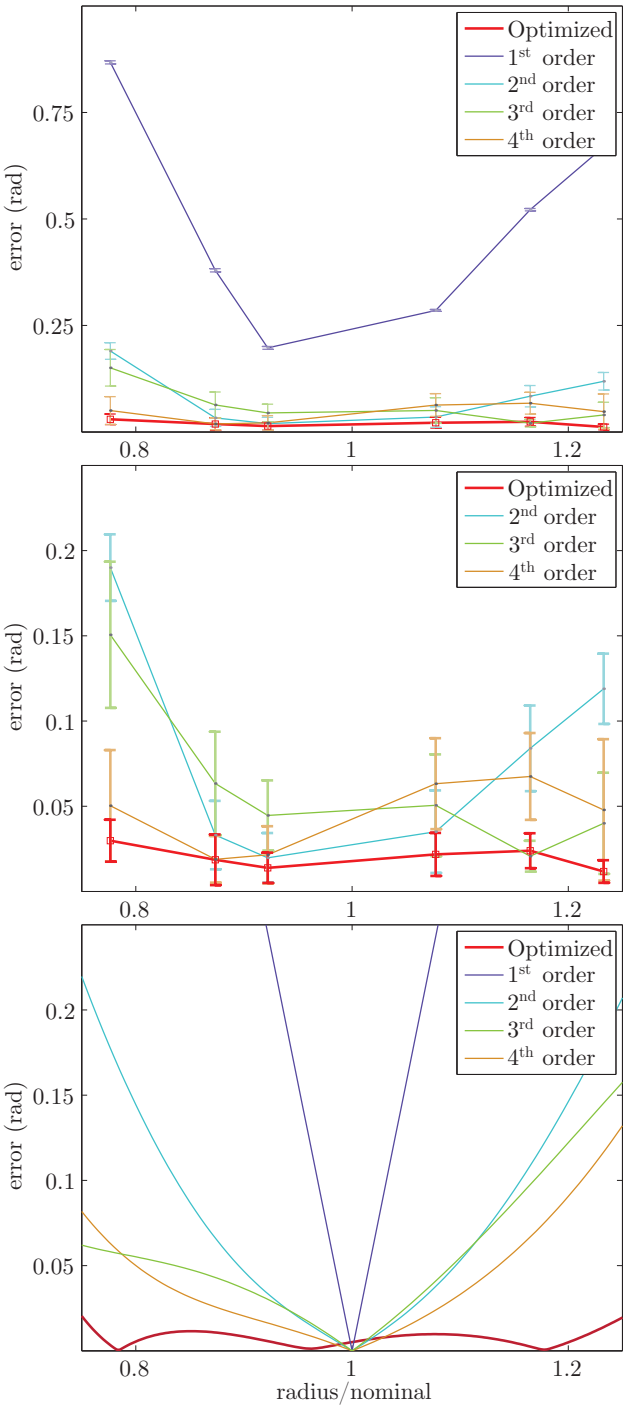


Figure 3.12: Experimental results from applying the paths shown in Figs. 3.6 and 3.8. Six spheres with diameter [16,18,19,22,24,25.4] mm were tested. For each sphere, the optimized path was tested 25 times and each primitive-based path was tested 10 times. Error bars show ± one standard deviation. The last plot shows the theoretical error.

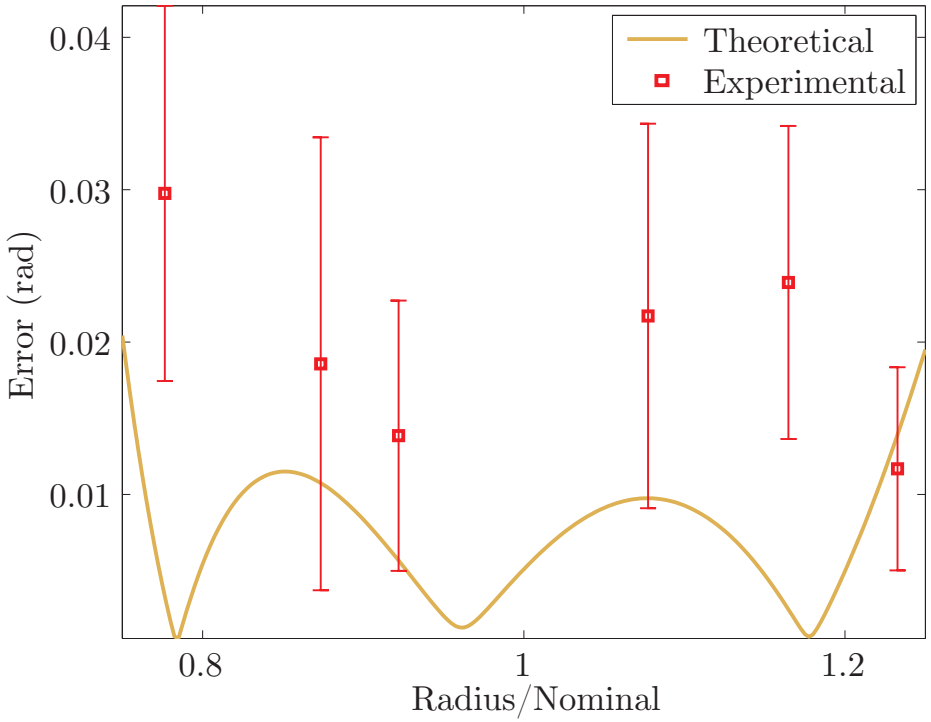


Figure 3.13: Accuracy test for six spheres with diameters [16,18,19,22,24,25.4] mm. The spheres were commanded to follow the 10-segment path in Fig. 3.8 designed by optimization. This test was repeated 25 times. The yellow line is the theoretical error. The red line shows the mean errors for each sphere tested ± one standard deviation.

sults from using this path are highlighted in Fig. 3.13. The Taylor series-based paths increased in accuracy as the order increased from 1^{st} to 3^{rd} order, but the 4^{th} order path decreased in accuracy. The accuracy gained by the higher-order series approximation may have been cancelled by the process-induced drift shown in Fig 3.11.

2. **Following a Path Sequence:** To demonstrate how motion primitives can be sequenced to complete a more complicated task, we concatenated three rolls that respectively approximate the rotations $R_x(\pi)$, $R_z(\pi/2)$, and $R_x(\pi)$. The compound path is 577mm long and shown in Fig 3.8. We then selected six different-sized spheres and printed four letters on appropriate sides of each sphere to illustrate the movement. Fig. 3.9 shows four snapshots of these spheres while rolling along the three move path. See `http://www.youtube.com/watch?v=-9go60KxtRI` for a video of this rotation sequence.

3. **Generating an Arbitrary Rotation:** Tests 3 and 4 were implemented using a single 25.4mm diameter sphere with 20 marked features. For each test we generate 15 different paths for spheres with an assumed radius equally spaced $0.75\times$ to $1.25\times$ the actual radius.

   This experiment demonstrates how a Taylor series approximate rotation, scaled and rotated appropriately, can be applied twice to approximate an arbitrary rotation in SO(3). We then repeat the same test using a 12-segment optimized path. To demonstrate the method, we chose a rotation of 2 radians about the $[3, 2, 1]$ axis. This corresponds to a rotation $R_z(\theta)R_x(\phi)R_z(\psi)$ with $\{\theta \approx 0.982426, \phi \approx 1.89125, \psi \approx -0.19358\}$. As shown in section 3.4

$$R_z(\theta)R_x(\phi)R_z(\psi) = R_p\left(r(\phi-\pi), \theta\right) R_p\left(r\pi, (\theta-\psi)/2\right),$$

   and so we design primitives to first roll a distance πr in the direction 0.588 radians with the x-axis, followed by a roll of length $1.24r$ in the direction 0.98 radians with the x-axis. We then use a 12-segment optimized path for the same desired rotation and the additional constraint $(\Delta x, \Delta y) = (0, 0)$ with position error gain $K = 0.98$.

   These desired paths are shown in Fig. 3.14. The actual paths for 15 different assumed ball radii are shown in Fig. 3.15. Our vision sys-

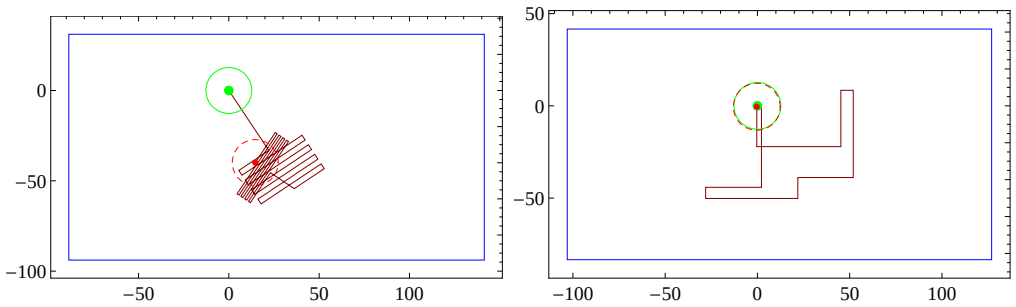


Figure 3.14: Paths for rotating 2 radians about the axis $[3, 2, 1]$. The left path was generated by a two-step rotation using a 2nd-order Taylor series approximation. On the right is a 12-segment optimized path. The path lengths are $\{2182, 326\}$ mm. The start point is shown in green and the stop point in red. The 230×125 mm workspace for our manipulator is represented by a blue rectangle.

tem tracked angular error during these experiments, and the error as a function of path distance is shown in Fig. 3.16. Note how the Taylor series solution oscillates around the desired rotation, while the optimized solution only converges at the end. The final error for each test is shown in Fig. 3.17. The 12-segment optimized path outperforms the path designed using Taylor series approximation.

4. **Movement in $\mathbb{R}^2 \times SO(3)$:** Our final tests compare movement in both position and orientation. We use two 12-segment optimized paths that respectively approximate ensemble movements of

$$\begin{aligned} \left([\Delta x, \Delta y]^\top, \Delta R\right) &= \left([0, 0]^\top, R_z(\pi)\right) \\ \text{and } \left([\Delta x, \Delta y]^\top, \Delta R\right) &= \left([2r, 0]^\top, R_x(0)\right). \end{aligned}$$

The desired and actual paths are shown in Fig. 3.18 and Fig. 3.19. The path angular error is shown in Fig. 3.20 and the ending angular error shown in Fig. 3.21.

3.6 Conclusion

We began with the problem of manipulating a plate-ball system where the sphere has an unknown but bounded radius. We modeled the sphere as an

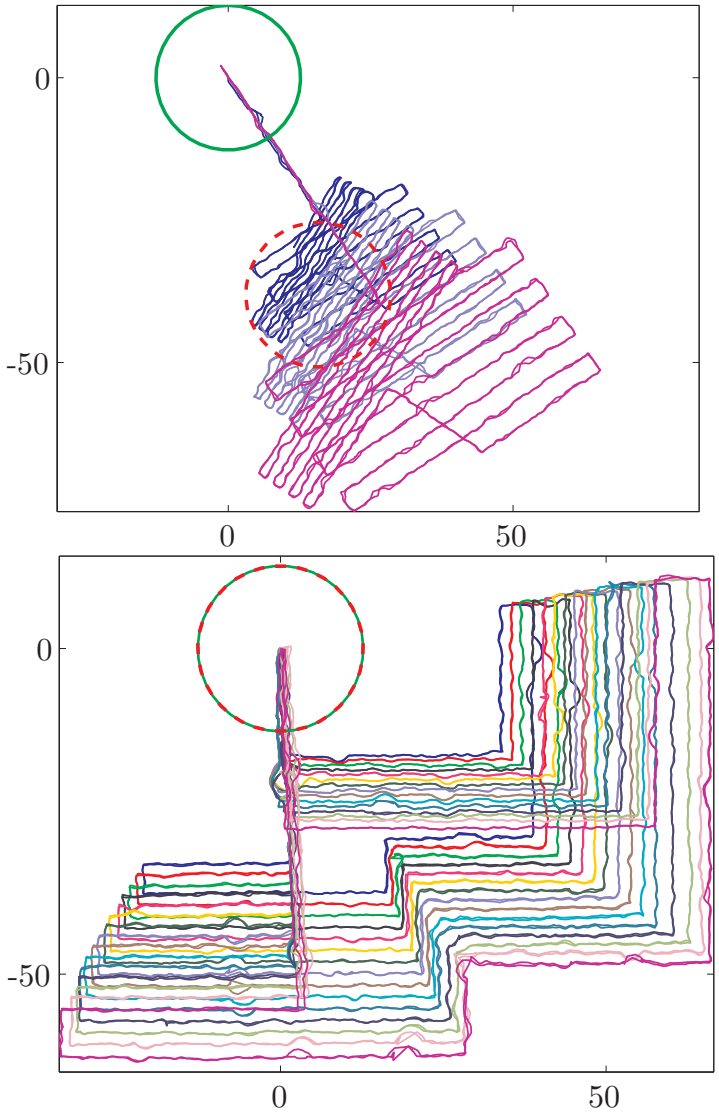


Figure 3.15: Experimental paths followed for rotating 2 radians about the axis $[3, 2, 1]^\top$. Top, path using Taylor series. Bottom, 12-segment optimized path. The path of contact point is shown. Units are mm.
(Consistent color for each assumed radius in Figs. 3.15-3.17)

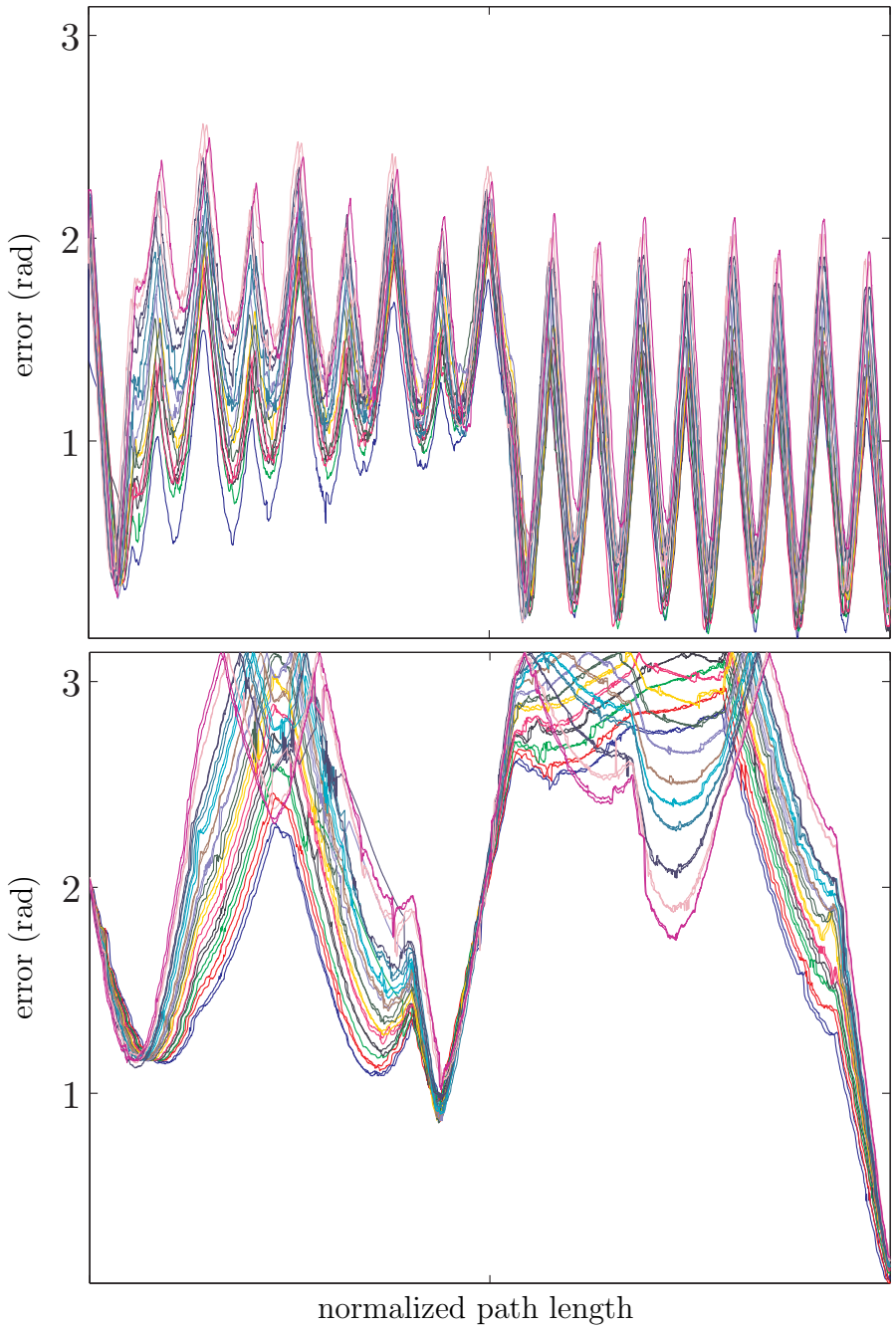


Figure 3.16: Measured angular error as a function of path length for different assumed radii while rotating 2 radians about the axis $[3, 2, 1]^\top$. Top, error using Taylor series approximate rotation, bottom error using 12-segment optimized-path.
(Consistent color for each assumed radius in Figs. 3.15-3.17)

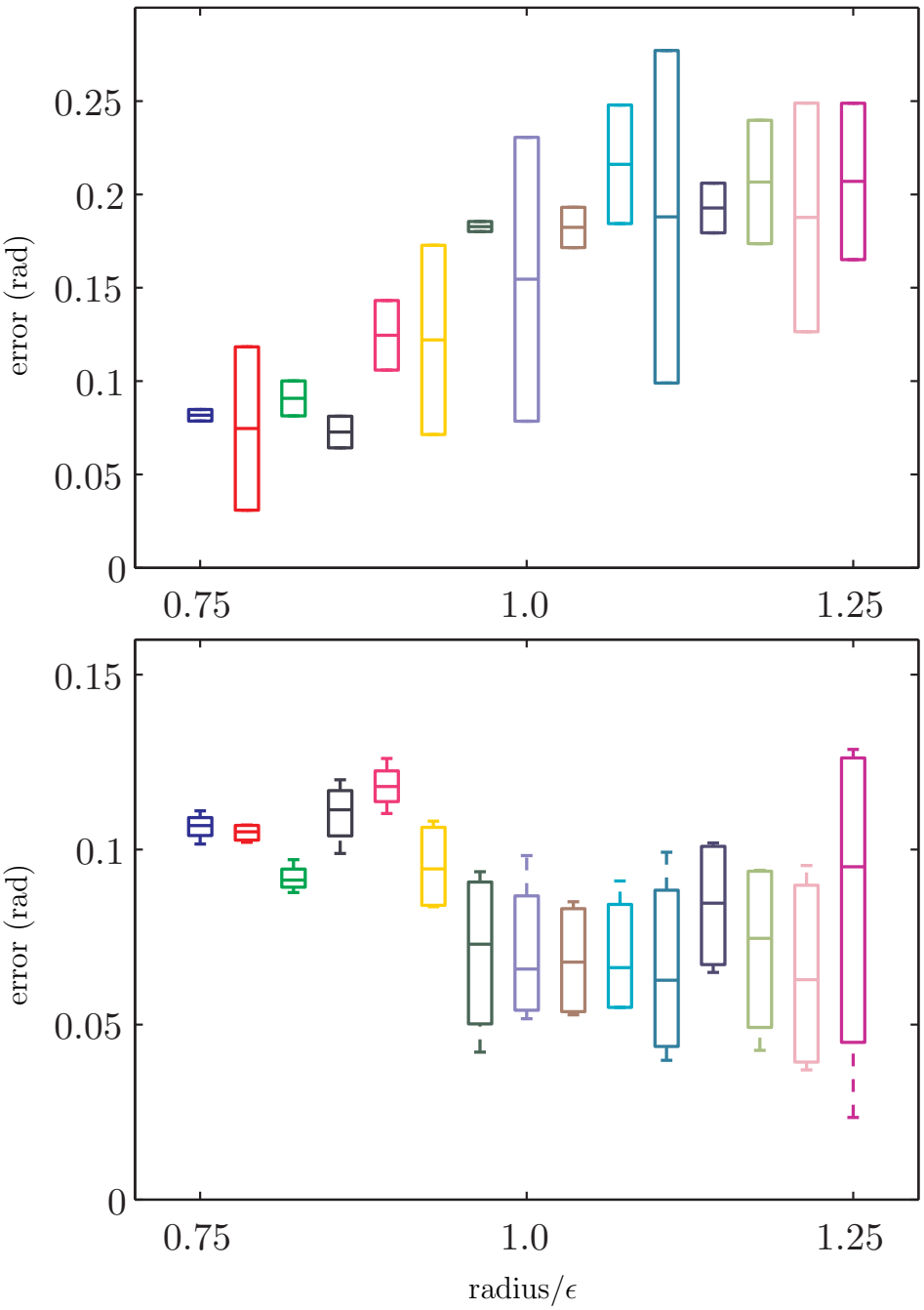


Figure 3.17: Measured ending angular error for rotating 2 radians about the axis $[3, 2, 1]^\top$. Top, error using Taylor series approximate rotation, bottom error using 12-segment optimized path. Note the different y scales—following the optimized path results in 1/2 to 1/3 the error of the primitive-based path.
(Consistent color for each assumed radius in Figs. 3.15-3.17)

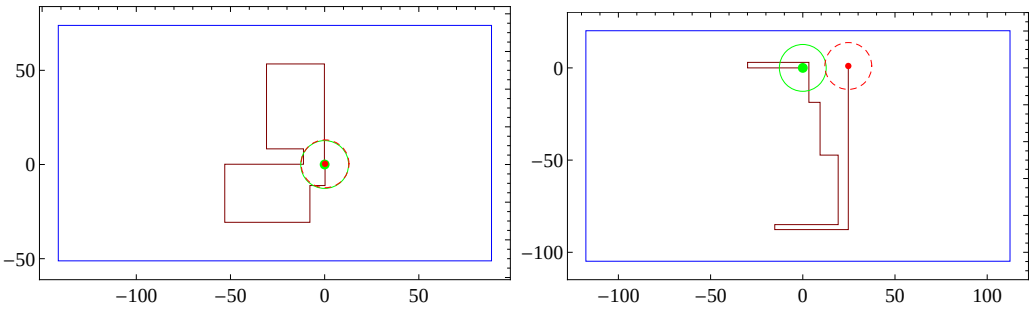


Figure 3.18: Paths for the experiments rolling to a goal in $\mathbb{R}^2 \times SO(3)$. Left, $\left([\Delta x, \Delta y]^\top, \Delta R\right) = \left([0,0]^\top, R_z(\pi)\right)$ and right $\left([2r,0]^\top, R_x(0)\right)$. The path lengths are $\{314, 336\}$mm. The start point is shown in green and the stop point in red. The 230×125mm workspace for our manipulator is represented by a blue rectangle.

ensemble control system, showed that this system is ensemble controllable, and derived an approximate steering algorithm to brings the sphere to within an arbitrarily small neighborhood of any given position and orientation in $\mathbb{R}^2 \times SO(3)$. We applied our work to manipulate spheres with unknown but bounded diameters, and validated our approach with hardware experiments that simultaneously reoriented multiple spheres.

Our solutions consisted of open-loop paths that could be precomputed. We demonstrated that the hardware system has low noise that was $\approx$0.04% of commanded inputs. This noise introduces a drift term to the state evolution that cannot be countered by open-loop control. In many application environments it is practical to add a camera system to sense the ball orientations. We implemented and described such a vision system in section 3.5.2.

We also showed that a continuum of different-sized spheres are approximately controllable by a shared input. Thus, a finite number of different-sized spheres are also approximately controllable by a shared input. Using methods similar to Chapter 4 with differential-drive robots and [14] with the Bloch system, future work will apply feedback techniques to our plate-ball system.

Finally, an important contribution of this chapter was a new experimental platform, so we provided a characterization of the system reliability and described the system so it can be replicated. The magnetic actuation makes the mechanism easy to implement, allows an unobstructed view of the ball on the plane, and enables simultaneous manipulation of multiple balls with different radii.

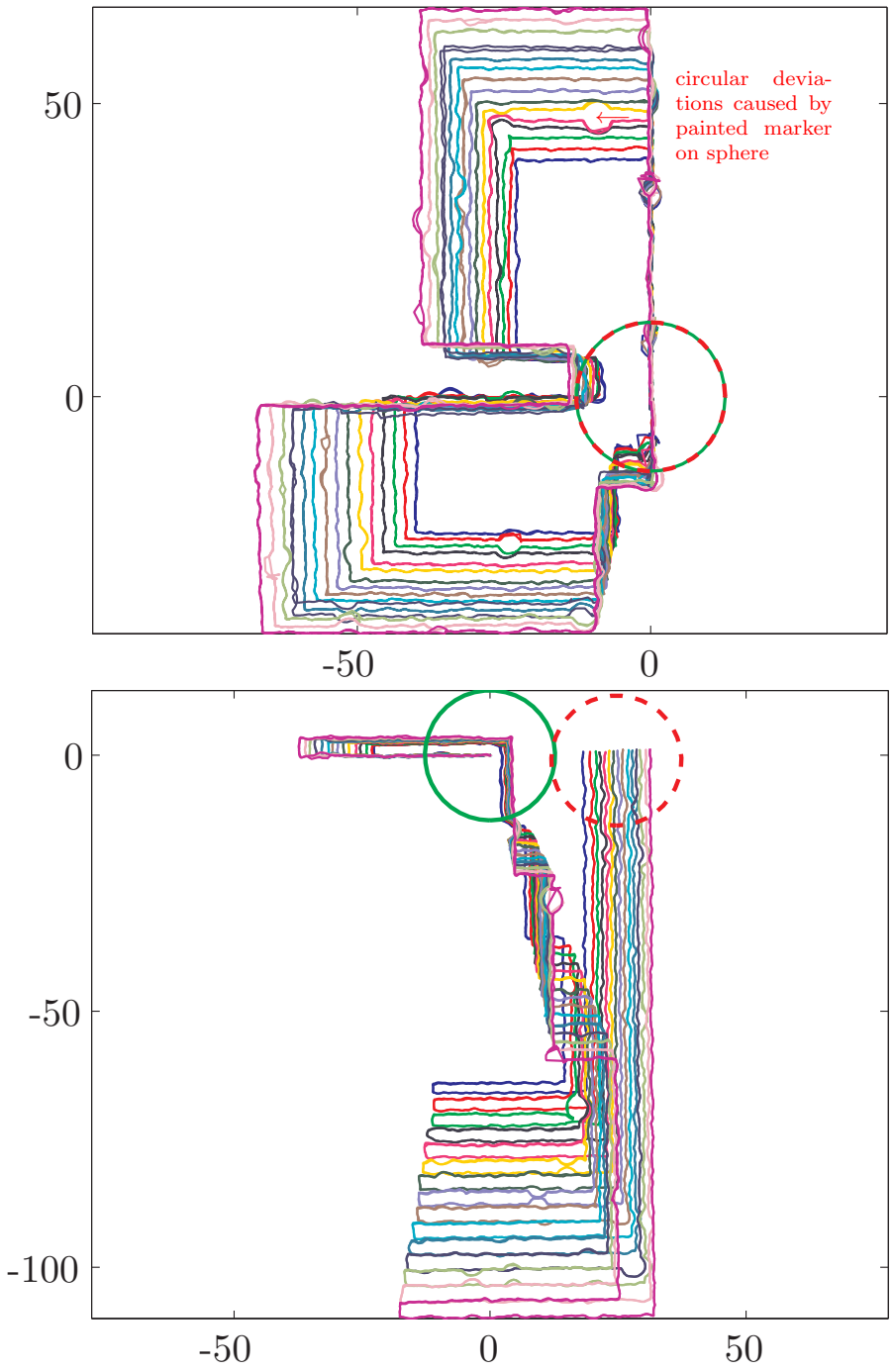


Figure 3.19: Measured paths for rolling to a goal in $\mathbb{R}^2 \times SO(3)$. Top, $\left([\Delta x, \Delta y]^\top, \Delta R\right) = \left([0,0]^\top, R_z(\pi)\right)$ and bottom $\left([2r,0]^\top, R_x(0)\right)$. The path of contact point is shown. Units are mm.
(Consistent color for each assumed radius in Figs. 3.19-3.21)

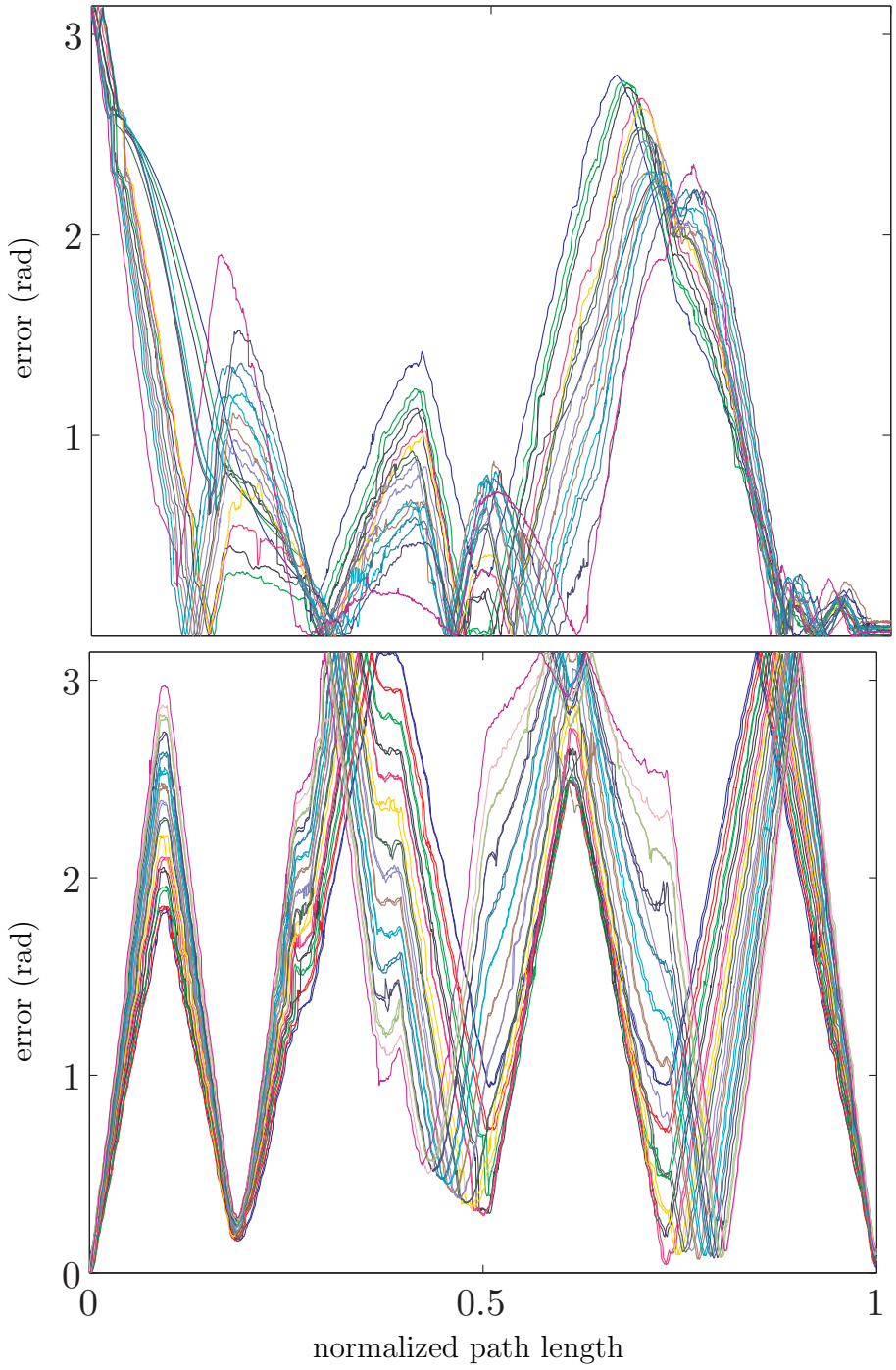


Figure 3.20: Measured angular error as a function of path length for different assumed radii. Top, $\left([\Delta x, \Delta y]^\top, \Delta R\right) = \left([0,0]^\top, R_z(\pi)\right)$ and bottom $\left([2r,0]^\top, R_x(0)\right)$.
(Consistent color for each assumed radius in Figs. 3.19-3.21)

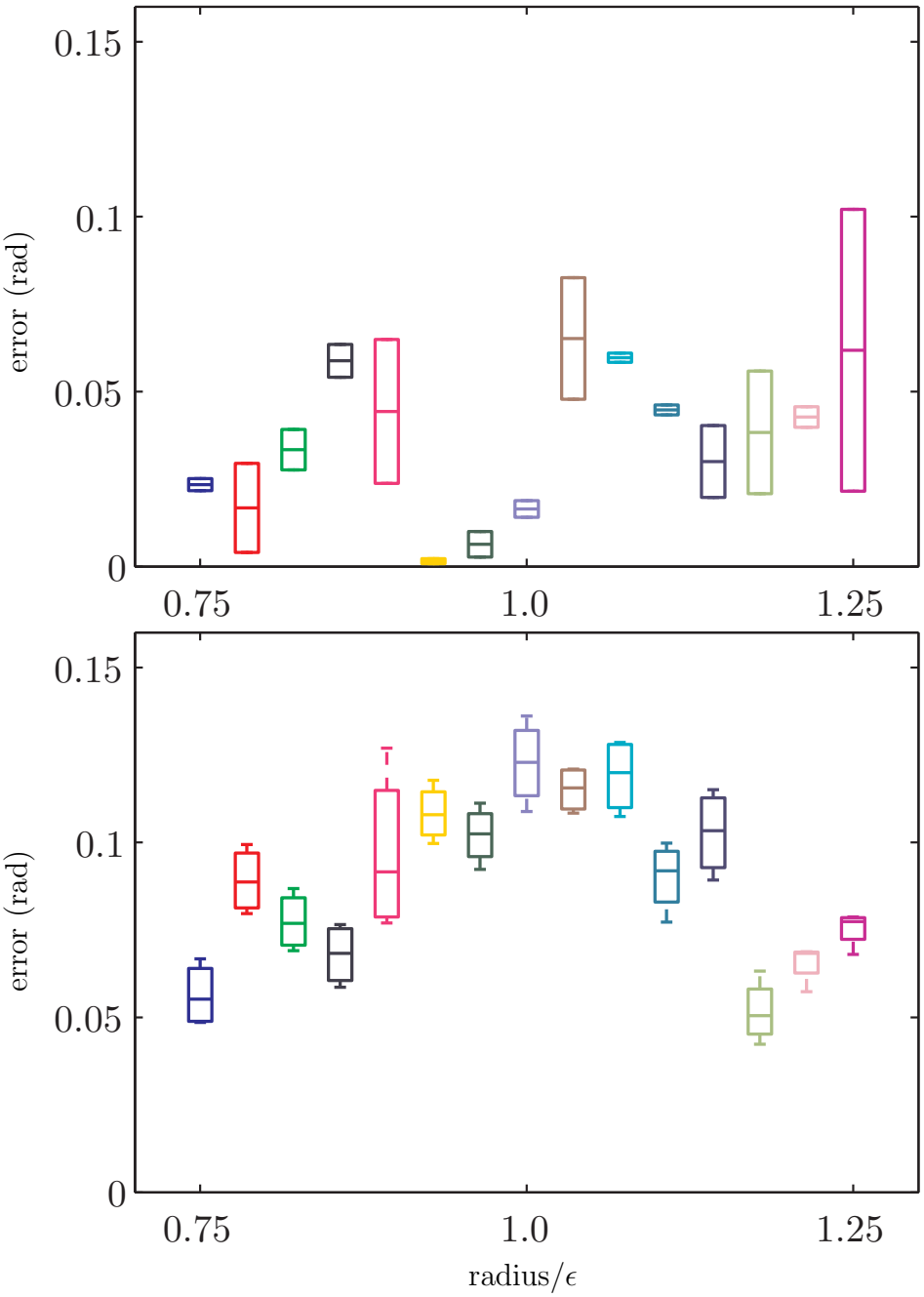


Figure 3.21: Measured ending angular error for rolling to a goal in $\mathbb{R}^2 \times SO(3)$. Top, $\left([\Delta x, \Delta y]^\top, \Delta R\right) = \left([0,0]^\top, R_z(\pi)\right)$ and bottom $\left([2r, 0]^\top, R_x(0)\right)$.
(Consistent color for each assumed radius in Figs. 3.19-3.21)

Chapter 4

Feedback Control of Many Differential-Drive Robots with Uniform Control Inputs

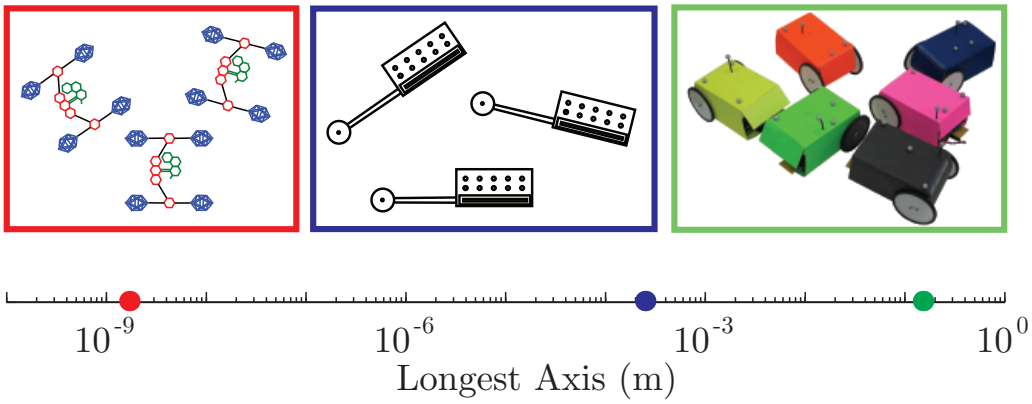


Figure 4.1: Three robotic systems with uniform inputs. On the left, light-driven nanocars [91]. In the middle, scratch-drive micro-robots [92]. On the right, six differential-drive robots commanded by a broadcast control signal.

In this chapter, we derive a globally asymptotically stabilizing feedback control policy for a collection of differential-drive robots under the constraint that every robot receives exactly the same control input. We begin by assuming that each robot has a slightly different wheel size, which scales each robot's forward speed and turning rate by a constant. These constants may be found by offline or online calibration. The resulting feedback policy is easy to implement, is robust to standard models of noise, and scales to an arbitrary number (even a continuous ensemble) of robots. We validate this policy with hardware experiments, which additionally reveal that our feedback policy still works when the wheel sizes are unknown and even when the wheel sizes are all approximately identical. These results have possible future application to control of micro- and nano-scale robotic systems, which are often subject to similar constraints.

Material from this chapter appeared in a preliminary conference version as [93].

4.1 Introduction

Next-generation micro-scale and nano-scale robotic systems have little-to-no onboard computation, and most are designed such that all robots in the system receive uniform control inputs. Two robotic systems of this type are shown in Fig. 4.1: light-driven nanocars and scratch-drive micro-robots.

The light-driven nanocar [91, 94] is a synthesized molecule 1.7×1.38 nm in size containing a uni-directional molecular motor, actuated by a certain wavelength of light. Future work by Tour et al. aims to add controllable steering to this molecule.

The scratch-drive micro-robot, from Donald and Paprotny et al. [92,95,96], is a device 60×250 μm in size actuated by varying the electric potential across a substrate; multiple scratch-drive robots on the same substrate are controlled by this single uniform control input. To independently control each micro-robot, their system is designed with unique species of robots such that individual species can be actuated while the others are immobilized or spin in place.

The motion of both systems can be roughly approximated by a nonholonomic unicycle. A common question is therefore—how do we steer a collection of unicycles under the constraint that every one receives exactly the same control inputs? This question is the one we address here.

We will investigate a collection of differential-drive robots under this same constraint—that every robot receives exactly the same control inputs. Nominally, a system of n differential-drive robots is not controllable. The path followed by each robot will be a rigid-body transformation of the path followed by every other robot. In practice, however, each robot is slightly different, and this inhomogeneity can be exploited in a systematic way in order to recover controllability. In particular, we will show that if each robot has a different wheel size, then we can derive a globally asymptotically stabilizing feedback control policy that steers the position of all robots (independently) between given start and goal configurations, despite the fact that they all receive the same control inputs. Similar inhomogeneities can be found in the systems of Fig. 4.1 (and in other micro/nano-scale robotic systems). For example, small imperfections in their scratch-drive actuators lead to speed variations between different scratch-drive micro-robots. Donald et al. reported speed variations of ±40% of the commanded speed. This data is

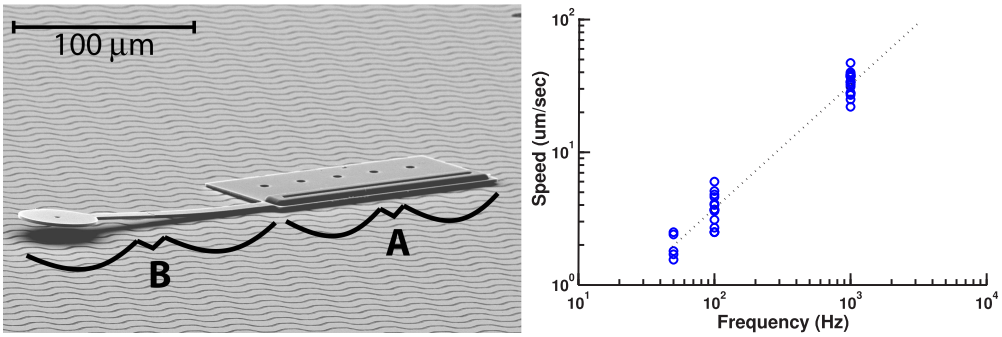


Figure 4.2: Left, a *scratch-drive microrobot*, a microrobot with unicycle kinematics actuated by toggling the voltage levels of the substrate. These robots can be commanded to move forward by duty cycling the substrate voltage. Larger voltage pulses can snap the actuator arm (B) down, creating a pivot point for turning, while low pulses can return the arm to an up position. Right, a plot showing speed variation for a set of scratch-drive robots. This range of velocities could enable controllability of multiple robots using a common control signal.
Figure reprinted, with permission, from Donald et al. [92] © 2008 IEEE.

reprinted in Fig. 4.2.

Our approach is based on the application of ensemble control, which we used in Chapter 2 to derive an approximate (open-loop) steering algorithm for a nonholonomic unicycle despite model perturbation (e.g., unknown wheel size) that scales both the forward speed and turning rate by an unknown but bounded constant [56]. Rather than steer one unicycle with an unknown parameter, we chose to steer an infinite collection of unicycles, each with a particular value of this parameter in some bounded set. The idea was that if the same control inputs steered the entire ensemble from start to goal, then surely they would steer the particular unicycle of interest from start to goal, regardless of its wheel size.

Here, we take advantage of this idea in a slightly different way. Rather than trying to mitigate the effects of bounded model perturbation (i.e., of inhomogeneity), we are trying to exaggerate these effects. Basic controllability results carry over from Chapter 2. The main contribution of this chapter is to derive a closed-loop feedback policy that guarantees exact asymptotic convergence of the ensemble to any given position. We note that, for single robots, it is possible to build a robust feedback controller that regulates position and orientation [21]. It is not obvious that the same can be done for an infinite collection of robots.

This chapter proceeds as follows. In Section 4.2, we provide a globally asymptotically stabilizing feedback control policy to control an ensemble of differential-drive robots. We discuss implementation details in Section 4.3. We demonstrate the convergence of our policy under a standard noise model in simulation (Section 4.4) and in hardware experiments (Section 4.5). These experiments revealed surprising results: *(1)* our policy still works when the wheel sizes are incorrectly specified and *(2)* our policy still works if all robots are approximately identical.

4.2 Global Asymptotic Stabilization of an Ensemble of Unicycles

In this section, we provide a control policy that globally asymptotically stabilizes an infinite ensemble of unicycles. This control policy sets the linear velocity $u_1(t)$ to decrease the position error. There exist configurations at which no $u_1(t)$ can decrease the position error; however, we prove that at any such configuration, except the origin, the ensemble can always be rotated in place until there exists some $u_1(t)$ that will decrease the position error.

Consider a single unicycle that rolls without slipping. We describe its configuration by $q = [x, y, \theta]^\top$ and its configuration space by $\mathcal{Q} = \mathbb{R}^2 \times \mathbb{S}^1$. The control inputs are the forward speed $u_1 \in \mathbb{R}$ and turning rate $u_2 \in \mathbb{R}$. The kinematics of the unicycle are given by

$$\dot{q}(t) = u_1(t) \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} + u_2(t) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{4.1}$$

Given $q(0), q_{\text{goal}} \in \mathcal{Q}$, the control problem for regulating position is to find inputs $u_1(t)$ and $u_2(t)$ such that for any $q(0)$ and q_{goal},

$$\lim_{t\to\infty} \left\| \begin{bmatrix} q_1(t) \\ q_2(t) \end{bmatrix} - \begin{bmatrix} q_{\text{goal},}1(t) \\ q_{\text{goal},}2(t) \end{bmatrix} \right\|_2 = 0.$$

If such inputs always exist, then we say that the system is *globally asymptotically stabilizable*.

We will solve this control problem under model perturbations which scale

u_1 and u_2 by some unknown, bounded constant $\epsilon \in [1-\delta, 1+\delta]$ for some $0 \leq \delta < 1$.

As in Chapter 2, our approach is to steer an uncountably infinite collection of unicycles parametrized by ϵ, each one governed by

$$\dot{q}(t,\epsilon) = \epsilon \left(u_1(t) \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} + u_2(t) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right). \tag{4.2}$$

We choose $u_2(t) = 1$ so that

$$\begin{aligned} \dot{x}(t,\epsilon) &= \epsilon u_1(t) \cos(\epsilon t) \\ \dot{y}(t,\epsilon) &= \epsilon u_1(t) \sin(\epsilon t). \end{aligned} \tag{4.3}$$

Theorem 5. *The ensemble* (4.3) *with* $0 \leq \delta < 1$ *is globally asymptotically stabilizable.*

Proof: We will prove the origin is globally asymptotically stabilizable by using a control-Lyapunov function [97]. A suitable Lyapunov function is the mean squared distance of the ensemble from the origin:

$$\begin{aligned} V(t,x,y) &= \int_{1-\delta}^{1+\delta} \frac{1}{2\epsilon} \left(x^2(t,\epsilon) + y^2(t,\epsilon) \right) d\epsilon \\ \dot{V}(t,x,y) &= \int_{1-\delta}^{1+\delta} \frac{1}{\epsilon} \left(x(t,\epsilon)\dot{x}(t,\epsilon) + y(t,\epsilon)\dot{y}(t,\epsilon) \right) d\epsilon \\ &= u_1(t) \int_{1-\delta}^{1+\delta} \left(x(t,\epsilon)\cos(\epsilon t) + y(t,\epsilon)\sin(\epsilon t) \right) d\epsilon \\ &= u_1(t) F(t,x,y) \end{aligned} \tag{4.4}$$

Here, $F(t,x,y)$ is the integral term which is finite as long as $x(t,\epsilon)$ and $y(t,\epsilon)$ are square integrable over ϵ. To ensure $F(t,x,y)$ is square integrable we will require that the initial configurations $x(0,\epsilon)$ and $y(0,\epsilon)$ be piecewise continuous. We note here that $V(t,x,y)$ is positive definite and radially unbounded, and $V(t,x,y) \equiv 0$ only at $(x(t,\epsilon), y(t,\epsilon)) = (0,0)$.

4.2.1 Designing a Control Policy

To make $\dot{V}(t,x,y)$ negative semi-definite, we choose

$$\begin{aligned} u_1(t) &= -F(t,x,y) \\ &= -\int_{1-\delta}^{1+\delta} \left(x(t,\epsilon)\cos(\epsilon t) + y(t,\epsilon)\sin(\epsilon t)\right) d\epsilon. \end{aligned} \tag{4.5}$$

For such a $u_1(t)$,

$$\dot{V}(t,x,y) = -\left(F(t,x,y)\right)^2.$$

Note here that $\dot{V}(t,x,y) \leq 0$, but there exists a subspace of $(x(t,\epsilon), y(t,\epsilon))$ such that $\dot{V}(t,x,y) = 0$. Because $\dot{V}(t,x,y)$ is negative semi-definite, we can only claim stability, not asymptotic stability. To gain a proof of asymptotic stability, we will use an approach similar to that of Beauchard et al. [14] to extend LaSalle's invariance principle [98] to this infinite-dimensional system. We will proceed by showing the invariant set contains only the origin.

4.2.2 Finding the Invariant Set

Define the set S as all configurations where no $u_1(t)$ exists that can decrease the Lyapunov function:

$$\begin{aligned} S &= \left\{ x(t,\epsilon), y(t,\epsilon) \middle| \dot{V}\left(t, x(t,\epsilon), y(t,\epsilon)\right) = 0 \right\} \\ &= \left\{ x(t,\epsilon), y(t,\epsilon) \middle| -\left(F(t,x,y)\right)^2 = 0 \right\} \\ &= \left\{ x(t,\epsilon), y(t,\epsilon) | F(t,x,y) = 0 \right\}. \end{aligned}$$

Define the time the ensemble enters S as t_0, the orientation of each robot at t_0 as $\theta_0(\epsilon)$, and $t' = t - t_0$. We then define all configurations that remain identically in S as the invariant set S_{inv}. Any configuration that enters this set will never modify its position because $u_1(t) = -F(t,x,y) = 0$ for any configuration in S_{inv}. Therefore we can drop the time-dependence of $x(t,\epsilon)$ and $y(t,\epsilon)$:

$$S_{inv} = \left\{ x(\epsilon), y(\epsilon) \middle| \int_{1-\delta}^{1+\delta} \Big(x(\epsilon)\cos(\epsilon t' + \theta_0(\epsilon)) \right.$$

$$+y(\epsilon)\sin(\epsilon t' + \theta_0(\epsilon))\Big)d\epsilon \equiv 0, \quad \forall t' \geq 0\Big\}.$$

Because there is no time dependence, this property applies for all t':

$$S_{inv} = \Big\{x(\epsilon), y(\epsilon)\Big| \int_{1-\delta}^{1+\delta} \Big(x(\epsilon)\cos(\epsilon t' + \theta_0(\epsilon)) + y(\epsilon)\sin(\epsilon t' + \theta_0(\epsilon))\Big)d\epsilon \equiv 0, \quad \forall t'\Big\}.$$

We can remove $\theta_0(\epsilon)$ with the following change of coordinates

$$\begin{bmatrix} x^*(\epsilon) \\ y^*(\epsilon) \end{bmatrix} = \begin{bmatrix} \cos\left(\theta_0(\epsilon)\right) & \sin\left(\theta_0(\epsilon)\right) \\ -\sin\left(\theta_0(\epsilon)\right) & \cos\left(\theta_0(\epsilon)\right) \end{bmatrix} \begin{bmatrix} x(\epsilon) \\ y(\epsilon) \end{bmatrix},$$

giving the invariant set

$$S_{inv} = \Big\{x(\epsilon), y(\epsilon)\Big| \int_{1-\delta}^{1+\delta} \Big(x^*(\epsilon)\cos(\epsilon t') + y^*(\epsilon)\sin(\epsilon t')\Big)d\epsilon \equiv 0, \quad \forall t'\Big\}.$$

We must show that no configuration except $(x(\epsilon), y(\epsilon)) \equiv (0,0)$ is in S_{inv}. We do this by applying the Fourier transform in t'.

$$\begin{aligned}
&\int_{1-\delta}^{1+\delta} (x^*(\epsilon)\cos(\epsilon t') + y^*(\epsilon)\sin(\epsilon t'))\, d\epsilon \equiv 0, && \forall t' \\
&\mathcal{F}\left[\int_{1-\delta}^{1+\delta} (x^*(\epsilon)\cos(\epsilon t') + y^*(\epsilon)\sin(\epsilon t'))\, d\epsilon\right]\{\omega\} \equiv \mathcal{F}[0]\{\omega\}, && \forall \omega \\
&\mathcal{F}\left[\int_{1-\delta}^{1+\delta} (x^*(\epsilon)\cos(\epsilon t') + y^*(\epsilon)\sin(\epsilon t'))\, d\epsilon\right]\{\omega\} \equiv 0, && \forall \omega \quad (4.6)
\end{aligned}$$

By linearity of the Fourier transformation (4.6) is

$$\int_{1-\delta}^{1+\delta} \Big(\mathcal{F}[x^*(\epsilon)\cos(\epsilon t')]\{\omega\} + \mathcal{F}[y^*(\epsilon)\sin(\epsilon t')]\{\omega\}\Big)d\epsilon \equiv 0, \quad \forall \omega$$

We then apply the Fourier transform of $\sin(\cdot)$ and $\cos(\cdot)$ as follows.

$$\int_{1-\delta}^{1+\delta} \sqrt{\frac{\pi}{2}}\Big(x^*(\epsilon)\left(\underline{\boldsymbol{\delta}}(-\epsilon+\omega) + \underline{\boldsymbol{\delta}}(\epsilon+\omega)\right) + \mathrm{i}y^*(\epsilon)\left(\underline{\boldsymbol{\delta}}(-\epsilon+\omega) - \underline{\boldsymbol{\delta}}(\epsilon+\omega)\right)\Big)d\epsilon \equiv 0, \quad \forall \omega, \qquad (4.7)$$

where $\underline{\boldsymbol{\delta}}(\cdot)$ is the Dirac-delta operator. The Dirac-delta operator is non-zero only when $\epsilon = \pm\omega$. Because $\epsilon \in [1-\delta, 1+\delta]$ and $0 \leq \delta < 1$, we can integrate (4.7) for all $\omega \in [1-\delta, 1+\delta]$ to show that in the invariant set

$$x^*(\omega) + \mathrm{i}y^*(\omega) \equiv 0, \quad \forall \omega \in [1-\delta, 1+\delta]. \tag{4.8}$$

Because x^* and y^* are both real-valued, (4.8) reduces to

$$x(\epsilon) \equiv 0, y(\epsilon) \equiv 0, \quad \forall \epsilon \in [1-\delta, 1+\delta].$$

We have shown that V is positive-definite and radially unbounded, $\dot{V}$ is negative semi-definite, and the only invariant point where $\dot{V} = 0$ is the origin. Therefore, we conclude the origin of the system (4.3) is globally asymptotically stable under the control policy

$$\begin{aligned} u_1(t) &= -\int_{1-\delta}^{1+\delta} \left(x(\epsilon)\cos(\epsilon t) + y(\epsilon)\sin(\epsilon t)\right) d\epsilon \\ u_2(t) &= 1. \end{aligned} \tag{4.9}$$

■

4.3 Implementation

In this section, we explain extensions of our control policy to unidirectional inputs and to discrete-time, finite ensembles, and describe a standard noise model. These extensions are useful for implementing our policy.

4.3.1 Extension to Unidirectional Vehicles

Our proof of GAS stability uses a control policy with positive angular velocity commands but positive and negative linear velocity commands. The nano- and micro-robots we are using for inspiration [91, 92, 94–96] cannot produce negative linear velocities. To extend the control law to robots that cannot move backwards, we modify it as follows:

$$u_1(t) = \max\left(0, -\int_{1-\delta}^{1+\delta} \left(x_{t,\epsilon}\cos(\epsilon t) + y_{t,\epsilon}\sin(\epsilon t)\right) d\epsilon\right).$$

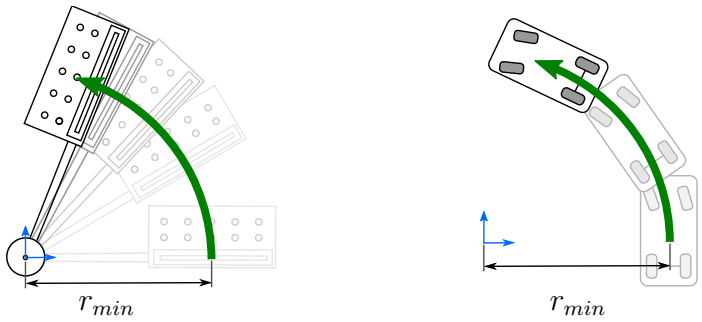


Figure 4.3: Because our control policy requires only non-negative turns, it can be extended to vehicles with a minimum turning radius (e.g. scratch-drive robots and automobiles) by defining the robot center to be the center of rotation.

$$u_2(t) = 1. \tag{4.10}$$

In this case, the invariant set is

$$\begin{aligned} S_{inv} &= \{x(\epsilon), y(\epsilon) | F(t,x,y) \leq 0, \quad \forall t' \geq 0\} \\ &= \left\{ x(\epsilon), y(\epsilon) \middle| \int_{1-\delta}^{1+\delta} \Big(x^*(\epsilon)\cos(\epsilon t') + y^*(\epsilon)\sin(\epsilon t') \Big) d\epsilon \geq 0, \quad \forall t' \geq 0 \right\}. \end{aligned} \tag{4.11}$$

We have already shown that the only solution identically equal to zero for all $t' \geq 0$ is $x(\epsilon) = y(\epsilon) = 0$. However, S_{inv} now allows negative $F(t,x,y)$ values and our proof of Theorem 5 does not apply.

Empirically in both simulation and hardware experiments, the control policy 4.10 is still asymptotically stable. In our experiments the Lyapunov function error under the unidirectional control policy is a constant factor greater than the error under the bidirectional control policy.

We note briefly that scratch-drive robots cannot turn in place, but instead pivot around the tip of their steering arm. Because our control policy requires only non-negative turns, it can be extended to vehicles with a minimum turning radius by the following transformation. Given a minimum turning radius r_{min}, define the new robot center to be the center of rotation. This is a translation $(-r_{min}, 0)$ in the robot reference frame, as shown in Fig. 4.3.

4.3.2 Extension to a Finite Ensemble of Robots

Thus far we have worked in continuous time for an ensemble of robots. Many real-life applications, including the micro- and nano-robots we discussed above, involve a finite number of robots which are controlled and measured in discrete time. We call an ensemble with a finite number of robots a *finite ensemble*.

To model a finite ensemble of n robots, we redefine the system kinematic model from (4.3) as

$$\begin{aligned} \dot{x}_i &= \epsilon_i u_1(t)\cos(\theta_i(t)) \\ \dot{y}_i &= \epsilon_i u_1(t)\sin(\theta_i(t)) \\ \dot{\theta}_i &= \epsilon_i u_2(t), \end{aligned} \tag{4.12}$$

and in the control policy (4.9), we replace the integration over ϵ with a finite sum from 1 to n:

$$\begin{aligned} u_1(t) &= -\frac{1}{n}\sum_{i=1}^{n}\left(x_i\cos(\theta_i(t)) + y_i\sin(\theta_i(t))\right) \\ u_2(t) &= 1, \end{aligned} \tag{4.13}$$

where for the ith robot, ϵ_i is the variable parameter, (x_i, y_i) is the position at time t, and $\epsilon_i t$ is the orientation at time t.

Theorem 6. *The finite ensemble 4.12 under control law 4.13 is globally asymptotically stable.*

Proof: A suitable Lyapunov function is the mean squared distance of the finite ensemble from the origin:

$$V(t,x,y) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2\epsilon_i}\left(x_i^2 + y_i^2\right) \tag{4.14}$$

$$\begin{aligned} \dot{V}(t,x,y) &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\epsilon_i}\left(x_i\dot{x}_i + y\dot{y}_i\right) \\ &= u_1(t)\frac{1}{n}\sum_{i=1}^{n}\left(x_i\cos\left(\epsilon_i t + \theta_i(0)\right) + y_i\sin\left(\epsilon_i t + \theta_i(0)\right)\right) \\ &= u_1(t)F(t) \end{aligned}$$

The invariant set is now

$$S_{inv} = \left\{ \mathbf{x}, \mathbf{y} \middle| \frac{1}{n} \sum_{i=1}^{n} \left(x_i^* \cos(\epsilon_i t') + y_i^* \sin(\epsilon_i t') \right) \equiv 0, \quad \forall t' \geq 0 \right\}.$$

As in 4.2, we then apply the Fourier transform as follows.

$$\frac{1}{n} \sum_{i=1}^{n} \sqrt{\frac{\pi}{2}} \Big(x_i^* \left(\underline{\boldsymbol{\delta}}(-\epsilon_i + \omega) + \underline{\boldsymbol{\delta}}(\epsilon_i + \omega) \right) + \mathrm{i} y_i^* \left(\underline{\boldsymbol{\delta}}(-\epsilon_i + \omega) - \underline{\boldsymbol{\delta}}(\epsilon_i + \omega) \right) \Big) \equiv 0, \quad \forall \omega. \tag{4.15}$$

Again x_i^* and y_i^* are real-valued. By setting $\omega = \epsilon_i$ for $i \in [1, n]$ we can show that in the invariant set

$$(x_i, y_i) = (0, 0) \quad \forall i \in [1, n],$$

and therefore the finite ensemble is globally asymptotically stabilizable with control policy (4.13). ■

4.3.3 Extension to Discrete-Time

To simplify implementation of (4.13) on a robotic testbed with actuation and sensing at discrete times, we split each ΔT time step into two stages with piecewise constant inputs. During the first stage we command the robots to turn in place, and during the second stage we apply a linear velocity.

$$\begin{aligned} F(k) &= \frac{1}{n} \sum_{i=1}^{n} \left(x_i(k) \cos(\theta_i(k)) + y_i(k) \sin(\theta_i(k)) \right) \\ k &= \frac{t}{\Delta T} - \mathrm{mod}\left(t, \Delta T\right) \\ \Big[u_1(t), u_2(t) \Big] &= \begin{cases} \frac{2}{\Delta T} \left[-F(k), 0 \right] & t - k\Delta T < \frac{\Delta T}{2} \\ \frac{2}{\Delta T} \left[0, \quad \phi \right] & \text{else} \end{cases} \end{aligned} \tag{4.16}$$

We can then write the kinematics as

$$\begin{bmatrix} x_i(k+1) \\ y_i(k+1) \end{bmatrix} = \begin{bmatrix} x_i(k) \\ y_i(k) \end{bmatrix} + \begin{bmatrix} \epsilon_i \cos(\theta_i(0) + \epsilon_i k\phi) \\ \epsilon_i \sin(\theta_i(0) + \epsilon_i k\phi) \end{bmatrix} u_1(k), \tag{4.17}$$

for $i = 1, 2, \ldots, n$ and $k \in \mathbb{Z}$. Eq. (4.17) is a discrete-time linear time-varying system. Obviously, as $\phi \to 0$, the discrete-time ensemble (4.17) approaches the continuous-time model (4.12). To prove (4.16) stabilizes (4.17), we show the system is uniformly k-step controllable, as in [99, chap 25.3].

We write (4.17) in standard notation as

$$q_i(k+1) = A_i(k)q_i(k) + B_i(k)u_1(k). \tag{4.18}$$

Here $A_i(k)$ is the identity matrix for all i, k. We can calculate $B_i(k)$ as

$$\begin{aligned}
B_i(0) &= \begin{bmatrix} \cos(\theta_i(0)) \\ \sin(\theta_i(0)) \end{bmatrix} \\
B_i(1) &= \begin{bmatrix} \cos(\theta_i(0) + \epsilon_i\phi) \\ \sin(\theta_i(0) + \epsilon_i\phi) \end{bmatrix} \\
&\vdots \\
B_i(k) &= \begin{bmatrix} \cos\left(\theta_i(0) + \epsilon_i k\phi\right) \\ \sin\left(\theta_i(0) + \epsilon_i k\phi\right) \end{bmatrix} \\
\mathbf{B}(k) &= \begin{bmatrix} B_1(k) \\ B_2(k) \\ \vdots \\ B_n(k) \end{bmatrix}
\end{aligned}$$

We define the controllability matrix $\mathcal{C}_k$ as

$$\mathcal{C}_k = \left[\mathbf{B}_0, \mathbf{B}_1 \ldots \mathbf{B}_{k-1}\right].$$

The finite ensemble with n robots has $2n$ degrees of freedom. To control each robot's x, y position requires $\mathcal{C}$ to be rank $2n$. This matrix is almost always full rank provided that $k \gg 2n$ and a suitable choice of ϕ. In our simulations and hardware experiments we use $\phi = \frac{\pi}{2}$. If $\mathcal{C}_k$ is full rank, then for any starting state $\mathbf{q}_0$ and desired final state $\mathbf{q}_1$, the control sequence is derived by solving in the least squares sense the overdetermined system of equations

$$\mathcal{C}_k \mathbf{u}_{[0,\ldots,k-1]} = (\mathbf{q}_1 - \mathbf{q}_0). \tag{4.19}$$

We note that for $k = 2n$, $\mathcal{C}$ is almost always ill-conditioned, leading to very large control commands and poor convergence. Better results are obtained for $k = 5n$, as shown in Fig. 4.4, with control effort 15 orders of magnitude less than that for $k = 2n$ and exact convergence to the goal.

The control policy (4.16) is easy to implement, never increases the summed distance of the ensemble from the goal, and is robust to standard models of noise.

4.3.4 Applying a Standard Noise Model

To model noise in our simulations we apply the noise model in [22, Chap. 5.4.2] by Thrun et al. This model defines each discrete-time motion as a rotation, a translation, and a second rotation, each perturbed by Gaussian noise. It uses the four parameters $\alpha_1, \alpha_2, \alpha_3$, and α_4 to weight the correlation of noise between rotation and translation commands. If the desired rotation, translation, and second rotation are given by $[\delta_{\text{rot1}}, \delta_{\text{trans}}, \delta_{\text{rot2}}]$, then we can sample a realistic motion by the sequence $[\hat{\delta}_{\text{rot1}}, \hat{\delta}_{\text{trans}}, \hat{\delta}_{\text{rot2}}]$ as follows

$$\begin{aligned}
\hat{\delta}_{\text{rot1}} &= \delta_{\text{rot1}} - \text{sample}(\alpha_1\delta_{\text{rot1}}^2 + \alpha_2\delta_{\text{trans}}^2) \\
\hat{\delta}_{\text{trans}} &= \delta_{\text{trans}} - \text{sample}(\alpha_3\delta_{\text{trans}}^2 + \alpha_4\delta_{\text{rot1}}^2 + \alpha_4\delta_{\text{rot2}}^2) \\
\hat{\delta}_{\text{rot2}} &= \delta_{\text{rot2}} - \text{sample}(\alpha_1\delta_{\text{rot2}}^2 + \alpha_2\delta_{\text{trans}}^2),
\end{aligned} \tag{4.20}$$

where sample(x) generates a random sample from the zero-centered normal distribution with variance x. We use this noise model for all discrete-time simulations with a finite number of robots.

4.4 Simulation Results

Here, we present our simulation methodology and results for both continuous- and discrete-time simulations.

4.4.1 Continuous Time Simulation

We implemented the finite ensemble (4.12) with control policy (4.13) in MATLAB to simulate $n = \{1000, 2000\}$ robots in continuous time for two different

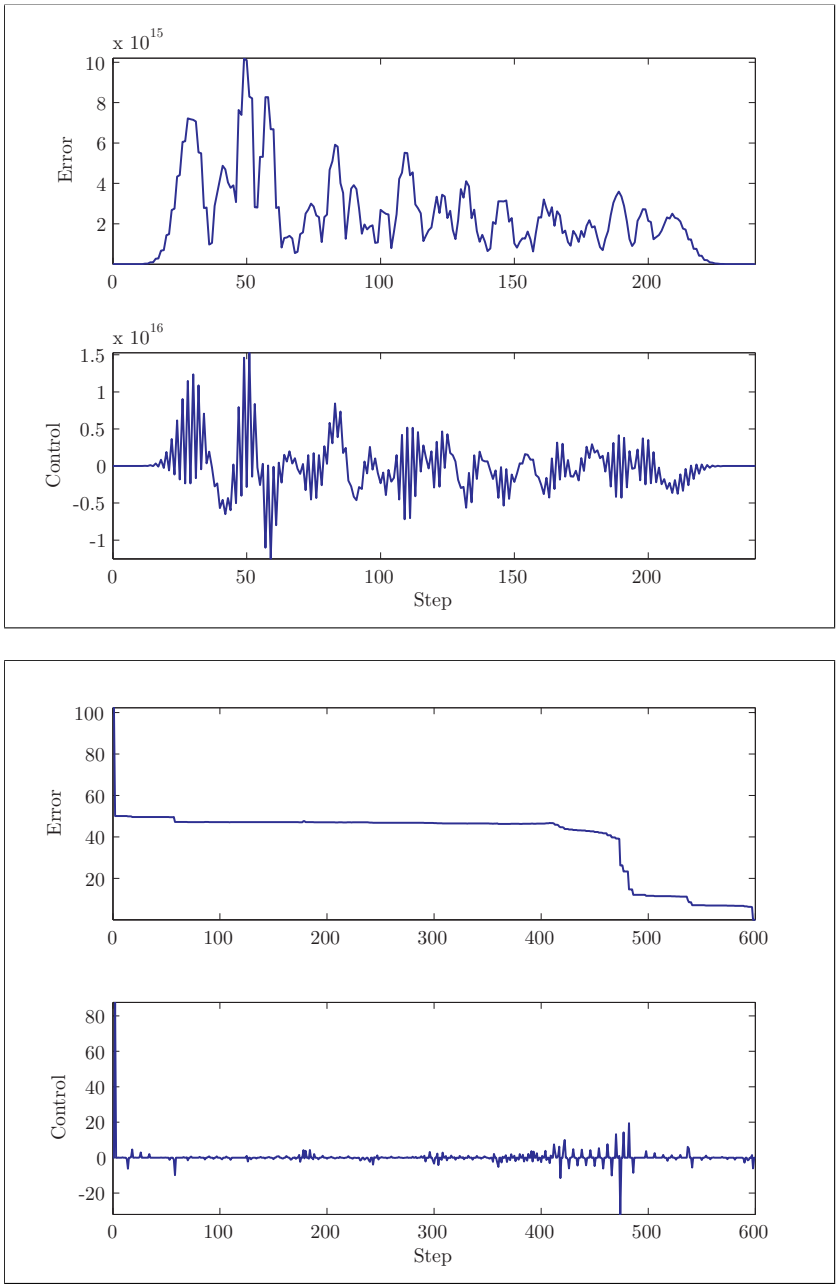


Figure 4.4: Error and control effort of a discrete-time finite ensemble of $n = 120$ robots under control (4.19). At top are results for $k = 2n$. The controllability matrix $\mathcal{C}_k$ is ill conditioned, leading to poor convergence and large control efforts. The bottom plot shows $k = 5n$, leading to control effort 15 orders of magnitude less and convergence to the goal. The initial error for each simulation is 100, but with $k = 2n$ the final error is 58, while the final error for $k = 5n$ is zero.

test cases. For these tests $\delta = 1/2$, and $\epsilon_i = 1 - \delta + \frac{2\delta}{n} i$. We also compared bidirectional policy with a unidirectional policy. In these simulations $u_2(t) = \cos(\sqrt{t})$ because the finite ensemble poorly approximates the continuum for large t when $u_2(t) = 1$.

Point to Point: Robots are initialized to $(x_i, y_i, \theta_i) = (1, 1, 0)$ and steered to the origin. Results are shown in Fig. 4.5.

Path to Point: Robots are initialized to

$$\theta_i = 2\pi i/n, \quad \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) \\ \sin(\theta_i) \end{bmatrix},$$

a circle of radius 1, and steered to the origin. Results are shown in Fig. 4.6.

In each simulation, under our control policy the error converges asymptotically to zero. Additionally, the system errors and trajectories for $n = 1000$ and 2000 are identical, suggesting that this level of discretization accurately represents the ensemble ($n = \infty$) kinematics. Results are shown in Fig. 4.6.

Unidirectional vs. Bidirectional Control Inputs: robots are initialized in the same manner as *Point to Point*, but simulated with the bidirectional control policy (4.13) and the unidirectional control policy (4.10). The error for unidirectional policy is on average twice that of the bidirectional policy. Results are shown in Fig. 4.7.

4.4.2 Discrete Time Simulation

We simulated a discrete-time collection of 120 robots under various levels of noise with both differing and identical values of ϵ. Sample trajectories are shown in Fig. 4.8. We explored three different cases:

Different ϵ Values: Simulating with differing ϵ, we found that with no noise, the position error of our robot collection converged to zero error. When the noise model (4.20) was applied, the error converged to a non-zero value for small values of noise, and diverged for large values of noise, as shown in Fig. 4.9.

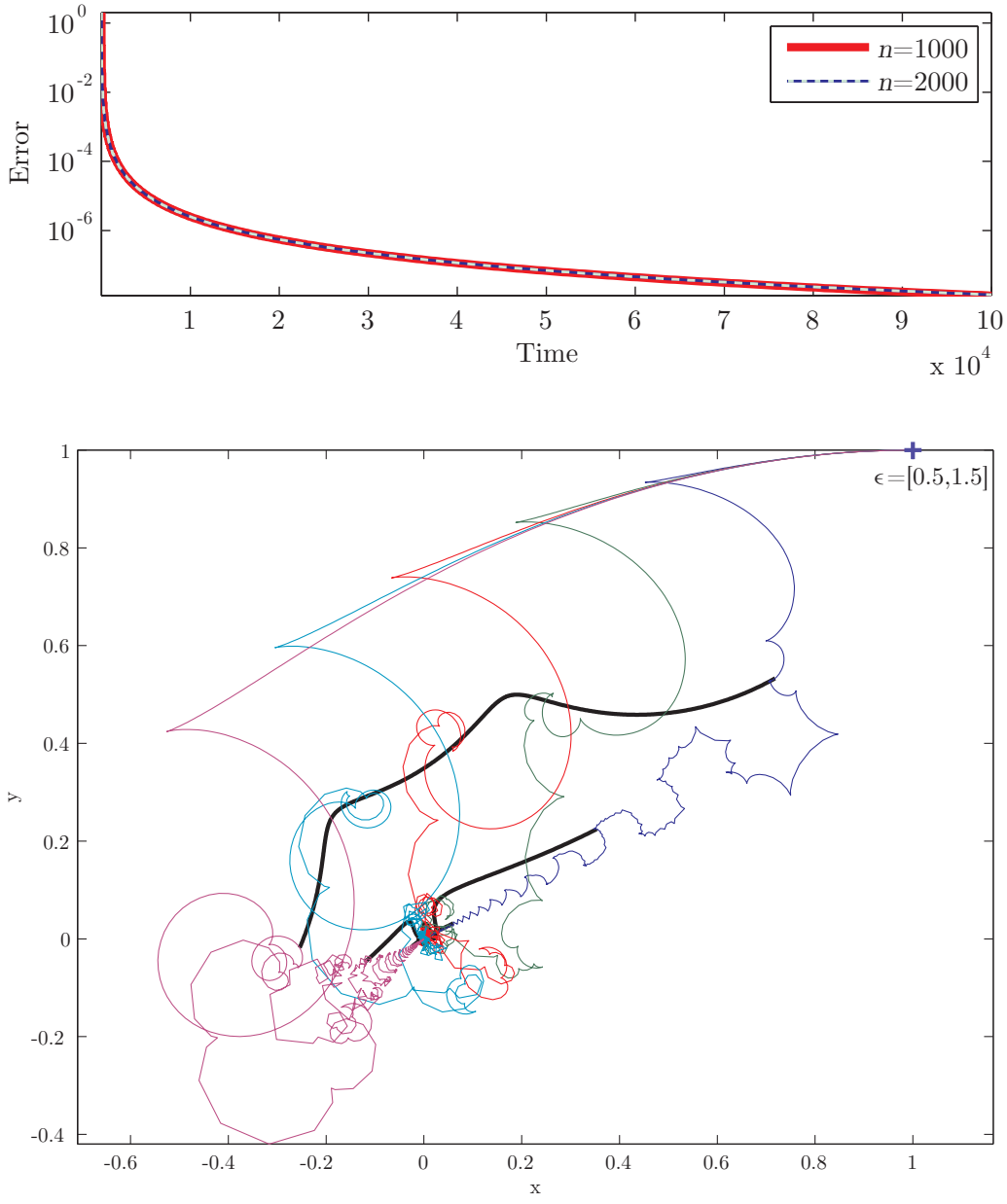


Figure 4.5: Continuous time simulation of n robots, with $\epsilon \in [0.5, 1.5]$, all initialized to $(1, 1)$ and steered to $(0, 0)$ using control policy (4.13). The simulation was run with $n = \{1000, 2000\}$. Each trial achieved the same error, as shown in the top plot. State trajectories of the system are shown in the bottom plot. Lines show the path followed for five particular values of ϵ. Thick black lines show the entire ensemble at instants of time.

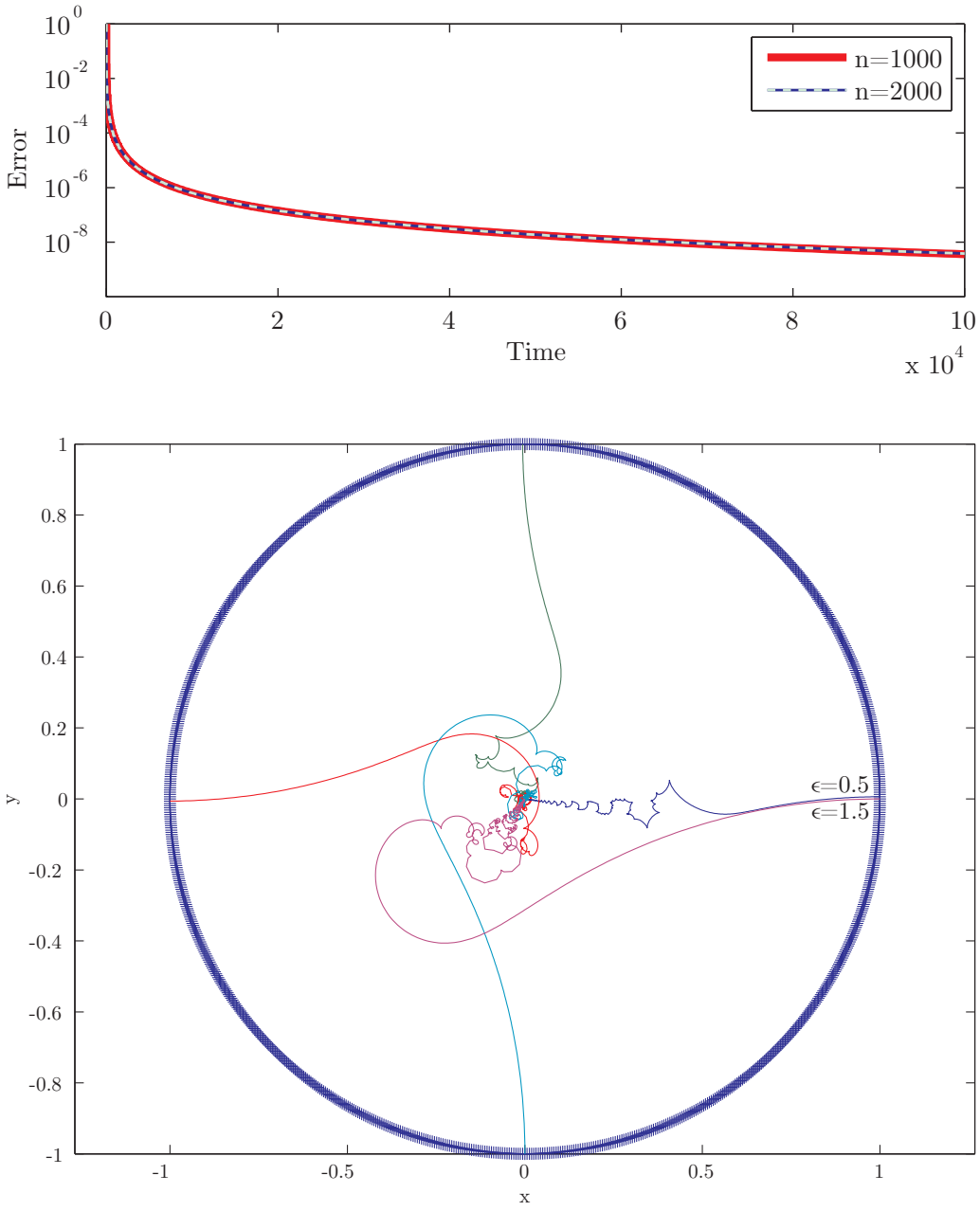


Figure 4.6: Continuous time simulation of n robots, with $\epsilon \in [0.5, 1.5]$, initially evenly distributed about the unit circle and steered to $(0, 0)$ using control policy (4.13). The simulation was run with $n = \{1000, 2000\}$. Each trial achieved the same ending error, as shown in the top plot. State trajectories of the system are shown in the bottom plot. Lines show the path followed for five particular values of ϵ.
(Video online: http://www.youtube.com/watch?v=PAdmASfMqAk).

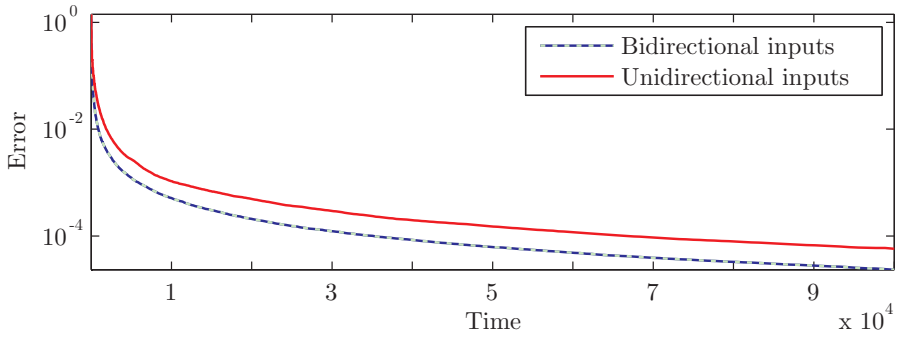


Figure 4.7: Lyapunov function for a continuous time simulation of $n = 1000$ robots, with $\epsilon \in [0.5, 1.5]$, all initialized to $(1, 1)$ and steered to $(0, 0)$ using the bidirectional control policy (4.13) and the unidirectional control policy (4.10). The error for unidirectional policy is on average twice that of the bidirectional policy.

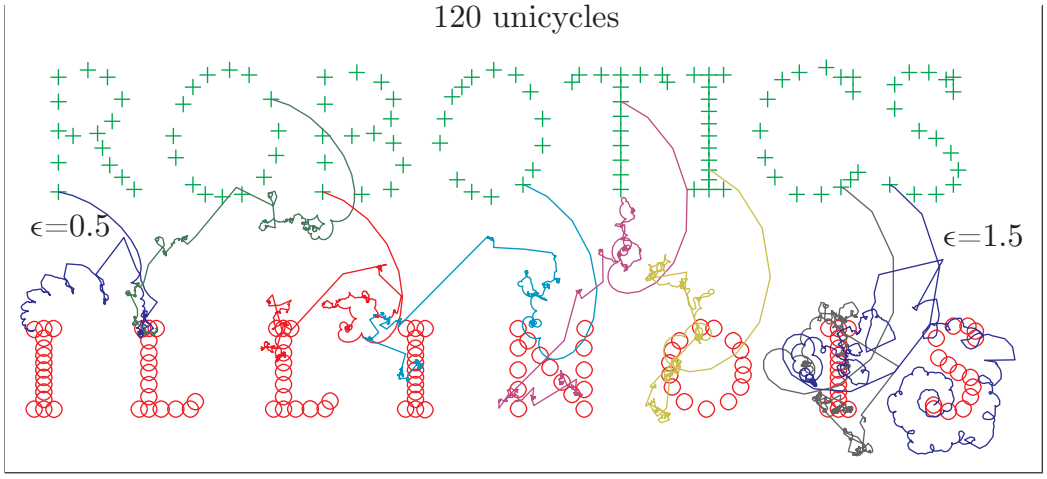


Figure 4.8: Simulation results from applying the control policy from Eq. (4.16) for 120 robots with unicycle kinematics. Wheel size (ϵ) was evenly distributed from 0.5 to 1.5. The plot shows the starting '+' and ending 'o' positions along with 8 selected state trajectories.
(Video online: http://www.youtube.com/watch?v=btWFQrBhrLI)

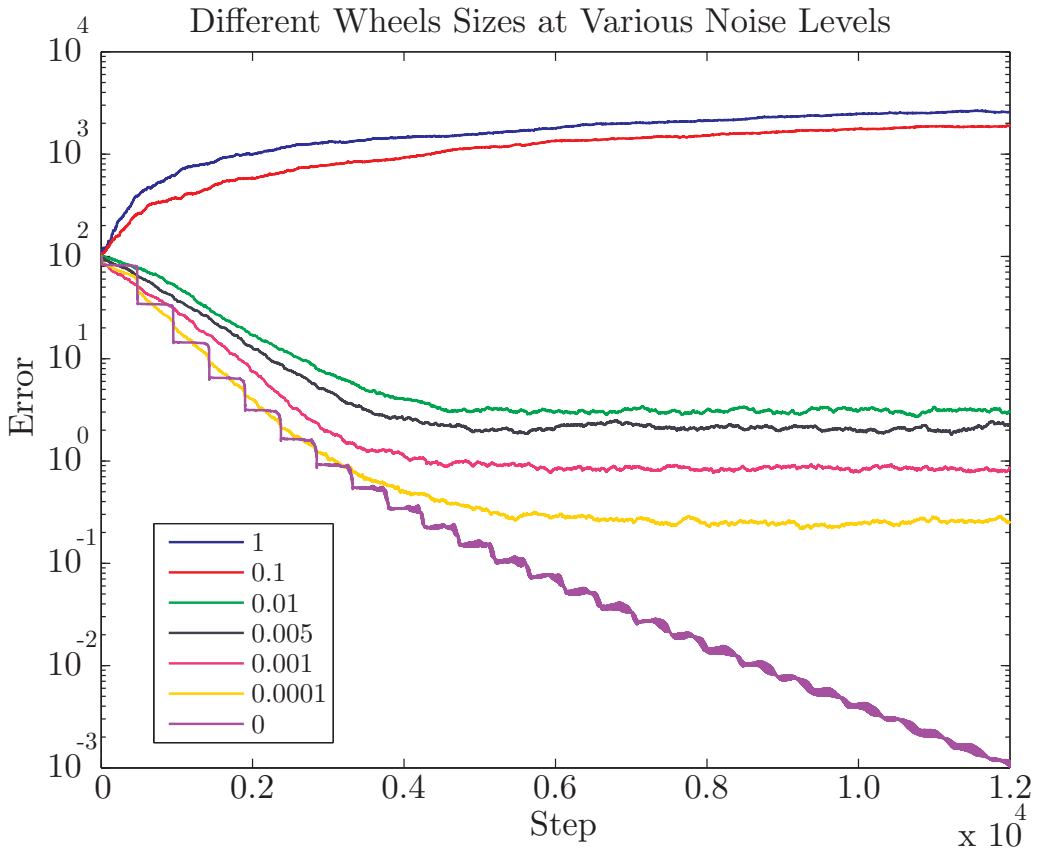


Figure 4.9: Error of a discrete-time, finite collection of 120 robots simulated under a standard noise model (4.20) with $\epsilon \in [0.5, 1.5]$ under different levels of noise parametrized by α; all α are equal.

Identical Robots: When all 120 robots are identical, the smallest position error is achieved within a specific intermediate range of noise values. Large α values caused the error to diverge, while small α values led to very slow convergence. This result is shown in Fig. 4.10.

Effect of Rotational Noise: Again with identical robots, we held the translational and cross-term noise at 0.01, a value which converged quickly in the previous simulation, and varied the rotational noise, α_1. We found that convergence rate increased with α_1, up to a limit of approximately $\alpha_1 = 1$. This result is shown in Fig. 4.11.

These results show that noise is necessary for a finite collection of identical robots to be controllable. This is a subset of a larger class of problems for which noise is beneficial, or even necessary, for stability and control.

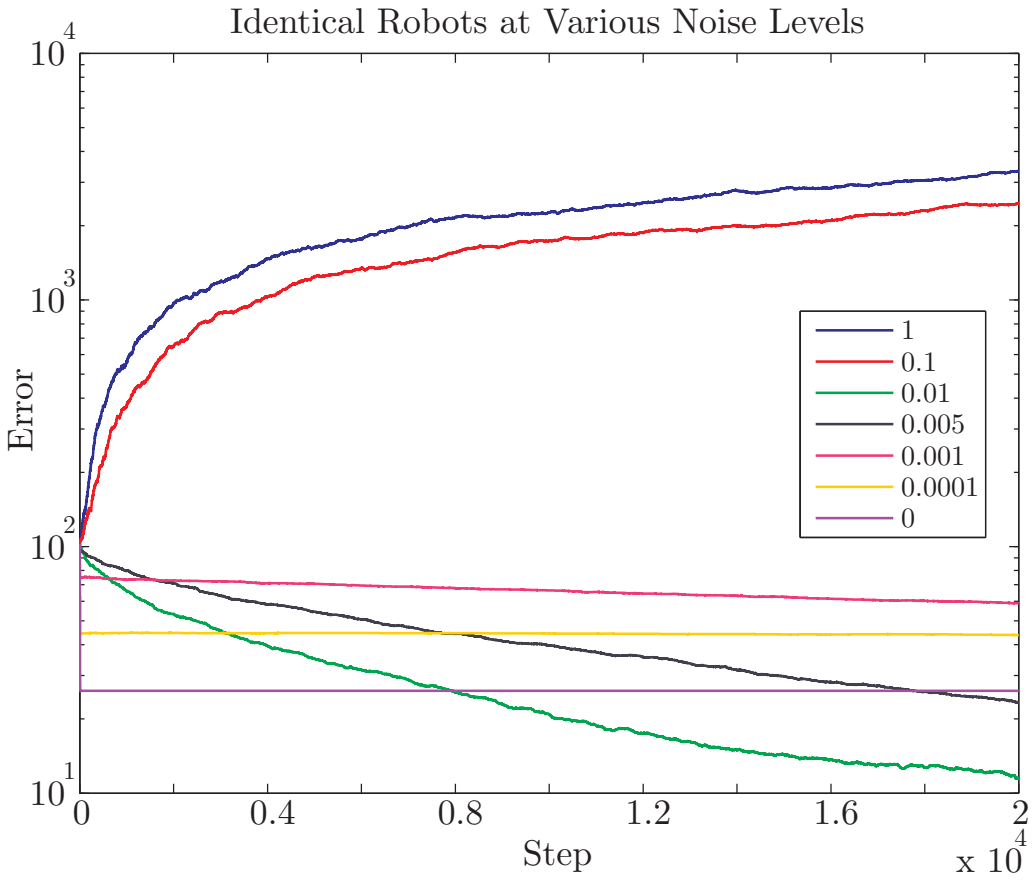


Figure 4.10: Error of a discrete-time, finite collection of 120 robots simulated under a standard noise model (4.20) with all ϵ set to 1. $\alpha = 0.01$ provides the best convergence.

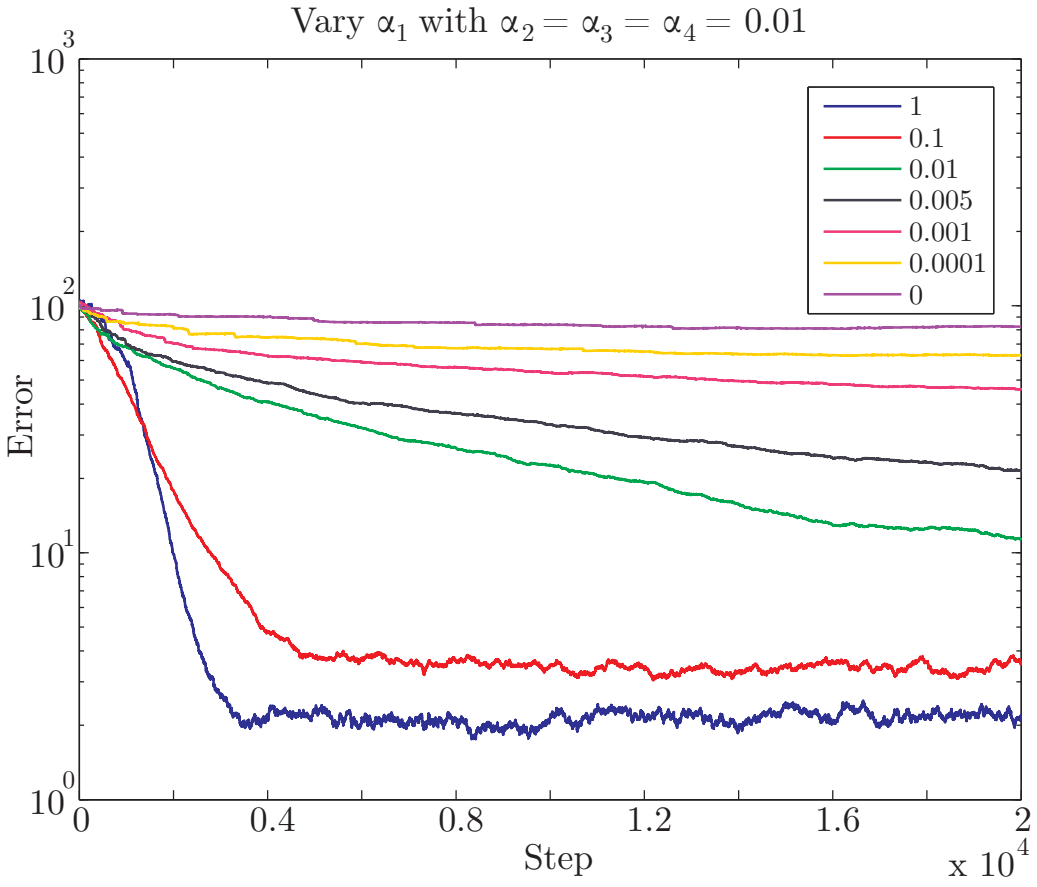


Figure 4.11: Error of a discrete-time, finite collection of 120 robots simulated under a standard noise model (4.20) with all ϵ set to 1. Focusing the noise in the rotation (α_1) improves the convergence for identical robots.

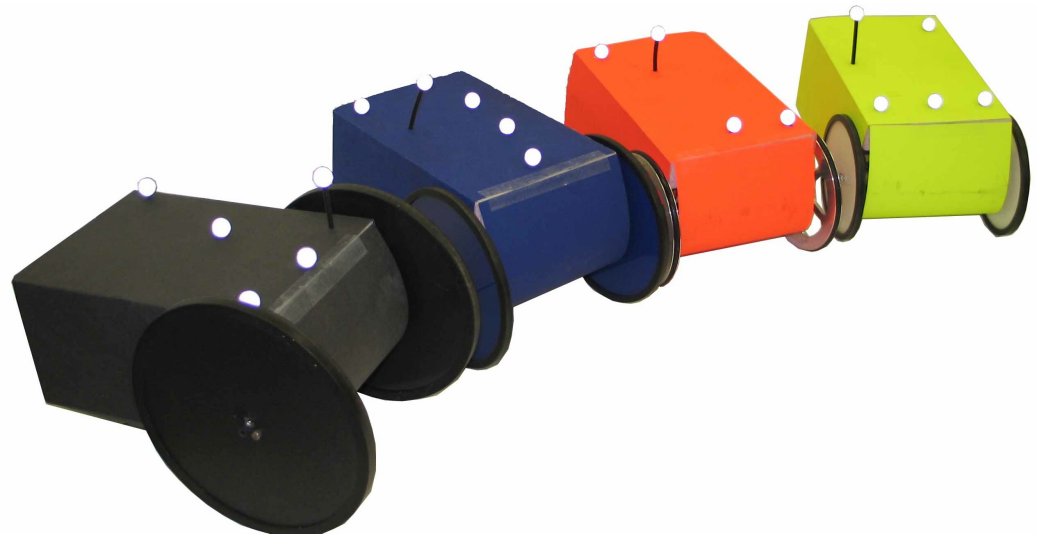


Figure 4.12: Four differential-drive robots with wheel diameters in the set $\{102, 108, 127, 152\}$ mm. Each robot receives the same broadcast control signal, but the different wheel sizes scale the commanded linear and angular velocities. Robots courtesy of College of Engineering Control Systems Laboratory [100].

4.5 Hardware Experiments

Here, we describe our hardware system and explain our experimental procedures and results.

4.5.1 Differential-Drive Robots

Our differential robots have two large direct-drive wheels in the back, and a free-wheeling ball caster in the front, as shown in Fig. 4.12. In the experiments shown in this chapter, we use wheels with diameters in the set $\{102, 108, 127, 152\}$ mm. Trials with identical-size wheels all used 102 mm wheels, as shown in Fig. 4.1.

4.5.2 System Overview

A block diagram of our system in Fig. 4.13 shows the relevant hardware. The robots are commanded to either move linearly or turn in place in units of encoder ticks. These commands are broadcast over 900 MHz radio using an AeroComm 4490 card.

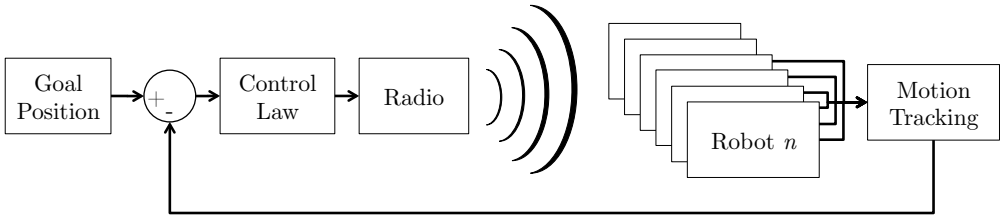


Figure 4.13: Block diagram for steering a multi-robot system with a broadcast uniform control input.

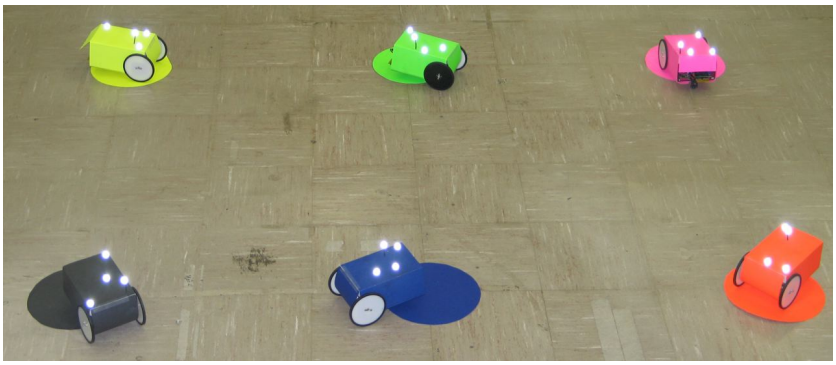


Figure 4.14: Hardware experiment setup showing six differential-drive robots at their target locations.

Four or five tracking dots are fixed to the top of each robot. These dots can be seen in Fig. 4.14. Position and orientation data for each vehicle are uniquely measured by an 18-camera NaturalPoint OptiTrack system with reported sub-millimeter accuracy. A MATLAB program computes the control policy (4.16) and sends the broadcast uniform control input.

4.5.3 Online Calibration

Calibration is not necessary for successful implementation of the controller, but it improves performance. In our hardware experiments, for every translation command $u(k)$, we record beginning and ending positions to calculate d_i, the distance traveled, and update each ϵ_i value according to the following rule:

$$\epsilon_i(k+1) = \epsilon_i(k) + K\frac{|u(k)|}{M}\left(\frac{d_i}{|u(k)|} - \epsilon_i(k)\right).$$

K is the weighting we give new measurements of ϵ, and M is the maximum possible distance we may command the robot to move. For the experiments shown here $K = 0.1$ and $M = 0.7$.

4.5.4 Results

We conducted a series of experiments to show that our control policy converges in a real system. Fig. 4.15 shows frames from a video of two of these experiments. We show results for unique wheel sizes with online calibration, for unique wheel sizes without online calibration, and for identical wheels.

Unique Wheel Sizes with Online Calibration: Initially, each robot was assumed to have $\epsilon = 1$, and the actual values of ϵ were learned through online calibration. The robots were successfully commanded from a horizontal line, to a box formation, to a vertical line, and finally to a tight box formation. The results in Fig. 4.16 show convergence both in position and in ϵ values. Online calibration requires persistent excitation, so convergence slows as the robots approach their targets.

Unique Wheel Sizes without Calibration: Surprisingly, it is not necessary to know or to learn the ϵ values. For this entire experiment ϵ was set to 1. Four robots were successfully commanded from a horizontal line to a box formation, and then to a vertical line. For each formation the summed error converged to less than half a meter, as shown in Fig. 4.17.

Identical Wheel Sizes: Even with identical ϵ values, a collection of robots is still controllable due to process noise. Fig. 4.18 shows successful convergence results of four robots with identical wheel sizes commanded to the same formations as the previous experiment.

4.5.5 Applications Enabled by Position Control

The ability to control position enables many tasks. In a video at `http://www.youtube.com/watch?v=z-t1rl0C4ic`, we demonstrate robot rendezvous using six robots with identical-sized wheels. Robot rendezvous robustly collects all the robots to one position; this primitive operation could be useful for

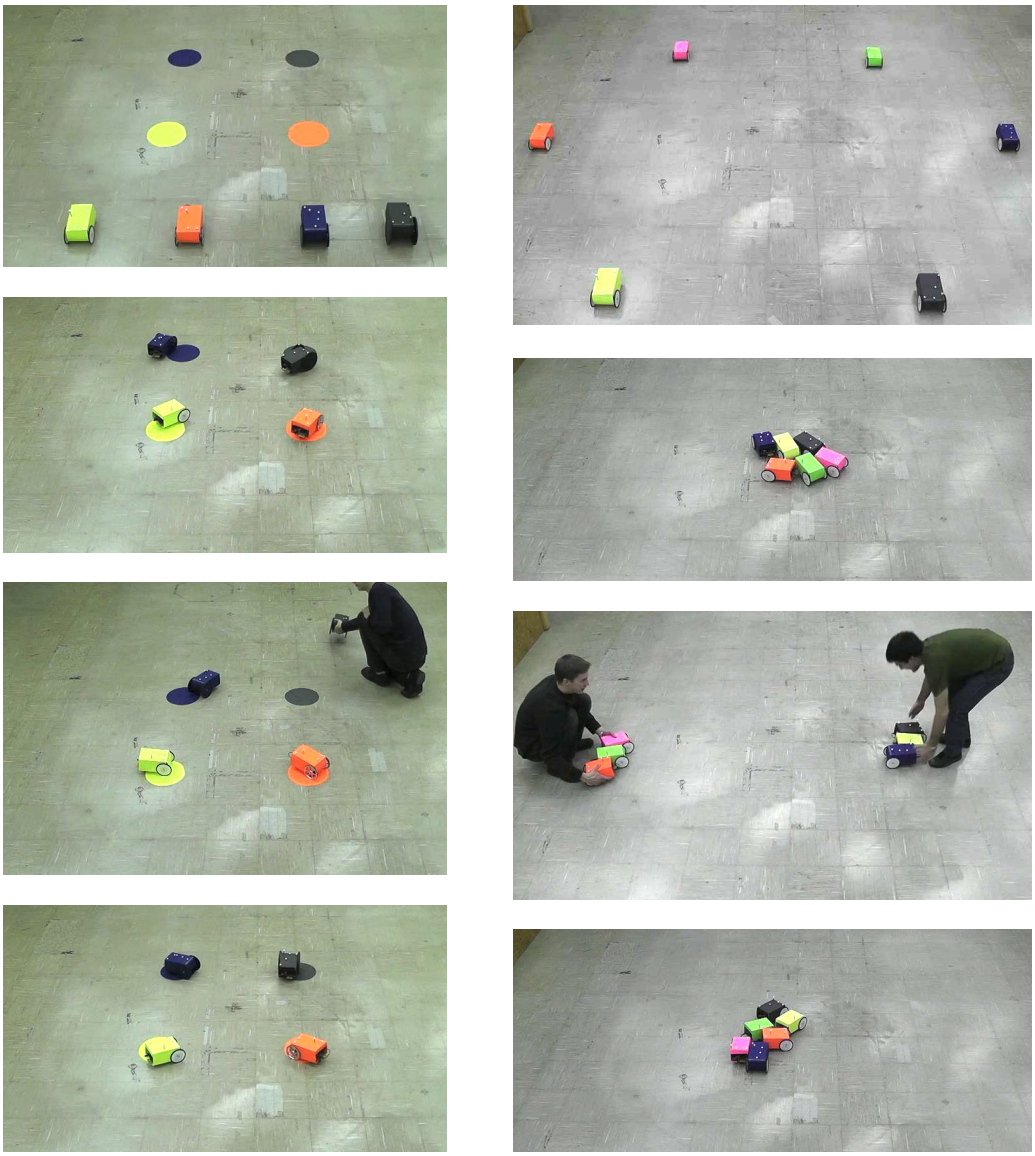


Figure 4.15: Photographs of hardware experiments steering four differential-drive robots with different wheel sizes (left column) and six differential-drive robots with identical wheel sizes (right column). The robots are initialized in a straight line and all receive the same control input from a wireless signal. A motion capture system is used for feedback to steer the four robots to the colored targets and the six robots to rendezvous. In the third frame a disturbance is injected by moving a single robot away from its target (left) and by splitting the ensemble (right). (Video online: http://www.youtube.com/watch?v=50gb5WMqJbY)

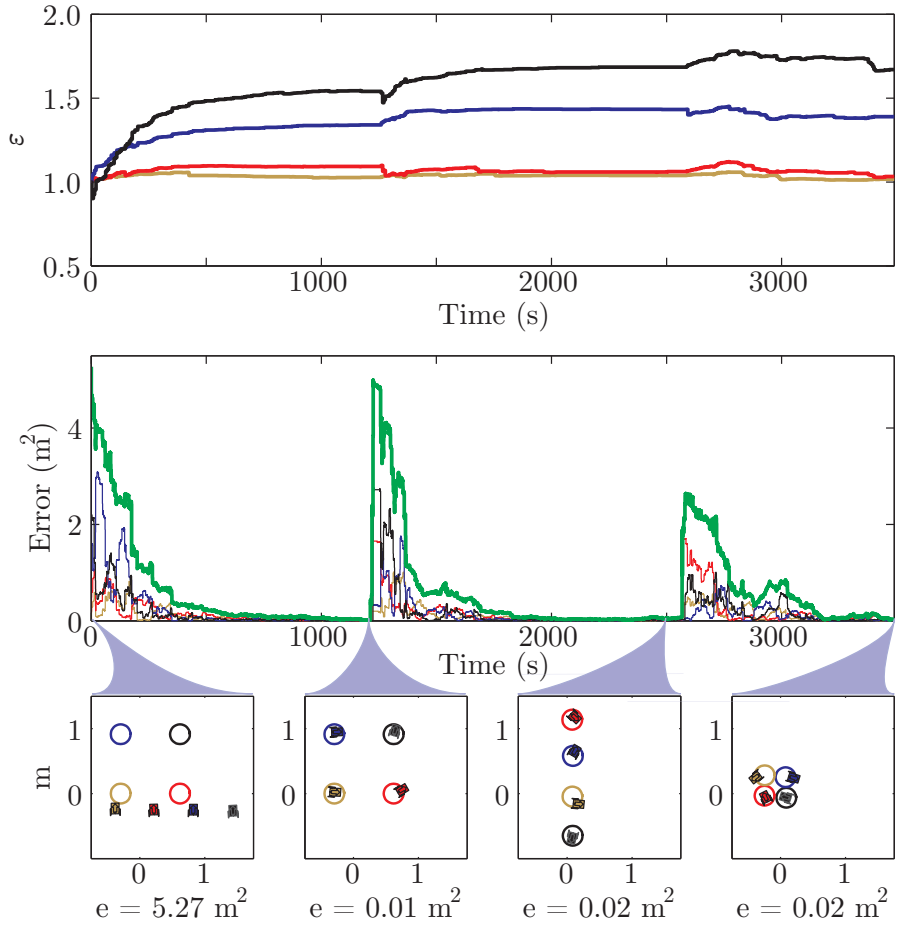


Figure 4.16: Hardware experiment with unique wheel sizes and online calibration. The top plot shows the convergence of ϵ values estimated by online calibration. The bottom plot shows the summed distance error as the robots were successfully steered through the sequence of formations shown.

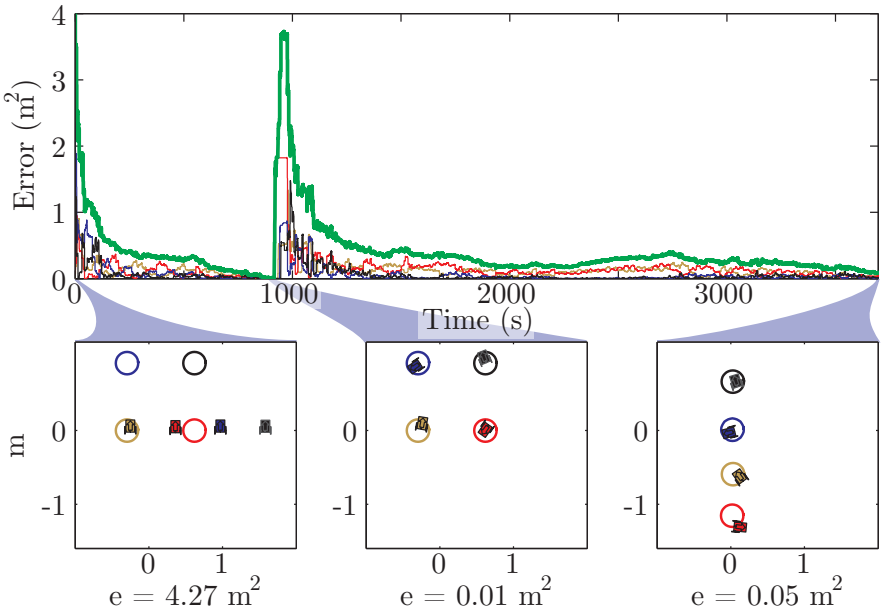


Figure 4.17: Hardware experiment with unique wheel sizes and no calibration. The plot shows the summed distance error as the robots were successfully steered through the sequence of formations shown.

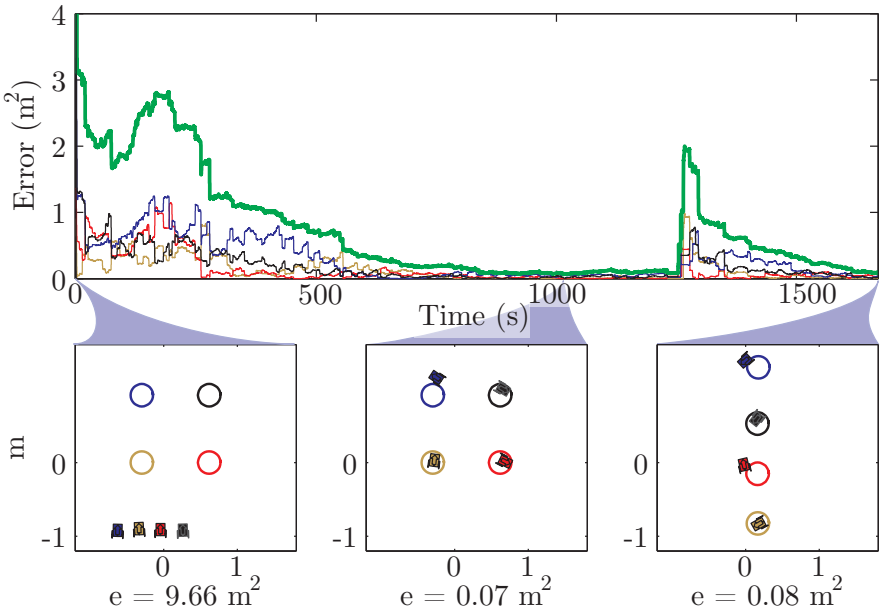


Figure 4.18: Hardware experiment with identical wheel sizes. The plot shows the summed distance error as the robots were successfully steered through the sequence of formations shown.

alignment of micro- and nano-robots. To achieve robot rendezvous, the goal position of each robot is set to the mean position of the ensemble.

Dispersion is the opposite of gathering. To achieve dispersion, the goal position of each robot is set to the mean of the ensemble, but the control policy is set to $u_1(t) = F(t)$, which will repel robots from each other. Dispersion may be useful for distributing micro- and nano-robots over a substrate.

Other tasks include forming subgroups, path- and trajectory-following, and pursuit/avoidance. Each can be implemented by a suitable selection of time-varying target locations in (4.13). A simulation of trajectory-following is shown in Fig. 4.19.

In a similar manner we can incorporate collision and obstacle avoidance by adding a potential field to the control policy (4.9) as in [101, Chap. 4]. A sample run of a simulation using potential fields to avoid two obstacles is shown in Fig. 4.20.

Micro/nano-manipulation and assembly are the focus of considerable micro and nano-robotics research. Sitti, Yu, and Cecil provide surveys of nano-manipulation and assembly [102–104]. Savia and Koivo focus on a survey of contact strategies for micro manipulation [105]. Micro/nano-manipulation refers to manipulating objects at the micro- and nano-scale, while assembly describes building structures from smaller components. The ability to control position and track trajectories enables an ensemble of micro- and nano-robots to be used as a manipulator. One advantage of using an ensemble control method over other methods, e.g. an atomic force microscope tip, is that ensemble methods allows the simultaneous manipulation of multiple objects. Fig. 4.21 shows frames from a simulation where six robots assemble a structure from three smaller components. The simulated robots receive exactly the same force and torque input commands, but each robot scales these commands by a unique constant. These simulations were performed using Box2D, version 2.0.1 (available: `http://box2d.org/`).

4.6 Conclusion

In this chapter we investigated ensembles of unicycles that share a uniform control input. Through Lyapunov analysis, we derived a globally asymptotic stabilizing controller for a continuous-time, infinite ensemble. We extended

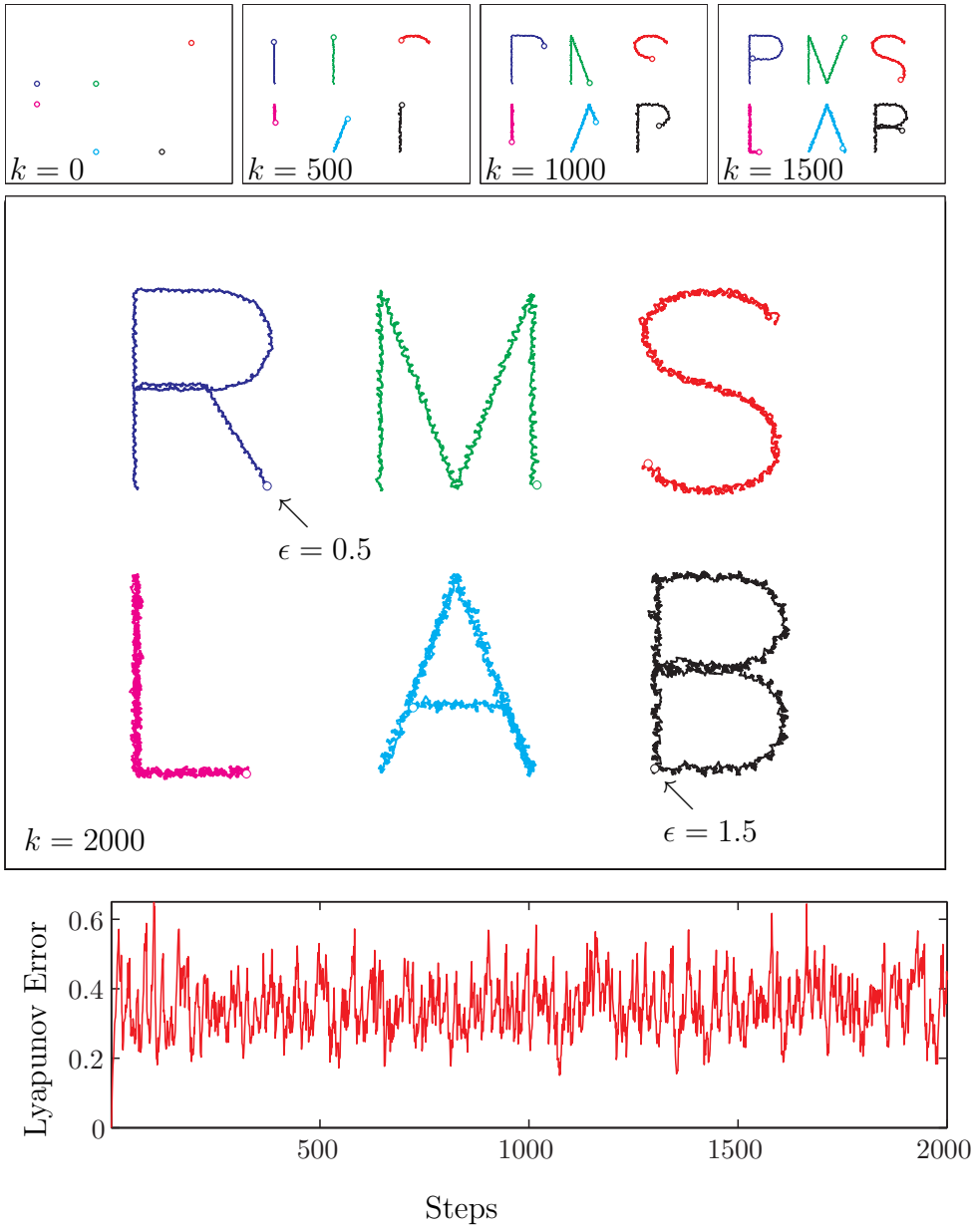


Figure 4.19: Simulation of trajectory-following. Six differential-drive robots with wheel sizes ranging from 0.5 to 1.5 of nominal are steered with a common control signal to follow trajectories that spell out 'RMSLAB'. The top left robot (blue) has the smallest wheels while the lower right robot (**black**) has the largest wheels. The bottom plot shows that the Lyapunov function stabilizes around 0.37.
(Video online: http://www.youtube.com/watch?v=z-t1rl0C4ic)

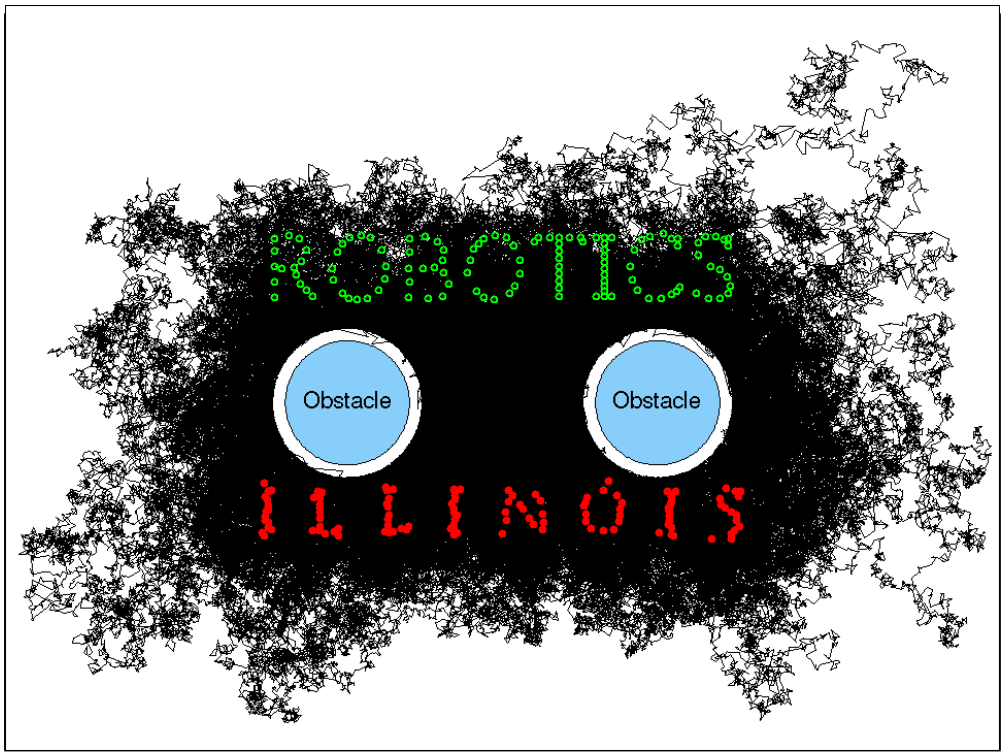


Figure 4.20: Simulation using potential fields for obstacle avoidance. 120 robots with varying wheels sizes $\epsilon \in [0.5, 1.5]$ are steered from the start position in green to the goal position spelling the word "ILLINOIS". The current robot position is drawn in red, and the path from start to goal for each robot is drawn in black.
(Video online: http://www.youtube.com/watch?v=5W8Sc2FwhxM)

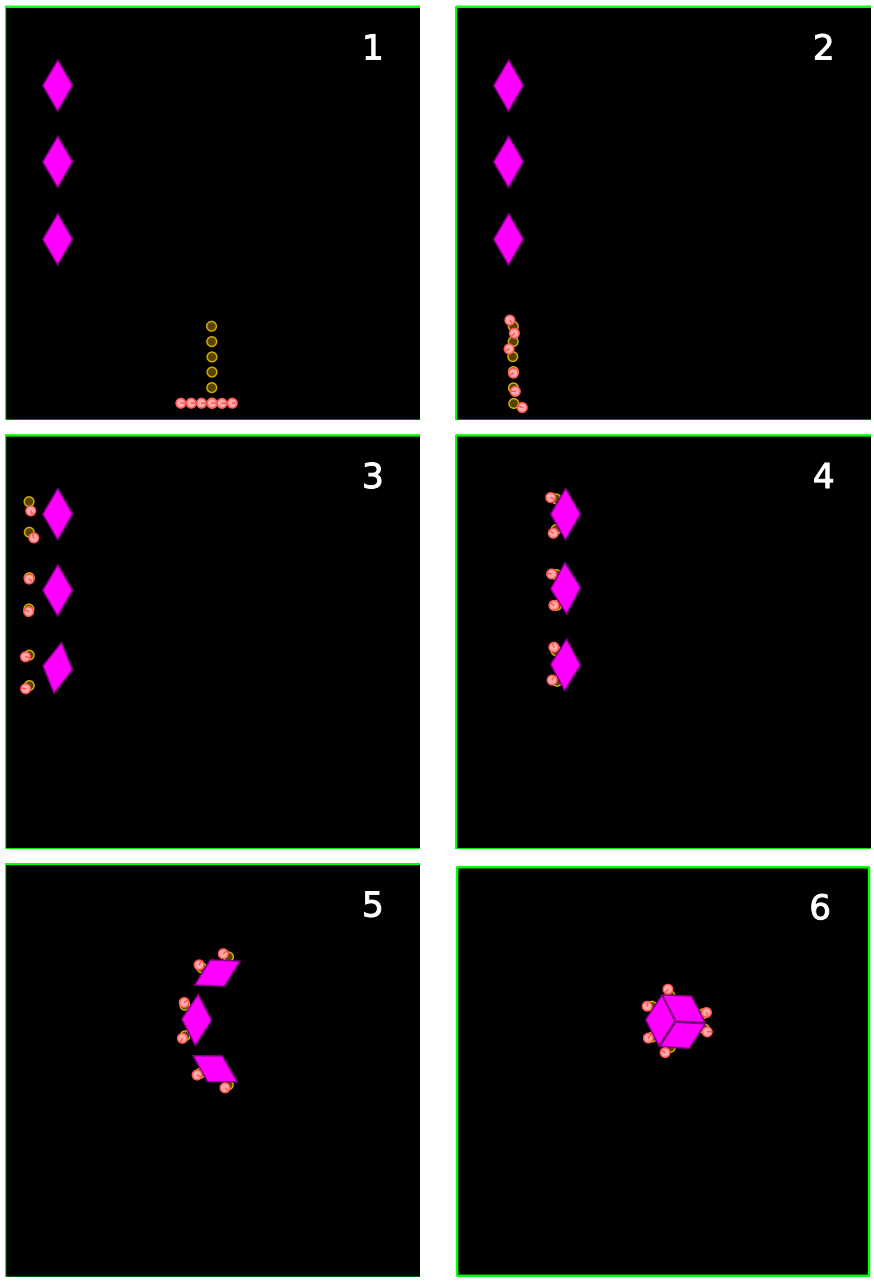


Figure 4.21: Image frames from an assembly and manipulation simulation. Six differential-drive robots are drawn with pink circles and they simultaneously manipulate three diamond-shaped objects. The goal trajectories are represented with yellow circles. In 5,000 steps the robots approach the objects (frames 1–3), push them into position (4-5), and then orient them to assemble the final object (6).
(Video online: http://www.youtube.com/watch?v=VeZF6aT-hog)

this controller to finite collections of unicycles in continuous and discrete time. In simulation, we showed that a discrete-time, finite ensemble of unicycles converges asymptotically and rejects disturbances from a standard noise model. In hardware experiments, we demonstrated online calibration which learned the unknown parameter for each robot. These experiments led to surprising results that *(1)* our controller still works when all wheel sizes are wrong and *(2)* in the presence of noise, our controller works even when all wheel sizes are the same.

This work shows that a collection of unicycles with uniform inputs to all robots can be regulated to arbitrary positions, reject disturbances from a standard noise model, and converge to goals with global asymptotic stability. This work may be particularly relevant to systems of micro- and nano-robots, which are often constrained to uniform inputs.

Chapter 5

Future Work

Figure 5.1: In this dissertation we applied the framework of ensemble control theory to control the three robotic systems shown above.

This dissertation applied the framework of ensemble control theory to derive approximate steering algorithms for two classical robotic systems under bounded model perturbation. We applied this same framework for feedback control of a multi-robot system where every robot receives exactly the same control input. We validated our approach on each system with the hardware experiments shown in Fig. 5.1.

This research provides many promising avenues for future work. Outlined below are opportunities related to control of many differential-drive robots, control of many balls with a plate-ball manipulator, and feedback control of canonical ensemble control systems.

Feedback Control of Many Differential-Drive Robots

We demonstrated position control and trajectory following with this multi-robot system. The planar control law we provided could be extended to three dimensions for swimming and flying robots with the addition of a second input for rotation. Another useful approach may be to treat each robot

as a finger for manipulating objects in a planar workspace. These fingers could be used for object alignment, transport, form closure, and assembly. Although we showed that the orientation of each robot in the ensemble is not controllable, this limitation does not apply to the objects being manipulated. Provided the object to be manipulated is larger than the individual robots, the ensemble can apply arbitrary forces and torques to steer objects along trajectories in $SE(2)$. Possible applications include targeted therapy, material removal, assembly, and telemetry for remote sensing.

Moreover, the final orientation of each robot is easy to compute. By assigning desired ending positions as a function of the final orientation, the ensemble can be steered to control the final orientation of the robots relative to a fixed object. This could enable the robots to surround a target and deliver material or to construct a boundary.

We proved that a finite ensemble of differential-drive robots is globally asymptotically stable. Our simulations indicate that it may be possible to prove the stronger result that our feedback policy provides exponential convergence of a finite ensemble.

Finally, the requirements for our control policy are modest. We showed that the policy applies even to robots with non-negative linear and angular velocities and minimum turning radii. This work may be particularly relevant to systems of micro- and nano-robots, which are often constrained to uniform inputs. Applying this policy to a micro- or nano-robotic system would demonstrate if this approach is practical. Additionally, our analysis of behavior under uncertainty provided insight on which tolerances are helpful for control and which are detrimental. This may prove helpful for the engineers who design these robots.

Feedback Control of Many Spheres with a Plate-Ball Manipulator

We designed piecewise-constant input trajectories to move a sphere between start and goal configurations in $\mathbb{R}^2 \times SO(3)$ despite an unknown sphere radius. This is a case-study for robotic-manipulation.

On a different scale, the results in this dissertation may find application in a manufacturing environment where an industrial robot manipulates many parts in parallel. In particular, the control-Lyapunov approach in Chapter 4 for differential-drive robots could be applied to manipulating many spheres.

Preliminary results with spheres painted with one hemisphere black and the other hemisphere white indicate similar convergence properties as in Chapter 4.

Finally, the optimized paths for the plate-ball problem in Chapter 3 only searched a limited space of n interleaved x- and y-axis rotations. This search returned a local minimum. The optimal shortest-length paths for a single ball are in general smooth curves. Future work could apply optimal control tools (perhaps pseudospectral optimization [67]) to construct better paths with guarantees of optimality.

Opportunities in Ensemble Control

For the purposes of this dissertation we restricted ourselves to three robotic systems. There are many other classical systems that can be shown to be ensemble controllable, e.g. a mass-spring-damper system with unknown parameters, planar bodies with thrusters with unknown thruster power/position, and the general problem of rolling bodies in contact.

Lastly, the feedback control law presented in this dissertation has a simple form that is easy to compute. It may be possible to apply similar feedback control techniques to ensemble control problems outside of robotics.

References

[1] J.-P. Laumond, "Finding collision-free smooth trajectories for a non-holonomic mobile robot," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 1987, pp. 1120–1123.

[2] R. W. Brockett and L. Dai, *Nonholonomic Motion Planning*. Springer, 1993, ch. 1. Non-holonomic kinematics and the role of elliptic functions in constructive controllability, pp. 1–19.

[3] R. W. Brockett and N. Khaneja, "On the control of quantum ensembles," in *System Theory: Modeling, Analysis and Control*, T. Djaferis and I. Schick, Eds. Kluwer Academic Publishers, 1999.

[4] N. Khaneja, "Geometric control in classical and quantum systems," Ph.D. dissertation, Harvard University, 2000.

[5] J.-S. Li and N. Khaneja, "Ensemble controllability of the bloch equations," in *IEEE Conf. Dec. Cont.*, San Diego, CA, Dec. 2006, pp. 2483–2487.

[6] J.-S. Li, "Control of inhomogeneous ensembles," Ph.D. dissertation, Harvard University, May 2006.

[7] J.-S. Li and N. Khaneja, "Control of inhomogeneous quantum ensembles," *Physical Review A (Atomic, Molecular, and Optical Physics)*, vol. 73, no. 3, p. 030302, 2006.

[8] J.-S. Li and N. Khaneja, "Ensemble control of linear systems," in *IEEE Conf. Dec. Cont.*, New Orleans, LA, USA, Dec. 2007, pp. 3768–3773.

[9] J.-S. Li and N. Khaneja, "Ensemble control of bloch equations," *IEEE Trans. Autom. Control*, vol. 54, no. 3, pp. 528–536, Mar. 2009.

[10] J.-S. Li, "Ensemble control of finite-dimensional time-varying linear systems," *IEEE Trans. Autom. Control*, vol. 56, no. 2, pp. 345–357, Feb. 2011.

[11] J. Ruths, "Optimal control of inhomogeneous ensembles," Ph.D. dissertation, Washington University in St. Louis, St. Louis, Missouri, United States, June 2011.

[12] J. Ruths and J.-S. Li, "A multidimensional pseudospectral method for optimal control of quantum ensembles," *The Journal of Chemical Physics*, vol. 134, no. 4, pp. 044 128–8, 01 2011.

[13] K. Beauchard, J.-M. Coron, and P. Rouchon, "Controllability issues for continuous-spectrum systems and ensemble controllability of Bloch equations," *Comm. Math. Phys.*, vol. 296, no. 2, pp. 525–557, June 2010.

[14] K. Beauchard, J.-M. Coron, and P. Rouchon, "Time-periodic feedback stabilization for an ensemble of half-spin systems," in *IFAC Sym. Nonlin. Cont. Sys.*, Bologna: Italy, Sep. 2010.

[15] R. M. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[16] S. V. D. Meer, "Stochastic cooling and the accumulation of antiprotons," in *Nobel Lectures in Physics*, T. Frängsmyr, Ed. Singapore: World Scientific, 1993.

[17] N. Michael and V. Kumar, "Controlling shapes of ensembles of robots of finite size with nonholonomic constraints," in *Robotics: Science and Systems*, Zurich, Switzerland, 2008.

[18] L. U. Odhner and H. Asada, "Stochastic recruitment control of large ensemble systems with limited feedback," *Journal of Dynamic Systems, Measurement, and Control*, vol. 132, no. 4, p. 041008, 2010.

[19] P. Rouchon, "Motion planning, equivalence, infinite dimensional systems," *Int. J. Appl. Math. Comput. Sci.*, vol. 11, no. 1, pp. 165–188, 2001.

[20] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Mobile robots," in *Robotics: Modelling, Planning and Control*, 2nd ed., ser. Advanced Textbooks in Control and Signal Processing. Springer, 2009, ch. 11.

[21] P. Lucibello and G. Oriolo, "Robust stabilization via iterative state steering with an application to chained-form systems," *Automatica*, vol. 37, pp. 71–79, 2001.

[22] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, Sep. 2005.

[23] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed., ser. Intelligent robotics and autonomous agents series. Cambridge, MA: MIT Press, 2011.

[24] J. Borenstein and L. Feng, "UMBmark: a benchmark test for measuring odometry errors in mobile robots," in *Proceedings SPIE 2591, 113*, 1995, pp. 113–124.

[25] G. Antonelli and S. Chiaverini, "Linear estimation of the physical odometric parameters for differential-drive mobile robots," *Autonomous Robots*, vol. 23, no. 1, pp. 59–68, July 2007.

[26] A. Martinelli, N. Tomatis, and R. Siegwart, "Simultaneous localization and odometry self calibration for mobile robot," *Autonomous Robots*, vol. 22, no. 1, pp. 75–85, Jan. 2007.

[27] H. Date, M. Sampei, M. Ishikawa, and M. Koga, "Simultaneous control of position and orientation for ball-plate manipulation problem based on time-state control form," *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 465–480, June 2004.

[28] T. Das and R. Mukherjee, "Exponential stabilization of the rolling sphere," *Automatica*, vol. 40, no. 11, pp. 1877–1889, 11 2004.

[29] P. Morin and C. Samson, "Stabilization of trajectories for systems on lie groups. application to the rolling sphere," in *Int. Fed. of Auto. Cont.*, Seoul Korea, July 2008, pp. 508–513.

[30] D. Casagrande, A. Astolfi, and T. Parisini, "Switching-driving Lyapunov function and the stabilization of the ball-and-plate system," *Automatic Control, IEEE Transactions on*, vol. 54, no. 8, pp. 1881–1886, Aug 2009.

[31] G. Oriolo, M. Vendittelli, A. Marigo, and A. Bicchi, "From nominal to robust planning: the plate-ball manipulation system," in *IEEE Int. Conf. Rob. Aut.*, vol. 3, Sep. 2003, pp. 3175–3180.

[32] G. Oriolo and M. Vendittelli, "A framework for the stabilization of general nonholonomic systems with an application to the plate-ball mechanism," *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 162–175, Apr. 2005.

[33] G. E. Dullerud and F. G. Paganini, *A Course in Robust Control Theory: a Convex Approach*. New York: Springer, 2000, vol. 36.

[34] N. C. Singer and W. P. Seering, "Preshaping command inputs to reduce system vibration," *Journal of Dynamic Systems, Measurement, and Control*, vol. 112, no. 1, pp. 76–82, 1990.

[35] M. Fischer and M. L. Psiaki, "Robustified control design by an approximate min-max technique," in *AIAA Guidance, Navigation and Control Conference*, Monterey, CA; USA, Aug 1993, pp. 949–957.

[36] S. Li, "A new perspective on control of uncertain complex systems," in *IEEE Conf. Dec. Cont.*, Dec. 2009, pp. 708–713.

[37] M. T. Mason, *Mechanics of Robotic Manipulation*. MIT Press, 2001.

[38] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[39] M. Erdmann, "Using backprojections for fine motion planning with uncertainty," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 19–45, 1986.

[40] R. Alterovitz, T. Siméon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty," in *Robotics: Science and Systems III*, 2008, pp. 233–241.

[41] M. Erdmann and M. Mason, "An exploration of sensorless manipulation," *IEEE J. Robot. Autom.*, vol. 4, no. 4, pp. 369–379, Aug. 1988.

[42] K. Y. Goldberg, "Orienting polygonal parts without sensors," *Algorithmica*, vol. 10, no. 2, pp. 201–225, 1993.

[43] J. F. Canny and K. Y. Goldberg, "industrial robotics: recent results and open problems," in *IEEE Int. Conf. Rob. Aut.*, vol. 3, May 1994, pp. 1951–1958.

[44] K. M. Lynch, M. Northrop, and P. Pan, "Stable limit sets in a dynamic parts feeder," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 608–615, Aug. 2002.

[45] T. D. Murphey, J. Bernheisel, D. Choi, and K. M. Lynch, "An example of parts handling and self-assembly using stable limit sets," in *Int. Conf. Int. Rob. Sys.*, Aug. 2005, pp. 1624–1629.

[46] O. C. Goemans, K. Goldberg, and A. F. van der Stappen, "Blades: a new class of geometric primitives for feeding 3d parts on vibratory tracks," in *Int. Conf. Rob. Aut.*, May 2006, pp. 1730–1736.

[47] T. H. Vose, P. Umbanhowar, and K. M. Lynch, "Friction-induced velocity fields for point parts sliding on a rigid oscillated plate," in *Robotics: Science and Systems*, Zurich, Switzerland, June 2008.

[48] S. Akella, W. H. Huang, K. M. Lynch, and M. T. Mason, "Parts feeding on a conveyor with a one joint robot," *Algorithmica*, vol. 26, no. 3, pp. 313–344, 2000.

[49] A. van der Stappen, R.-P. Berretty, K. Goldberg, and M. Overmars, "Geometry and part feeding," *Sensor Based Intelligent Robots*, pp. 259–281, 2002.

[50] K. Goldberg, B. V. Mirtich, Y. Zhuang, J. Craig, B. R. Carlisle, and J. Canny, "Part pose statistics: estimators and experiments," *IEEE Trans. Robot. Autom.*, vol. 15, no. 5, pp. 849–857, Oct. 1999.

[51] M. Moll and M. Erdmann, "Manipulation of pose distributions," *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 277–292, 2002.

[52] S. Akella and M. T. Mason, "Using partial sensor information to orient parts," *The International Journal of Robotics Research*, vol. 18, no. 10, pp. 963–997, 1999.

[53] T. D. Murphey and J. W. Burdick, "Feedback control methods for distributed manipulation systems that involve mechanical contacts," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 763–781, 2004.

[54] H. J. Sussmann and V. Jurdjevic, "Controllability of nonlinear systems," *J. of Diff. Eqn.*, vol. 12, no. 1, pp. 95–116, July 1972.

[55] G. Lafferriere and H. J. Sussmann, "A differential geometric approach to motion planning," in *Nonholonomic Motion Planning*. Kluwer, 1993, pp. 235–270.

[56] A. Becker and T. Bretl, "Approximate steering of a unicycle under bounded model perturbation using ensemble control," *IEEE Trans. Robot.*, vol. 28, no. 3, pp. 580–591, June 2012.

[57] A. Becker and T. Bretl, "Motion planning under bounded uncertainty using ensemble control," in *Robotics: Science and Systems (RSS)*, February 2010. [Online]. Available: http://www.roboticsproceedings.org/rss06/p38.pdf

[58] H. L. Royden, *Real Analysis*, 3rd ed. New Jersey: Prentice Hall, 1988.

[59] J.-P. Laumond, P. Jacobs, M. Taix, and R. Murray, "A motion planner for nonholonomic mobile robots," *IEEE Trans. Robot. Autom.*, vol. 10, no. 5, pp. 577–593, Oct. 1994.

[60] D. Block, "The segbot," June 2005, College of Engineering Control Systems Laboratory. [Online]. Available: http://coecsl.ece.illinois.edu/ge423/spring04/group9/index.htm

[61] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 869–880, Dec 1996.

[62] A. Becker and T. Bretl, "Approximate steering of a plate-ball system under bounded model perturbation using ensemble control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012 (under review).

[63] A. A. G. Requicha, "Toward a theory of geometric tolerancing," *The International Journal of Robotics Research*, vol. 2, no. 4, pp. 45–60, Dec. 1983.

[64] S. Akella and M. T. Mason, "Orienting toleranced polygonal parts," *The International Journal of Robotics Research*, vol. 19, no. 12, pp. 1147–1170, Dec. 2000.

[65] V. Jurdjevic, "The geometry of the plate-ball problem," *Archive for Rational Mechanics and Analysis*, vol. 124, pp. 305–328, 1993.

[66] D. Garg, M. Patterson, C. Francolin, C. Darby, G. Huntington, W. Hager, and A. Rao, "Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a radau pseudospectral method," *Computational Optimization and Applications*, vol. 49, no. 2, pp. 335–358, 2011-06-01.

[67] A. V. Rao, D. A. Benson, C. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, "Algorithm 902: GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method," *ACM Trans. Math. Softw.*, vol. 37, no. 2, pp. 1–39, 2010.

[68] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao, "Direct trajectory optimization and costate estimation via an orthogonal collocation method," *Journal of Guidance Control and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006.

[69] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A unified framework for the numerical solution of optimal control problems using pseudospectral methods," *Automatica*, vol. 46, no. 11, pp. 1843–1851, 11 2010.

[70] D. Garg, W. W. Hager, and A. V. Rao, "Pseudospectral methods for solving infinite-horizon optimal control problems," *Automatica*, vol. 47, no. 4, pp. 829–837, 4 2011.

[71] Z. Li and J. Canny, "Motion of two rigid bodies with rolling constraint," *IEEE Trans. Robot. Autom.*, vol. 6, no. 1, pp. 62–72, Feb. 1990.

[72] A. Marigo and A. Bicchi, "Rolling bodies with regular surface: controllability theory and applications," *IEEE Trans. Autom. Control*, vol. 45, no. 9, pp. 1586–1599, Sep. 2000.

[73] A. Becker, ""Iterative Feedback Control of Plate-Ball Orientation and Position." MATLAB Central File Exchange," April 2012. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/36252

[74] M. Svinin and S. Hosoe, "Planning of smooth motions for a ball-plate system with limited contact area," in *IEEE Int. Conf. Rob. Aut.*, May 2008, pp. 1193–1200.

[75] M. Svinin and S. Hosoe, "Motion planning algorithms for a rolling sphere with limited contact area," *IEEE Trans. Robot.*, vol. 24, no. 3, pp. 612–625, June 2008.

[76] A. Bicchi, R. Sorrentino, and C. Piaggio, "Dexterous manipulation through rolling," in *IEEE Int. Conf. Rob. Aut.*, vol. 1, May 1995, pp. 452–457.

[77] P. Choudhury and K. Lynch, "Rolling manipulation with a single control," in *IEEE Int. Conf. Cont. App.*, Sep. 2001, pp. 1089–1094.

[78] W. Ding, A. J. Howard, M. M. Peri, and C. Cetinkaya, "Rolling resistance moment of microspheres on surfaces: contact measurements," *Philosophical Magazine*, vol. 87, no. 36, pp. 5685–5696, Dec. 2007.

[79] R. R. Agayan, R. G. Smith, and R. Kopelman, "Slipping friction of an optically and magnetically manipulated microsphere rolling at a glass-water interface," *J. of App. Phys.*, vol. 104, no. 5, pp. 054 915–11, Sep. 2008.

[80] A. Becker, "Re-Orient a Sphere with Two Straight Rolls," June 2012, Wolfram Demonstrations Project. [Online]. Available: http://demonstrations.wolfram.com/ReOrientASphereWithTwoStraightRolls/

[81] B. J. Pryor, "Fourier synthesis methods for identification and control of ensembles," Ph.D. dissertation, Harvard University, 2007.

[82] B. Pryor and N. Khaneja, "Fourier methods for control of inhomogeneous quantum systems," in *IEEE Conf. Dec. Cont.*, Dec. 2007, pp. 6340–6345.

[83] B. Shapiro, "Sisyphus," sand trace kinetic sculpture, 1998.

[84] J.-P. Hébert, "Bessel function with stones," sand trace sculpture, 1999.

[85] K. M. Lynch, "Determining the orientation of a painted sphere from a single image: A graph coloring problem," unpublished, 2001. [Online]. Available: http://lims.mech.northwestern.edu/~lynch/papers/sphere/sphere.pdf

[86] R. Zimmermann, Y. Gasteuil, M. Bourgoin, R. Volk, A. Pumir, and J.-F. Pinton, "Tracking the dynamics of translation and absolute orientation of a sphere in a turbulent flow," *Rev. Scien. Instr.*, vol. 82, no. 3, pp. 033 903–03 313, Mar. 2011.

[87] D. Stein, E. Scheinerman, and G. Chirikjian, "Mathematical models of binary spherical-motion encoders," *IEEE/ASME Trans. Mechatronics*, vol. 8, no. 2, pp. 234–244, June 2003.

[88] D. Bradley and G. Roth, "Tracking a sphere with six degrees of freedom," Nat. Res. Council of CN, Ottawa, Ont., Tech. Rep. 47397, October 2004.

[89] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross, "High-quality single-shot capture of facial geometry," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 40:1–40:9, July 2010.

[90] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, no. 4, pp. 629–642, Apr. 1987.

[91] P.-T. Chiang, J. Mielke, J. Godoy, J. M. Guerrero, L. B. Alemany, C. J. Villagómez, A. Saywell, L. Grill, and J. M. Tour, "Toward a light-driven motorized nanocar: Synthesis and initial imaging of single molecules," *ACS Nano*, vol. 6, no. 1, pp. 592–597, Feb. 2011.

[92] B. Donald, C. Levey, and I. Paprotny, "Planar microassembly by parallel actuation of MEMS microrobots," *J. of MEMS*, vol. 17, no. 4, pp. 789–808, Aug. 2008.

[93] A. Becker, C. Onyuksel, and T. Bretl, "Feedback control of many differential-drive robots with uniform control inputs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012 (under review).

[94] Y. Shirai, A. J. Osgood, Y. Zhao, K. F. Kelly, and J. M. Tour, "Directional control in thermally driven single-molecule nanocars," *Nano Letters*, vol. 5, no. 11, pp. 2330–2334, Feb. 2005.

[95] B. Donald, C. Levey, C. McGray, D. Rus, and M. Sinclair, "Power delivery and locomotion of untethered microactuators," *J. Microelectromech. Syst.*, vol. 12, no. 6, pp. 947–959, Dec. 2003.

[96] B. Donald, C. Levey, C. McGray, I. Paprotny, and D. Rus, "An untethered, electrostatic, globally controllable MEMS micro-robot," *J. of MEMS*, vol. 15, no. 1, pp. 1–15, Feb. 2006.

[97] A. M. Lyapunov, translated and edited by A.T. Fuller, *The General Problem of the Stability of Motion.* London: Tayor & Francis, 1992.

[98] J. P. LaSalle, "Some extensions of Liapunov's second method," *IRE Transactions on Circuit Theory*, vol. CT, no. 7, pp. 520–527, Dec. 1960.

[99] W. S. Levine, Ed., *The Control Handbook*. United States of America: CRC Press, Inc., 1996, ch. 25.3 Discrete-Time Linear Time-Varying Systems, pp. 459–463.

[100] D. Block, "The segboy," June 2012, College Of Engineering Control Systems Laboratory. [Online]. Available: http://coecsl.ece.illinois.edu/projects.html

[101] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion*. The MIT Press, 2005.

[102] M. Sitti, "Survey of nanomanipulation systems," in *IEEE-NANO, IEEE Conference on Nanotechnology*, 2001, pp. 75–80.

[103] M.-F. Yu, "Fundamental studies of nanoscale sensing and actuation based on nanomanipulation and assembly," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2003, pp. 2365–2370.

[104] J. Cecil, D. Powell, and D. Vasquez, "Assembly and manipulation of micro devices—a state of the art survey," *Robotics and Computer-Integrated Manufacturing*, vol. 23, no. 5, pp. 580–588, Oct. 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0736584506000871

[105] M. Savia and H. Koivo, "Contact micromanipulation—survey of strategies," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 4, pp. 504–514, Aug. 2009.