CONTROLLING A SWARM OF SIMPLE ROBOTS USING GLOBAL INPUTS

A Dissertation

Presented to

the Faculty of the Department of Electrical and Computer Engineering

University of Houston

in Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in Electrical Engineering

by
Shiva Shahrokhi
August 2018

CONTROLLING A SWARM OF SIMPLE ROBOTS USING GLOBAL INPUTS

Shiva Shahrokhi

Approved:

Chair of the Committee
Aaron T. Becker, Assistant Professor
Department of Electrical and Computer Engineering

Committee Members:

Lydia Kavraki, Noah Harding Professor
Department of Computer Science, Rice University

Stanko Brankovic, Associate Professor
Department of Electrical and Computer Engineering

David Mayerich, Assistant Professor
Department of Electrical and Computer Engineering

Rose Faghih, Assistant Professor
Department of Electrical and Computer Engineering

Suresh K. Khator, Associate Dean,
Cullen College of Engineering

Badrinath Roysam, Professor and Chair,
Electrical and Computer Engineering

Acknowledgements

I thank God that I spent some of my best years during the Ph.D. program at the University of Houston. I grew up and became more capable. I would like to take this opportunity to express my gratitude to those who were part of this journey.

First, I would like to express my appreciation to my advisor, Dr. Aaron T. Becker. I truly appreciate his support, kindness, intellectual comments and caring for not only about my research but also many aspects of my life. Dr. Becker was a perfect match for me as an advisor. He is a nice person and full of energy and he makes the lab a happy and energetic place. I thank him for everything he has done for me.

I would like to thank my thesis committee members: Prof. Lydia Kavraki, Dr. David Mayerich, Dr. Rose Faghih, Dr. Stanko Brankovic and Dr. Nicaolas Tsekos for their insightful comments and suggestions and their kindness and supportiveness. I also enjoyed their classes and learned a lot from them.

I was very fortunate to be the first student of swarm control lab. I am thankful to my great collaborators: Lillian Lin, Arun Mahadev, Mable Wan, Chris Ertel, Jingang Shi, and Benedict Isichei. I have always benefited from our discussions. I also would like to thank my other fabulous lab mates: Mary Burbage, Haoran Zhao, Julien Leclerc, Li Huang, Sheryl Manzoor, Mohammad Sultan, An Nguyen, Shriya Bhatnagar, Daniel Bao, Srikanth K.V.S, Parth Joshi, Javier Garcia, Steban Soto, Victor Montano, and Jarret Lonsford. I enjoyed being with all of them.

I would also like to express my sincere appreciation to my parents, Nasrin and Hadi. Their belief in me made me brave enough to go for my dreams. I appreciate them for my whole life. I miss my father who was a great role model for me so much and hope he rests in peace. I believe he is looking at me and hope that I have made him proud.

I am very fortunate to have my mother and deeply appreciate her encouragement and self-sacrifice for me. I also like to thank my brothers and my sister, Ali, Amir and Shima, who always backed me up and encouraged me. I appreciate them all for being such a great family.

Finally, I would like to express my most gratitude and appreciation to the love of my life, my husband, Ali. I cannot imagine how I could finish my Ph.D. if he was not there to encourage and support me. Each dark day he was there to show me the tiny light at the end of the road. Each bright day he was there to make my smile bigger and make my heart warmer. He made the path possible for me, and never left me alone in the path. He is smart and hardworking and looking at him motivated me to work smarter and harder. I am forever grateful for each and every day of my life with him.

CONTROLLING A SWARM OF SIMPLE ROBOTS USING GLOBAL INPUTS

An Abstract

of a

Dissertation

Presented to

the Faculty of the Department of Electrical and Computer Engineering

University of Houston

in Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in Electrical Engineering

by
Shiva Shahrokhi
August 2018

Abstract

Microrobotics has the potential to revolutionize many applications including targeted material delivery, assembly, and surgery. The same properties that promise breakthrough solutions—small size and large populations—present unique challenges for controlling motion. When there are more particles than control inputs the system is underactuated and requires new control techniques. Rather than focusing on a specific microrobotic system, this dissertation designs control laws and algorithms for steering many particles controlled by global fields.

First, we identify key parameters for particle manipulation by using a collection of online games where players steer swarms of up to 500 particles to complete manipulation challenges. Inspired by techniques where human operators performed well, we investigate controllers that only use the mean and variance of the swarm. We next derive automatic controllers for these and a hysteresis-based switching control to regulate the first two moments of the particle distribution. Torque control is also necessary for manipulating objects as well as for aligning sensors, emitters, or redirecting an incoming signal.

Second, this dissertation proves that swarm torque control is possible, then presents algorithms to automate the task. Torque control enables us to control the position and orientation of an object.

Finally, this dissertation investigates particle control with uniform magnetic gradients (the same force is applied everywhere in the workspace). We provide position control algorithms that only require non-slip wall contact in 2D. The walls of in vivo and artificial environments often have surface roughness such that the particles do not move unless actuation pulls them away from the wall. We assume that particles in contact with the boundaries have zero velocity if the shared control input pushes the particle into the wall. All the results are validated with simulations and hardware implementations.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Controlling a swarm of particles with global inputs

Large populations of micro- and nanorobots are being produced in laboratories around the world, with diverse potential applications in drug delivery and construction, see [1–3]. These activities require robots that behave intelligently. Our vision is for large swarms of robots to be remotely guided 1) through the human body, to cure disease, heal tissue, and prevent infection shown in Fig. 1.1 [4] and 2) ex vivo to assemble structures in parallel. For each application, large numbers of micro robots are required to deliver sufficient payloads. Often particles are difficult or impossible to sense individually due to their size and location. For example, microrobots are smaller than the minimum resolution of a clinical MRI-scanner, see [5], however it is often possible to sense global properties of the group such as mean position and variance. Limited computation and communication at small scales also makes autonomous operation or direct control over individual robots difficult. Instead, this dissertation treats the robots that are steered by a global control signal broadcast to the entire population. The tiny robots themselves are often just rigid bodies, and it may be more accurate to define the robot as the *system* that consists of particles, a uniform control field, and sensing. Particle swarms propelled by a uniform field, where each particle receives the same control input, are common in applied mathematics, biology, and computer graphics [1–3]. Such systems are severely underactuated, having 2 degrees of freedom in the shared planar control input, but $2n$ degrees of freedom for the n-particle swarm. Figure 1.2 shows an example of simple particles when they all get the same control input. Techniques are needed that can handle this underactuation.

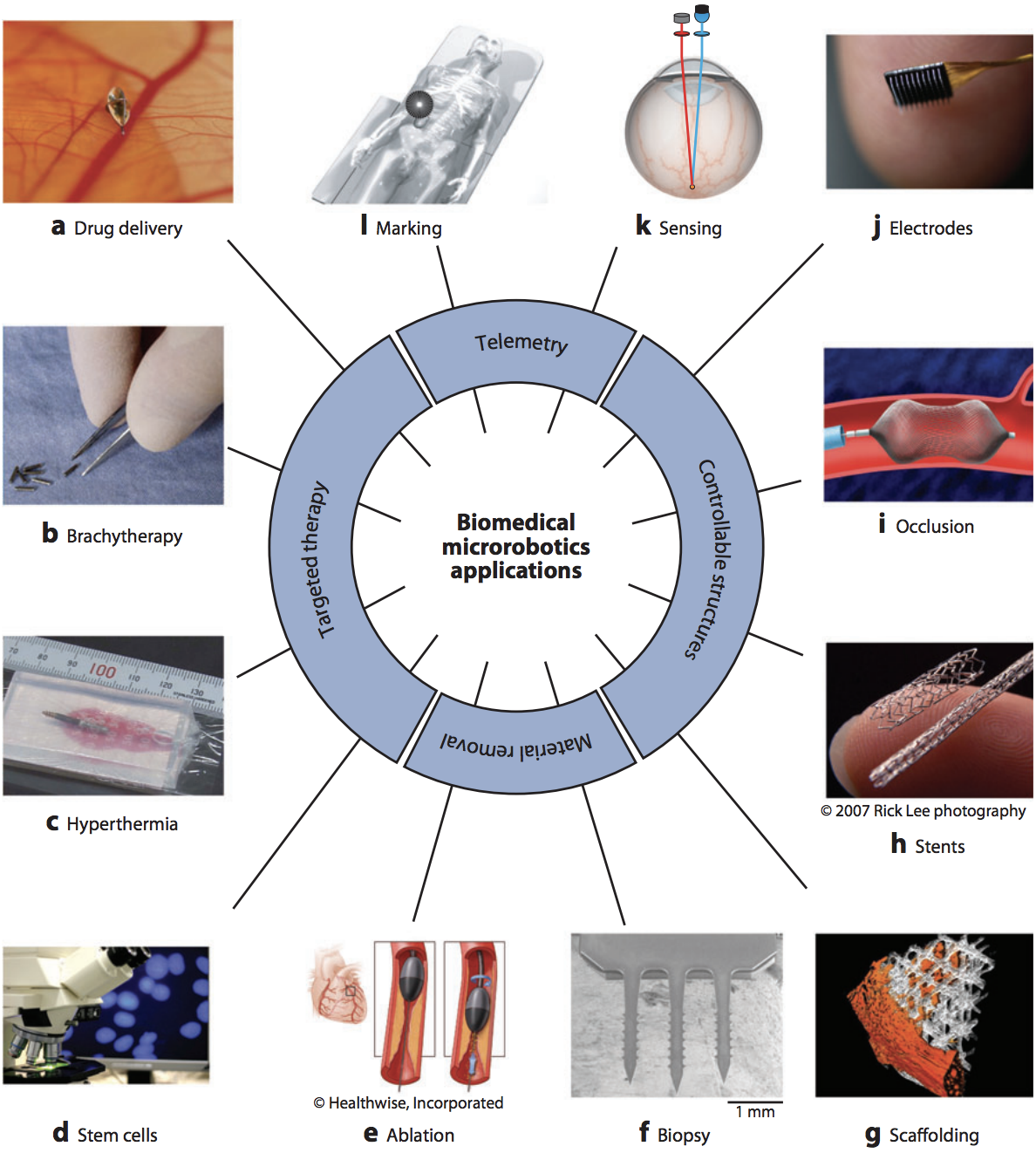


Figure 1.1: Microrobots has the potential to revolutionize medical applications. Used with permission from [4].

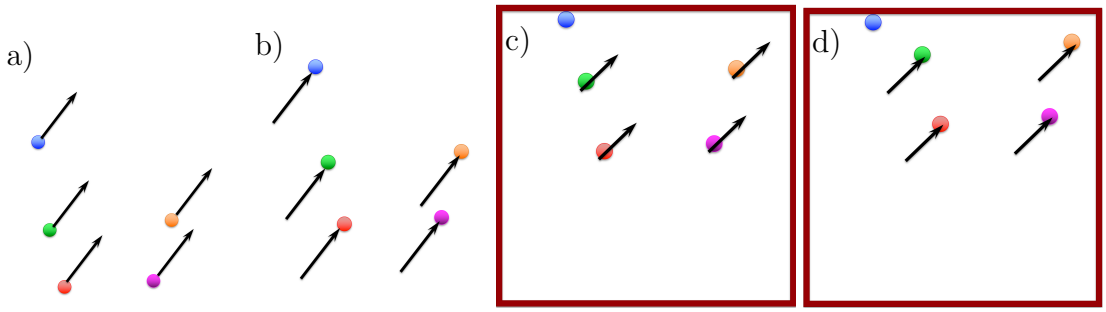


Figure 1.2: a) The robots are actually simple particles that can be controlled by a global force. b) All the particles get the same control input. c,d) Breaking the symmetry is only possible with external boundary or obstacle.

To make progress in automatic control with global inputs, this dissertation presents swarm manipulation controllers inspired by our online experiments that require only mean and variance measurements of the particles' positions. We prove that the mean position of a swarm is controllable and that, with an obstacle, the swarm's position variance orthogonal to rectangular boundary walls is also controllable (these are σ_x^2 and σ_y^2 for a workspace with axis-aligned walls). The usefulness of these techniques is demonstrated by several automatic controllers. One controller steers a swarm of robots to push a larger block through a 2D maze [6].

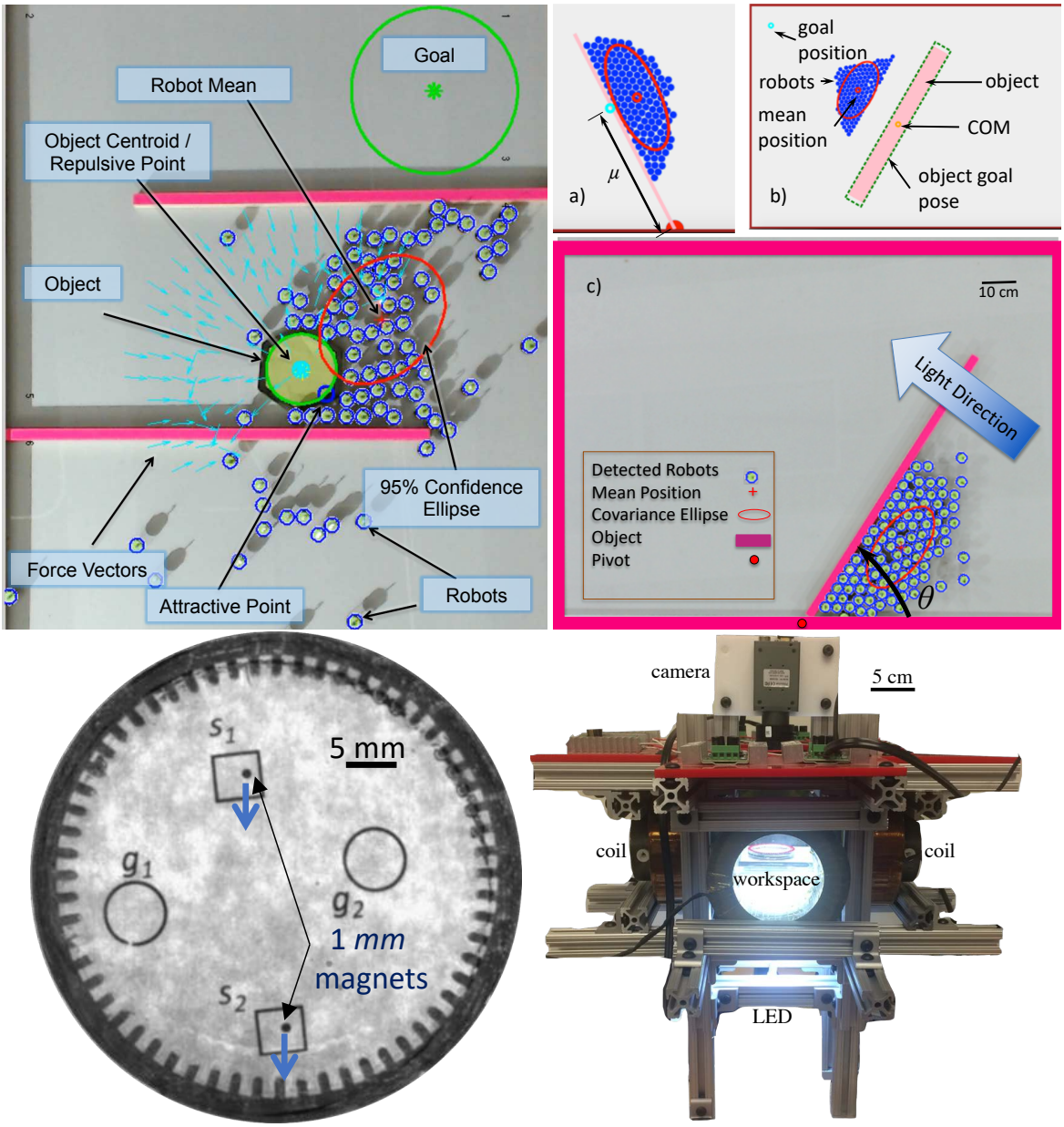
Meanwhile, object manipulation often also requires controlling torque on an object for a variety of alignment tasks including retroreflectors, and targeted radiation therapy. Accurate torque control is difficult. The swarm, when steered toward an object, begins interacting with the object at different times. The number of robots touching this object as a function of time is difficult to predict and often impossible to directly measure. Stochastic effects make long-term prediction challenging. Even when it is possible to predict which agents will hit the object first, as agents interact with the object, the swarm's configuration changes. The challenge is not only limited to swarm-object interaction, but also to swarm-swarm interactions when the swarm self-collides or is split into multiple components. As a result, the force the swarm will exert on the object is hard to predict. This dissertation studies the torque applied by a swarm of particles on

a long aspect-ratio rod. We calculate the force and torque generated by a swarm that has assumed a characteristic shape as a function of the direction the swarm moves. We generalize these results for three canonical position distributions of a swarm: uniform, triangular, and normal. The model shows that for a pivoted rod the uniform distribution produces the maximum torque for small swarm standard deviations, but the normal distribution maximizes torque for large standard deviations.

Positioning is a foundational capability for a robotic system, e.g. placement of brachytherapy seeds. 2D position of each particle in such a swarm is controllable if the workspace contains a single obstacle the size of one particle [7]. However, requiring a single, small, rigid obstacle suspended in the middle of the workspace is often an unreasonable constraint, especially in 3D. This dissertation relaxes that constraint, and provides position control algorithms for two particles that only require non-slip wall contacts. We assume that particles in contact with the boundaries have zero velocity if the uniform control input pushes the particle into the wall.

1.2 Dissertation organization

The dissertation is arranged as follows: Chapter 2 discusses steering a swarm of particles using global inputs and swarm statistics. Chapter 3 goes deep into the torque control of the swarm and orientation control of the object. Chapter 4 demonstrates algorithms to control position of two robots in any convex workspace when the walls have non-slip contact. We conclude in Chapter 5. Figure 1.3 shows a demonstration of each chapter.



workspace

magnetic setup

Figure 1.3: Top left: Chapter 2 discusses steering a swarm of particles using global inputs and swarm statistics. Top right: Chapter 3 goes deep into the torque control. Bottom: Chapter 4 demonstrates algorithms to position two robots.

Chapter 2

Steering a Swarm of Particles Using Global Inputs and Swarm Statistics

2.1 Introduction

Large populations of micro- and nanorobots are being produced in laboratories around the world, with diverse potential applications in drug delivery and construction, see [1–3]. Limited computation and communication at small scales makes autonomous operation or direct control over individual robots difficult. Instead, this chapter treats the robots as particles that are steered by a global control signal broadcast to the entire population. This chapter examines object manipulation by a swarm of particles, as illustrated in Fig. 2.1. The transportation methodology is similar to that in [8], but rather than using onboard computation or sensing, the particles all move in the same direction.

Many promising applications for particle swarms require direct human control, but user interfaces to thousands—or millions—of particles is a daunting human-swarm interaction (HSI) challenge. Human control of large swarms is possible over a hundred hardware robots and thousands of simulated particles [9].

This chapter presents swarm manipulation controllers inspired by our online experiments that require only mean and variance measurements of the positions of the particles. To perform the object manipulation task illustrated in Fig. 2.1, we use these controllers as primitives, policy iteration for path planning, handle outliers by partitioning the workspace, and minimize pushing the object backwards with potential field navigation.

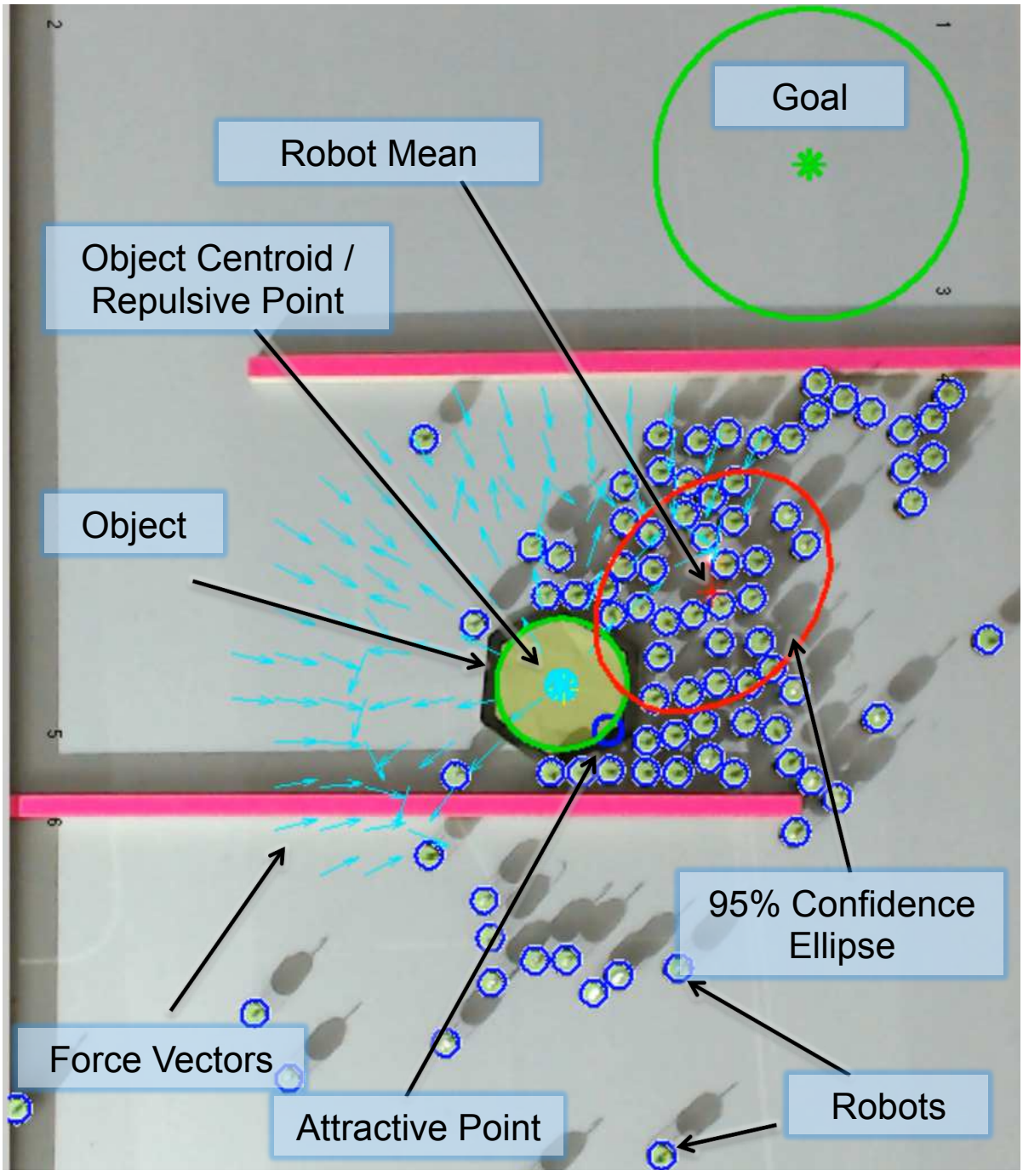


Figure 2.1: A swarm of particles, all actuated by a uniform control input can be effectively manipulated by a control law that uses only the mean and variance of the robot distribution. Here a swarm of kilobots pushes a green hexagon toward the goal.

This chapter is an extended version of [10] and is organized as follows. After a discussion of related work in Sec. 4.2, we describe our experimental methods for an online human-user experiment and their results in Sec. 2.3. Next we prove that the mean and variance of a particle swarm are controllable in Sec. 4.3, and present automatic controllers in Sec. 4.7. We use these controllers as primitives and present a framework for manipulating an object through a maze in Sec. 2.6. We implement these controllers in our hardware robots and use them to complete an object manipulation task with 100 kilobots in Sec. 2.7, and conclude in Sec. 4.9.

2.2 Related work

This section describes global control challenges and reviews highlights of human-swarm interaction, block pushing, and compliant manipulation.

2.2.1 Global control of microrobots

This dissertation investigates global control of particles that have no onboard computation. This prevents us from applying controllers that require computation on the agents, as in [11–13]. Another control paradigm is to construct robots with physical heterogeneity so that they respond differently to a global broadcast control signal. Examples include *scratch-drive microrobots*, actuated and controlled by a DC voltage signal from a substrate by [14, 15]; magnetic structures with different cross-sections that can be independently steered by [16, 17]; *MagMite* microrobots with different resonant frequencies and a global magnetic field by [18]; and magnetically controlled nanoscale helical screws constructed to stop movement at different cutoff frequencies of a global magnetic field by [19] and [1], similarly, exploiting inhomogeneity between robots [20, 21]. These control algorithms theoretically apply to any number of robots, even robotic continuums. However, all these works never controlled more than twelve robots at a time because

process noise cancels the differentiating effects of robot inhomogeneity. We desire control algorithms that extend to many thousands of robots. Limited position control was achieved by [22] and [23], but both used robots commanded in their local coordinate frame. This chapter focuses on a more common paradigm: particles commanded in a global coordinate frame.

While it is now possible to create many microrobots, there remain challenges in control, sensing, and computation:

Control—global inputs Many micro- and nanorobotic systems, see [1–3, 14–19, 24] rely on global inputs, where each robot receives an exact copy of the control signal. Our experiments follow this global model.

Sensing—large populations n differential-drive robots in a 2D workspace require $3n$ state variables. Even holonomic robots require $2n$ state variables. Numerous methods exist for measuring this state in microrobotics [1, 3, 5]. These solutions use computer vision systems to sense position and heading angle, with corresponding challenges of handling missed detections and image registration between detections and robots. These challenges increase at small scales where sensing competes with control for communication bandwidth. We examine control when the operator has access to partial feedback, including only the first and/or second moments of a population's position, or only the convex-hull containing the robots.

Computation—calculating the control law There are controllers that require at best a summation over all the robot states, see [21] and at worst a matrix inversion, see [20]. These operations become intractable for large populations of robots. By focusing on *human* control of large robot populations, we accentuate computational difficulties because the controllers are implemented by the unaided human operator.

2.2.2 Human-swarm interaction

Most humans are able to, with practice, steer a swarm of robots controlled by a global input. Prior to this work, no algorithm existed. Using human input to learn how to control a dynamic system is a line of research with a rich history [25, 26]. This chapter exploits insights gained from *SwarmControl.net*, particularly the fact that having a swarm's mean and variance is sufficient for object manipulation through an obstacle field.

A user interface enabling an operator to maneuver a swarm of robots through a cluttered workspace by specifying the bounding prism for the swarm and then translating or scaling this prism is designed in [27]. This chapter shares the concept of a global control input, but our robots have no onboard computation and cannot track a virtual boundary.

Human *fanout*, the number of robots a single human user could directly control is studied in [28]. They postulated that the optimal number of robots was approximately the autonomous time divided by the interaction time required by each robot. Their sample problem involved a multi-robot search task, where users could assign goals to robots. Their user interaction studies with simulated planar robots indicated a *fanout plateau* of about 8 robots, with diminishing returns for more robots. They hypothesized the location of this plateau is highly dependent on the underlying task. Indeed, this chapter indicates there are tasks without plateaus. Their research investigated robots with three levels of autonomy. We use robots without autonomy, corresponding with their first-level robots.

Several user studies compare methods for controlling large swarms of simulated robots, for example [29–31]. These studies provide insights but are limited by cost to small user studies; have a closed-source code base; and focus on controlling intelligent, programmable agents. For instance, the studies [29], [30], and [31] were limited to a pool of 5, 18, and 32 participants. Using an online testing environment, we conduct similar

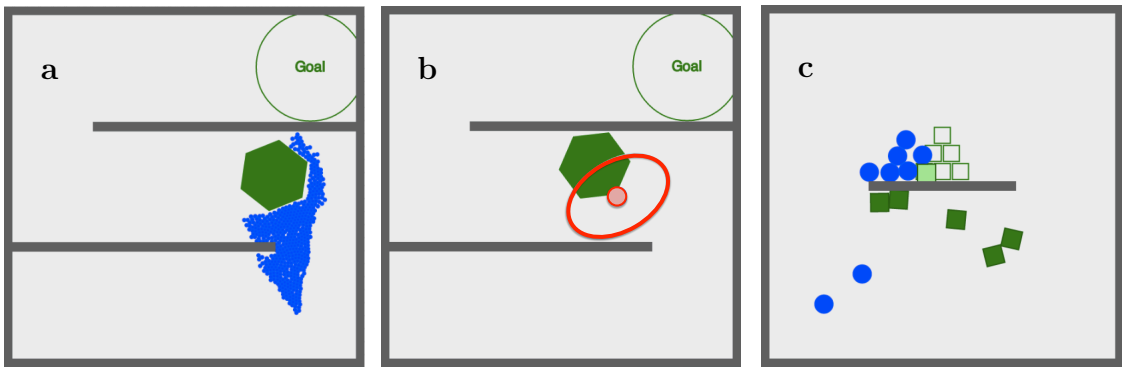


Figure 2.2: Screenshots from our online experiments controlling multi-particle systems with limited, global control. **(a)** Varying the number of particles from 1-500 **(b)** Comparing 4 levels of visual feedback **(c)** Varying noise from 0 to 200% of control authority.

studies but with sample sizes three orders of magnitude larger.

2.2.3 Block pushing and compliant manipulation

Unlike *caging* manipulation, where robots form a rigid arrangement around an object, as in [32, 33], our swarm of robots is unable to grasp the blocks they push, and so our manipulation strategies are similar to *nonprehensile manipulation* techniques, e.g. [34], where forces must be applied along the center of mass of the moveable object. A key difference is that our robots are compliant and tend to flow around the object, making this similar to fluidic trapping as in [35] and [36].

Our n-robot system with 2 control inputs and $4n$ states is inherently under-actuated, and superficially bears resemblance to compliant, under-actuated manipulators. Our swarms conform to the object to be manipulated, but lack the restoring force provided by flexures in [37] or silicone in [38]. Our swarms tend to disperse and so to regroup them we require artificial forces like the variance control primitives in Sec. 2.4.3.

2.3 Online experiment

The goal of these online experiments is to test several scenarios involving large-scale human-swarm interaction (HSI), and to do so with a statistically-significant sample size. Towards this end, we have created SwarmControl.net: an open-source, online testing platform suitable for inexpensive deployment and data collection on a scale not yet seen in swarm robotics research. Screenshots from this platform are shown in Fig. 2.2. All code and experimental results are online at [39].

We developed a flexible testing framework for online human-swarm interaction studies. Over 5,000 humans performed over 20,000 swarm-robotics experiments with this framework, logging almost 700 hours of experiments. These experiments indicated three lessons used for designing automatic controllers for object manipulation with particle swarms:

1) When the number of particles is large (> 50), varying the number of particles does not significantly affect the performance.

2) Swarm control is robust to independent and identically distributed (IID) noise.

3) Controllers that only use the mean and variance of the swarm can perform better than controllers with full feedback.

2.3.1 Implementation

Our web server generates a unique identifier for each participant and sends it along with the landing page to the participant. A script on the participant's browser runs the experiment and posts the experiment data to the server. Anonymized human subject data was collected under IRB #14357-01.

We designed six experiments to investigate human control of large swarms for manipulation tasks. Screenshots of representative experiments are shown in Fig. 2.2. Each

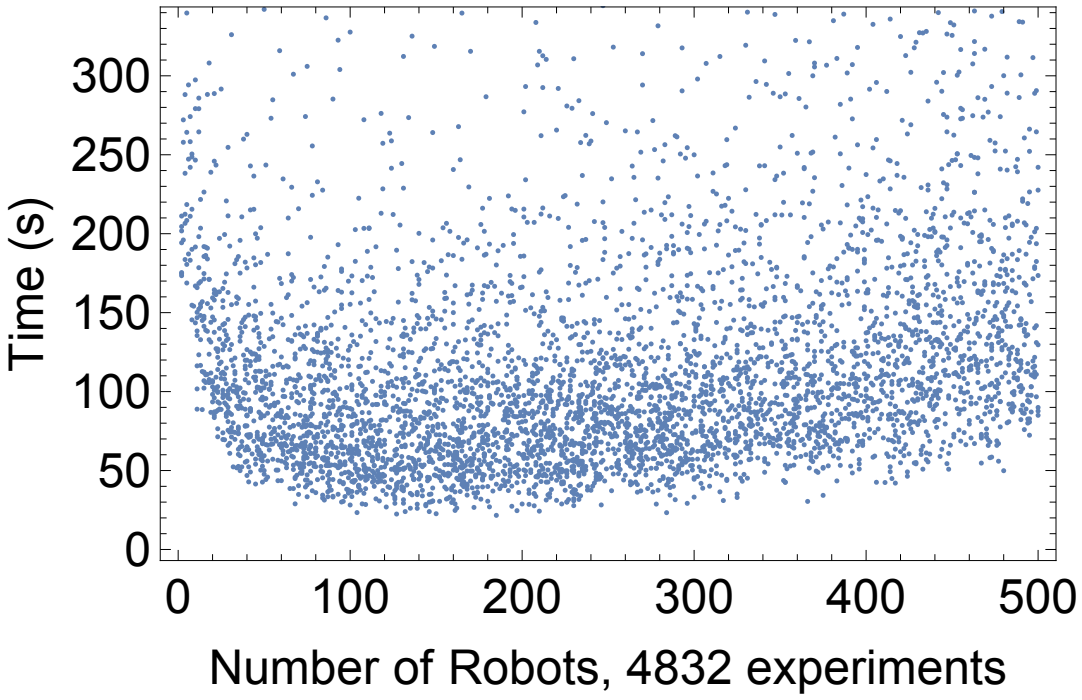


Figure 2.3: Data from *Varying Number* using particles to push an object through a maze to a goal location.

experiment examined the effects of varying a single parameter: population of particles for manipulation, four levels of visual feedback, different levels of Brownian noise. The users could choose which experiment to try, and our architecture randomly assigned a parameter value for each trial. We recorded the completion time and the participant ID for each successful trial.

2.3.2 Varying number

This experiment varied from 1 to 500 the population of particles used to transport an object. The total area, maximum particle speed, and total net force the swarm could produce were constant. The particles pushed a large hexagonal object through an S-shaped maze. We hypothesized participants would complete the task faster with more particles. The results, shown in Fig. 2.3, do not support our hypothesis, indicating a minimum around 130 particles, but only a gradual increase in completion time from 50 to 500.

2.3.3 Varying visualization

This experiment explores manipulation with varying amounts of sensing information: **full-state** sensing provides the most information by showing the position of all particles; **convex-hull** draws a convex hull around the outermost particles; **mean** provides the average position of the population; and **mean + variance** adds a confidence ellipse. Fig. 2.4 shows screenshots of the same particle swarm with each type of visual feedback. Full-state requires $2n$ data points for n particles. Convex-hull requires at worst $2n$, but according to Har [40], the expected number is $O(2n^{1/3})$. Mean requires two, and variance three, data points. Because they do not increase with population size, mean and mean + variance are convenient even with millions of particles.

Our hypothesis predicted a steady decrease in performance as the amount of visual feedback decreased. Our experiment indicated the opposite: players with just the mean completed the task faster than those with full-state feedback. As Fig. 2.5 shows, the levels of feedback arranged by increasing completion time are [mean, mean + variance, full-state, convex-hull]. All experiments lasting over 300 s were removed, under the assumption that the user stopped playing. Using ANOVA analysis, we rejected the null hypothesis that all visualization methods are equivalent, with p-value 2.69×10^{-19}. Anecdotal evidence from beta-testers who played the game suggests that tracking 100 particles is overwhelming—similar to schooling phenomenons that confuse predators—

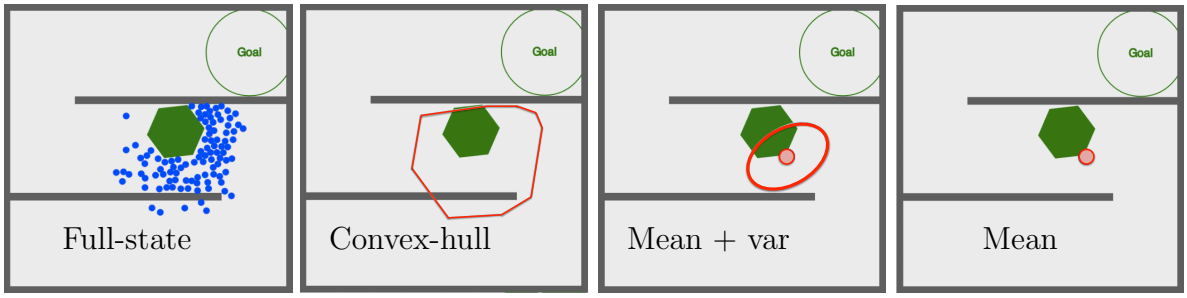


Figure 2.4: Screenshots from task *Vary Visualization*. This experiment challenges players to quickly steer 100 particles (blue discs) to push an object (green hexagon) into a goal region. We record the completion time and other statistics.

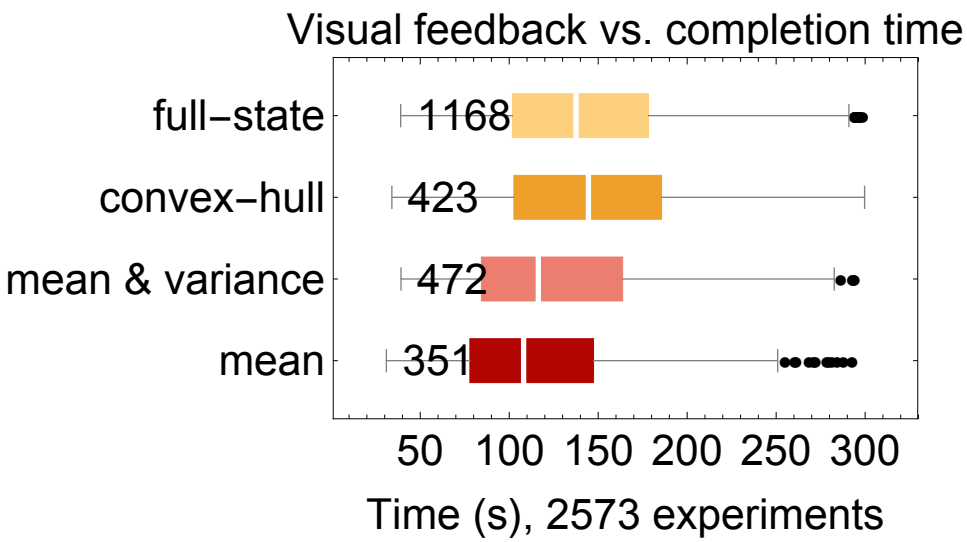


Figure 2.5: Completion-time results for the four levels of visual feedback shown in Fig. 2.4. Players performed better with limited feedback.

while working with just the mean + variance is like using a "spongy" manipulator. However, our beta-testers described convex-hull feedback as confusing and irritating since it is not robust to outliers. A single particle left behind an obstacle will stretch the entire hull, obscuring the majority of the swarm.

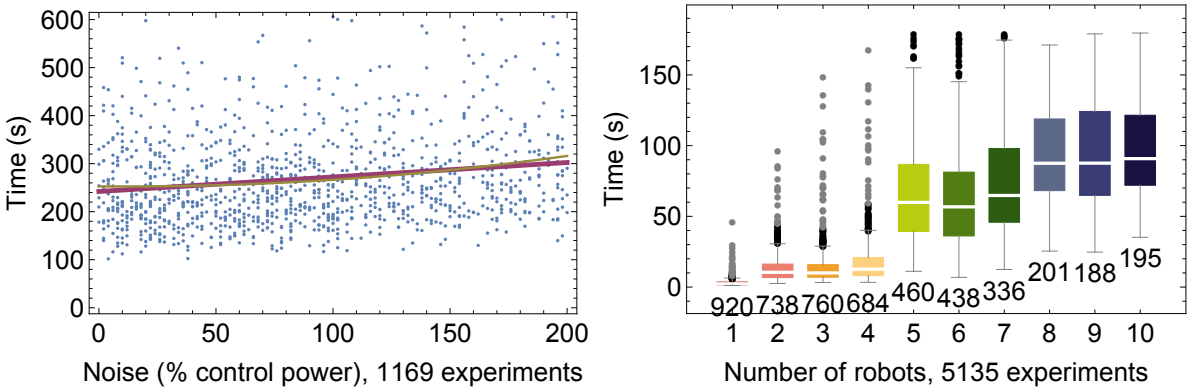


Figure 2.6: Left: Varying the noise from 0 to 200% of the maximum control input resulted in only a small increase in completion time. Right: For position control, increasing the number of particles resulted in longer completion times.

2.3.4 Varying noise

This experiment varied the strength of disturbances to study how noise (disturbance inputs) affects human control of large swarms. Noise was applied at every time step as follows:

$$\dot{x}_i = u_x + m_i \cos(\psi_i)$$

$$\dot{y}_i = u_y + m_i \sin(\psi_i).$$

Here, m_i and ψ_i were uniformly IID, with $m_i \in [0, M]$ and $\psi_i \in [0, 2\pi]$. M was a constant for each trial ranging from 0 to 200% of the maximum control power ($u_{\max}$).

We hypothesized 200% noise was the largest a human could be expected to control—at 200% noise, the particles move erratically. Disproving our hypothesis, the results in Fig. 2.6a show only a 40% increase in completion time for the maximum noise. This indicates swarm control is robust to IID noise.

2.4 Global control laws for a holonomic swarm

Emboldened by the three lessons from our online experiments, this section presents automatic controllers for large numbers of particles that only rely on the first two moments of the swarm position distribution.

We represent particles as holonomic robots that move in the 2D plane. ??? define this term We want to control position and velocity of the particles. First, assume a noiseless system containing one particle with mass m. Our inputs are global forces $[u_x, u_y]^\top$. We define our state vector $\mathbf{x}(t)$ as the x position, x velocity, y position and y velocity. The state-space representation in standard form is:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t). \tag{2.1}$$

and our state space representation is:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}. \tag{2.2}$$

We want to find the number of states that we can control, which is given by the rank of the *controllability matrix*:

$$\mathcal{C} = [B, AB, A^2B, A^3B]. \tag{2.3}$$

$$\text{Here } \mathcal{C} = \left[\begin{array}{cc|cc|cc|cc} 0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right], \tag{2.4}$$

$$\text{rank}(\mathcal{C}) = 4, \tag{2.5}$$

and thus all four states are controllable. This section starts by proving independent position control of many particles is not possible, but the mean position can be controlled. We then provide conditions under which the variance of many particles is also controllable.

2.4.1 Independent control of many particles is impossible

In this model, a single particle is fully controllable. For holonomic particles, movement in the x and y coordinates are independent, so for notational convenience without loss of generality we will focus only on movement in the x axis. Given n particles to be controlled in the x axis, there are $2n$ states: n positions and n velocities. Without loss of generality, assume $m = 1$. Our state-space representation is:

$$\begin{bmatrix} \dot{x}_1 \\ \ddot{x}_1 \\ \vdots \\ \dot{x}_n \\ \ddot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ \vdots \\ x_n \\ \dot{x}_n \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_x. \tag{2.6}$$

However, just as with one particle, we can only control two states because the controllability matrix $\mathcal{C}_n$ has rank two:

$$\mathcal{C}_n = \left[\begin{array}{c|c|c|c|c} 0 & 1 & 0 & 0 & \\ 1 & 0 & 0 & 0 & \\ \vdots & \vdots & \vdots & \vdots & \dots \\ 0 & 1 & 0 & 0 & \\ 1 & 0 & 0 & 0 & \end{array}\right], \quad \text{rank}(\mathcal{C}_n) = 2. \tag{2.7}$$

2.4.2 Controlling the mean position

This means *any* number of particles controlled by a global command have just two controllable states in each axis. We cannot arbitrarily control the position and velocity of two or more particles, but have options on which states to control. We create the following reduced order system that represents the mean x position and velocity of the n particles:

$$\begin{bmatrix} \dot{\bar{x}} \\ \ddot{\bar{x}} \end{bmatrix} = \frac{1}{n} \begin{bmatrix} 0 & 1 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ \vdots \\ x_n \\ \dot{x}_n \end{bmatrix}$$

$$+ \frac{1}{n} \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_x. \tag{2.8}$$

Thus:

$$\begin{bmatrix} \dot{\bar{x}} \\ \ddot{\bar{x}} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \dot{\bar{x}} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_x. \tag{2.9}$$

We again analyze the controllability matrix $\mathcal{C}_\mu$:

$$\mathcal{C}_\mu = \left[\begin{array}{c|c} 0 & 1 \\ 1 & 0 \end{array}\right], \quad \text{rank}(\mathcal{C}_\mu) = 2. \tag{2.10}$$

Thus the mean position and mean velocity are controllable.

There are several techniques for breaking the symmetry of the control input to allow controlling more states, for example by using obstacles as in [9], or by allowing independent noise sources as in [23].

We control mean position with a PD controller that uses the mean position and mean velocity. $[u_x, u_y]^\top$ is the global force applied to each particle:

$$\begin{aligned} u_x &= K_p(x_{\text{goal}} - \bar{x}) + K_d(0 - \dot{\bar{x}}), \\ u_y &= K_p(y_{\text{goal}} - \bar{y}) + K_d(0 - \dot{\bar{y}}). \end{aligned} \tag{2.11}$$

K_p is the proportional gain, and K_d is the derivative gain.

2.4.3 Controlling the variance

The variance, σ_x^2, σ_y^2, of n particles' position is:

$$\begin{aligned} \overline{x}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad \sigma_x^2(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2, \\ \overline{y}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} y_i, \qquad \sigma_y^2(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2. \end{aligned} \tag{2.12}$$

Controlling the variance requires being able to increase and decrease the variance. We will list a sufficient condition for each. Microscale systems are affected by unmodelled dynamics. These unmodelled dynamics are dominated by Brownian noise, as described in [41]. To model this (2.1) must be modified as follows:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + W\boldsymbol{\varepsilon}(t), \tag{2.13}$$

where $W\boldsymbol{\varepsilon}(t)$ is a random perturbation produced by Brownian noise with magnitude W. Given a large obstacle-free workspace with $\mathbf{u}(t) = 0$, a *Brownian noise* process increases the variance linearly with time.

$$\dot{\sigma}_x^2(\mathbf{x}(t), \mathbf{u}(t)) = W\boldsymbol{\varepsilon}, \quad \sigma_x^2(t) = \sigma_x^2(0) + W\boldsymbol{\varepsilon}t. \tag{2.14}$$

If faster dispersion is needed, the swarm can be pushed through obstacles such as a diffraction grating or Pachinko board as in [9].

If particles with radius r are in a bounded environment with sides of length $[\ell_x, \ell_y]$, the unforced variance asymptotically grows to the variance of a uniform distribution,

$$[\sigma_x^2, \sigma_y^2] = \frac{1}{12}[(\ell_x - 2r)^2, (\ell_y - 2r)^2]. \tag{2.15}$$

A flat obstacle can be used to decrease variance. Pushing a group of dispersed particles against a flat obstacle will decrease their variance until the minimum-variance (maximum density) packing is reached. For large n, Graham and Sloane [42] showed that the minimum-variance packing for n circles with radius r is

$$\sigma^2_{\text{optimal}}(n, r) \approx \frac{\sqrt{3}}{\pi} n r^2 \approx 0.55 n r^2. \tag{2.16}$$

Thus, to control this variance, we choose

$$u(t) = \begin{cases} \text{move to wall} & \text{if } \sigma^2(\mathbf{x}) > \sigma^2_{\text{goal}} \\ \text{move from wall} & \text{if } \sigma^2(\mathbf{x}) \leq \sigma^2_{\text{goal}}. \end{cases} \tag{2.17}$$

Similar to the PD controller in Eq. (3.31) that controls the mean particle position, a controller to regulate the variance to σ^2_{ref} is:

$$u_x = K_p(x_{\text{goal}}(\sigma^2_{\text{ref}}) - \bar{x}) - K_d \bar{v}_x + K_i(\sigma^2_{\text{ref}} - \sigma_x^2), \tag{2.18}$$

$$u_y = K_p(y_{\text{goal}}(\sigma^2_{\text{ref}}) - \bar{y}) - K_d \bar{v}_y + K_i(\sigma^2_{\text{ref}} - \sigma_y^2). \tag{2.19}$$

We call the gain scaling the variance error K_i because the variance, if unregulated, integrates over time. This controller requires a vertical and a horizontal wall. Eq. (2.18) assumes the nearest wall is to the left of the particle at $x = 0$, and chooses a reference goal position such that the swarm, if uniformly distributed between 0 and ℓ, would have the correct variance according to (2.15):

$$x_{\text{goal}}(\sigma^2_{\text{ref}}) = \ell/2 = r + \sqrt{3\sigma^2_{\text{ref}}}. \tag{2.20}$$

If a wall to the right is closer, the signs of $[K_p, K_i]$ are inverted, and the location x_{goal} is translated. A similar argument applies to (2.19).

2.4.4 Controlling both mean and variance

The mean and variance of the swarm cannot be controlled simultaneously. However, if the variance gained while moving from a corner to the target position is less than some $\sigma^2_{\max} - \sigma^2_{\min}$, we can adopt the hybrid, hysteresis-based controller shown in Alg. 1 to regulate the mean and variance. Such a controller normally controls the mean position, but switches to minimizing variance if the variance exceeds $\sigma^2_{\max}$. Variance is reduced until less than $\sigma^2_{\min}$, then control again regulates the mean position. This technique satisfies control objectives that evolve at different rates as in [43], and the hysteresis avoids rapid switching between control modes. The process is illustrated in Fig. 2.7.

Algorithm 1 Hybrid mean and variance control

Require: Knowledge of swarm mean $[\bar{x}, \bar{y}]$, variance $[\sigma_x^2, \sigma_y^2]$, the locations of the rectangular boundary $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$, and the target mean position $[x_{\text{target}}, y_{\text{target}}]$.

$x_{\text{goal}} \leftarrow x_{\text{target}}$, $y_{\text{goal}} \leftarrow y_{\text{target}}$
loop
  if $\sigma_x^2 > \sigma^2_{\max}$ **then**
    $x_{\text{goal}} \leftarrow x_{\min}$
  else if $\sigma_x^2 < \sigma^2_{\min}$ **then**
    $x_{\text{goal}} \leftarrow x_{\text{target}}$
  end if
  if $\sigma_y^2 > \sigma^2_{\max}$ **then**
    $y_{\text{goal}} \leftarrow y_{\min}$
  else if $\sigma_y^2 < \sigma^2_{\min}$ **then**
    $y_{\text{goal}} \leftarrow y_{\text{target}}$
  end if
  Apply (3.31) to move toward $[x_{\text{goal}}, y_{\text{goal}}]$
end loop

A key challenge is to select proper values for $\sigma^2_{\min}$ and $\sigma^2_{\max}$. The optimal packing variance was given in (2.16). The random packings generated by pushing our particles into corners are suboptimal, so we choose the conservative values:

$$\sigma^2_{\min} = 2.5r + \sigma^2_{\text{optimal}}(n, r),$$
$$\sigma^2_{\max} = 15r + \sigma^2_{\text{optimal}}(n, r). \tag{2.21}$$

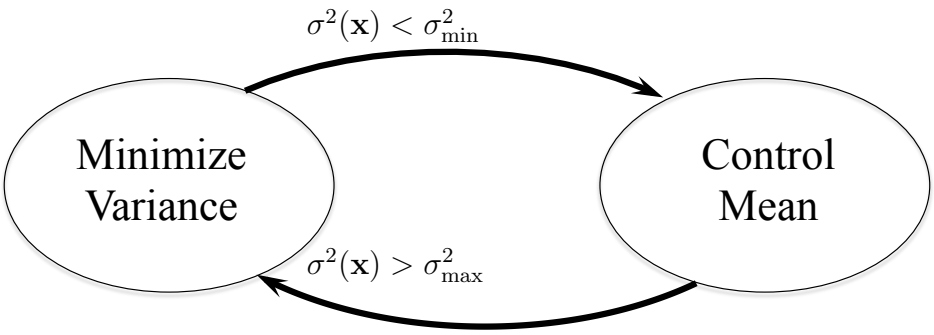


Figure 2.7: Hysteresis to control swarm mean and variance.

2.5 Simulation of control laws

Our simulations use a Javascript port of Box2D, a popular 2D physics engine with support for rigid-body dynamics, including collision, density, and friction, and fixed time step simulation [44]. All experiments in this section ran on a Chrome web browser on a 2.6 GHz Macbook. All code is available at [45].

Controlling the mean position We performed a parameter sweep using the PD controller (3.31) to identify the best control gains. Representative experiments are shown in Fig. 2.8. 100 particles were used and the maximum speed was 3 m/s. As shown in Fig. 2.8, we can achieve an overshoot of 1% and a rise time of 1.52 s with $K_p = 4$, and $K_d = 1$.

Controlling the variance Variance control uses the control law (2.18) with $K_{p,i,d} = [4, 1, 1]$. Results are in Fig. 2.9.

Hybrid control of mean and variance Fig. 2.10 shows a simulation run of the hybrid controller in Alg. 1 with 100 particles in a square workspace containing no internal obstacles.

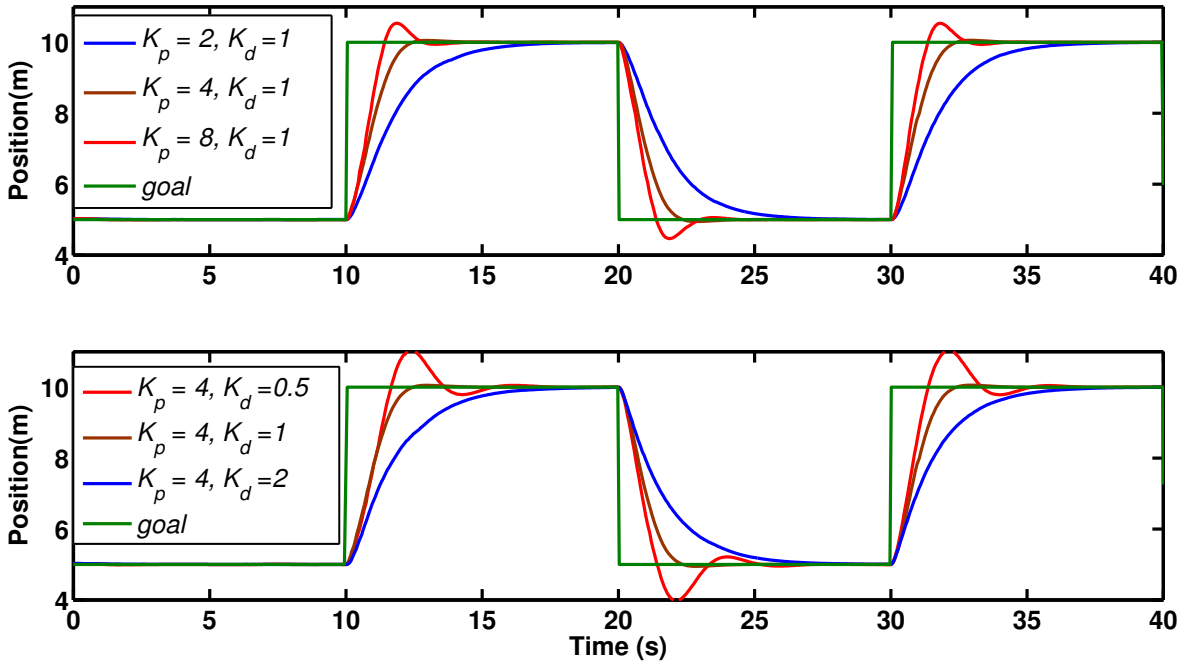


Figure 2.8: In simulation, tuning proportional (K_p, top) and derivative (K_d, bottom) gain values in (3.31) improves performance with $n = 100$ particles.

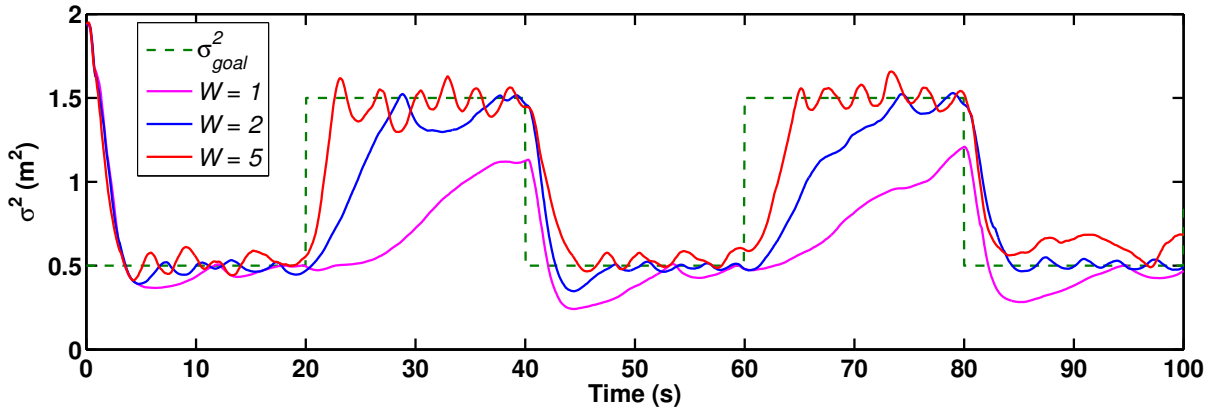


Figure 2.9: In simulation, increased noise results in more responsive variance control because stronger Brownian noise causes a faster increase of variance.

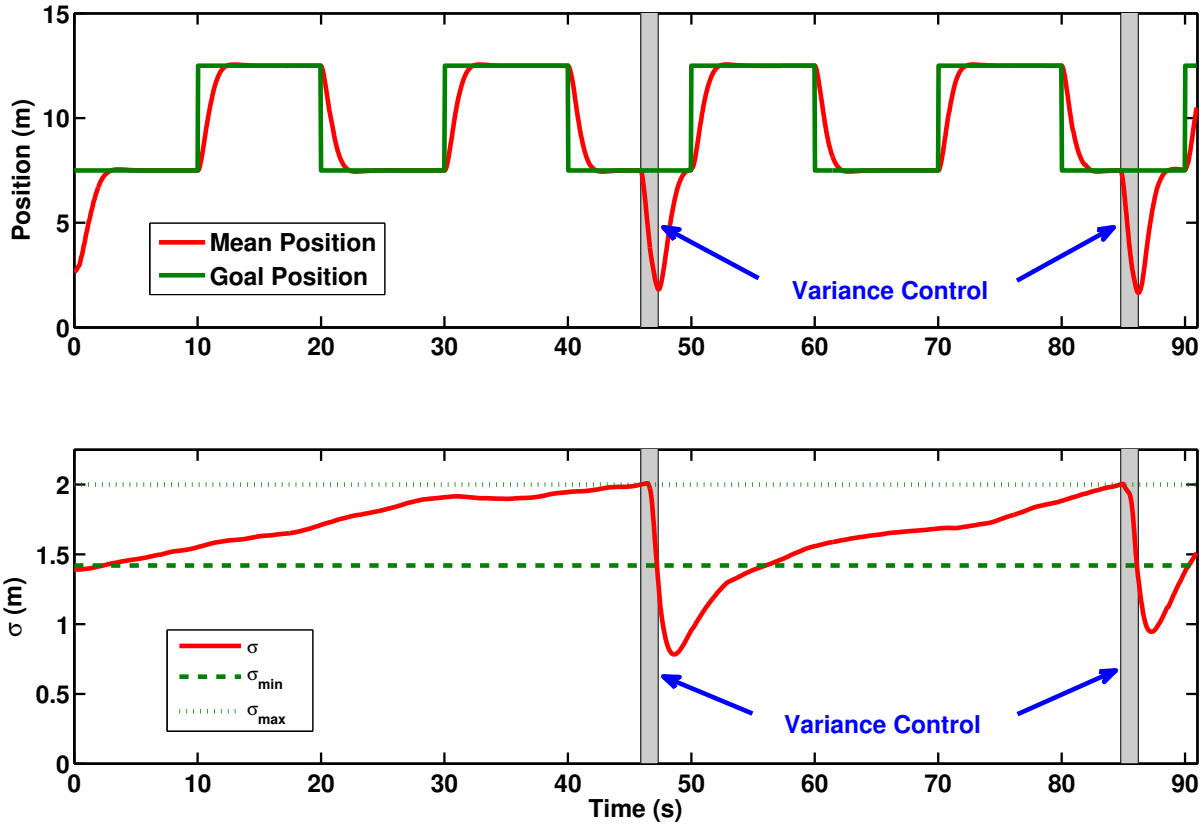


Figure 2.10: Simulation result with 100 particles under hybrid control Alg. 1, which controls both the mean position (top) and variance (bottom). For ease of analysis, only x position and variance are shown.

2.6 Particle swarm object manipulation

Table 2.1: Summary, Object Manipulation Results

Method	Result (10 trials), mean±std (s)
Value Iteration (VI)	367 ± 253
VI + Potential Fields (PF)	271 ± 267
VI + Outlier Rejection (OR)	245 ± 135
BFS + PF + OR	183 ± 179
VI + PF + OR	90 ± 35

This section analyzes an *object manipulation* task attempted by our hybrid, hysteresis-based controllers. The swarm must deliver the object to the goal region. We assume Coulomb and viscous friction parameters such that the object can be moved by particle

motion. Increasing the number of pushing particles increases the object speed. To solve this object manipulation task we divide the task into three components: 1) designing a policy for the object, 2) pushing the object with a compliant swarm, and 3) managing outliers. Table 2.1 summarizes results for successful simulation trials.

2.6.1 Learning a policy for the object

To design the policy we first discretize the environment. In [6], we used breadth-first search (BFS) on this discretized grid, but using workspace BFS fails to account for the hull of the object and will suggest moves that can cause collisions with the workspace. A configuration-space BFS approach avoids that problem but still fails to model uncertain actuation of the object by the swarm.

To solve both these problems, this paper models object movement as a Markov Decision Process (MDP) with non-deterministic movement. Value iteration is used to learn an *optimal policy* [46]. At each state the object can be commanded to move in one of eight directions with a small probability of moving in a wrong direction.

The reward function $r(x, \mathbf{u})$ is defined as

$$r(x, \mathbf{u}) = \begin{cases} +100, & \text{if } \mathbf{u} \text{ leads to goal state} \\ -100, & \text{if } \mathbf{u} \text{ leads to an obstacle state} \\ -1, & \text{otherwise} \end{cases} \tag{2.22}$$

where x is the current state and $\mathbf{u}$ is the action. Value iteration computes $\hat{V}(x)$, the expected discounted sum reward if the optimal policy is implemented, for the object starting in each state x. The optimal policy is

$$\mathbf{D}(x) = \arg\max_{\mathbf{u}} \left[r(x, \mathbf{u}) + \sum_{j=1}^{N} \hat{V}(x_j) p(x_j | x, \mathbf{u}) \right]. \tag{2.23}$$

The value function $\hat{V}(x_i)$ is calculated by computing the value $\hat{V}$ for all N states and iterating until convergence:

for $i = 1$ to N do

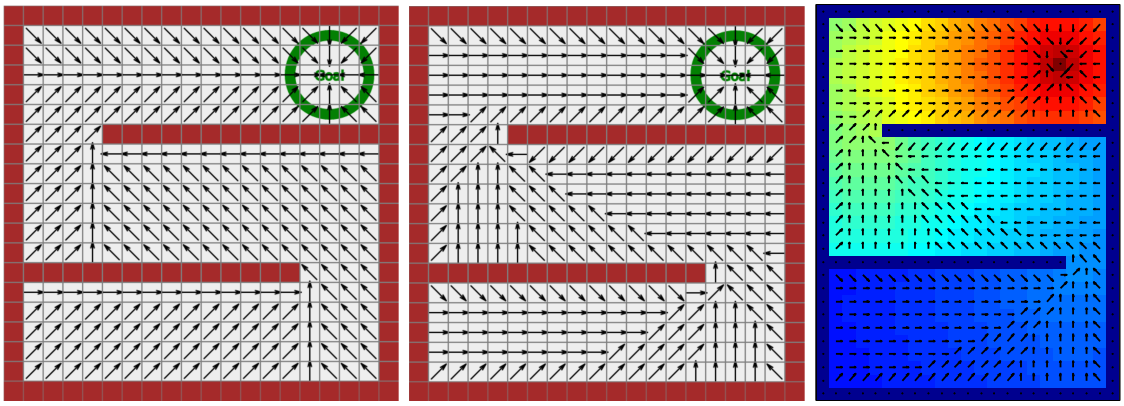


Figure 2.11: BFS finds the shortest path for the moveable object to compute gradient vectors (left). Modeling as an MDP enables encoding penalties for being near obstacles. (Middle) The control policy from value iteration. (Right) The vision algorithm detects obstacles in the hardware setup.

$$\hat{V}(x_i) = \gamma \max_{\mathbf{u}} \left[r(x_i, \mathbf{u}) + \sum_{j=1}^{N} \hat{V}(x_j) p(x_j | x_i, \mathbf{u}) \right]$$

$$\text{end} \tag{2.24}$$

Our probabilistic motion model $p(x_j|x, \mathbf{u})$ assumed the object moved in the commanded direction $\mathbf{u}$ half of the time but $+45°$ with probability 0.25 and $-45°$ with probability 0.25. In our experiments $\gamma = 0.97$, and (2.24) was iterated 200 times. A MATLAB implementation is available at [47].

$\mathbf{M}_{\text{BFS}}$ and the value function are shown in Fig. 2.11. In 10 simulations with 100 particles, pushing the object to goal using BFS required 183±179 s while value iteration required 90±35 s (mean±std).

2.6.2 Potential fields for swarm management with a compliant manipulator

When the swarm is in front of the object, control law (3.31) pushes the object backwards. To fix this, we implement a potential field approach that attracts the swarm to the intermediate goal, but repulses the swarm from in front of the object, as shown in Fig. 2.12. The approach is similar to [48], chapter 5. The repulsive potential field is centered at $\mathbf{b}$, the object's COM, and is active in a circular sector of angular width 2θ

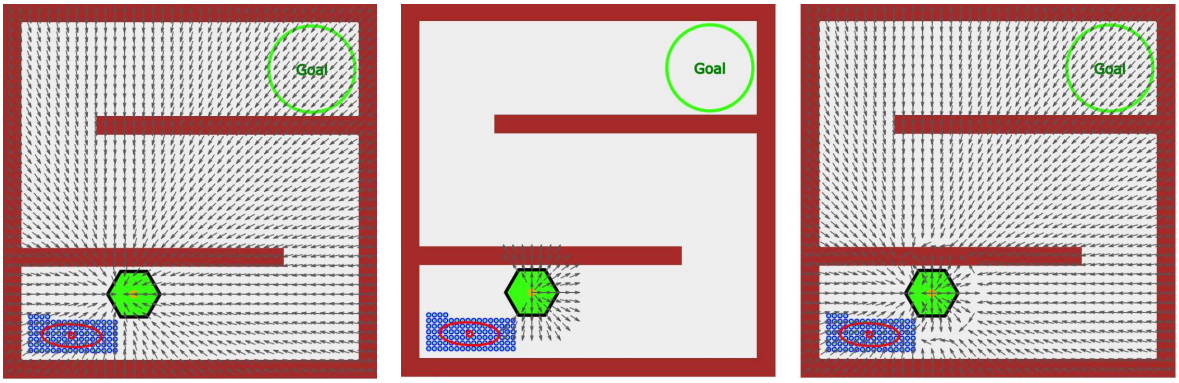


Figure 2.12: (Left) The attractive field is centered behind the object's COM. (Middle) The repulsive field is centered at the object's COM. (Right) Combining these forces prevents the swarm from pushing the object backwards.

and radius d_0 aligned with $\mathbf{D}(\mathbf{b})$. $\mathbf{D}(\mathbf{b})$ is the desired direction of motion from (2.23).

$$\mathbf{d} = [\bar{x}, \bar{y}] - \mathbf{b}, \qquad \phi = \cos^{-1}\left(\frac{\mathbf{D}(\mathbf{b}) \cdot \mathbf{d}}{\|\mathbf{D}(\mathbf{b})\| \cdot \|\mathbf{d}\|}\right),$$

$$F_{\text{att}} = -\zeta \frac{\mathbf{d}}{\|\mathbf{d}\|},$$

$$F_{\text{rep}} = \begin{cases} \eta(\frac{1}{\|\mathbf{d}\|} - \frac{1}{d_0})\frac{1}{\|\mathbf{d}\|^2}\mathbf{d}, & \|\mathbf{d}\| \leq d_0 \ \wedge \ \phi < \theta \\ 0, & \text{otherwise} \end{cases},$$

$$F_{\text{pot}} = F_{\text{att}} + F_{\text{rep}},$$

$$[x_{\text{goal}}, y_{\text{goal}}] = [\bar{x}, \bar{y}] + \frac{F_{\text{pot}}}{\|F_{\text{pot}}\|}. \tag{2.25}$$

Here η and ζ are positive parameters that scale the forces and $\|\mathbf{d}\|$ is the distance from the swarm mean $[\bar{x}, \bar{y}]$ to the object COM. In simulations, $\theta = \pi/2$, $\eta = 75$, $\zeta = 2$ and $d_0 = 3$. Because the kilobots have a slower time constant, they use $\theta = \pi/2$, $\eta = 50$, $\zeta = 1$ and $d_0 = 7.5$.

In 10 simulations with 100 particles, pushing the object to goal without a repulsive potential field failed in two of twelve runs. No failures occurred with the repulsive potential field. Of successful trials, completion time without repulsive potential fields required 245±135 s while using repulsive potential fields required 90±35 s (mean±std).

2.6.3 Outlier rejection

The variance controller in Alg. 1 is a greedy algorithm that is susceptible to outliers. Our controller in [6] failed in 14% trials, in each failure some particles were unable to reach the object because workspace obstacles were blocking them. This failure rate increases if object weight increases or ground-particle friction increases. The mean and covariance calculations (2.12) included all particles in the workspace. Particles that cannot reach the object due to obstacles skew these calculations. The state machine in Fig. 2.13a solves this problem by creating two states for the maze: either main or transfer. Each state has a set of regions representing a discretized visibility polygon. Whenever the object crosses a region boundary the state toggles. The *main* regions are generated by extending obstacles until they meet another obstacle. The *transfer* regions are perpendicular to obstacle boundaries, and act as a buffer between two main regions.

Fig. 2.13b shows the regions for the main state. The object is in region 1. An indicator function is applied to (2.12) so only particles inside region 1 are counted. This filtering increases experimental success because the mean calculation only includes nearby particles that can directly interact with the object. When the object leaves main region 1 the state switches to transfer. The transfer regions are shown in Fig. 2.13c. The object is in transfer region 1, so only particles in transfer region 1 are included in the mean and covariance calculations. The particles should push the object to the left. Without filtering using regions, the red circle is the mean and the algorithm would instruct the particles to push the object up. The black circle shows the filtered mean and the algorithm instructs the particles to push the object directly left.

In 10 simulations with 100 particles, completion time without outlier rejection required 271±267 s while using outlier rejection required 90±35 s (mean±std).

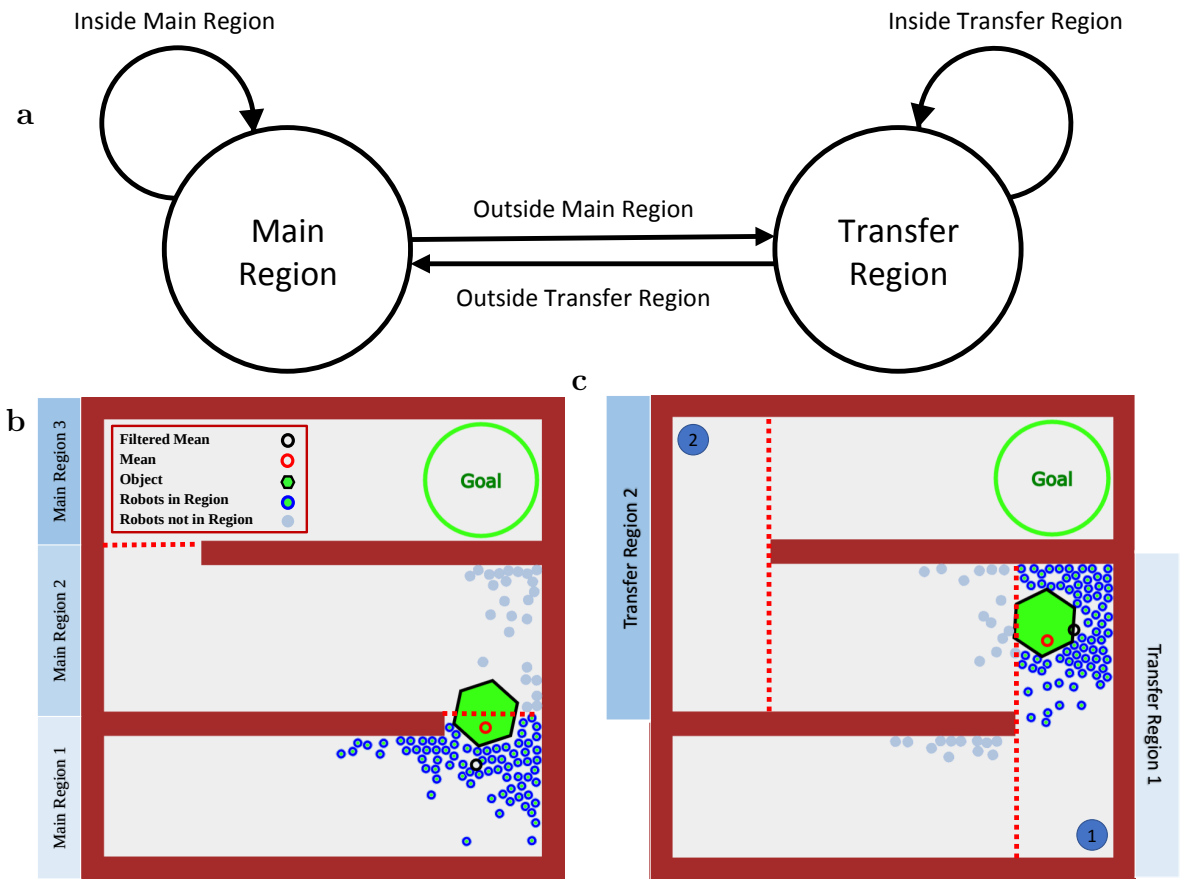


Figure 2.13: Outlier rejection state machine and regions.

2.6.4 Simulation results

We use the hybrid hysteresis-based controller in Alg. 1 to track the desired position, while maintaining sufficient particle density to move the object by switching to minimize variance whenever variance exceeds a set limit: $0.003W$ and $0.006W$ were added to the min and max variance limits from (2.21), where W is the magnitude of the Brownian noise. The minimize variance control law (2.18) is slightly modified to choose the nearest corner further from the goal than the object with an obstacle-free straight-line path to the object. The control algorithm for object manipulation is listed in Alg. 2.

In rare cases during simulations the swarm may become trapped in a local minimum of (2.25). If the swarm mean position does not change for five seconds, the swarm is

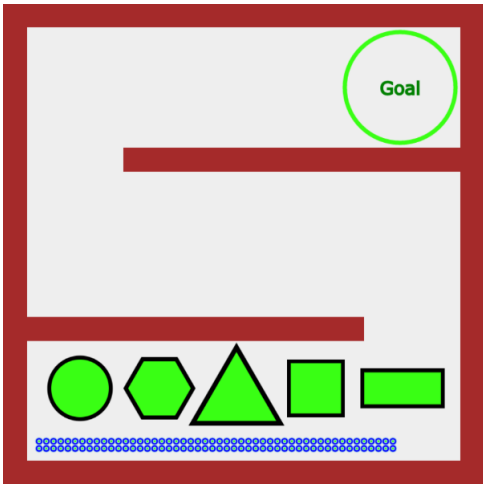


Figure 2.14: The six equal-area objects tested in simulation.

assumed to be in a local minimum and is commanded to move toward the previous corner. As soon as the mean position changes, normal control resumes.

Algorithm 2 Object-manipulation controller for a particle swarm.

Require: Knowledge of moveable object's center of mass $\mathbf{b}$; swarm mean $[\bar{x}, \bar{y}]$ and variance $[\sigma_x^2, \sigma_y^2]$, each calculated using the regions function from §2.6.3; map of the environment

Compute optimal policy for object, according to §2.6.1
while $\mathbf{b}$ is not in goal region **do**
  $\sigma^2 \leftarrow \max(\sigma_x, \sigma_y)$
  if $\sigma^2 > \sigma^2_{\max}$ **then**
    while $\sigma^2 > \sigma^2_{\min}$ **do**
      $[x_{\text{goal}}, y_{\text{goal}}] \leftarrow$ nearest corner in region
      Apply (3.31) to move toward $[x_{\text{goal}}, y_{\text{goal}}]$
    end while
  else
    Calculate $\mathbf{D}(\mathbf{b})$ ▷ direction for object at $\mathbf{b}$
    Apply (2.25) ▷ potential field for swarm
  end if
end while

Fig. 4.23 shows snapshots during an execution of this algorithm in simulation. To illustrate the flexibility of the algorithm we tested two additional workspaces, *E-shaped* and *Spiral*, without changing the algorithm. These are shown in Fig. 2.16. More complicated workspaces could be generated by composing these workspaces. Fig. 2.17

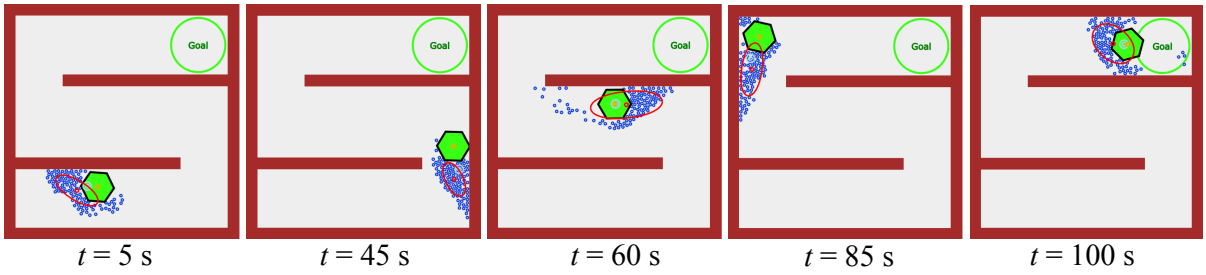


Figure 2.15: Snapshots showing an object manipulation simulation with 100 particles under automatic control (see also Extension 1).

shows the results of all three mazes. The E-shaped maze required the least average time because the path to the goal is shorter. Experimental results of parameters sweeps are summarized in Fig. 2.18. Each trial measured the time to deliver the object to the goal location. The default parameter settings used 100 particles, a normalized weight of 1, a hexagon shape, and Brownian noise (applied once each simulation step) with $W = 5$.

The interaction between the particles and object is impulsive so, like the study of impulsive pulling in [49], adding additional particles decreases completion time, but with diminishing returns. The effect of adding particles diminishes asymptotically because additional particles have difficulty interacting with the object. Brownian noise adds stochasticity. This randomness can break the object free if it is stuck, but diminishes the effect of the control input. Increasing noise increases completion time. The particles have limited force, so increasing the object weight increases completion time. Each shape was designed to have the same mass and area. Rectangles and squares tend to get stuck in the 90° workspace corners, and cause longer completion times than circles, triangles, and hexagons.

2.7 Object manipulation with hardware robots

Our experiments use centimeter-scale hardware systems called *kilobots*. While those are far larger than the micro scale devices we model, using kilobots allows us to emulate a variety of dynamics, while enabling a high degree of control over robot function, the

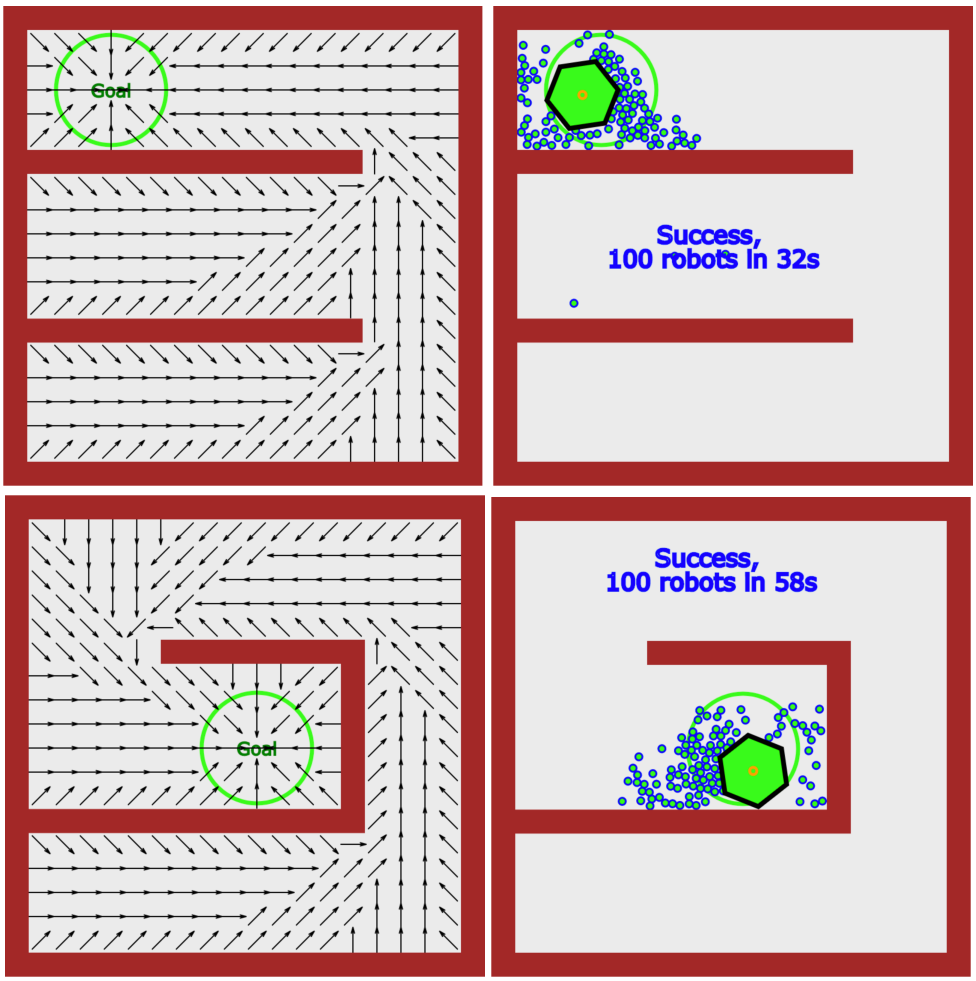


Figure 2.16: We tested three workspaces. The control policies from value iteration for an E-shaped and a spiral workspace are shown in the left column.

environment, and data collection. The kilobot is a nonholonomic, low-cost robot designed for testing collective algorithms with large numbers of robot [50, 51]. It is available as an open-source platform or commercially [52]. Each robot is approximately 3 cm in diameter, 3 cm tall, and uses two vibration motors to move on a flat surface at speeds up to 1 cm/s. Each robot has one ambient light sensor that is used to implement *phototaxis*, moving towards a light source.

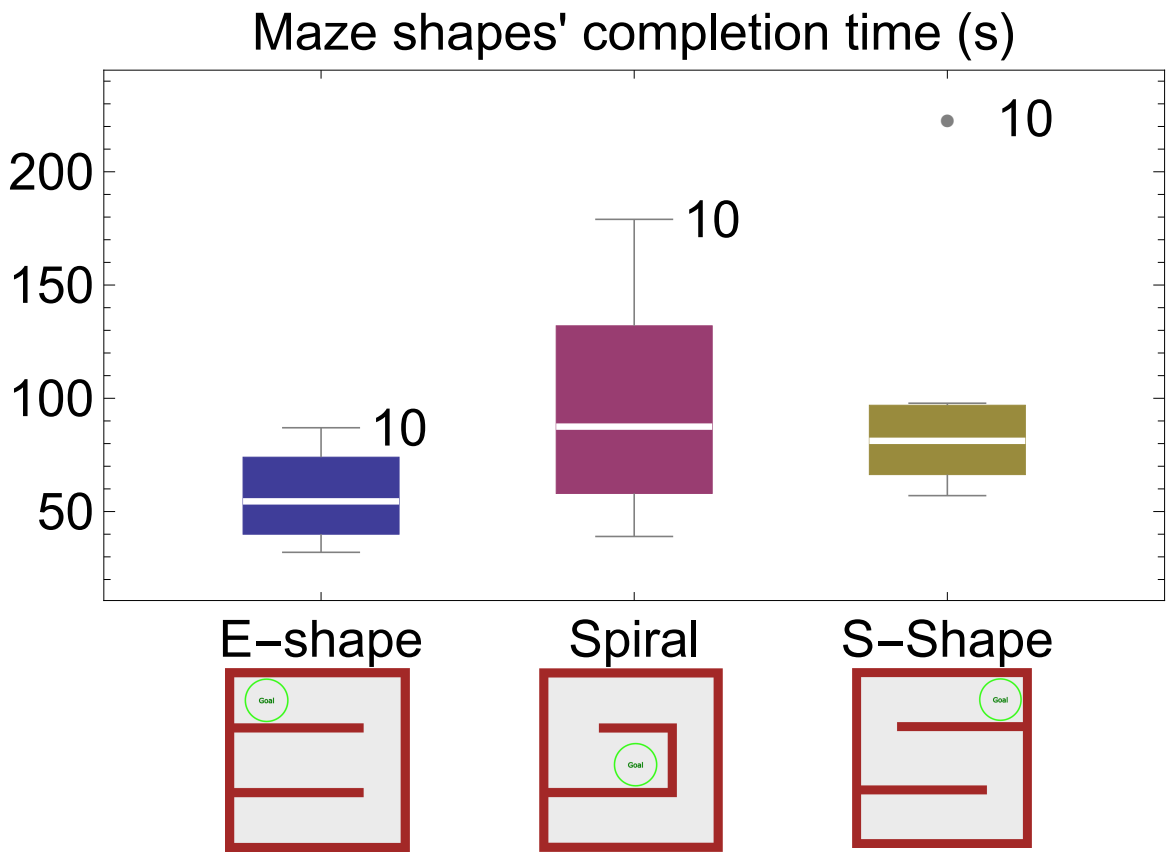


Figure 2.17: Completion times for three workspaces.

2.7.1 Environmental setup

In these experiments as shown in Fig. 3.9, we used n=100 kilobots and a 1.5 m×1.2 m whiteboard as the workspace. LED floodlights were placed 1.5 m above the table on the sides and corners of a square with 6 m sides. An Arduino Uno connected to an 8-relay shield controlled the lights.

Above the table, an overhead machine vision system tracks the swarm. The vision system identifies obstacles and the object by color segmentation, determines the corners of the maze, and identifies robots using a circular Hough transform.

The objects were 3D printed from ABS plastic with a paper overlay. Shapes included a 325 cm^2 equilateral triangle, 324 cm^2 square, 281 cm^2 hexagon, 254 cm^2 circle,

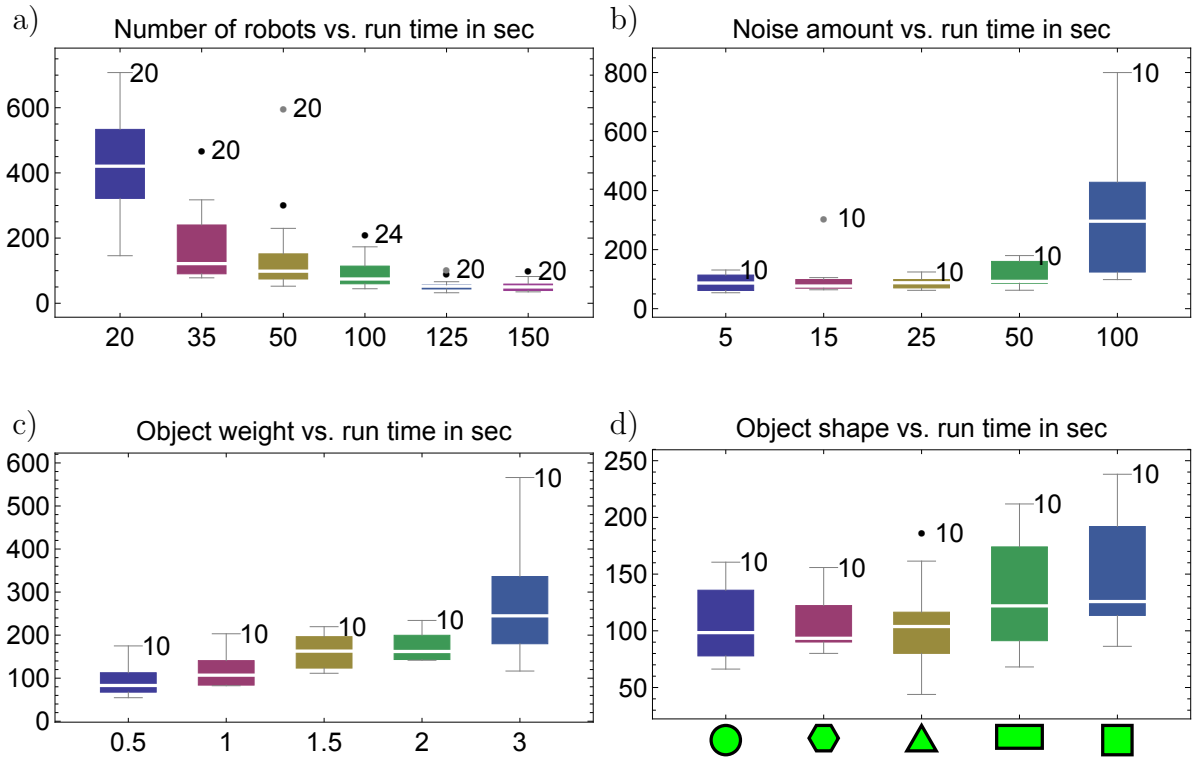
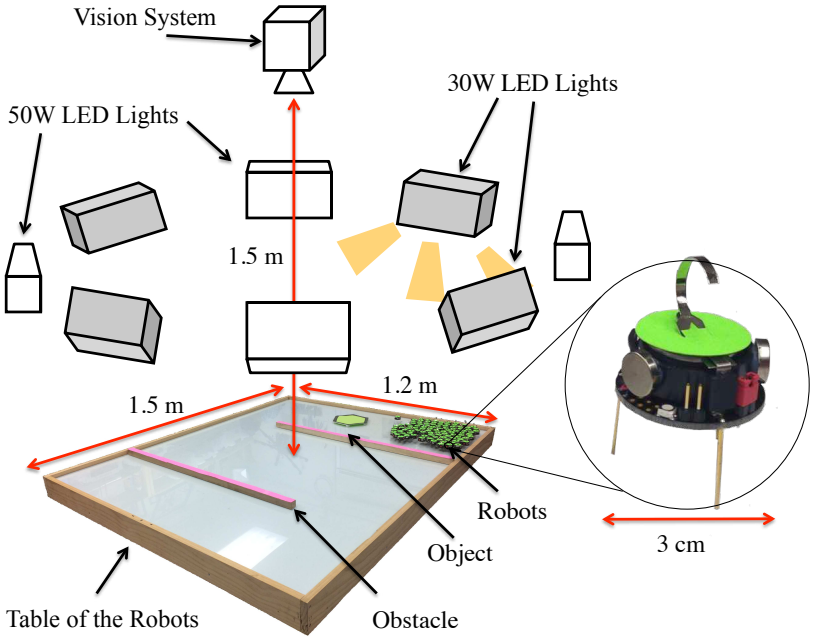


Figure 2.18: Parameter sweep simulation studies for a) number of particles, b) different noise values, c) object weight, and d) object shape. Each bar is labelled with the number of trials. Completion time is in seconds.

and a 486 cm^2 rectangle, all shown in Fig. 3.9. The laser-cut patterns for the neon green fiducial markers on the robots and 3D files for objects are available at our github repository [45].

Swarm mean control (hardware experiment) Unlike the PD controller (3.31), we cannot command a force input to the kilobots. Instead, control is given by turning on one of eight lights. The kilobots run a phototaxis routine where they search for an orientation that aligns them with the light source, and then move with an approximately constant velocity toward this light. The kilobots oscillate along this orientation because they only have one light detector.

We use the sign of (3.31), and choose the closest orientation to $\mathbf{D}(\mathbf{b})$ among the eight light sources. Fig. 2.20 shows that this limited, discretized control still enables



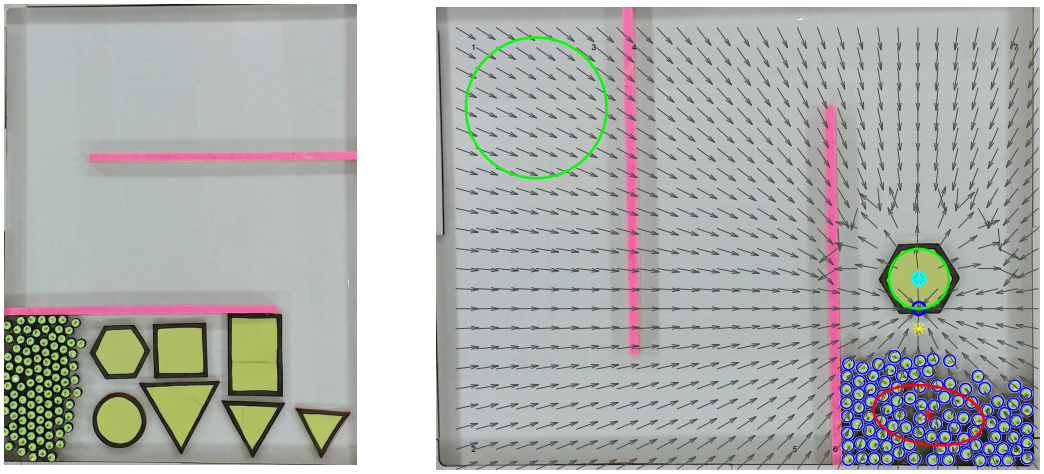


Figure 2.19: Hardware platform. At right are the shapes used for hardware experiments and a visualization of the potential field.

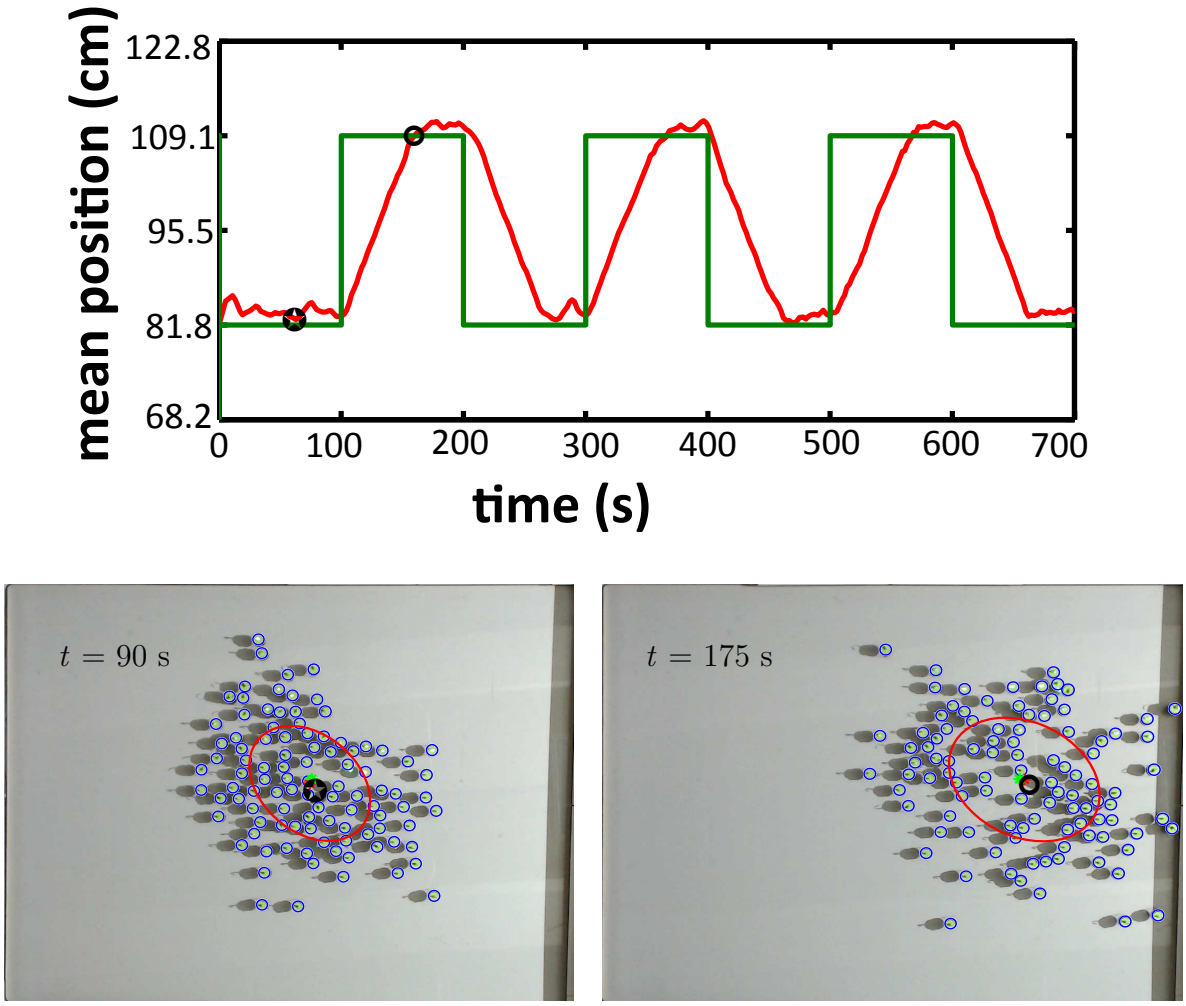


Figure 2.20: Regulating mean x position of 100 kilobots using control law (3.31).

regulating the mean position of a swarm of 100 robots.

2.7.2 Automated object manipulation (hardware experiment)

Even though kilobots are nonholonomic, they performed five successful runs manipulating a hexagonal object through an obstacle maze. Videos of these runs are in Extension 2. These hardware experiments represent the results of over 100 hours of trials. Each trial used 100 kilobots. Trials two through five were performed in a row with no failures in between. For each trial, fully charged kilobots were placed in the lower left-hand of the workspace, as shown in Fig. 2.21. The moveable object was placed in the

lower center of the workspace. MATLAB code for vision processing, the value iteration of Sec. 2.6.1 and the algorithm of Sec. 2.6.4 is available on MATLAB Central at [53]. Trials were run until the object COM entered the goal region. The trials ran for {1465, 3457, 3000, 2162, 2707} s. This is 2558 ± 771 s (mean$\pm$std).

We also tested other object shapes. A circular object completed in 3155 s. A square object completed in 6871 s. A rectangle and three equilateral triangle objects of varying sizes failed in a total of nine runs. Manipulation failures occurred when the object was pushed into a corner, requiring torque to be unstuck. Swarm torque control is the subject of our ongoing research begun in [54].

2.8 Conclusion

The small size of micro and nano particles makes individual control and autonomy challenging, so currently these particles are steered by global control inputs such as magnetic fields or chemical gradients. To investigate this control challenge, this chapter introduced SwarmControl.net, an open-source tool for large-scale user experiments where human users steer swarms of robots to accomplish tasks. Analysis of the gameplay results revealed benefits of measuring and controlling statistics of the swarm rather than full state feedback, robustness to IID noise, and small effects of varying population size of large swarms.

Inspired by the three lessons from SwarmControl.net, this paper designed controllers and controllability results using only the mean and variance of a particle swarm. We developed a hysteresis-based controller to regulate the position and variance of a swarm. We designed a controller for object manipulation using value iteration for path planning, regions for outlier rejection, and potential fields for minimizing moving the object backwards. All automatic controllers were implemented using 100 kilobots steered by the direction of a global light source. These experiments culminated in an object

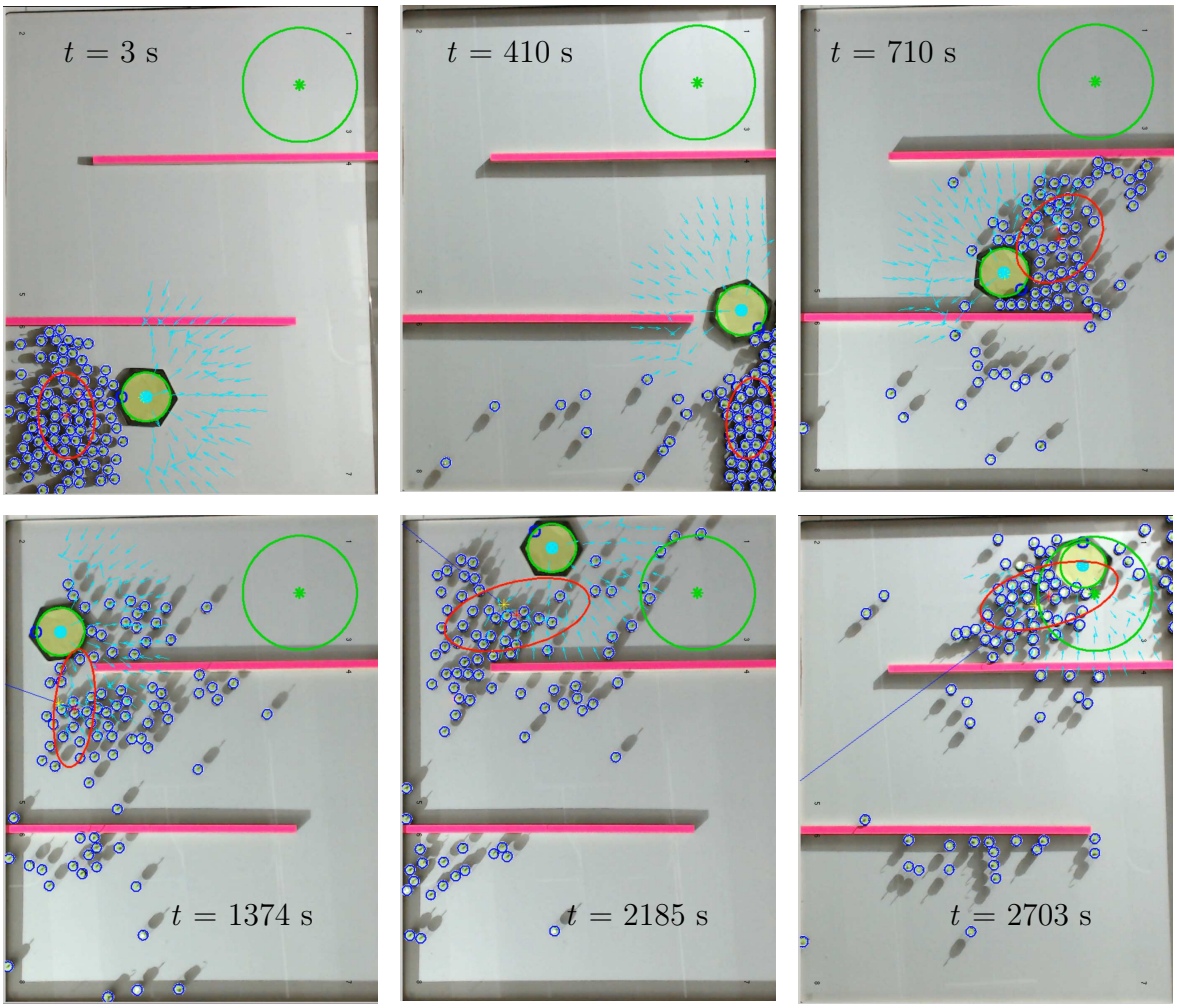


Figure 2.21: Snapshots showing object manipulation experiment with 100 kilobots under automatic control. The automatic controller generates a policy to the goal, (see Extension 2).

manipulation task in a workspace with obstacles.

Our goal in the next chapter is to apply force and torque to components and manipulating them through obstacles and each other. This work provides foundational algorithms and techniques for steering swarms, object manipulation, and addressing obstacle fields.

Chapter 3

Object Manipulation and Position Control Using a Swarm With Global Inputs

3.1 Introduction

This chapter investigates maximizing torque applied by a large number of particles, hereafter called a *swarm*, when the swarm has non-slip contact with a rigid, 2D body. The under-actuated swarm is steered by a shared signal that consists of a vector direction for movement. The robotic system is comprised of the swarm of particles, the shared control signal, and an external sensor that measures the swarm position. This chapter examines analytically two representative aspects of swarm torque control: first, pushing a pivoted rod, and second pushing a free body. We conclude with hardware experiments with centimeter-scale robots. Maximizing torque improves the efficiency of a particle swarm.

With a single agent, torque control is straightforward: the agent simply maximizes the length of the movement arm to maximize torque. To make an agent push open a door, it should push on the edge furthest from the hinge. The optimal solution for a swarm of particles is not straightforward because they cannot all push at one position.

This chapter focuses on maximizing torque using the swarm's position distribution. Representative results are shown in Fig. 3.1: torque control using 56 mobile robots on a pivoted and a free object.

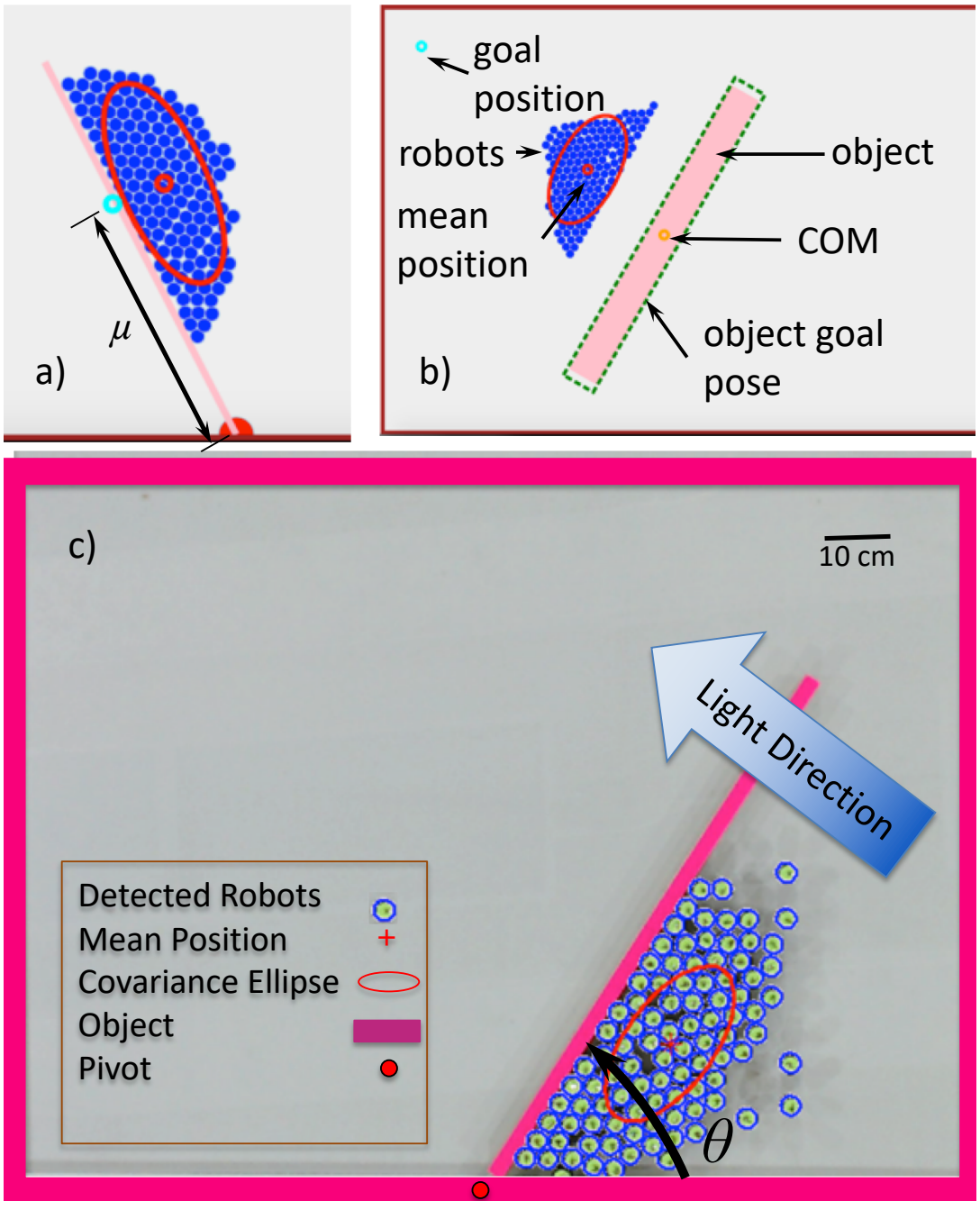


Figure 3.1: (a) Simulation of robots exerting torque on a hinged "door". (b) Orientation control of a free long rod. (c) Hardware robots applying torque to an object. See video attachment.

3.2 Related work

Unlike *caging* manipulation, where robots form a rigid arrangement around an object [32, 55], our swarm of particles is unable to grasp the blocks they push, and so manipulation requires *nonprehensile manipulation* techniques, e.g. [34].

Robotic manipulation by pushing has a long and successful history [34,56–58]. Key developments introduced the notion of stable pushes and a friction cone. A *stable push* is a pushing operation by a robot with a flat-plate pushing element in which the object does not change orientation relative to the pushing robot [34]. The *friction cone* is the set of vector directions a robot in contact with an object can push that object with a stable push. Stable pushes can be used as primitives in a rapidly-expanding random tree to form motion plans. A key difference is that our robots are compliant and tend to flow around the object, with similarities to fluidic trapping [35, 36].

Though some particles self aggregate, e.g. ferrous particles tend to clump in a magnetic field, many do not. The magnetotactic bacteria of [59, 60] are directed by the orientation of the magnetic field and do not suffer from magnetic aggregation.

Controlling the *shape*, or relative positions, of a swarm of robots is a key ability for a myriad of applications. Correspondingly, it has been studied from a control-theoretic perspective in both centralized, e.g. virtual leaders in [61], and decentralized approaches, e.g. decentralized control-Lyapunov function controllers in [62]. Most approaches assume a level of intelligence and autonomy in the individual robots that exceeds the capabilities of current micro- and nano-robots [59, 63]. Instead, this paper focuses on a centralized technique that applies the same control input to each member of the swarm, as in [9].

3.3 Torque control

We derive inspiration from recent work on pulling with a swarm [64]. The contribution of this work is to map swarm distributions to torque production. To change the output torque τ in (3.1), we can choose the direction and magnitude of the force applied F, and the moment arm from the object's center of mass (COM) O to the point of contact P. We define a coordinate frame rooted at the COM, with the x-axis parallel to the object's longest axis. The resulting torque is:

$$\tau = F \times (P - O). \tag{3.1}$$

We assume that the forces produced by individual particles add linearly. As [64] indicates, this is often a simplification of the true dynamics. The swarm version of (3.1) is the summation of the forces contributed by n individual particles:

$$\tau_{\text{total}} = \sum_{i=1}^{n} \rho_i F_i \times (P_i - O), \tag{3.2}$$

$$F_{\text{total}} = \sum_{i=1}^{n} \rho_i F_i. \tag{3.3}$$

Here F_i is the force that the i^{th} particle applies. Not all particles are in contact with the object. ρ_i is an indicator variable: $\rho_i = 1$ if the particle is in direct contact with the object or touching a chain of particle where at least one particle is in contact with the object. Otherwise $\rho_i = 0$. The moment arm is the particle's position P_i to the object's COM $O = [O_x, O_y]^\top$. If all particles are identical and the control input is uniform, the force is equivalent for every particle and so F_i equals some constant.

3.4 Instantiating torque from swarm distribution

Consider a swarm of particles whose marginal distribution along x has probability density $p(x)$, where x is defined as perpendicular to the object's long axis. This section

Table 3.1: Main results from Section 3.4 for maximizing torque with three common distributions.

Distribution	The best μ to push			Maximum possible torque		
Pivoted Uniform	$\mu_{u_{\max}} =$	$1-\sqrt{3}\sigma$	for $\sigma < \frac{1}{2\sqrt{3}}$	$\tau_{u_{\max}} =$	$1-\sqrt{3}\sigma$	for $\sigma \le \frac{1}{2\sqrt{3}}$
		$1-\sqrt{3}\sigma < \mu < \sqrt{3}\sigma$	for $\sigma > \frac{1}{2\sqrt{3}}$		$\frac{1}{4\sqrt{3}\sigma}$	for $\sigma > \frac{1}{2\sqrt{3}}$
Pivoted Triangular	$\mu_{t_{\max}} =$	$\sqrt{12\sigma^2+1}-\sqrt{6}\sigma$	for $0 < \sigma < \frac{1}{2\sqrt{3}}$	$\tau_{t_{\max}} =$	$\frac{(1+12\sigma^2)^{\frac{3}{2}}-1}{18\sigma^2} - \sqrt{6}\sigma$	for $\sigma \le \frac{1}{2\sqrt{3}}$
		$\frac{\sqrt{2}}{2}$	for $\sigma \ge \frac{1}{2\sqrt{3}}$		$\frac{\sqrt{2}-2+3\sqrt{6}\sigma}{36\sigma^2}$	for $\sigma > \frac{1}{2\sqrt{3}}$
Free Uniform	$\mu_{u_{f_{\max}}} =$	$1-\sqrt{3}\sigma$	for $\sigma < \frac{1}{2\sqrt{3}}$	$\tau_{u_{f_{\max}}} =$	$1-\sqrt{3}\sigma$	for $\sigma \le \frac{1}{2\sqrt{3}}$
		$\sqrt{3}\sigma$	for $\sigma > \frac{1}{2\sqrt{3}}$		$\frac{1}{4\sqrt{3}\sigma}$	for $\sigma > \frac{1}{2\sqrt{3}}$
Free Triangular	$\mu_{t_{f_{\max}}} =$	$\sqrt{12\sigma^2+1}-\sqrt{6}\sigma$	for $\sigma < \frac{2}{\sqrt{6}}$	$\tau_{t_{f_{\max}}} =$	$\frac{(1+12\sigma^2)^{\frac{3}{2}}-1}{18\sigma^2} - \sqrt{6}\sigma$	for $\sigma < \frac{2}{\sqrt{6}}$
		$1 < \mu < \sqrt{12\sigma^2+1}-\sqrt{6}\sigma$	for $\sigma \ge \frac{2}{\sqrt{6}}$		$\frac{1}{9\sigma^2}$	for $\sigma \ge \frac{2}{\sqrt{6}}$

considers three canonical probability distributions: uniform, triangular, and normal, all parameterized by mean μ and standard deviation σ. They are plotted in Fig. 3.2 and described by:

$$p_u(x) = \begin{cases} \frac{1}{2\sqrt{3}\sigma}, & \text{for } \mu - \sqrt{3}\sigma \le x \le \mu + \sqrt{3}\sigma \\ 0, & \text{otherwise} \end{cases}, \tag{3.4}$$

$$p_t(x) = \begin{cases} \frac{x-\mu+\sqrt{6}\sigma}{6\sigma^2}, & \text{for } \mu - \sqrt{6}\sigma \le x \le \mu \\ \frac{-x+\mu+\sqrt{6}\sigma}{6\sigma^2}, & \text{for } \mu < x \le \mu + \sqrt{6}\sigma \\ 0, & \text{otherwise} \end{cases}, \tag{3.5}$$

$$p_n(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}. \tag{3.6}$$

The next sections examine where to steer the mean of the probability distribution to maximize torque. We discuss two problems: pivoted object torque and free object torque. All the results are summarized in Table 3.1, and the calculations are included in the Mathematica file in the multimedia attachment.

3.4.1 Pivoted object torque

In this problem, the torque applied to a rod of length 1 pivoted at 0 when $\theta = 0$ is

$$\tau_p = \int_0^1 x\, p(x)\, dx. \tag{3.7}$$

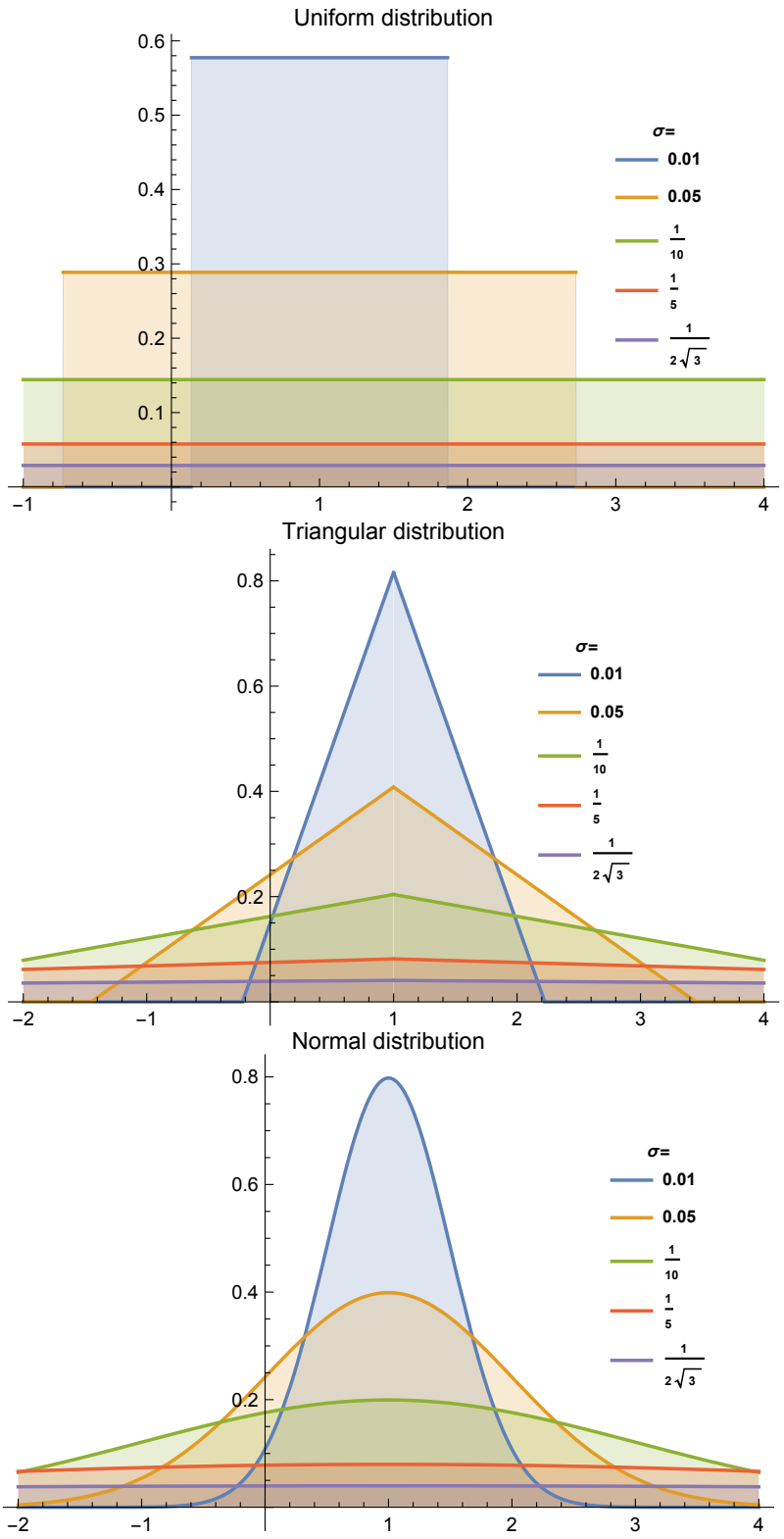


Figure 3.2: Three distributions are examined in this work: uniform, triangular, and normal. Each is plotted above with $\mu = 1$ for representative σ values.

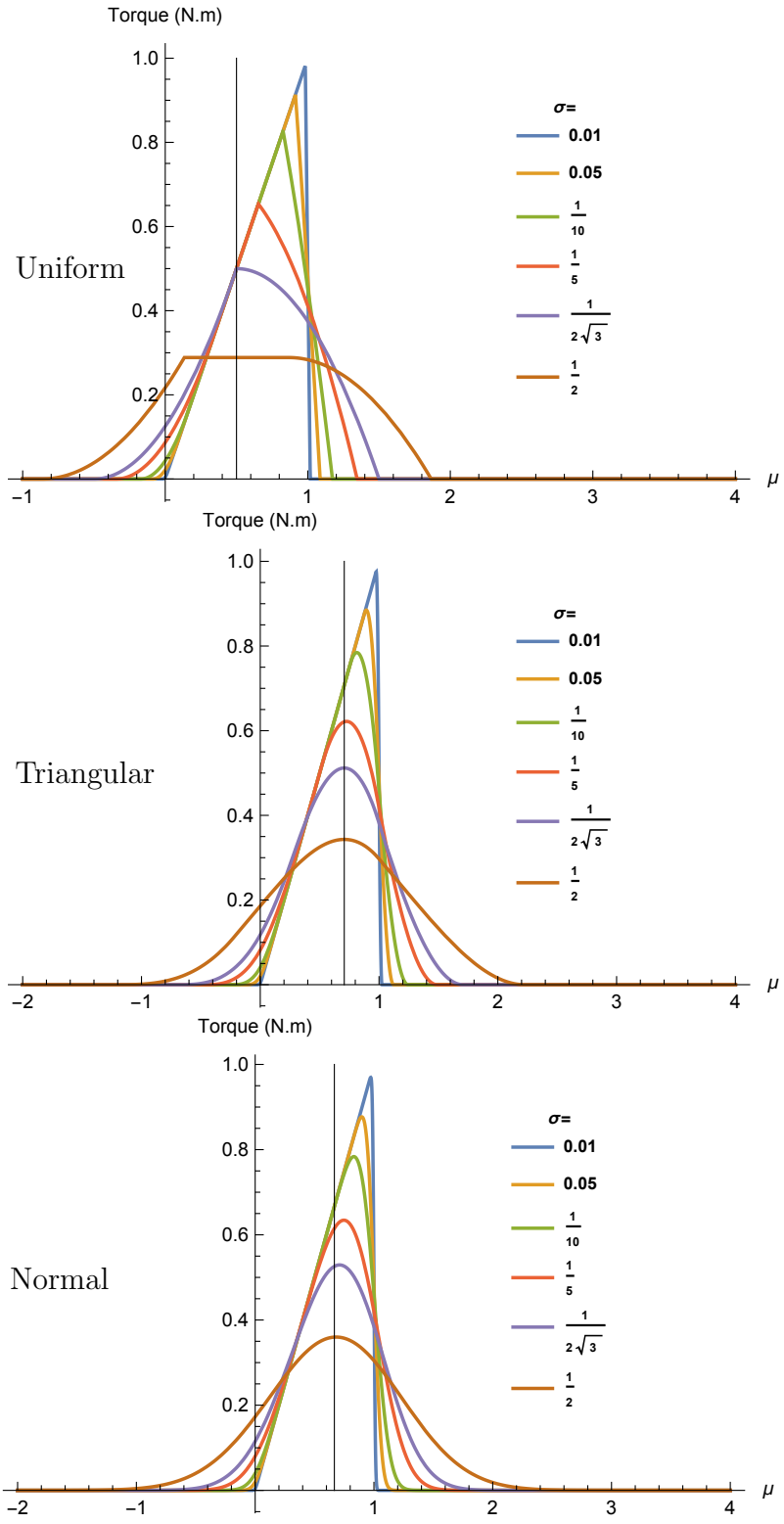


Figure 3.3: Torque as a function of mean position μ. Mean position is the pushing location along a rod extending from 0 to 1. For all distributions pushing at $\mu = 1$ is not optimal unless $\sigma = 0$ or the swarm is uniformly distributed with $\sigma > \frac{1}{2\sqrt{3}}$.

Defining torque for distributions First, for simplicity of the following derivations, the lower l and upper bound u are defined for the uniform distribution as

$$l = \max(0, \mu - \sqrt{3}\sigma)\,, u = \min(1, \mu + \sqrt{3}\sigma). \tag{3.8}$$

Torque by a uniformly distributed swarm is

$$\tau_{u_p} = \frac{u^2 - l^2}{4\sqrt{3}\sigma}. \tag{3.9}$$

To simplify derivations for the triangular distribution, we define the bounds of integration as

$$l_1 = \max(0, \mu - \sqrt{6}\sigma)\,, l_2 = \max(0, \mu), \tag{3.10}$$

$$u_1 = \min(1, \mu)\,, u_2 = \min(1, \mu + \sqrt{6}\sigma).$$

Torque by a triangularly distributed swarm is

$$\tau_{t_p} = \frac{\tau_l + \tau_r}{36\sigma^2}, \tag{3.11}$$

where τ_l and τ_r are defined as:

$$\begin{aligned} \tau_l &= {l_1}^2(2l_1 - 3(\mu - \sqrt{6}\sigma)) \\ &\quad + {u_1}^2(2u_1 - 3(\mu - \sqrt{6}\sigma)), \\ \tau_r &= {l_2}^2(2l_2 - 3(\mu + \sqrt{6}\sigma)) \\ &\quad - {u_2}^2(2u_2 - 3(\mu + \sqrt{6}\sigma)). \end{aligned} \tag{3.12}$$

Torque by a normally distributed swarm is:

$$\begin{aligned} \tau_{n_p} = &\frac{\left(e^{\frac{\mu^2}{2\sigma^2}} - e^{\frac{(1-\mu)^2}{2\sigma^2}}\right)\sigma}{\sqrt{2\pi}} \\ &+ \frac{\mu}{2}\left(\operatorname{erf}\left(\frac{1-\mu}{\sqrt{2}\sigma}\right) + \operatorname{erf}\left(\frac{\mu}{\sqrt{2}\sigma}\right)\right). \end{aligned} \tag{3.13}$$

The $\operatorname{erf}(\cdot)$ is the error function. Torque is plotted as a function of μ for representative σ values in Fig. 3.3.

Maximum torque for distributions For a uniformly distributed swarm, the mean position that maximizes torque on a pivoted object is:

$$\mu_{u_{\max}} = \begin{cases} 1 - \sqrt{3}\sigma & \text{for } \sigma < \frac{1}{2\sqrt{3}} \\ 1 - \sqrt{3}\sigma < \mu < \sqrt{3}\sigma & \text{for } \sigma > \frac{1}{2\sqrt{3}} \end{cases} . \tag{3.14}$$

As soon as the width of the uniformly distributed swarm is longer than the rod ($\sigma > \frac{1}{2\sqrt{3}}$), there exists a range of optimal solutions: any μ such that the swarm covers 0 to 1.

The resulting torque is:

$$\tau_{u_{\max}} = \begin{cases} 1 - \sqrt{3}\sigma & \text{for } \sigma \leq \frac{1}{2\sqrt{3}} \\ \frac{1}{4\sqrt{3}\sigma} & \text{for } \sigma > \frac{1}{2\sqrt{3}} \end{cases} . \tag{3.15}$$

For a triangularly distributed swarm, the mean position that maximizes torque is:

$$\mu_{t_{\max}} = \begin{cases} \sqrt{12\sigma^2 + 1} - \sqrt{6}\sigma & \text{for } 0 < \sigma < \frac{1}{2\sqrt{3}} \\ \frac{\sqrt{2}}{2} & \text{for } \sigma \geq \frac{1}{2\sqrt{3}} \end{cases} . \tag{3.16}$$

Thus the torque is:

$$\tau_{t_{\max}} = \begin{cases} \frac{(1+12\sigma^2)^{\frac{3}{2}} - 1}{18\sigma^2} - \sqrt{6}\sigma & \text{for } \sigma \leq \frac{1}{2\sqrt{3}} \\ \frac{\sqrt{2} - 2 + 3\sqrt{6}\sigma}{36\sigma^2} & \sigma > \frac{1}{2\sqrt{3}} \end{cases} . \tag{3.17}$$

For all distributions, the μ that maximizes torque if $\sigma = 0$ is 1. For the τ_p case, the optimal μ moves in the $-x$ direction, and for some distributions approaches a limit as σ increases. This limit is $\mu = \frac{\sqrt{2}}{2}$ for a triangularly distributed swarm. In a normal distribution, the first derivative of (3.13) is:

$$\frac{d_\tau}{d_\mu} = \frac{1}{2}\left(\operatorname{erf}\frac{1-\mu}{\sqrt{2}\sigma} + \operatorname{erf}\frac{\mu}{\sqrt{2}\sigma}\right) - \frac{e^{\frac{-(\mu-1)^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma}. \tag{3.18}$$

The limit as $\sigma \to \infty$ is $\frac{2-3\mu}{\sigma^3 6\sqrt{2\pi}}$ which goes to zero as the swarm variance increases. However, the zero of the numerator is $\mu = \frac{2}{3}$ which means as σ increases the best position to push the rod asymptotically reaches $\mu = \frac{2}{3}$.

The central limit theorem explains why systems where all the particles have independent noise form a normal distribution. However, optimizing torque for a normal distribution involves two error functions (erf). Instead, we have solved all the equations for the triangular distribution. The numerical study illustrated in the left plot of Fig. 3.4 shows the triangular and normal distributions perform similarly.

3.4.2 Free object torque

In this problem, the torque applied to a free rod of length 2 from $[-1, 0]$ to $[1, 0]$, is:

$$\tau_f = \int_{-1}^{1} x\, p(x)\, dx. \tag{3.19}$$

a) Defining torque to a free rod for distributions:

To calculate free object torque for a uniformly distributed swarm, for simplicity of derivations, l_f and u_f are defined:

$$l_f = \max(-1, \mu - \sqrt{3}\sigma),\ u_f = \min(1, \mu + \sqrt{3}\sigma). \tag{3.20}$$

The torque applied by a uniformly distributed swarm is:

$$\tau_{u_f} = \frac{u_f^2 - l_f^2}{4\sqrt{3}\sigma} \text{ for } u_f > l_f. \tag{3.21}$$

To simplify derivations for a triangularly distributed swarm, we define bounds of integration as:

$$l_{f_1} = \max(-1, \mu - \sqrt{6}\sigma),\ l_{f_2} = \max(-1, \mu), \tag{3.22}$$

$$u_{f_1} = \min(1, \mu),\ u_{f_2} = \min(1, \mu + \sqrt{6}\sigma).$$

The torque applied by a triangularly distributed swarm is:

$$\tau_{t_f} = \frac{\tau_{f_l} + \tau_{f_r}}{36\sigma^2}, \tag{3.23}$$

where τ_{f_l} and τ_{f_r} is defined as

$$\tau_{f_l} = {u_{f_1}}^2(2u_{f_1} - 3(\mu - \sqrt{6}\sigma))$$

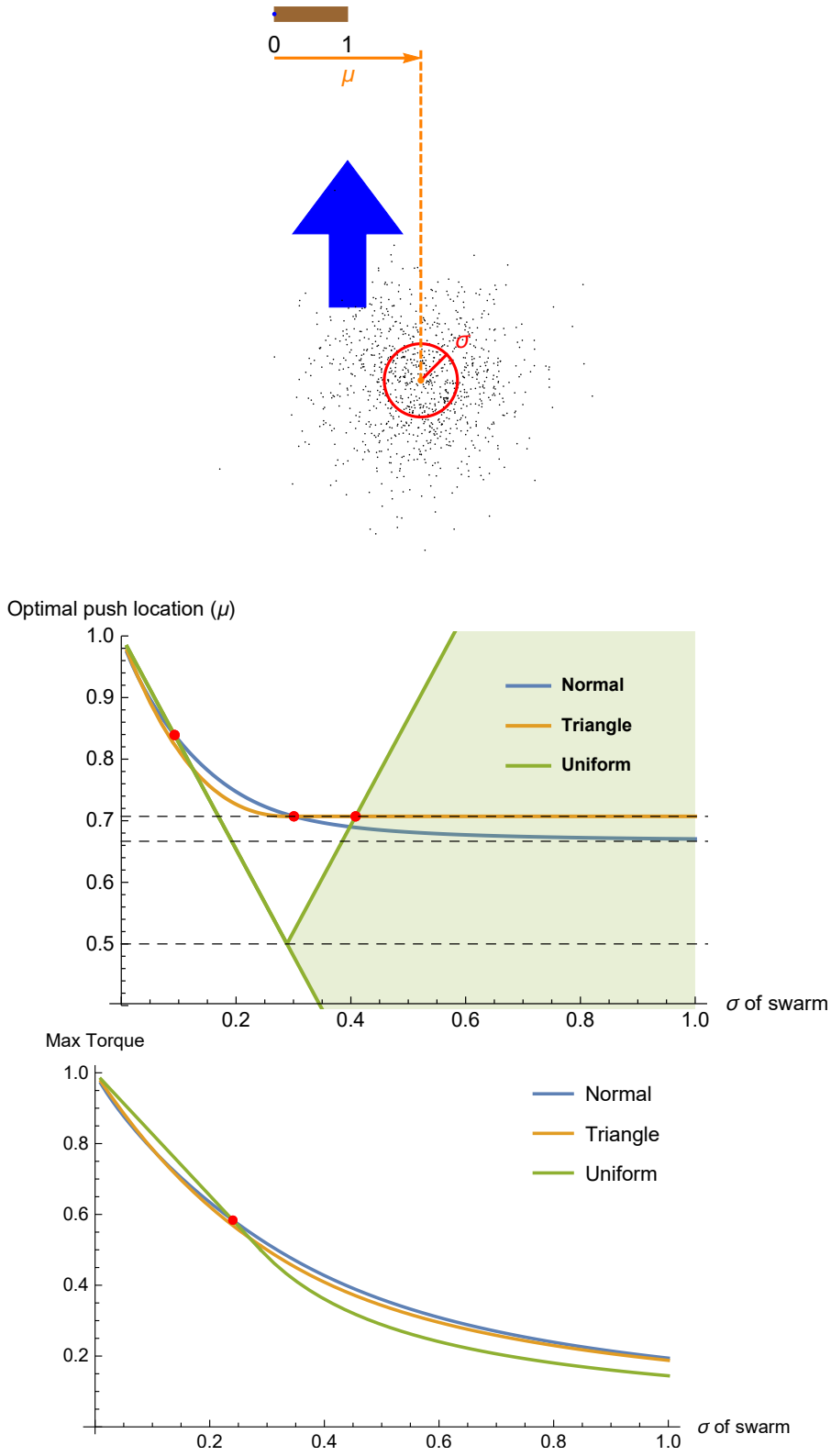


Figure 3.4: Best location to push and maximum torque plots for a pivoted object of length 1, pivoted at 0. Generating code is in the attachment.

$$- {l_{f_1}}^2(2l_{f_1} - 3(\mu - \sqrt{6}\sigma)),$$

$$\tau_{fr} = -{u_{f_2}}^2(2u_{f_2} - 3(\mu + \sqrt{6}\sigma),$$

$$+ {l_{f_2}}^2(2l_{f_2} - 3(\mu + \sqrt{6}\sigma)).$$

The torque applied by a normally distributed swarm is:

$$\tau_{n_f} = \frac{\left(e^{\frac{-(\mu+1)^2}{2\sigma^2}} - e^{\frac{-(\mu-1)^2}{2\sigma^2}}\right)\sigma}{\sqrt{2\pi}} + \frac{1}{2}\mu\left(\text{erf}\left(\frac{1-\mu}{\sqrt{2}\sigma}\right) + \text{erf}\left(\frac{1+\mu}{\sqrt{2}\sigma}\right)\right). \tag{3.24}$$

b) Maximum torque for distributions Maximum torque of a uniformly distributed swarm to a free object has the same upper solution as with a pivoted object:

$$\mu_{u_{f\max}} = \begin{cases} 1 - \sqrt{3}\sigma & \text{for } \sigma < \frac{1}{2\sqrt{3}} \\ \sqrt{3}\sigma & \text{for } \sigma > \frac{1}{2\sqrt{3}} \end{cases}. \tag{3.25}$$

The major difference is the existence of only one solution for σ values for torque on a free object. The left bound of the distribution must never be less than 0, or torque applied left of the center will cancel torque to the right of the center. The maximum torque produced is given by (3.15).

For a triangularly distributed swarm the optimal mean is:

$$\mu_{t_{f\max}} = \begin{cases} \sqrt{12\sigma^2 + 1} - \sqrt{6}\sigma & \text{for } \sigma < \frac{2}{\sqrt{6}} \\ 1 < \mu < \sqrt{12\sigma^2 + 1} - \sqrt{6}\sigma & \text{for } \sigma \geq \frac{2}{\sqrt{6}} \end{cases}. \tag{3.26}$$

As with the uniform distribution in (3.14) the optimal solution with $\sigma > \frac{2}{\sqrt{6}}$ has a range of solutions. Setting $\mu = 1$ maximizes force for the optimal torque. However,

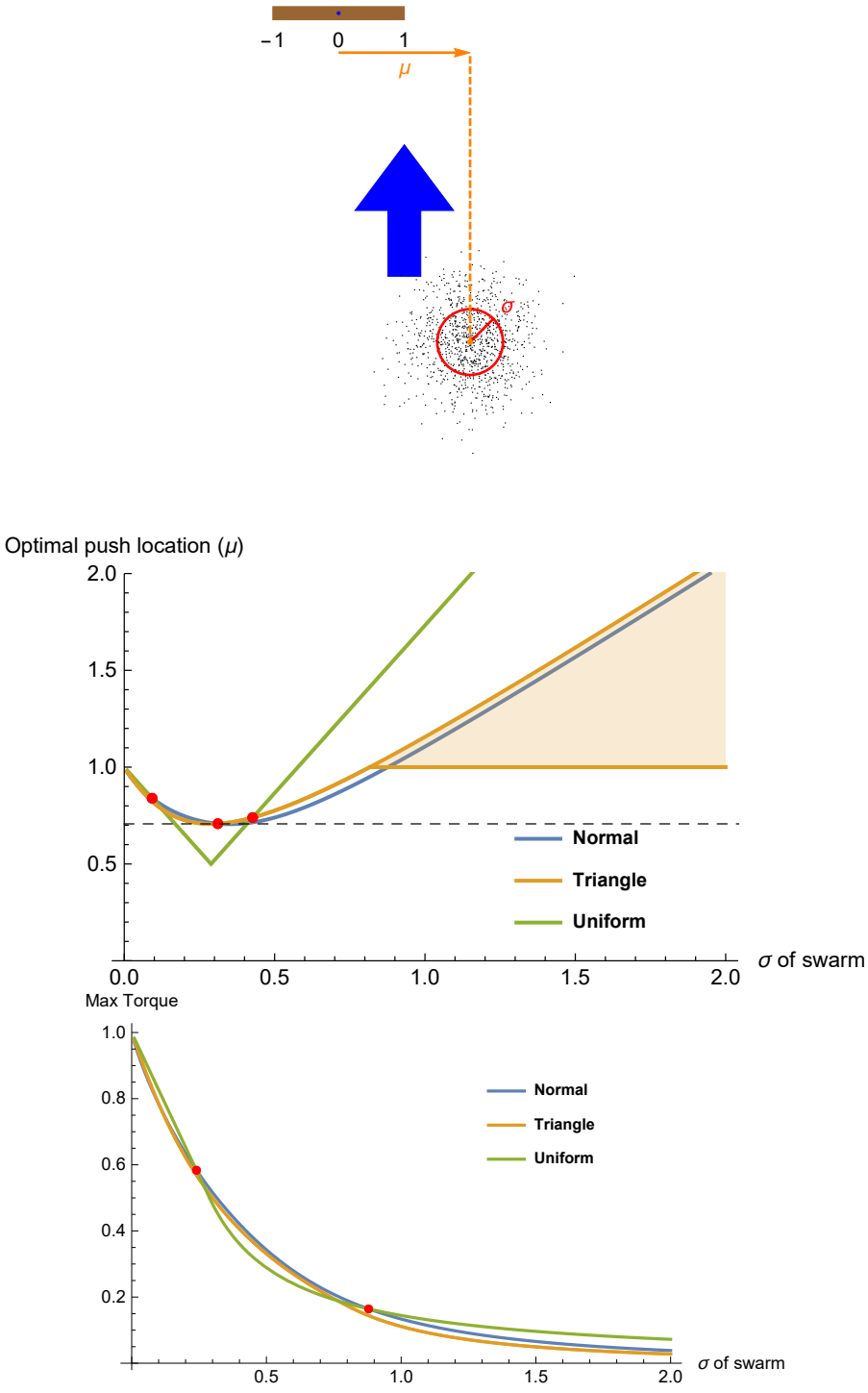


Figure 3.5: Best location to push and maximum torque plots for a free object of length 2, located from $x = -1$ to 1.

moving the mean right reduces the negative torque applied by particles to the left of 0, but this gain is canceled by a corresponding reduction in positive torque.

Therefore, the maximum torque is:

$$\tau_{t_{f_{\max}}} = \begin{cases} \frac{(1+12\sigma^2)^{\frac{3}{2}}-1}{18\sigma^2} - \sqrt{6}\sigma & \text{for } \sigma < \frac{2}{\sqrt{6}} \\ \frac{1}{9\sigma^2} & \text{for } \sigma \geq \frac{2}{\sqrt{6}} \end{cases} . \tag{3.27}$$

Again, for a normally distributed swarm, we have numerically found the maximum torque and optimal pushing location μ. The results are closely approximated by the upper bound solution for the triangular system. The results are plotted in Fig. 3.5.

3.5 Angle of repose

Consider a swarm of granular particles applying force to a rod. If the rod moves slower than the particles, these particles will build up behind the rod in a characteristic triangular shape defined by an apex angle. This piling up is common to all granular media, and the angle formed is a function of the *angle of repose*.By measuring the angle of repose for the particles shown in the top plot of Fig. 3.6, we can estimate the force and torque that the swarm is applying to the rod as a function of the rod's length, the angle of repose, and orientation of the rod. We define the angle of repose as α, the rod's orientation relative to 90° from the particle movement vector as θ, and the rod's length as ℓ. By integrating over the triangular shape, the force applied to the rod (when a unit area of particles produces 1 N of force) is:

$$F(\theta, \alpha, \ell) = \begin{cases} \frac{-\ell^2\Big(\cos(2\theta)-\cos(2\alpha)\Big)}{8\cos\alpha\sin\theta} & -\alpha < \theta < \alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.28}$$

The force for different angle of repose values are shown in the middle plot of Fig. 3.6.

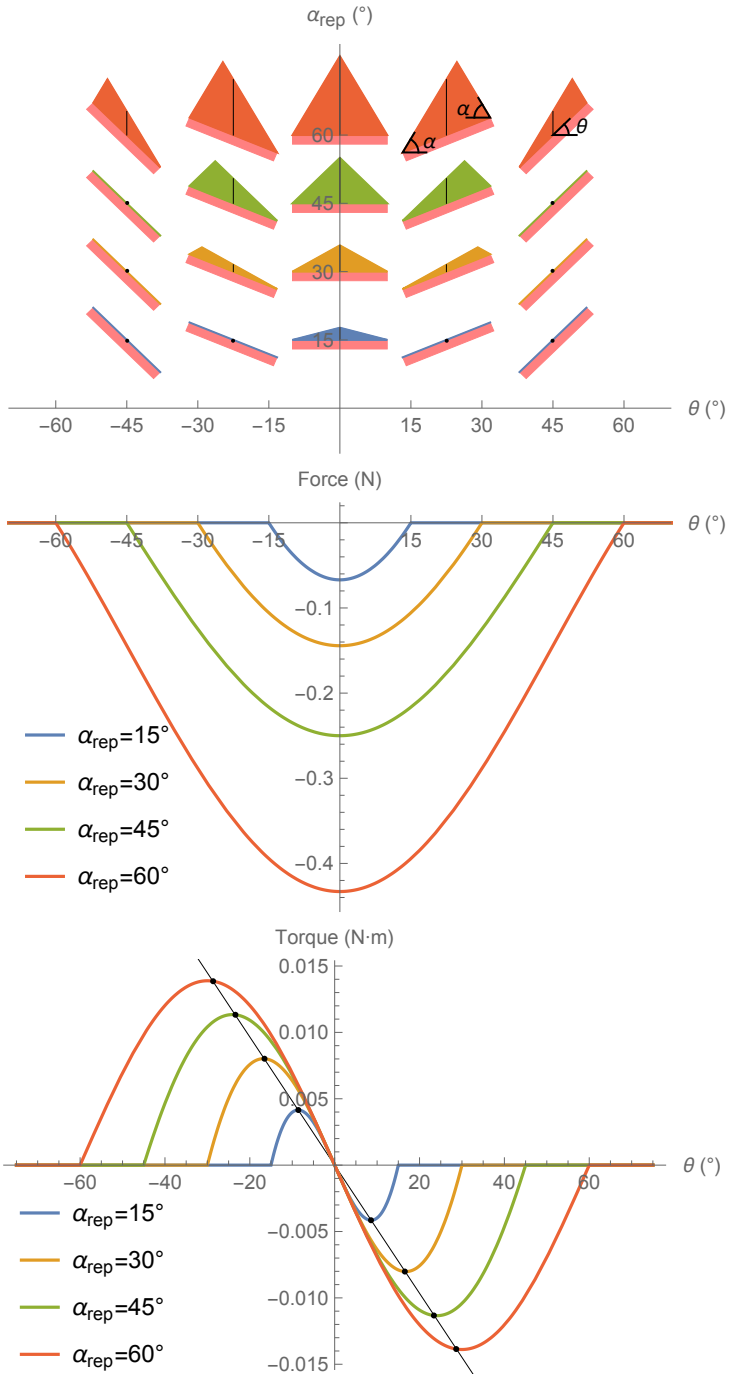


Figure 3.6: Top plot shows colored particulate heaped up on pink-colored long rods. Middle plot shows the force applied to the rod and bottom the torque as a function of θ for four angle of repose values. The maximum torque values are shown with black dots.

Torque will also be similarly defined as:

$$\tau(\theta, \alpha, \ell) = \begin{cases} \frac{\ell^3\left(\cos(2\alpha)-\cos(2\theta)\right)\sin\theta}{48\sin^2\alpha} & -\alpha < \theta < \alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.29}$$

Torque is shown in the bottom plot of Fig. 3.6. Given sufficient particles to pile up to the angle of repose, this torque tends to stabilize the object to be perpendicular to the pushing direction. Force is maximized with $\theta = 0$, but the θ value that maximizes torque is a function of α:

$$\theta_{t_{\max}} = \frac{\sin(\alpha)}{\sqrt{3}}. \tag{3.30}$$

To maximize the torque a particulate swarm applies on a thin rod, the swarm should move in the direction $-\theta_{t_{\max}} - 90^\circ$ with respect to the long axis of the rod.

3.6 Simulation

This section examines three challenges for torque control of an object, arranged in increasing difficulty. Each task uses a PD controller that regulates the swarm's mean position, as in [6]. The control input is the global force applied to each robot:

$$\begin{aligned} u_x &= K_p(goal_x - \bar{x}) + K_d(0 - \bar{v}_x) \\ u_y &= K_p(goal_y - \bar{y}) + K_d(0 - \bar{v}_y) \end{aligned} \tag{3.31}$$

here K_p is the proportional gain, and K_d is the derivative gain. The swarm's average position is $[\bar{x}, \bar{y}]^T$ and mean velocity is $[\bar{v}_x, \bar{v}_y]^T$. Each task uses a different algorithm to select the swarm's goal position $[goal_x, goal_y]^T$. The derivative gain K_d limits overshoot.

Maximizing torque on a pivoted body An object with a pivot point can rotate, but not translate. A door is an example. If there was only one robot touching the object, the robot should push at the point which maximizes the moment arm, at the extreme end of the object furthest from the pivot point. The optimal pushing location provides

the maximum force, because it maximizes the distance $r = \|P - O\|$ in (3.1). However, given a swarm of robots, maximizing r is no longer the optimal solution. If the mean of the swarm hits the object at the extreme edge, half of the robots will miss the object and the swarm will be split. Because few robots remain, the force is significantly decreased and torque is not maximized. In our simulation, the swarm applies torque until the swarm's mean position is beyond the object. Then the swarm will regather in a corner before returning to torque control, a time-consuming task. The key parameter of interest for a hinged door of length L is C, the position along the door where the mean of the swarm will push. The goal position in (3.32) is set to:

$$\begin{aligned} goal_x &= O_x + C\cos(O_\theta) \\ goal_y &= O_y + C\sin(O_\theta), \end{aligned} \tag{3.32}$$

O_θ = the orientation of the object's major axis, measured from the world x-axis. Fig. 3.7 illustrates how different values of C result in different rates of turning. These simulations tested $C = \{1/2, 3/4, 5/8, 1\}L$. The fastest turning rates occurred with $C = 5/8L$. Code is available at [65].

Orientation of the object These simulations used a uniform density rectangle as the object. This object was 30× larger than the robots. Using the pure torque control discussed in the previous paragraph, the orientation of the object can be controlled by applying force. The rectangular object is not pivoted, so it moves in addition to rotating. The swarm still may split into multiple components. We use the hysteresis variance control from [6] to gather the swarm when its variance grows too large. The following control law chooses a goal position to regulate the orientation of the object.

$$\begin{aligned} goal_x &= O_x + K_{\text{orient}}(O_\theta - goal_\theta)\cos(O_\theta) \\ goal_y &= O_y + K_{\text{orient}}(O_\theta - goal_\theta)\sin(O_\theta) \end{aligned} \tag{3.33}$$

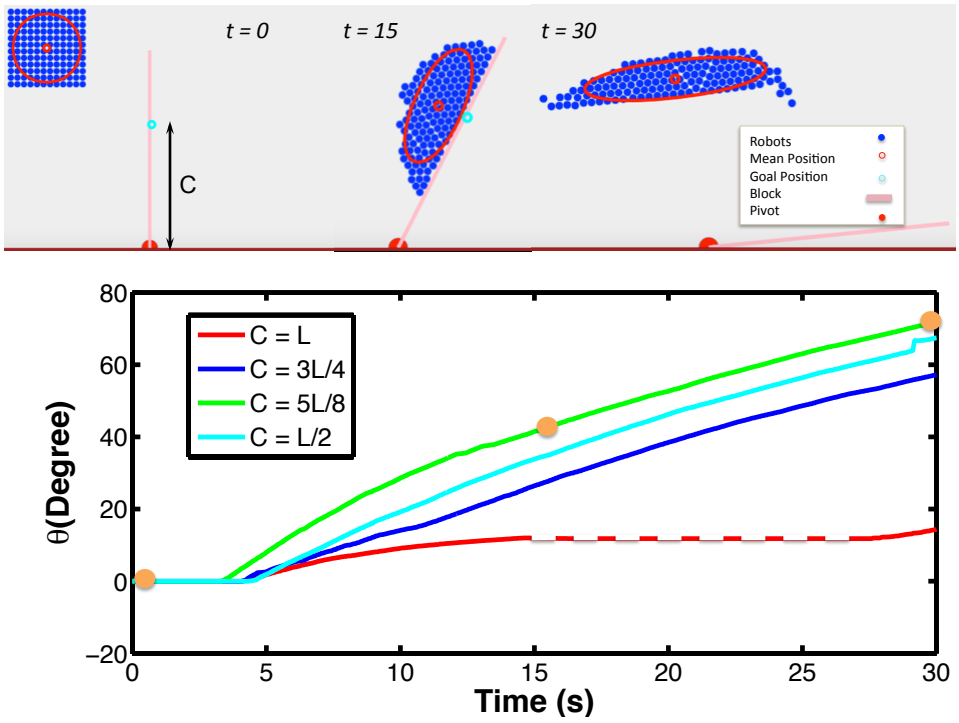


Figure 3.7: Simulation results from a swarm applying force to a hinged door. The swarm mean is steered toward a point C units along the object from the pivot point. The red dashed line indicates the times that the swarm was in variance control mode.

Here K_{orient} is a positive gain on the control input.

Fig. 3.8 illustrates this controller with different starting positions. When the plot traces are constant the swarm is no longer pushing the object and instead is being regathered in a corner of the workspace. Code is available at [66].

3.7 Experiments

The previous sections assumed perfect transmission of force from each particle to the object. Accurate swarm torque control is difficult. In 2D, a swarm of n particles has $2n$ degrees of freedom. The swarm, when steered toward an object, begins interacting with the object at different times. The number of particles touching this object as a function of time is difficult to predict and often impossible to directly measure. Stochastic effects make long-term prediction challenging. Even when it is possible to predict

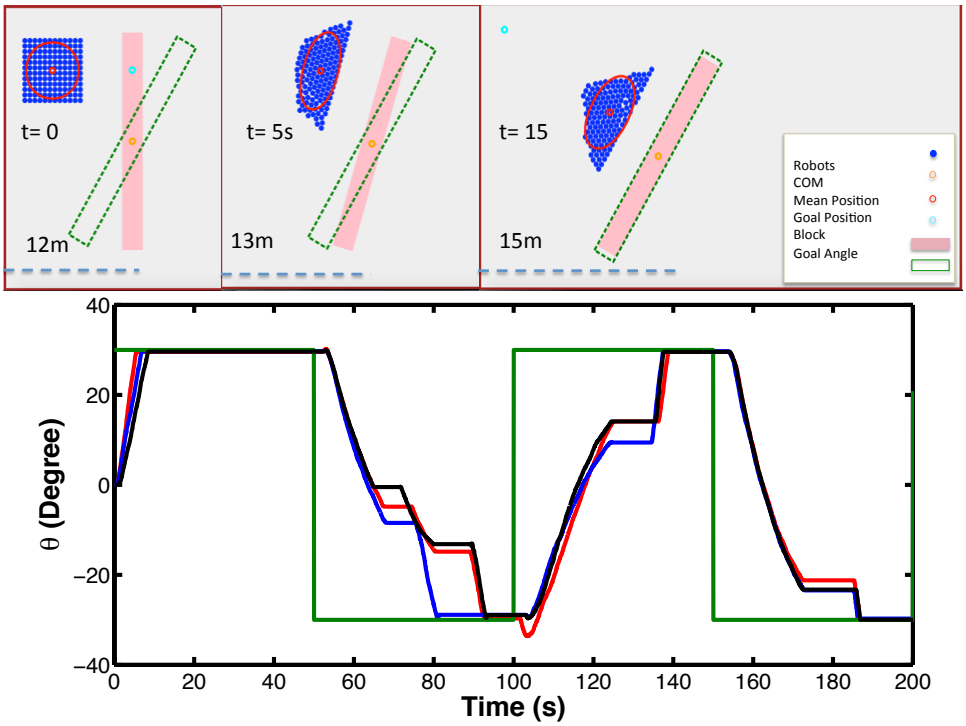


Figure 3.8: Plot demonstrating orientation control of a rectangular object. The green line is the goal orientation. Other lines show different random starting position of the swarm.

which particles will hit the object first, as particles interact with the object, the swarm's configuration changes. The challenge is not only limited to swarm-object interaction, but also to swarm-swarm interactions when the swarm self-collides or is split into multiple components. As a result, the force the swarm will exert on the object is not easy to predict. To demonstrate the analytical results from Section 3.4 hold, we performed experiments using centimeter-scale hardware robots called *Kilobots*.

These allow us to emulate a variety of dynamics, while enabling a high degree of control over robot function, the environment, and data collection. The Kilobot, from [50, 51], is a low-cost robot designed for testing collective algorithms with large numbers of robots. It is available in an open source platform or commercially from [52]. Each robot is approximately 3 cm in diameter, 3 cm tall, and uses two vibration motors to move on a flat surface at speeds up to 1 cm/s. Each robot has one ambient light sensor that is used to implement *phototaxis*, moving towards a light source.

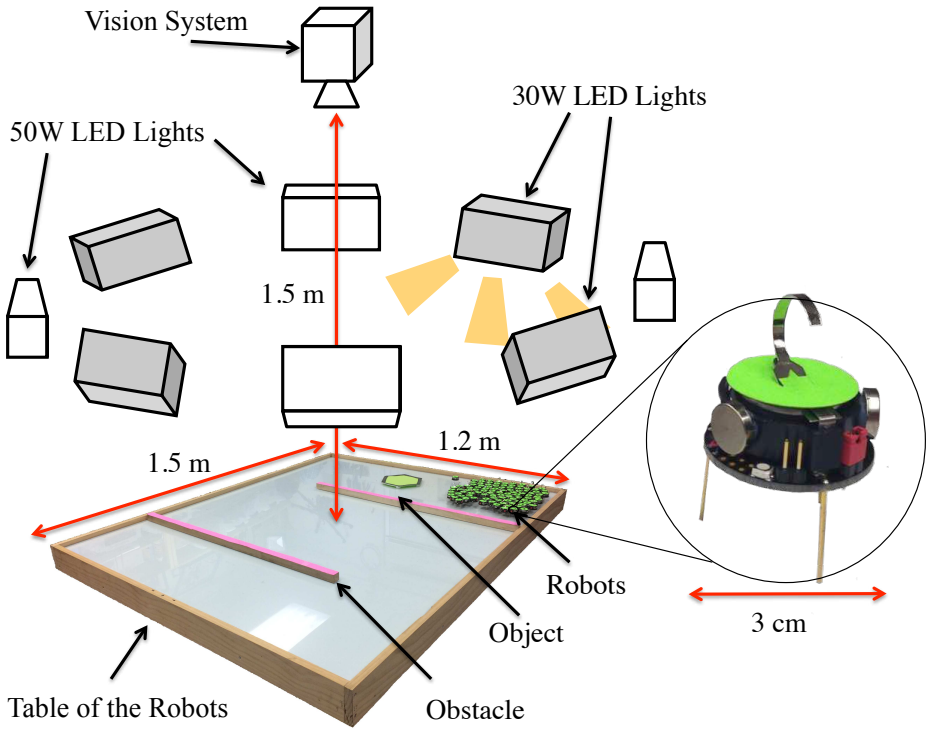


Figure 3.9: Hardware platform: table with 1.5×1.2 m workspace, surrounded by eight remotely triggered LED floodlights, with an overhead machine vision system.

These experiments used up to $n = 97$ Kilobots, a glass-covered 1.5 m×1.2 m whiteboard as the workspace, and 30 W and 50 W LED floodlights arranged 1.5 m above the plane of the table at the midpoint of the east side of a 6 m square centered on the workspace, as shown in Fig. 3.9. Above the table, an overhead machine vision system tracks the position of the swarm. For the last two experiments, we designed the object to rigidly hold the swarm, and programmed the robots to go straight.

3.7.1 Pushing a pivoted object

Fig. 3.11 shows snapshots of the experiment for a pivoted object at different mean positions for the swarm when they are uniformly distributed. The object was designed such that the Kilobots keep the starting uniform distribution with $\sigma = 0.05L = 6.8\,\text{cm}$.

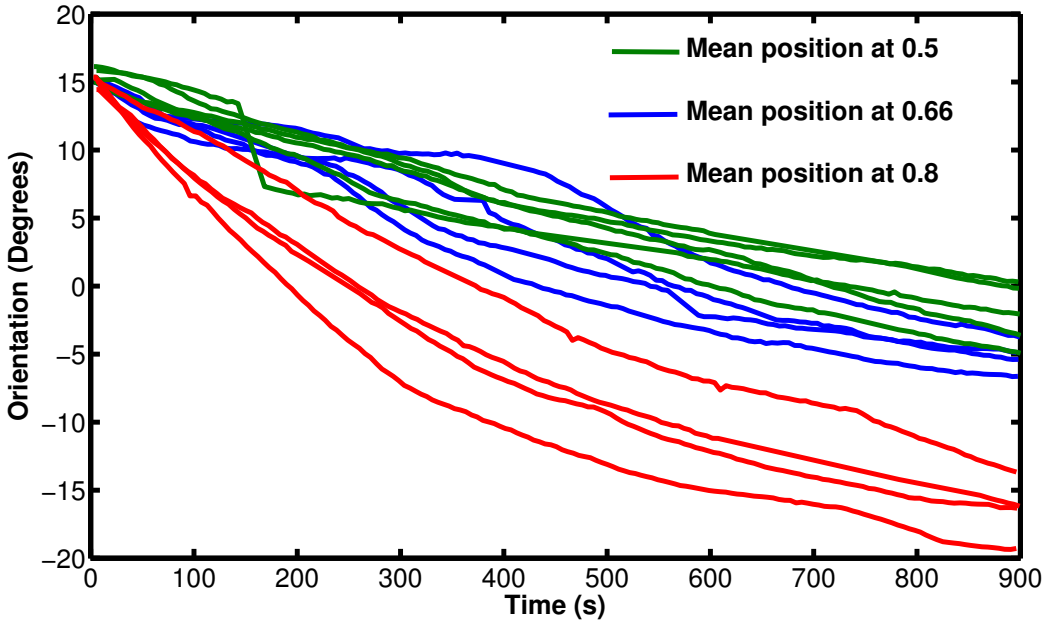


Figure 3.10: Pivoted object using uniformly distributed Kilobots with $\sigma = 0.05L = 6.8\,\text{cm}$, but different mean positions. Mean positions are normalized along a rod of length 1.42 m.

Kilobots were programmed to go straight so that they always apply force perpendicular to the object. The results of different mean positions are shown in Fig. 3.10 for four trials at $\mu = [0.5, 0.66, 0.8]$. Mean positions near the optimal solution of (3.14) increase torque.

The experiments from Section 4.7.a were manually demonstrated using this physical swarm. Fig. 3.11 illustrates an experiment showing pure torque control with a swarm of robots. In this figure a large-aspect-ratio rectangle (91×2 cm, colored pink in the image) was hinged to one side of the table. Like a door, this object could only be moved around this pivot. Two trials were performed. In each trial the swarm was initialized in the lower right side of the table, and then commanded to push the object with the mean position of the swarm directed toward a point distance C from the pivot point. Data was recorded for 150 seconds. In the first trial, $C = L$, so the robots were commanded to push the door at the extreme edge of the door from the pivot. In the second trial $C = 1/2L$, and

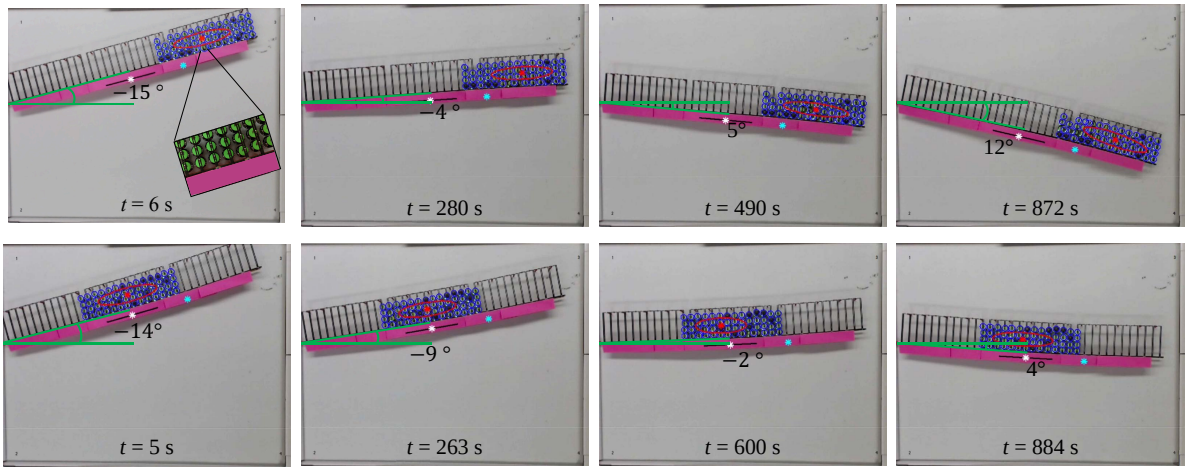


Figure 3.11: Snapshots showing the effect of pushing a pivoted rectangular object at different distances from the pivot point. The top row shows the frames with the mean position is at 0.75 of the object length. The bottom row show frames with swarm mean position $\mu = 1/2$.

so the swarm pushed the object in the center of the rectangle. As discussed in Section 4.7, the robots spread when commanded to push the object at the extreme end, and half of the robots flowed past the end of the rectangle without engaging the rectangle. This illustrates a key difference between robotic swarms and a single pusher robot. The swarm exerts the most torque when (3.2) is maximized. (3.2) is maximized when the majority of the swarm engages the object. For this reason in Fig. 3.12, the trial in the second row of screenshots moves the door further than the trial in the first row.

3.7.2 Orientation control of a free object

Figure 3.13 shows snapshots of orientation control of a free rod using 97 Kilobots. Kilobots were programmed to move toward the brightest light in the room. Therefore, the light is the global input. The goal orientation of the object changed periodically, and the robots will go to a corner when the goal orientation is achieved.

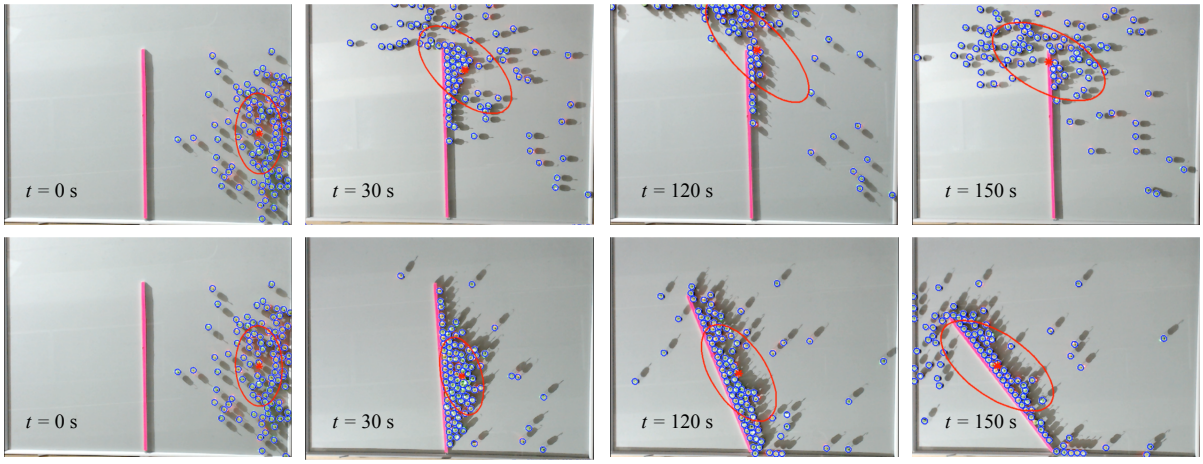


Figure 3.12: Snapshots showing the effect of pushing a pivoted rectangular object at different distances from the pivot point. The top row of snapshots illustrate the swarm pushing at the end of the object. The bottom row illustrates that when the swarm pushes at the middle of the object the force provided by the swarm remains constant.

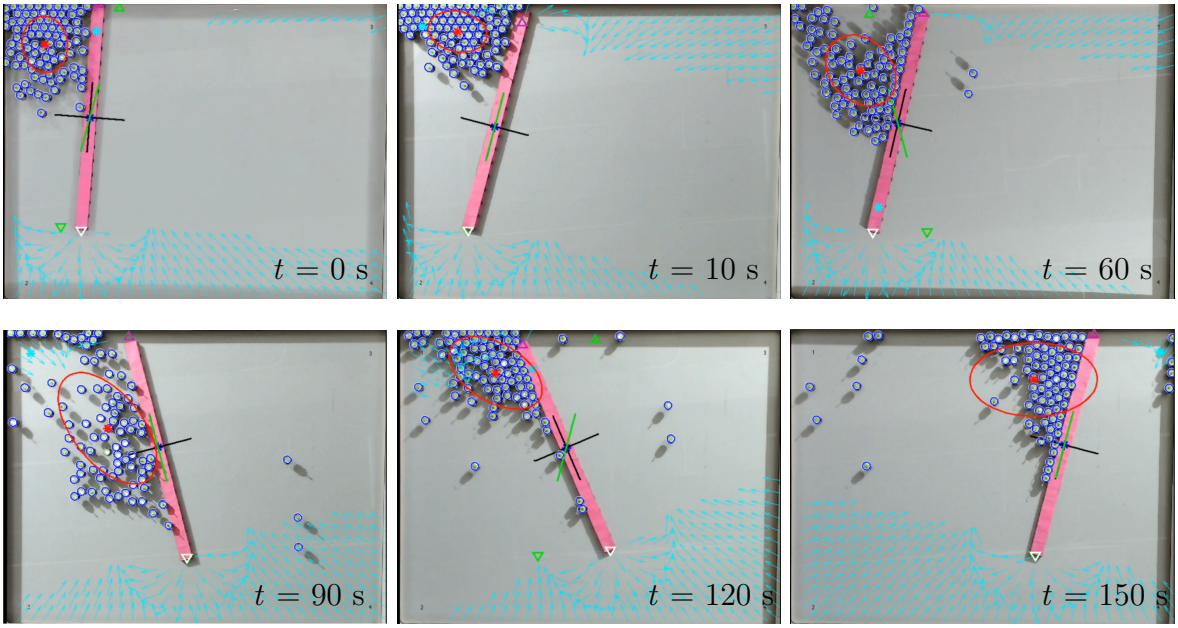


Figure 3.13: Snapshots showing orientation control of a free object using 97 hardware robots that all get the same control input. Light direction is the global control input and the robots are programmed to move toward the brightest light in the environment.

3.8 Conclusion

This paper presented an analysis of torque applied by a swarm to a long rod. Three distributions were studied: uniform, triangular and normal. Though particles are often normally distributed, optimizing torque from normal distributions involves error functions. We illustrated numerically that the triangular distribution well approximates the normal distribution, and gave analytic solutions with uniform and triangular distributions. Analytic results for force and torque produced by granular media in configurations defined by an angle of repose were also given.

Chapter 4

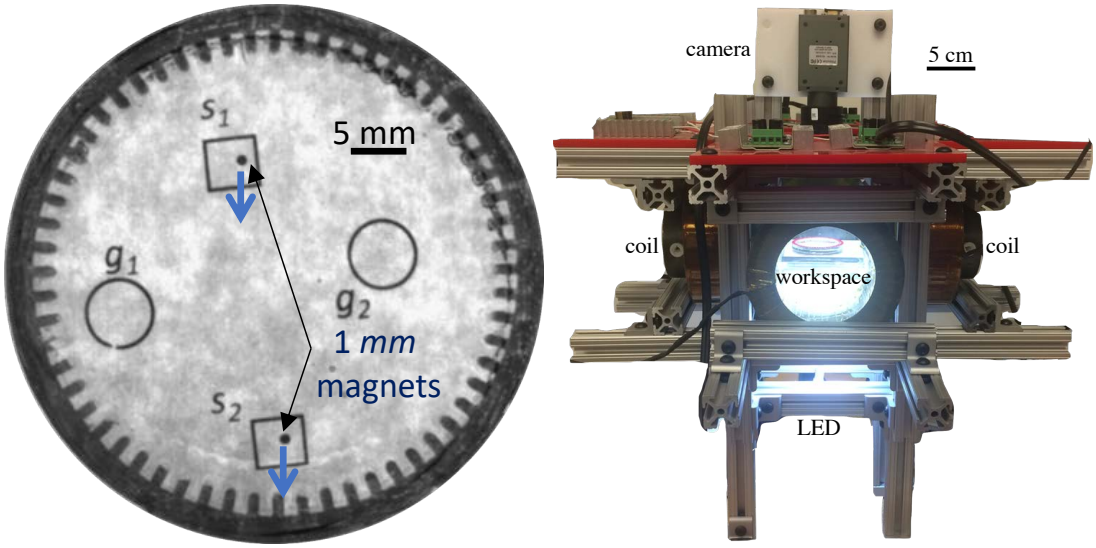
Exploiting Non-Slip Wall Contacts to Position Two Particles Using The Same Control Input

4.1 Introduction

Positioning is a foundational capability for a robotic system, e.g. placement of brachytherapy seeds. 2D position of each particle in such a swarm is controllable if the workspace contains a single obstacle the size of one particle [7]. However, requiring a single, small, rigid obstacle suspended in the middle of the workspace is often an unreasonable constraint, especially in 3D. This chapter relaxes that constraint, and provides position control algorithms that only require non-slip wall contacts. We assume that particles in contact with the boundaries have zero velocity if the uniform control input pushes the particle into the wall.

The paper is arranged as follows. After a review of recent related work in Sec. 4.2, Sec. 4.3 introduces a model for boundary interaction and two shortest path results for representative workspaces. We provide an algorithm to arbitrarily position two particles in Sec. 4.4. Section 4.7 describes implementations of the algorithms in simulation and Sec. 4.8 describes hardware experiments, as shown in Fig. 4.1. We end with directions for future research in Sec. 4.9.

This work analyses any convex workspaces and 3D positioning. This chapter also implements the algorithms using a hardware setup inspired by the anatomy of the gastrointestinal tract.



workspace magnetic setup

Figure 4.1: Workspace and magnetic setup for an experiment to position particles that receive the same control inputs, but cannot move while a control input pushes them into a boundary.

4.2 Related work

Controlling the *shape*, or relative positions, of a swarm of robots is a key ability for a range of applications. Correspondingly, it has been studied from a control-theoretic perspective in both centralized and decentralized approaches. For examples of each, see the centralized virtual leaders in [61], and the gradient-based decentralized controllers using control-Lyapunov functions in [62]. However, these approaches assume a level of intelligence and autonomy in individual robots that exceeds the capabilities of many systems, including current micro- and nano-robots. Current micro- and nano-robots, such as those in [59, 63, 67] lack onboard computation.

This chapter focuses on centralized techniques that apply the same control input to both particles. Precision control requires breaking the symmetry caused by the uniform input. Symmetry can be broken using particles that respond differently to the uniform control signal, either through agent-agent reactions [68], or engineered inhomogeneity

[23, 69, 70]. The magnetic gradients of MRI scanners are *uniform*, meaning the same force is applied everywhere in the workspace [71]. This work assumes a uniform control with homogenous particles, as in [7], and breaks the control symmetry using obstacles in the workspace.

Alternative techniques rely on non-uniform inputs, such as artificial force-fields. Applications have included techniques to design shear forces for sensorless manipulation of a single object by [72]. [73] demonstrated a collection of 2D force fields generated by six degree-of-freedom vibration inputs to a rigid plate. These force fields, including shear forces, could be used as a set of primitives for motion control to steer the formation of multiple objects.

Similarly, much recent work in magnet control has focused on exploiting inhomogeneities in the magnetic field to control multiple micro particles using gradient-based pulling [74, 75]. Unfortunately, using large-scale external magnetic fields makes it challenging to independently control more than one microrobot unless the distance between the electromagnetic coils is at the same length scales as the robot workspace [74–76]. In contrast, this chapter requires only a controllable constant gradient in orthogonal directions to position the particles.

If a control input causes the particles to collide with obstacles at different times, inverting the control input does not undo the action. Due to this lack of time-reversibility, techniques that require a bidirectional graph, e.g. PRM [77] and RRT* [78] are not suitable. Instead, this chapter employs a greedy search strategy.

4.3 Theory

In the absence of obstacles, uniform inputs move a swarm identically. Independent control requires breaking this symmetry. The following sections examine using non-slip boundary contacts to break the symmetry caused by uniform inputs. Our algorithms

rely on holding one particle stationary by pushing it into the boundary while moving the other particle. We start with a boundary interaction model in subsection 4.6.2.

For some configurations, we can obtain the optimal solution. Subsections 4.3.2 and 4.3.3 provide shortest-path results for two representative workspaces, squares and disks.

4.3.1 Boundary interaction model

The system dynamics represent particle swarms in low-Reynolds number environments, where viscosity dominates inertial forces and so velocity is proportional to input force [79]. In this regime, the input force command $\mathbf{u}(t)$ controls the velocity of the particles. If the i^{th} particle has position $\mathbf{x}_i(t)$ and velocity $\dot{\mathbf{x}}_i(t)$, we assume the following system model:

$$\dot{\mathbf{x}}_i(t) = \mathbf{u}(t) + F\left(\mathbf{x}_i(t), \mathbf{u}(t)\right), \ i \in [1, n]. \tag{4.1}$$

$$F(\mathbf{x}_i(t), \mathbf{u}(t)) = \begin{cases} -\mathbf{u}(t) & \mathbf{x}_i(t) \in \text{boundary } \textbf{and} \\ & \mathbf{N}(\text{boundary}_{\mathbf{x}_i(t)}) \cdot \mathbf{u}(t) \leq 0 \\ 0 & \text{else.} \end{cases},$$

Here $F(\mathbf{x}_i(t), \mathbf{u}(t))$ is the frictional force provided by the boundary, and $\mathbf{N}(\text{boundary}_{\mathbf{x}_i(t)})$ is the normal to the boundary at position $\mathbf{x}_i(t)$.

The same model can be generalized to particles moved by fluid flow where the vector direction of fluid flow $\mathbf{u}(t)$ controls the velocity of particles, or for a swarm of particles that move at a constant speed in a direction specified by a uniform input $\mathbf{u}(t)$ [50]. As in our model, fluid flowing in a pipe has zero velocity along the boundary. Similar mechanical systems exist at larger scales, e.g. all tumblers of a combination lock move uniformly unless obstructed by an obstacle. Our control problem is to design the control inputs $\mathbf{u}(t)$ to deliver two particles to goal positions.

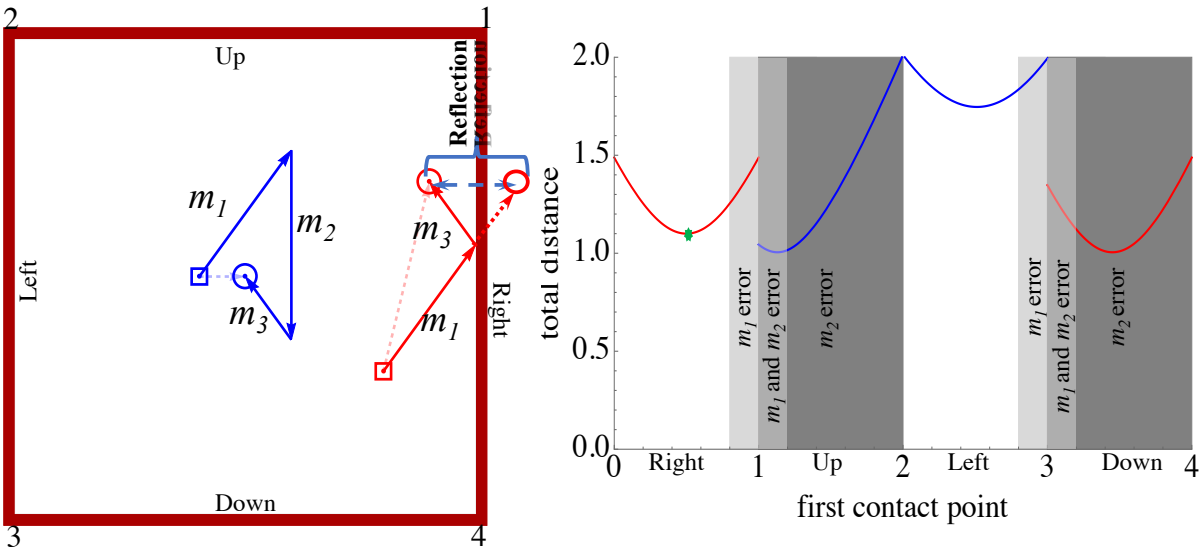


Figure 4.2: The shortest three-move path that reconfigures two particles has the property that the incident angle equals the reflected angle.

4.3.2 Example: shortest path in a square workspace

Changing the relative positions of particles in any workspace requires making one particle contact the boundary. If the goal configuration cannot be reached in one move but can be reached in three moves, the shortest path has a simple solution. The first move, m_1, makes one particle contact a wall, m_2 adjusts the relative spacing error to zero, and m_3 takes the particles to their final positions. m_2 cannot be shortened, so optimization depends on choosing the location where the particle contacts the wall. Since the shortest distance between two points is a straight line, reflecting the goal position across the boundary wall and plotting a straight line gives the optimal contact location, as shown in Fig. 4.2. There are four walls, and four candidate solutions, but some candidate solutions may be invalid because a different boundary is hit before the desired first contact position in move m_1 (light grey regions) or invalid because m_2 cannot generate the goal relative spacing (dark grey regions).

This optimization result was implemented in our previous work using a square workspace [80]. Fig. 4.3 shows solutions from a *Mathematica* implementation in a square

workspace for six representative configurations.

4.3.3 Shortest path in unit disk that intersects circumference

The shortest path between two points in the unit disk that intersects the circumference is composed of two straight line segments and has an optimal contact point, as shown in Fig. 4.4. The problem can be simplified by choosing the coordinate system carefully. We define the x-axis along the line from the circle center to the starting point: $S = (s, 0)$, and define the point of intersection by the angle θ from the x-axis: $P = (\cos\theta, \sin\theta)$. Define the final point E by a radius e and angle β: $E = e(\cos\beta, \sin\beta)$. Then the length of the two line segments is

$$\sqrt{(s - \cos\theta)^2 + \sin^2\theta} + \sqrt{(e\cos\beta - \cos\theta)^2 + (e\sin\beta - \sin\theta)^2}, \tag{4.2}$$

which is minimized by choosing an appropriate θ value.

The length of the two line segments as a function of θ is drawn in the right plot of Fig. 4.4. There are several simple solutions. If s is 1 or e is 0 or β is 0, the optimal angle θ^* is 0. If e is 1 or s is 0, the optimal angle is β. Label the origin O. The optimal path satisfies the law of reflection off the unit circle, with angle of incidence equal to angle of reflection. The angle $\angle OPS$ (from the origin to P to S) is the same as the angle $\angle OPE$ (from the origin to P to E). We name these angles α. This can be proved by drawing an ellipse whose foci are S and E. When the ellipse is tangent to the circle, the point of tangency is P. Since the distance from the origin to P is always 1, we can set up three equalities using the law of sines: From triangle OSP: $\frac{\sin\alpha}{s} = \frac{\sin(\alpha+\theta)}{1} = \frac{\sin\theta}{||SP||}$, and from triangle OEP: $\frac{\sin\alpha}{e} = \frac{\sin(\beta-\theta)}{||EP||}$. If we mirror the point S about line $\overline{OP}$ and label this point C, from triangle CEO: $\frac{\sin(\alpha+\theta)}{e} = \frac{\sin(2\theta-\beta)}{||CE||}$.

Simplifying this system of equations results in $s = e\csc\theta(s\sin(2\theta-\beta)+\sin(\beta-\theta))$. Solving this last equation results in a quartic solution that has a closed-form solution with four roots, each of which can be either a clockwise or a counterclockwise rotation θ,

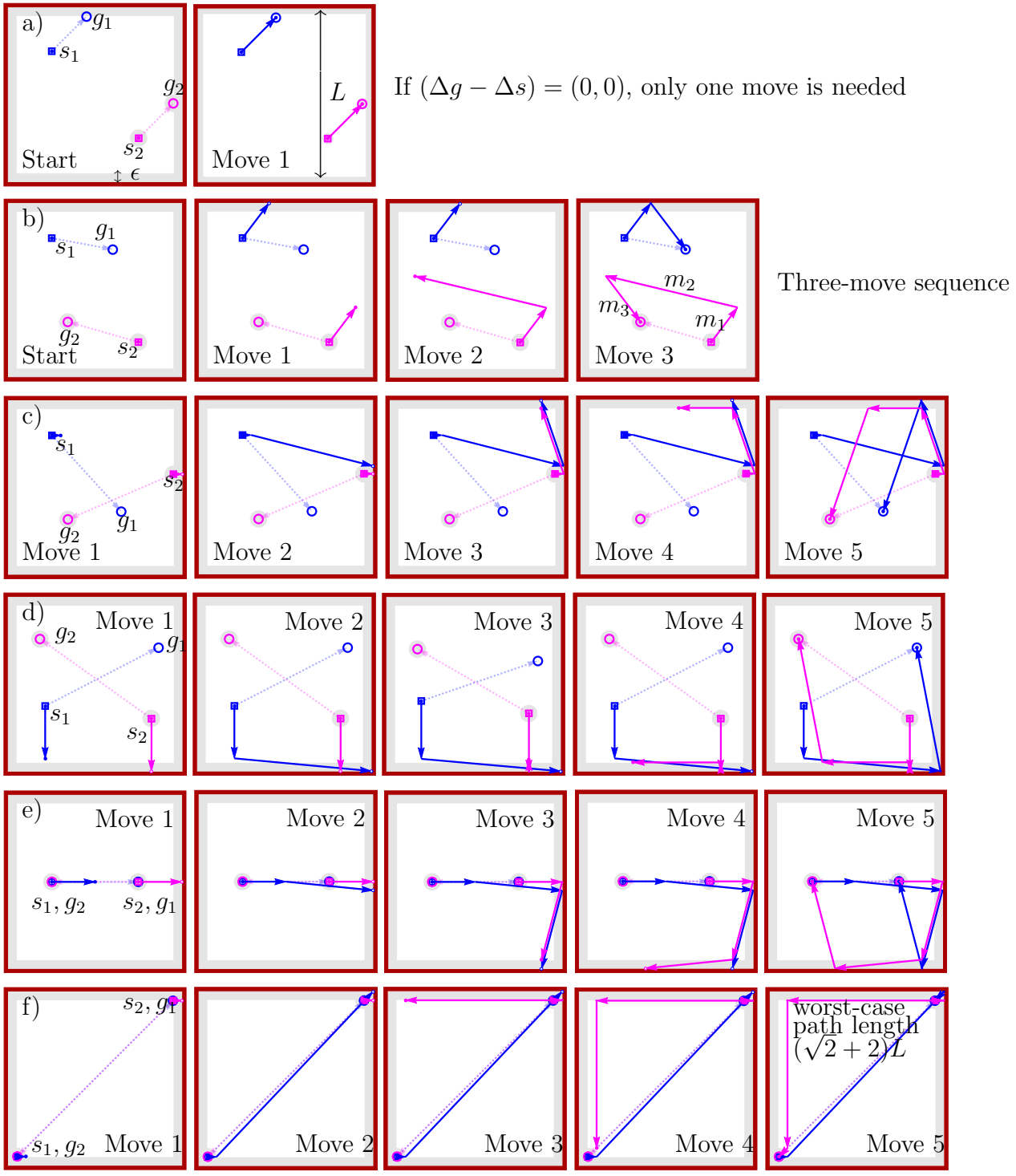


Figure 4.3: Frames from an implementation of Alg. 3: two particle positioning using walls with non-slip contacts.

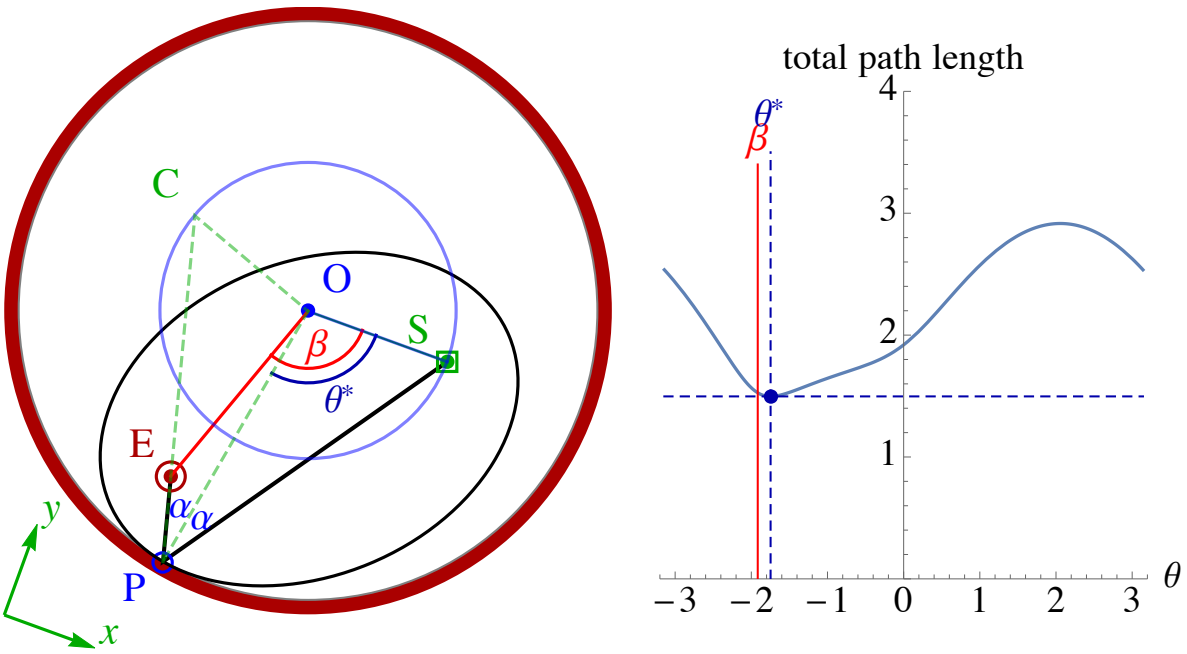


Figure 4.4: The shortest path between two points S to E in the unit disk that intersects the circumference. The path length as a function of intersection point, $P = (\cos\theta, \sin\theta)$ is shown at right.

depending on the sign of β, with $-\pi \leq \beta \leq \pi$. We evaluate each and select the solution that results in the shortest length path. For an interactive Mathematica demonstration of this shortest path, see [81]. Because the closed form solution is long, it is included in the paper attachments.

4.4 Position control of two particles using boundary interaction

This section presents algorithms that use non-slip contacts with walls to arbitrarily position two particles in a convex workspace. Workspaces are 2D convex polygons with no internal obstacles. Assume two particles are initialized at s_1 and s_2 with corresponding goal destinations g_1 and g_2. Denote the current positions of the particles p_1 and p_2. Values $.x$ and $.y$ denote the x and y coordinates, i.e., $p_1.x$ and $p_1.y$ denote the x and y locations of p_1. Algorithm 3 can now handle any convex workspace, including the special limit case of a circular workspace. In the last subsection we present techniques to control

3D positioning of two particles.

4.4.1 Δ configuration space

The configuration space for two particles is a four dimensional manifold. Translating both particles the same amount is a trivial operation, but changing the relative positions requires boundary interaction. For this reason, our algorithms use the two dimensional Δ configuration space. The Δ configuration space is a set of all possible Δp values, defined as the difference in position of the first particle from the second particle: $\Delta p = p_2 - p_1$. We use the Δ configuration space to plan move sequences that achieve the desired relative spacing. Once the particles have the correct relative spacing, they can be delivered to the goal configuration in one move.

The Δ configuration space for an n-sided convex polygon P can be constructed in a method analogous to computing configuration space obstacles for polygons [82]. Translate n copies of P so that each copy moves a different vertex of P to $(0, 0)$. Because P is convex, the convex-hull of all these translated vertices is the boundary of the Δ configuration space. For an n-sided convex polygon, the Δ configuration space is a $2n$-sided convex polygon. Even-sided regular polygons are a special case in which half the sides align and the Δ configuration space is n-sided. An example Δ configuration space construction is shown in Fig. 4.5: a four-sided workspace is on the left, the four translated copies with dashed lines outlining the convex hull is in the middle, and the resulting Δ configuration space is on the right.

4.4.2 Two particle path planning

The *2-move reachable set* is the locus of points in the Δ configuration space corresponding to any two-move sequence where the first move brings one particle into contact with the boundary, and the second move translates the second particle without moving

the first. For the given Δs (starting configuration), the rightmost image of Fig. 4.5 draws the 2-move reachable sets in transparent blue. Figure 4.6 shows the starting and ending relative positions as Δs and Δg in the Δ configuration space. The next subsections give procedures to compute the 2-move reachable set.

The goal is to move the particles within δ of the goal positions using a shared control input where δ is an arbitrary small number. We do this by first moving them within δ of the correct relative position and then translating the particles to the goal. The relative position is $||\Delta g - \Delta p|| = ||(g_2 - g_1) - (p_2 - p_1)||$.

Algorithm 3 assigns a uniform control input at every instance. It first computes the 2-move reachable set. If the goal relative position is in the 2-move reachable set, we move particles to achieve that relative position. If it is not in the 2-move reachable set, we move particles to achieve the closest point on this reachable set from Δg, which is Δg_c.

Achieving a Δg_c configuration requires two-moves, the first to move until one particle touches a wall, and the second to adjust the relative spacing. Once the correct *relative* position has been achieved, a final translation delivers both particles to their goal destinations. Otherwise, we iterate until we reach the goal.

4.4.3 Convex polygonal workspaces: 2-move reachable set

Figure 4.6 shows six workspaces, their Δ configuration spaces, and the 2-move reachable sets that correspond to representative initial conditions. Figure 4.7 highlights the construction of the 2-move reachable sets for a square workspace. There are four 2-move reachable sets, but the horizontal (and vertical) reachable sets are equivalent in the Δ configuration space so we can plan in this space and choose between the two options to minimize the total distance. Algorithm 4 computes the 2-move reachable set for any convex workspace. The set is constructed by considering each edge of the workspace. We

Algorithm 3 2-ParticlePathPlanning$(s_1, s_2, g_1, g_2, P, \epsilon)$

Require: knowledge of starting (s_1, s_2) and goal (g_1, g_2) positions of two particles. P is a description of the workspace. ϵ is a positive error bound.

1: $(p_1, p_2) \leftarrow (s_1, s_2)$ ▷ p_1, p_2 are current positions
2: moves $\leftarrow \{\}$
3: $\Delta p \leftarrow p_2 - p_1$
4: $\Delta g \leftarrow g_2 - g_1$
5: **while** $||\Delta p - \Delta g|| > \epsilon$ **do**
6: $R_{\text{SET}} \leftarrow$ Compute 2-move reachable set ▷ use Alg. 4 or 5
7: $\Delta g_c \leftarrow$nearest point in R_{SET} to Δg
8: $m \leftarrow$move-to-wall corresponding to Δg_c
9: moves $\leftarrow$ Append m to moves
10: $(p_1, p_2) \leftarrow$ ApplyMove m to (p_1, p_2)
11: $\Delta p \leftarrow p_2 - p_1$
12: **end while**
13: moves $\leftarrow$ Append $g_2 - p_2$ to moves ▷ translate to goal
14: **return** moves

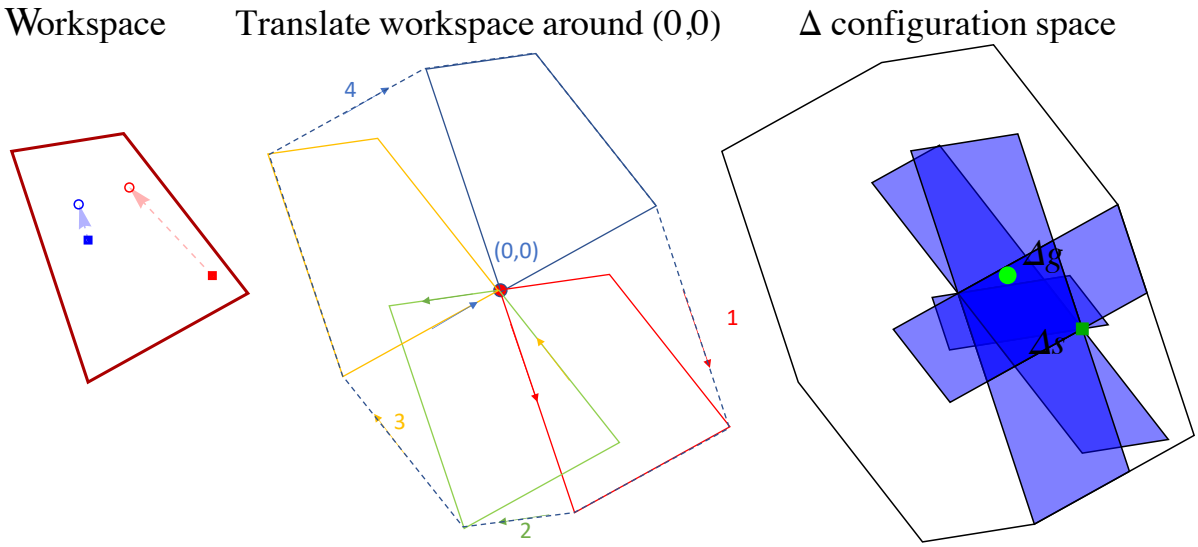


Figure 4.5: Workspace and Δ configuration space is shown for an arbitrary convex polygon with $n = 4$ sides. The 2-move reachable sets for this initial configuration (shown by square markers) are drawn in transparent blue.

name each vertex as p_i where $1 \leq i \leq n$. If one particle contacts edge $\overline{p_i p_{i+1}}$ before the other (one particle will always contact before the other unless the particles are parallel to the wall), the corresponding 2-move reachable set is a polygon, constructed in lines 2-13 of Alg. 4. The union of these polygons for all n sides is the 2-move reachable set of Δ configurations. Figure 4.8 illustrates the procedure to construct the 2-move reachable

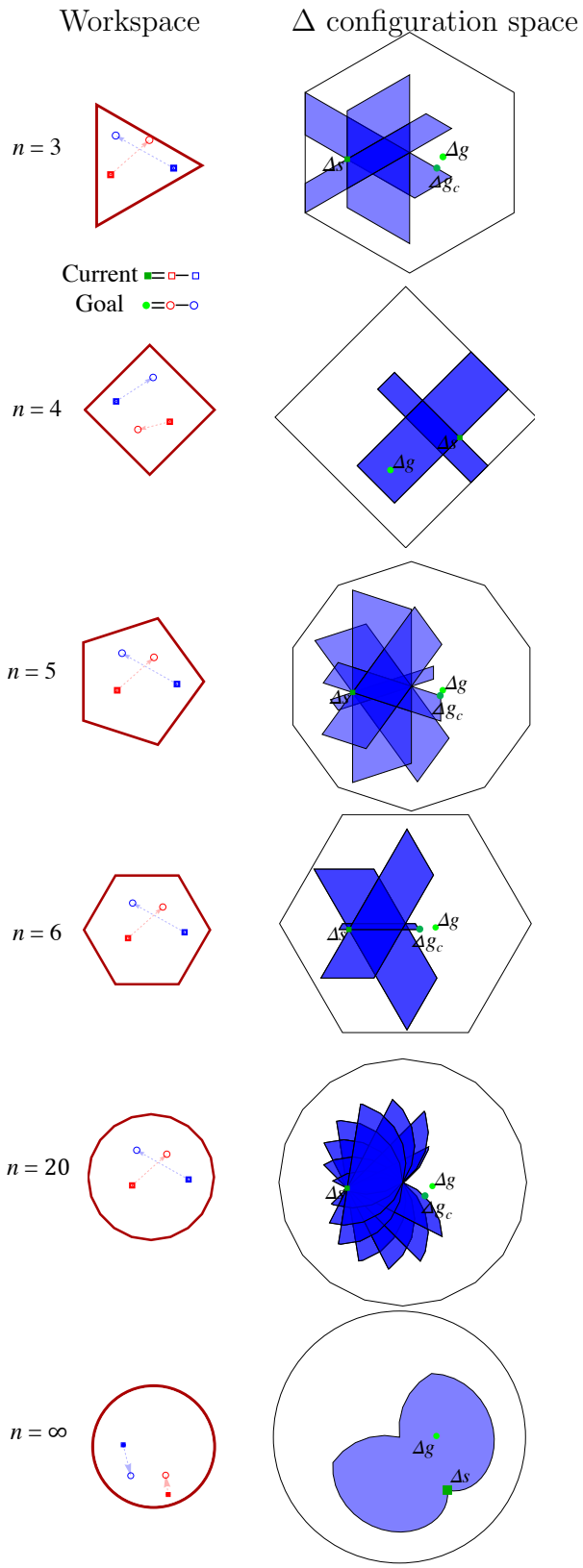


Figure 4.6: The Δ configuration space is all possible configurations of $p_2 - p_1$. The sets reachable in two moves, called *2-move reachable sets*, are drawn with transparent blue polygons.

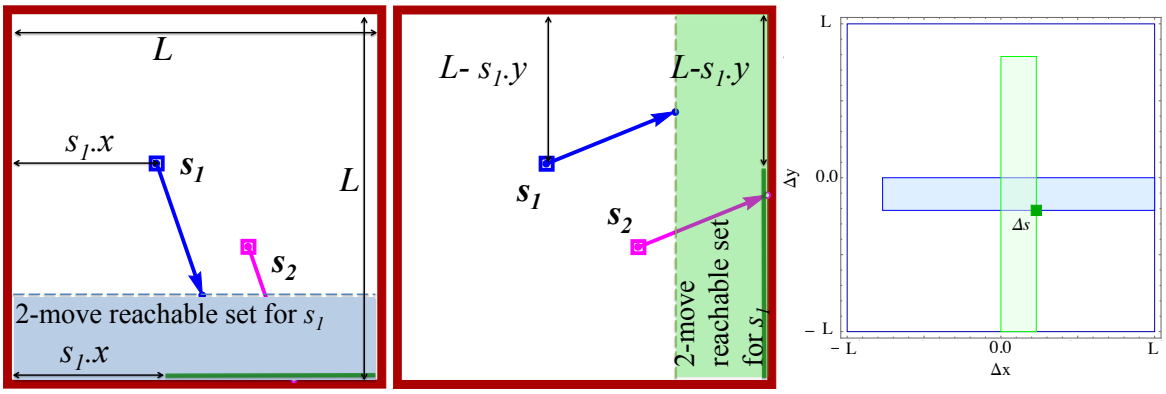


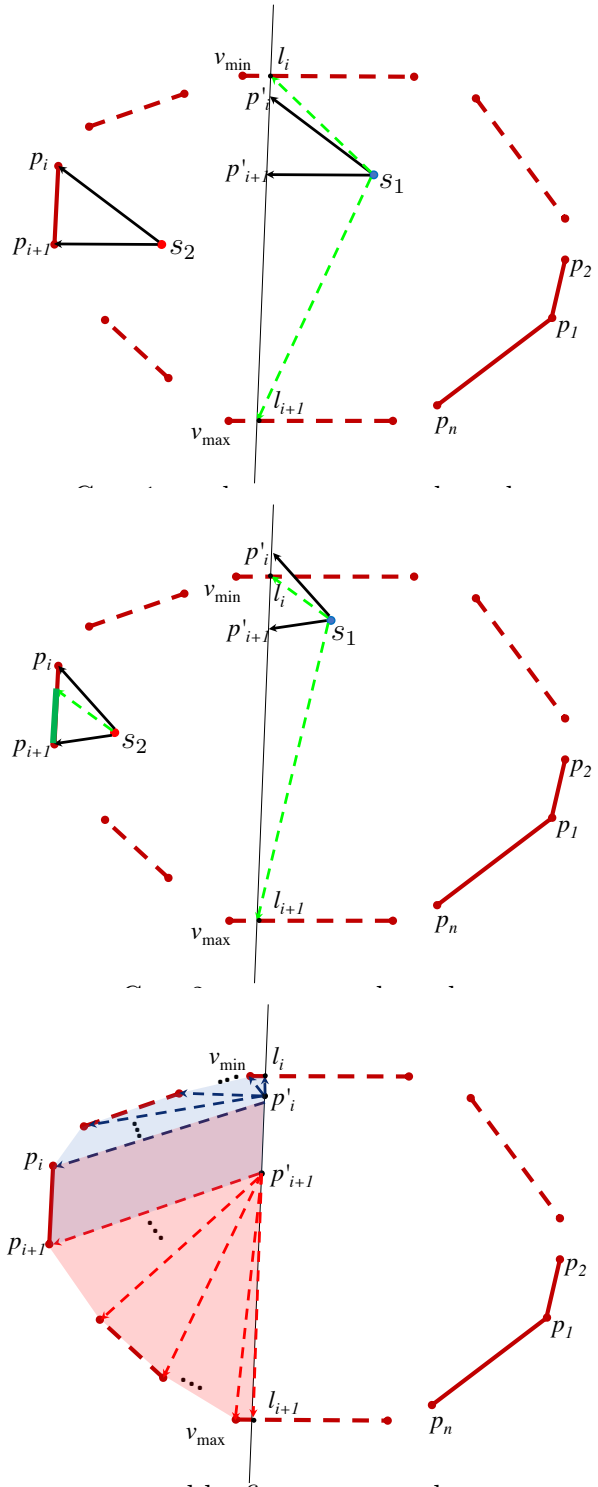
Figure 4.7: (left and middle) If particle 2 contacts the bottom or right wall before particle 1 reaches a wall, particle 2 can reach anywhere along the green line, and particle 1 can move to anywhere in the shaded area. (right) The 2-move reachable sets in the Δ configuration space is shown.

set generated by collisions with the $\overline{p_i p_{i+1}}$ edge.

Algorithm 4 ReachableSetPolygon(s_1, s_2, g_1, g_2, P)

Require: knowledge of starting (s_1, s_2) and goal (g_1, g_2) positions of two particles. P is a list of the vertices of a convex polygon.

$R_{\text{SET}} \leftarrow \{\}$
for p_i in P **do**
  $p'_i \leftarrow s_1 + s_2 - p_i$
  $p'_{i+1} \leftarrow s_1 + s_2 - p_{i+1}$
  $L \leftarrow \overline{p'_i p'_{i+1}}$ ▷ line (p'_i, p'_{i+1})
  $l_i, l_{i+1} \leftarrow$ intersections of L and polygon P
  if p'_i not inside polygon P **then**
    $p'_i \leftarrow l_i$
  end if
  if p'_{i+1} not inside polygon P **then**
    $p'_{i+1} \leftarrow l_{i+1}$
  end if
  $D \leftarrow s_2 - s_1 - ([l_i, v_{\min}, ..., p_i] - p'_i,$
    $[p_{i+1}, p_{i+2}, ..., v_{\max}, l_{i+1}] - p'_{i+1})$
  $R_{\text{SET}} \leftarrow$ Append polygon D to R_{SET}
end for
Return R_{SET}



Polygon generated by first contacts along $\overline{p_i p_{i+1}}$

Figure 4.8: Steps to generate the 2-move reachable set when one particle collides with edge $\overline{p_i p_{i+1}}$ of a convex polygonal workspace.

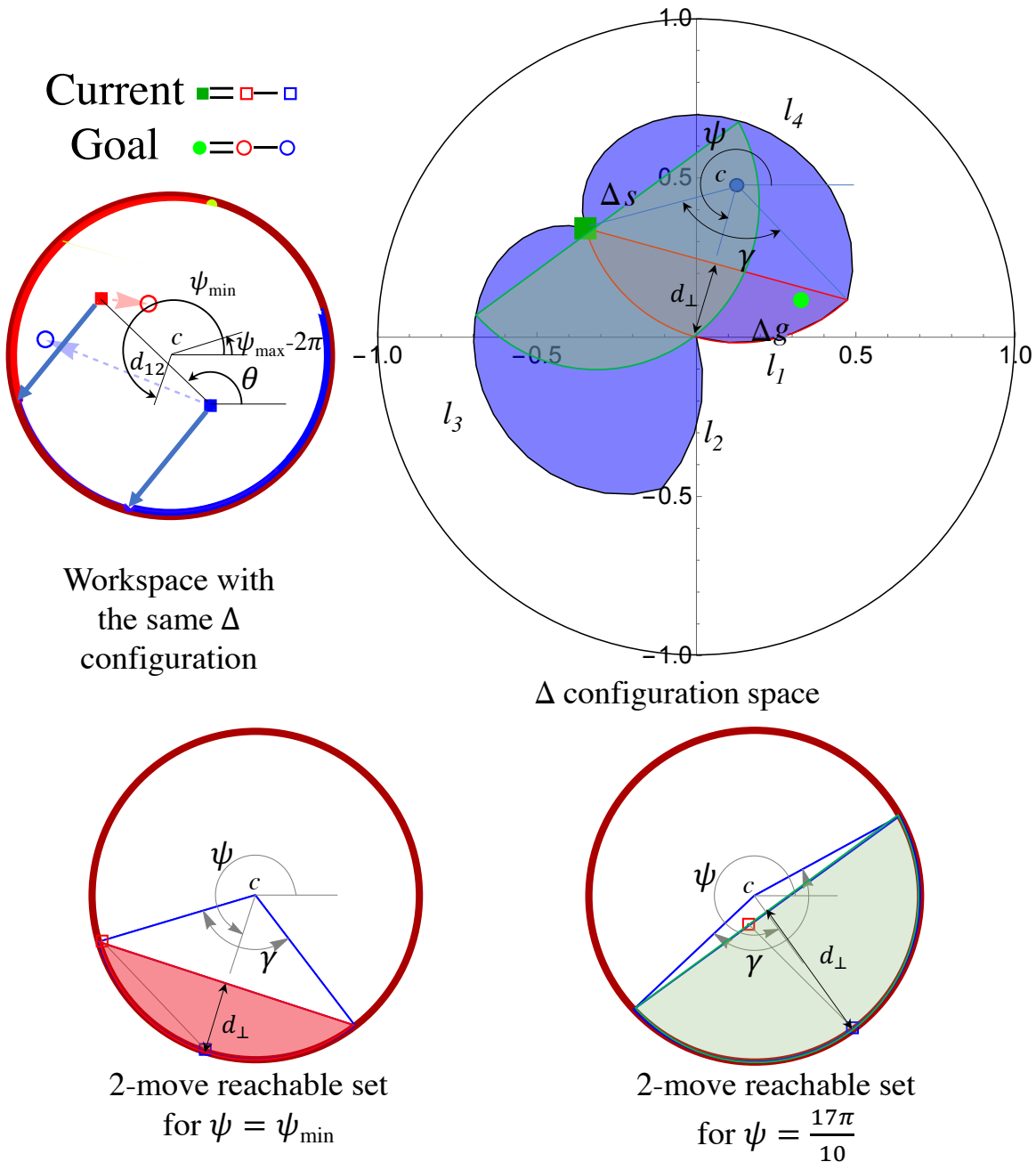


Figure 4.9: Left top: The possible first contact points for the blue and red particles are shown with blue and red arcs. Left bottom: if the blue particle touches the wall at $\psi_{\min}$ (blue square) the other particle (red square) can move anywhere in the red region. Right bottom: if the blue particle touches the wall at $\psi = \frac{17\pi}{10}$ (blue square) the other particle (red square) can move anywhere in the green region. Right: The Δ configuration space for the corresponding starting positions of the particles is shown. The possible 2-move reachable sets before contact are shown in the Δ configuration as a blue region. If the blue particle contacts the boundary at $\psi_{\min}$, the reachable Δ configuration is the red set, or the green set if $\psi = \frac{17\pi}{10}$.

4.4.4 Circular workspaces: 2-move reachable set

To compute the 2-move reachable set for a circular workspace, first consider all possible first contact locations. The set of boundary points that a particle can touch before the other particle touches are two arcs from $\psi_{\min}$ to $\psi_{\max}$ and from $\pi + \psi_{\min}$ to $\pi + \psi_{\max}$:

$$\psi \in [\psi_{\min}, \psi_{\max}] = \theta + \left[\sin^{-1}\left(\tfrac{d_{12}}{2r}\right) - \tfrac{\pi}{2}, \tfrac{\pi}{2} - \sin^{-1}\left(\tfrac{d_{12}}{2r}\right)\right], \tag{4.3}$$

where $d_{12} = ||s_1 - s_2||$, r is the radius of the workspace, and the angle between two particles is $\theta = \arctan(\frac{p_1.x - p_2.x}{p_1.y - p_2.y})$.

A circle has an infinite number of sides, thus infinite reachable sets. However, the 2-move reachable set can be parameterized by the angle of first contact location ψ, as shown in Fig. 4.9.

Each ψ value generates a 2-move reachable set that is a chord of the disk, with interior angle γ parameterized by ψ:

$$\gamma(\psi) = \cos^{-1}\left(1 - \frac{d_\perp(\psi)}{r}\right), \text{ where:} \tag{4.4}$$

$$d_\perp(\psi) = 2||s_1.p_\psi(\psi) - s_2.p_\psi(\psi)||, \tag{4.5}$$

$$p_\psi(\psi) = r[\cos(\psi), \sin(\psi)]. \tag{4.6}$$

The 2-move reachable sets with π difference in ψ value are equivalent in the Δ configuration space. The reachable Δ configuration set for any first contact point defined by ψ is the area under a chord from angle $\psi - \frac{\gamma(\psi)}{2}$ to $\psi + \frac{\gamma(\psi)}{2}$, for a circle of radius r centered at $c = r(\cos(\psi - \pi), \sin(\psi - \pi))$. Two such chords are drawn in red and green in Fig. 4.9.

The equations for the four lines outlining the union of two-move reachable sets are as follows:

$$l_1 = r\Big(\cos\psi_{\min} - \cos(\gamma + \psi_{\min}) \tag{4.7}$$

$$+ \sin\psi_{\min} - \sin(\gamma + \psi_{\min})\Big) \qquad 0 < \gamma < \gamma(\psi_{\min}),$$

$$
\begin{aligned}
l_2 = r\Big(& \cos\psi_{\max} - \cos(\gamma + \psi_{\max}) \\
& + \sin\psi_{\max} - \sin(\gamma + \psi_{\max})\Big) \qquad\qquad \gamma(\psi_{\max}) < \gamma < 0, \\
l_3 = r\Big(& \cos\psi - \cos(\psi + \gamma(\psi)) \\
& + \sin\psi - \sin(\psi + \gamma(\psi))\Big) \qquad\qquad \psi_{\min} < \psi < \psi_{\max}, \\
l_4 = r\Big(& \cos\psi - \cos(\psi - \gamma(\psi)) \\
& + \sin\psi - \sin(\psi - \gamma(\psi))\Big) \qquad\qquad \psi_{\min} < \psi < \psi_{\max}.
\end{aligned}
$$

We combine these boundaries to compute the 2-move reachable set summarized in Alg. 5. The motion-planner finds a ψ that would enable us to reach Δg_c, the nearest point in the 2-move reachable set to Δg. We first check if Δg_c is in the Δ configuration space chords defined by either $\psi_{\min}$ or $\psi_{\max}$ using the following two tests:

$$
\begin{aligned}
& (\Delta g_c.x - c.x)^2 + (\Delta g_c.y - c.y)^2 > r^2 \textbf{ and} \\
& (c.x - \Delta g_c.x)\cos\psi + (c.y - \Delta g_c.y)\sin\psi > r\cos\gamma.
\end{aligned} \tag{4.8}
$$

If Δg_c is not in either chord, we draw a line from Δg_c to the current relative position, Δp. This line is a chord of the circle centered at c. The ψ to this chord obeys:

$$
\psi = \tan^{-1}\Big(\frac{\Delta p.x - \Delta g_c.x}{\Delta p.y - \Delta g_c.y}\Big). \tag{4.9}
$$

The particles achieve Δg_c in two moves. The first move causes one particle to touch the wall at p_ψ, (4.6). The second move achieves the required relative position.

4.4.5 3D workspaces: cylinders and prisms

Extending path planning to 3D is possible only if the two particles do not initially have the same x and y positions. For ease of analysis, we assume the workspace boundaries extend in the $\pm z$ direction to form either right cylinders or right prisms. If the 3D

Algorithm 5 ReachableSetCircle(s_1, s_2, g_1, g_2)

Require: knowledge of starting (s_1, s_2) and goal (g_1, g_2) positions of two particles.

1: Calculate $\psi_{\min}$ and $\psi_{\max}$ ▷ use (4.3)
2: Calculate $\gamma(\psi)$ ▷ use (4.4)
3: Calculate l_1, l_2, l_3, l_4 ▷ use (4.7)
4: Return the union of (l_1, l_2, l_3, l_4)

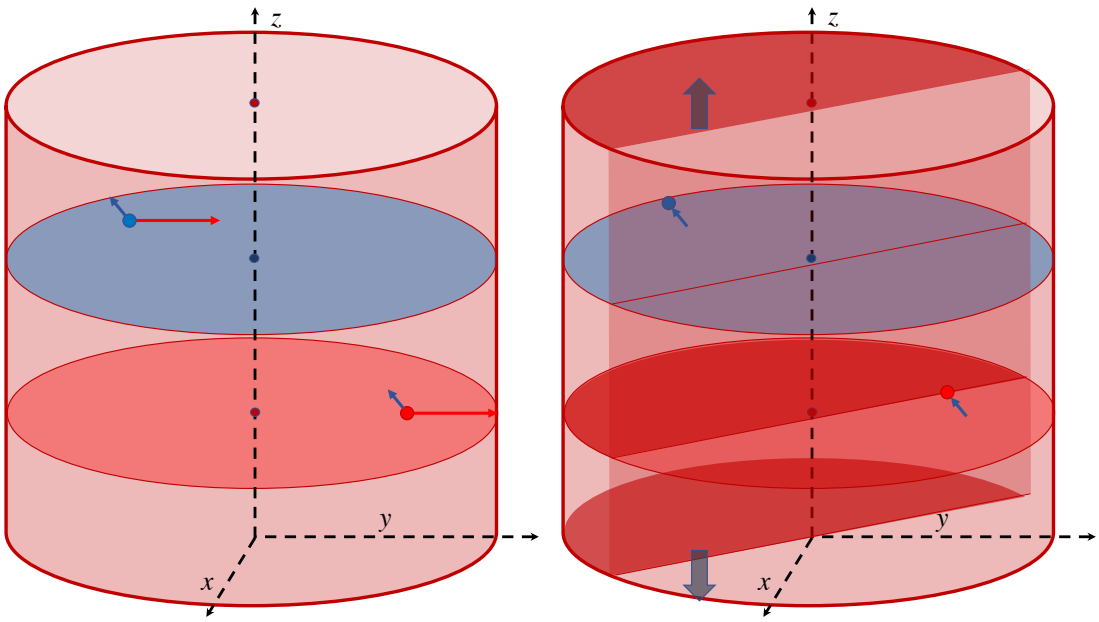


Figure 4.10: Illustration on how boundary contacts enable 3D positioning. Once one particle contacts a boundary, the other particle's 2-move reachable set is a prism formed by extending the 2D 2-move reachable set in the $\pm z$ direction.

projection is at a different angle, redefine the 2D workspace as a region perpendicular to the projection. First, we move the closest particle to the boundary, which prevents its z-coordinate from changing. We next apply actuation in either the $\pm z$ direction to achieve the desired Δz. Then the particles are actuated away from the boundary and to the appropriate z positions. Path planning continues using Alg. 3 to position the particles to the desired x and y positions. As an example, consider Fig. 4.10 which shows a cylindrical workspace. The blue particle starts in the blue disk and the red particle starts in the red disk. The two candidate shortest-length paths that touch the wall are shown with parallel arrows. Each arrow will cause one of the particles to touch the wall, enabling the other particle to move freely in the z-axis to achieve the required relative

position. This can be extended to other 3D workspaces if the workspace can be locally approximated as a 3D prism or cylinder. Other workspaces may be better handled by other path planners, such as [7], which used collisions with protrusions of the workspace to rearrange particles.

4.5 Position control of n robots using boundary interaction

The ideas from Alg. 3 can be extended to control the position of n particles using walls with non-slip contact. The solution is complete, but not optimal, and requires the starting and final configurations of particles to be disjoint. The solution described here is an iterative procedure with n loops. The k^{th} loop moves the k^{th} robot from a *staging zone* to the desired position in a *build zone*. All robots move according to the uniform input, but due to non-slip wall contacts, at the end of the k^{th} loop, robots 1 through k are in their desired final configuration in the build zone, and robots $k+1$ to n are in the staging zone. See Fig. 4.11 for a schematic of the build and staging zones.

Assume an open workspace with four axis-aligned walls with non-slip contact. The axis-aligned build zone of dimension (w_b, h_b) containing the final configuration of n robots must be disjoint from the axis-aligned staging zone of dimension (w_s, h_s) containing the starting configuration of n robots. Without loss of generality, assume the build zone is above the staging zone. Let d be the diameter of the particles. Furthermore, there must be at least ϵ space above the build zone, ϵ below the staging zone, and $\epsilon+d$ to the left of the build and staging zone. The minimum workspace is then $(\epsilon+d+\max(w_b, w_s), 2\epsilon+h_s, h_b)$.

The n robots position control algorithm relies on a DRIFTMOVE$(\alpha, \beta, \epsilon, \theta)$ control input, described in Alg. 7 and shown in Fig. 4.12. For $\theta = 0°$, a drift move consists of repeating a triangular movement sequence $\{(\beta/2, -\epsilon), (\beta/2, \epsilon), (-\alpha, 0)\}$. Any particle touching a top wall moves right β units, while every particle not touching the top moves right $\beta - \alpha$.

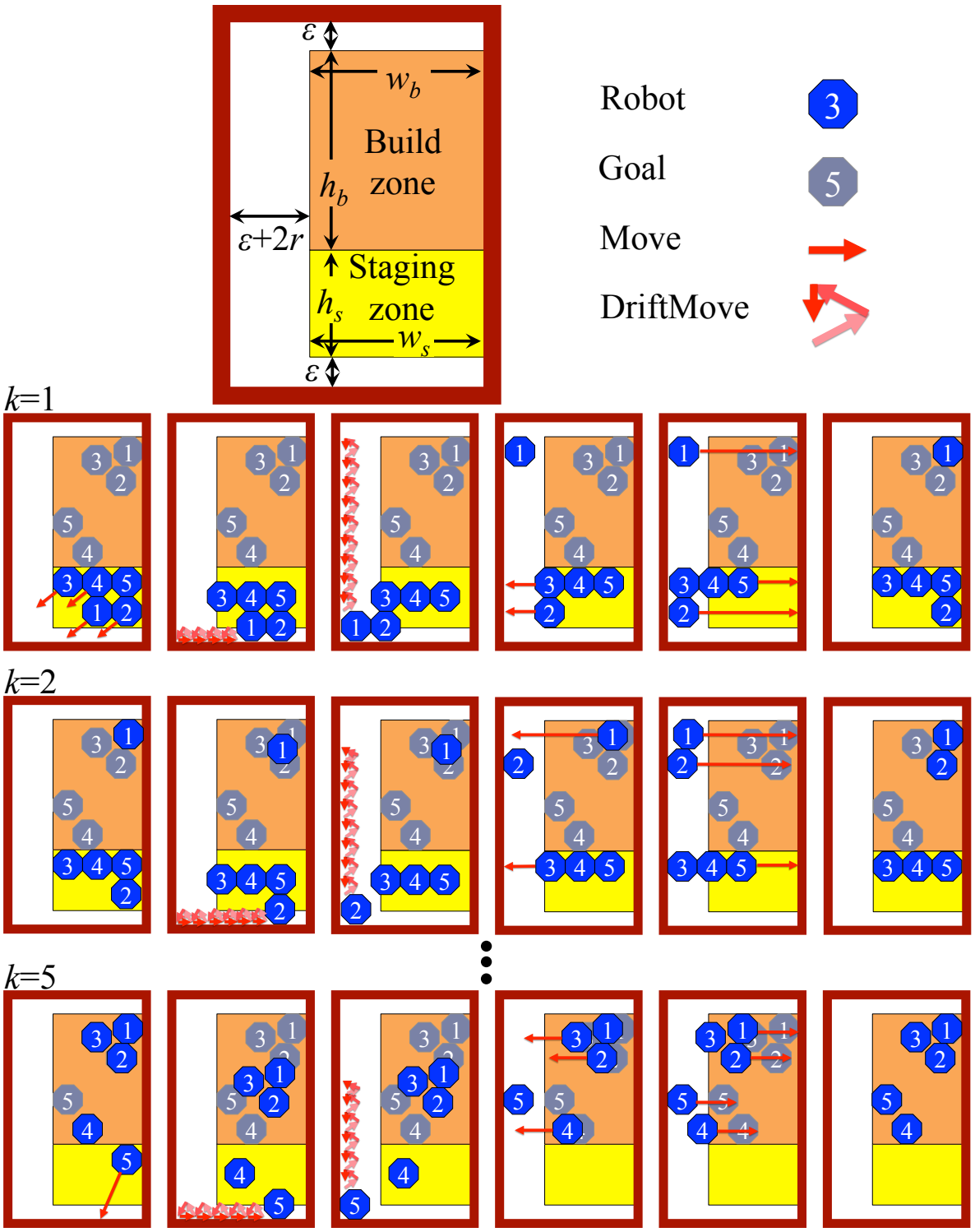


Figure 4.11: Illustration of Alg. 6, n robot position control using walls with non-slip contact.

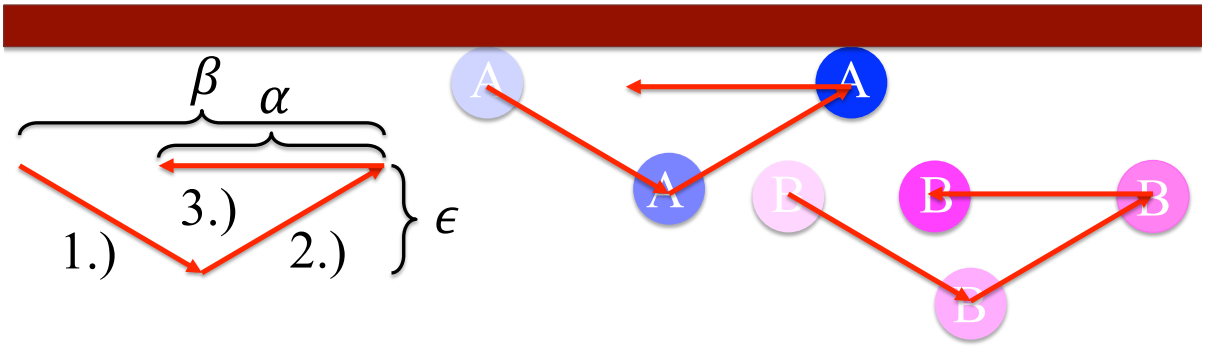


Figure 4.12: A DRIFTMOVE$(\alpha, \beta, \epsilon, 0°)$ repeats a triangular movement sequence $\{(\beta/2, -\epsilon),\ (\beta/2, \epsilon),\ (-\alpha, 0)\}$. At the sequence end, robot A has moved β units right, and robot B has moved $\beta - \alpha$ units right.

Let $(0, 0)$ be the lower left corner of the workspace, p_k the x, y position of the k^{th} robot, and f_k the final x, y position of the k^{th} robot. Label the robots in the staging zone from left-to-right and bottom-to-top, and the f_k configurations top-to-bottom and right-to-left as shown in Fig. 4.13.

Algorithm 6 PositionControl$n$Robots(k)

Move($-\epsilon, d/2 - p_{ky}$)
while $p_{kx} > d/2$ **do**
    DRIFTMOVE$(\epsilon, \min(p_{kx} - d/2, \epsilon), \epsilon, 180°)$
end while
$m \leftarrow \text{ceil}(\frac{f_{ky} - d/2}{\epsilon})$
$\beta \leftarrow \frac{f_{ky} - d/2}{m}$
$\alpha \leftarrow \beta - \frac{d/2 - p_{ky} - \epsilon}{m}$
for m iterations **do**
    DRIFTMOVE$(\alpha, \beta, \epsilon, 90°)$
end for
Move $(d/2 + \epsilon - f_{kx}, 0)$
Move $(f_{kx} - d/2, 0)$

Algorithm 7 DRIFTMOVE$(\alpha, \beta, \epsilon, \theta)$

particles touching the wall move β units, while particles not touching the wall move $\beta - \alpha$ units.

$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$
MOVE$(R \cdot [\beta/2, -\epsilon]^\top)$
MOVE$(R \cdot [\beta/2, \epsilon]^\top)$
MOVE$(R \cdot [-\alpha, 0]^\top)$

Alg. 6 proceeds as follows: First, the robots are moved left away from the right wall, and down so all robots in k^{th} row touch the bottom wall. Second, a set of DriftMoves are executed that move all robots in k^{th} row left until k touches the left wall, with no net movement of the other robots. Third, a set of DriftMoves are executed that move only robot k to its target height and return the other robots to their initial heights. Fourth, all robots except robot k are pushed left until robot k is in the correct relative x position compared to robots 1 to $k-1$. Finally, all robots are moved right until robot k is in the desired target position. Running time is $O(n(w+h))$.

The hardware platform depicted in Fig. 4.13 is an assembled practical setup that assumes that $\epsilon = 1$ cm. The workspace is a 7×7 cm grid space. All particles are 3D-printed plastic whose top is a 1cm diameter cylinder with a narrower base that encapsulates a steel bearing ball. Non-slip wall contact is generated by a toothed wall design to keep particles from moving out of place while implementing the drift move. The workspace boundary is mounted on top of a white sheet of cardboard. Underneath the cardboard, a grid of 3mm diameter magnets glued with 1 cm spacing to a thin board generates the uniform control input. A video attachment shows the algorithm at work. This discretized setup requires several modifications to Alg. 6. In this demonstration, all moves are 1 cm in length. All drift moves are a counterclockwise *square* move of size 1 cm$\times$1 cm. Once the k^{th} particle gets to its designated location in each loop, a correction step is implemented. This correction step increases by two the total number of moves required per particle. Fig. 4.11 shows there are only 6 stages per particle involved in Alg. 6. The fixed step algorithm requires 8 stages per particle as shown in Fig. 4.13.

A significant difference between Alg. 6 and the fixed move implementation of it is that Alg. 6 enables placing particles at arbitrary, non-overlapping locations, while the fixed move implementation requires goal locations at the center of grid cells.

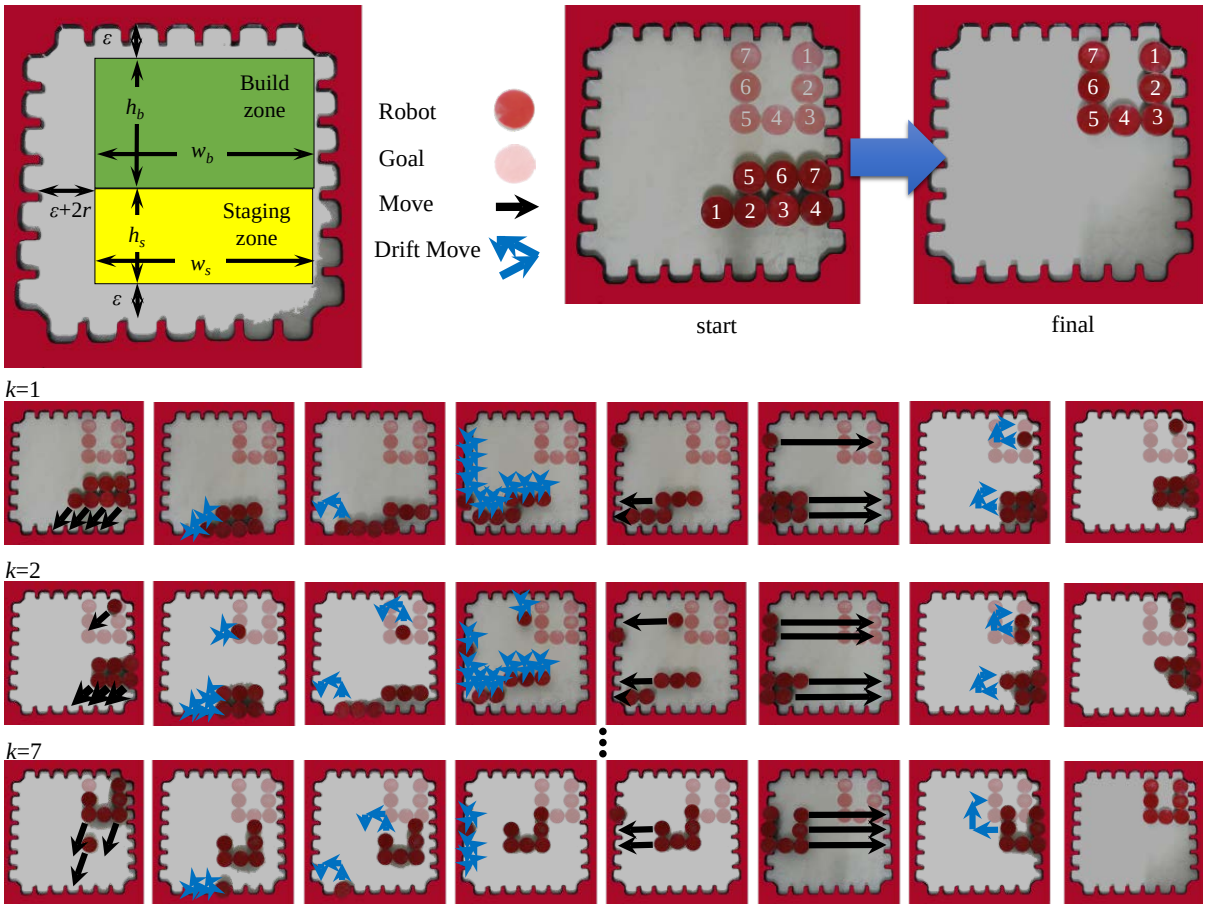


Figure 4.13: Illustration of Alg. 6, discretized n robot position control using walls that enforce non-slip contact.

4.6 Covariance control

Covariance control as well as mean position and variance control is needed when the swarm is moving through narrow passageways. This section discusses ideas to control covariance of the swarm using boundaries.

4.6.1 Using boundaries: stable configurations of a swarm

One method to control a swarm's shape in a bounded workspace is to simply push in a given direction until the swarm conforms to the boundary. Like fluid settling in a tank, the stable final configuration minimizes potential energy.

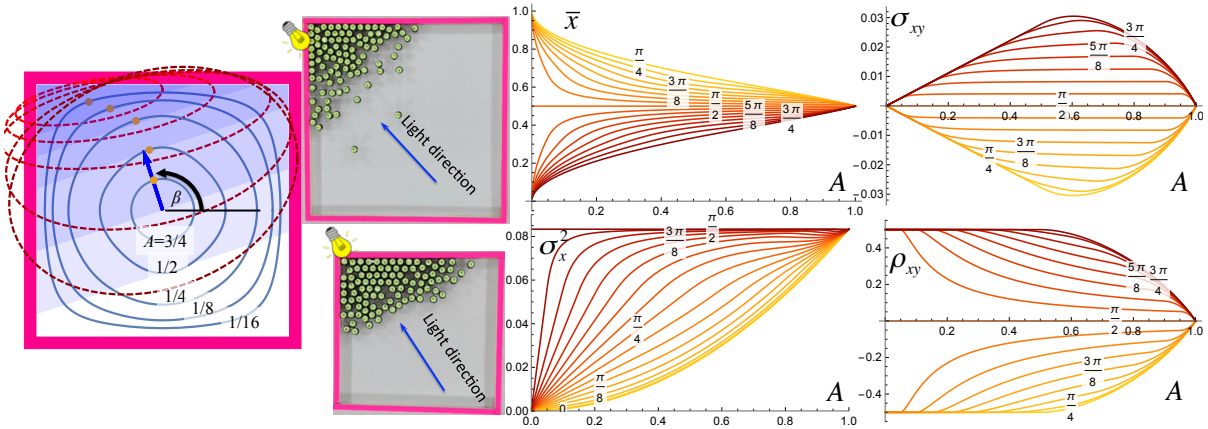


Figure 4.14: Pushing the swarm against a square boundary wall allows limited control of the shape of the swarm, as a function of swarm area A and the commanded movement direction β. Left plot shows locus of possible mean positions for five values of A. Center shows two corresponding arrangements of kilobots.

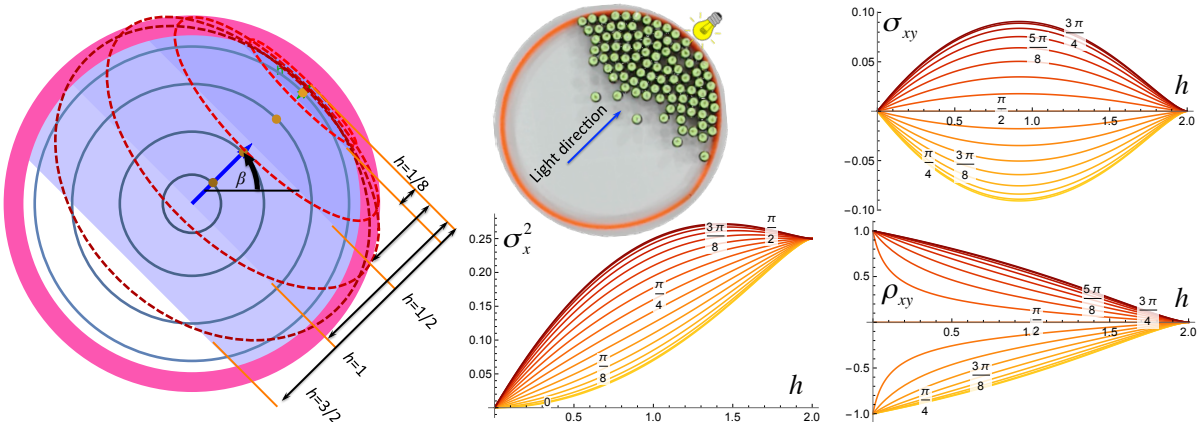


Figure 4.15: Pushing the swarm against a circular boundary wall allows limited control of the shape of the swarm, as a function of the fill level h and the commanded movement direction β. Left plot shows locus of possible mean positions for four values of h.

Square workspace We first examine the mean $(\bar{x}, \bar{y})$, covariance $(\sigma_x^2, \sigma_y^2, \sigma_{xy})$, and correlation ρ_{xy} of a large swarm of robots as they move inside a square workspace under the influence of a force pulling in the direction β. The swarm is large, but the robots are small in comparison, and together occupy a constant area A, $A \in [0, 1]$. Under a global input, the swarm moves to a side of the workspace and forms a polygonal shape that minimizes potential energy, as shown in Fig. 4.14, see also [83].

The range for the global input angle β is [0,2π). In this range, the swarm assumes eight different polygonal shapes. These shapes alternate between triangles and trapezoids when the area $A<1/2$, and between squares with one corner removed and trapezoids when $A>1/2$.

Computing means, variances, covariance, and correlation requires integrating over R, the region containing the swarm:

$$\bar{x} = \frac{\iint_R x\,dx\,dy}{A}, \qquad \bar{y} = \frac{\iint_R y\,dx\,dy}{A} \tag{4.10}$$

$$\sigma_x^2 = \frac{\iint_R (x-\bar{x})^2\;dx\,dy}{A},\ \sigma_y^2 = \frac{\iint_R (y-\bar{y})^2\;dx\,dy}{A} \tag{4.11}$$

$$\sigma_{xy} = \frac{\iint_R (x-\bar{x})\,(y-\bar{y})\;dx\,dy}{A},\ \rho_{xy} = \frac{\sigma_x^2}{\sigma_x\sigma_y} \tag{4.12}$$

The region of integration R is the polygon containing the swarm. For example, if $A < 1/2$ and the force angle is β, the mean when R is a triangular region in the lower-left corner is:

$$\begin{aligned}\bar{x}(A,\beta) &= \frac{\int_0^{\sqrt{2A\tan(\beta)}} \left(\int_0^{\cot(\beta)\left(\sqrt{2A\tan(\beta)}-x\right)} dy \right) x\,dx}{A} \\ &= \frac{\sqrt{2}}{3}\sqrt{A\tan(\beta)}\end{aligned} \tag{4.13}$$

$$\begin{aligned}\bar{y}(A,\beta) &= \frac{\int_0^{\sqrt{2A\tan(\beta)}} \left(\int_0^{\cot(\beta)\left(\sqrt{2A\tan(\beta)}-x\right)} y\,dy \right) dx}{A} \\ &= \frac{\sqrt{2}}{3}\sqrt{A\cot(\beta)}\end{aligned} \tag{4.14}$$

The full equations are included in the appendix [84], and are summarized in Fig. 4.14. A few highlights are that the correlation is maximized $\pm 1/2$ when the swarm is in a triangular shape. The covariance of a triangle is always $\pm(A/18)$. Variance is minimized

in the direction of β and maximized orthogonal to β when the swarm is in a rectangular shape. The range of mean positions are maximized when A is small.

Circular workspace Though rectangular boundaries are common in artificial workspaces, biological workspaces are usually rounded. Similar calculations can be made for a circular workspace. The workspace is a circle centered at (0,0) with radius 1 and thus area π. For notational simplicity, the swarm is parameterized by the global control input signal β and the fill-level h. Under a global input, the robot swarm fills the region under a chord with area

$$A(h) = \arccos(1-h) - (1-h)\sqrt{(2-h)h}. \tag{4.15}$$

For a circular workspace, the locus of mean positions are aligned with β and the mean position is at radius $r(h)$ from the center:

$$r(h) = \frac{2(-(h-2)h)^{3/2}}{3\left(\sqrt{-(h-2)h}(h-1) + \arccos(1-h)\right)} \tag{4.16}$$

Variance $\sigma_x^2(\beta, h)$ is maximized at $\beta = \pi/2 + n\pi$ and $h \approx 1.43$, while covariance is maximized at $\beta = \pi 3/4 + n\pi$ and $h \approx 0.92$. For small h values, correlation approaches ± 1. Results are displayed in Fig. 4.15, see also [85].

4.6.2 Using boundaries: friction and boundary layers

Global inputs move a swarm uniformly. Shape control requires breaking this uniform symmetry. A swarm inside an axis-aligned rectangular workspace can reduce variance normal to a wall by simply pushing the swarm into the boundary. If the swarm can flow around each other, pushing the swarm into a boundary produces the limited set of configurations presented in Sec. 4.6.1. Instead of pushing our robots directly into a wall, the following sections examine an oblique approach using boundaries that generate friction with the robots. These frictional forces are sufficient to break the symmetry caused

by uniform inputs. Robots touching a wall have a friction force that opposes movement along the boundary. This causes robots along the boundary to move more slowly than robots in free-space.

Let the control input be a vector force $\vec{F}$ with magnitude F and orientation θ with respect to a line perpendicular to and into the nearest boundary. N is the normal or perpendicular force between the robot and the boundary. The force of friction F_f is nonzero if the robot is in contact with the boundary and $\sin(\theta) < 0$. The resulting net force on the robot, $F_{forward}$, is aligned with the wall and given by

$$\begin{aligned} F_{forward} &= F\sin(\theta) - F_f \\ \text{where } F_f &= \begin{cases} \mu_f N, & \mu_f N < F\sin(\theta) \\ F\sin(\theta), & \text{else} \end{cases} \\ \text{and } N &= F\cos(\theta) \end{aligned} \tag{4.17}$$

Fig. 4.16 shows the resultant forces on two robots when one is touching a wall. Though each receives the same inputs, they experience different net forces. For ease of analysis, the following algorithms assume μ_f is infinite and robots touching the wall are prevented from sliding along the wall. This means that if one robot is touching the wall and another robot is free, the touching robot will not move when the control input is parallel or into the wall. There are many alternate models of friction that also break control symmetry. Fig. 4.16c shows fluid flow along a boundary. Fluid in the free-flow region moves uniformly, but flow decreases to zero in the boundary layer [86].

$$F_{forward}(y) = F - F_f \begin{cases} \frac{h-y}{h}, & y < h \\ 0, & \text{else} \end{cases} \tag{4.18}$$

The next section shows how a system in a rectangularly bounded workspace with friction model (4.17) can arbitrarily position two robots.

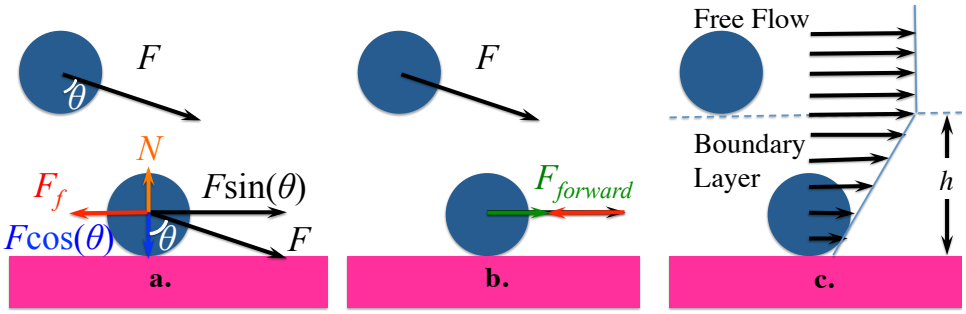


Figure 4.16: (a,b) Wall friction reduces the force for going forward $F_{forward}$ on a robot near a wall, but not for a free robot. (c) velocity of a fluid reduces to zero at the boundary.

4.6.3 Maximizing correlation using wall friction

Using the environment to individually move n particles to goal positions in Alg. 6 requires $O(n^2)$ time, while Alg. 3 can only control two particles. The controllers in Section 4.3 are efficient because they require only a single move but the range of possible configurations are limited. This section presents an efficient technique to generate desired correlations.

Assume an obstacle-free, bounded, unit-size, square workspace. As shown in Fig. 4.14, the maximum correlation occurs when the swarm is pushed in the direction $\beta = 3\pi/4$. This correlation as a function of swarm area A is never larger than $1/2$, and the maximum correlation decays to 0 as A grows to 1. By (4.12), this maximum correlation is:

$$\rho_{xy} = \begin{cases} \frac{1}{2}, & 0 \le A \le \frac{1}{2} \\ \frac{3A(2(A-2)A+1)}{4A^3-24A+\sqrt{2}(12A-12)\sqrt{1-A}+17} - 1, & \frac{1}{2} \le A \le 1 \end{cases} \tag{4.19}$$

If friction obeys the linear boundary layer model of (4.18) with boundary layer thickness h and maximum friction F_f equal to the maximum applied force F, we can generate larger correlations. If the swarm size is smaller than ≈ 0.43 and the boundary layer is sufficiently thick we can generate correlations larger than $1/2$ using boundary friction.

Assume that the swarm is initialized in the lower-left corner, in a rectangle of width w and height A/w. Such a rectangular configuration can be accomplished using

the variance controllers from [6]. If the swarm is then commanded to move a distance L to the right, components of the swarm outside the boundary friction layer of height h move further than components near the boundary. The swarm is contained in a region R composed of no more than three stacked components: at bottom a parallelogram inclined to the right top, at middle a rectangle, and at top a parallelogram inclined to the left top. These regions can be defined by the rectangle's left side, bottom, and top:

$$\begin{aligned} r_{\text{left}} &= \min(L, 1-w) \\ r_{\text{bottom}} &= \min\left(\frac{A}{w}, h\frac{r_{left}}{L}\right) \\ r_{\text{top}} &= \min\left(\frac{A}{w}, 1-h\frac{r_{left}}{L}\right) \end{aligned} \tag{4.20}$$

If $\frac{A}{w} \leq r_{\text{top}}$ the top parallelogram has no area. Similarly, if $r_{\text{top}} \leq r_{\text{bottom}}$ the rectangle has no area. The mean, variance, and correlation are calculated using (4.10), (4.11), and (4.12) over the region R:

$$\begin{aligned} \iint_R f(x,y)\,dx\,dy = &\int_0^{r_{\text{bottom}}} \int_{\frac{L}{h}y}^{\frac{L}{h}y+w} f(x,y)dx\,dy \\ &+ \int_{r_{\text{bottom}}}^{r_{\text{top}}} \int_{r_{\text{left}}}^{r_{\text{left}}+w} f(x,y)dx\,dy \\ &+ \int_{r_{\text{top}}}^{\frac{A}{w}} \int_{-\frac{L(y-r_{\text{top}})}{h}+r_{\text{left}}}^{r_{\text{left}}+w-\frac{L(y-r_{\text{top}})}{h}} f(x,y)dx\,dy \end{aligned} \tag{4.21}$$

Given an environment parameterized by A and h, efficient correlation control consists of choosing the w, L pair that generates the desired positive correlation. Negative correlations can be generated by initializing the swarm in the upper left, or lower right.

4.6.4 Efficient control of correlation

This section examines maximum correlation values as a function of w, L using (4.21) from Section 4.6.3. The maximum correlation using boundary layer friction

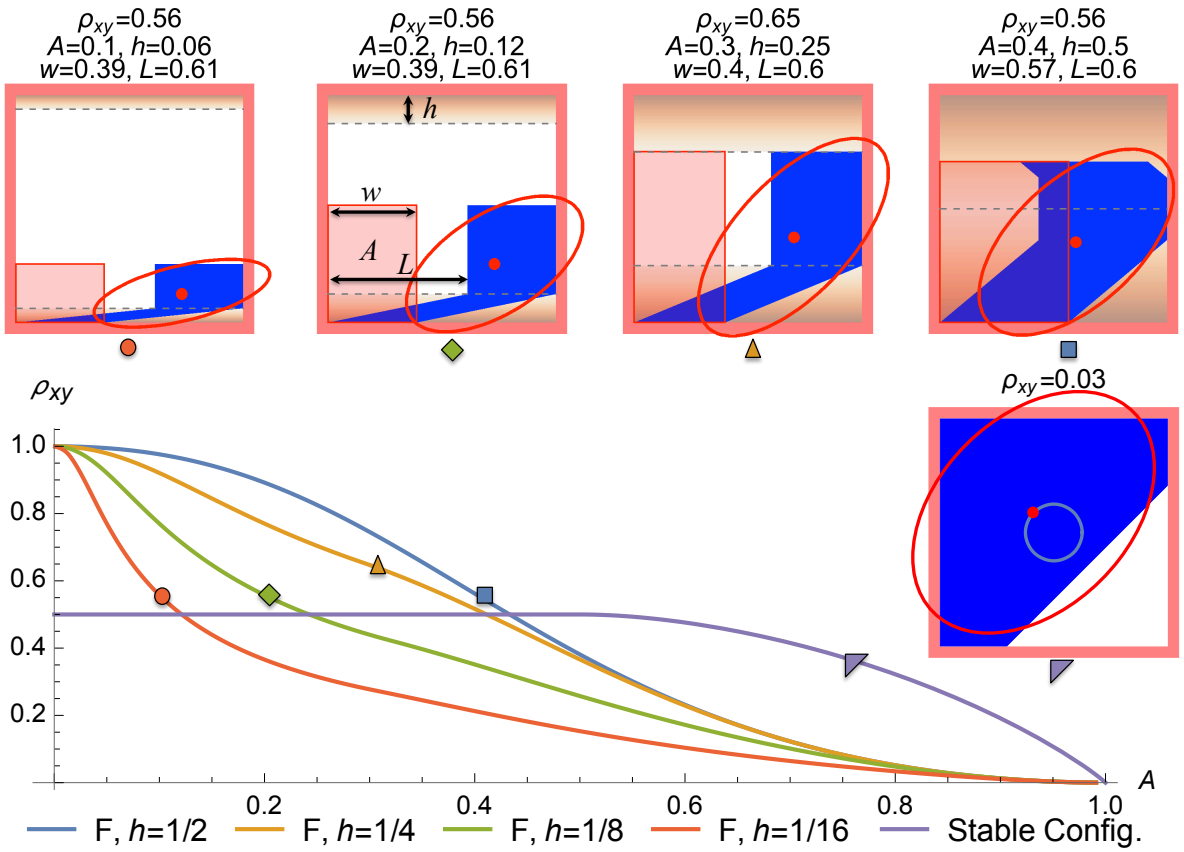


Figure 4.17: Analytical results comparing maximum correlation ρ_{xy} under the boundary layer friction model of (4.21) with four boundary layer thicknesses h and the stable triangular configuration (4.19).

$\max_{w,L}(\rho(A,h,w,L))$ can be found by gradient descent, as shown in Fig. 4.17. For swarms with small area, this method enables generating the full range of correlations ± 1, while the stable configuration method from Section 4.3 could only generate correlations $\pm 1/2$. As the swarm area A increases above ≈ 0.43, the stable configuration method is more effective. Larger boundary layers h enable more control of correlation. The multimedia attachment shows efficient control of correlation with simulations using the 2D physics engine Box2D [44] and 144 disc-shaped robots with boundary layer model (4.18).

4.6.5 Hardware experiment: control of covariance

To demonstrate covariance control, up to 100 kilobots were placed on the workspace and manually steered with lights, using friction with the boundary walls to vary the

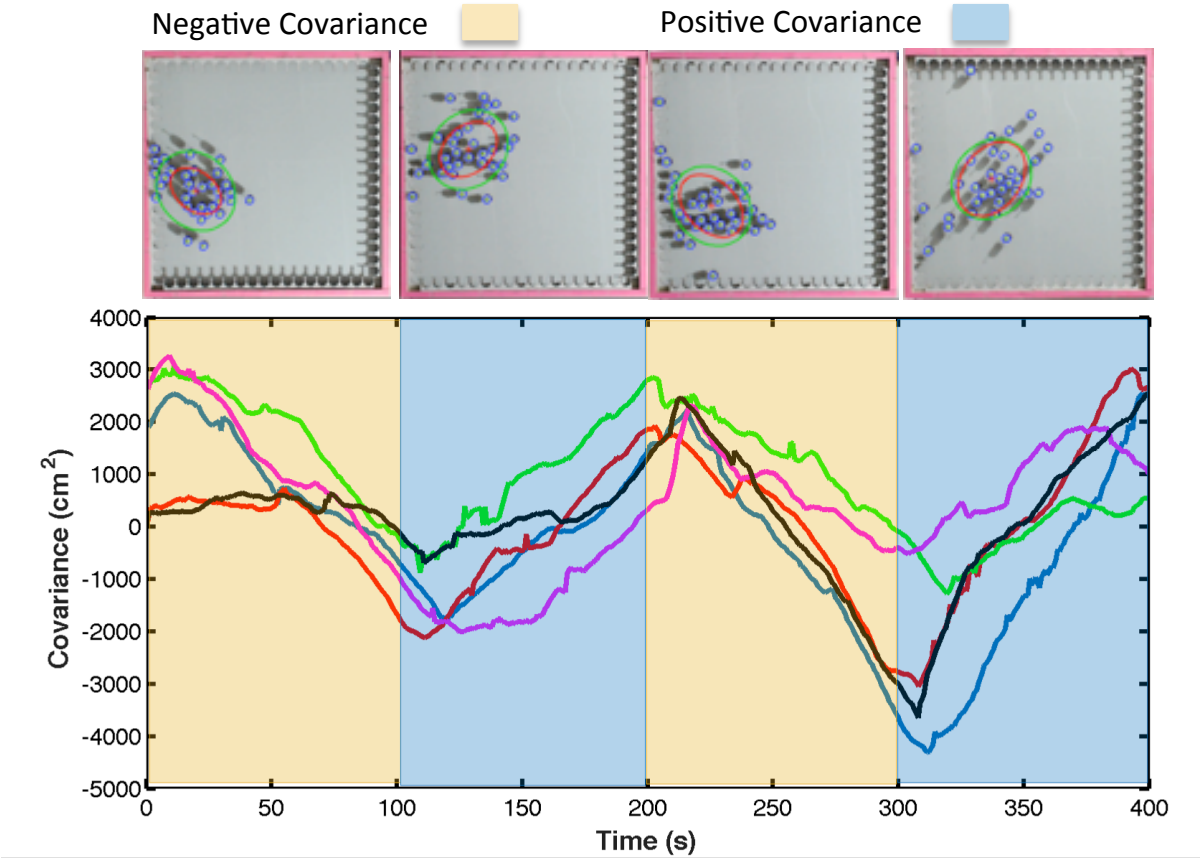


Figure 4.18: Hardware demonstration steering ≈ 50 kilobot robots to desired covariance. The goal covariance is negative in first 100 seconds and is positive in the next 100 seconds. The actual covariance is shown in different trials. Frames above the plot show output from machine vision system and an overlaid covariance ellipse.

covariance from -4000 to 3000 cm^2. The resulting covariance is plotted in Fig. 4.18, along with snapshots of the swarm.

4.7 Simulation

Algorithm 3 was implemented in Mathematica using particles with zero radius. Figure 4.19 shows frames of the algorithm in two representative workspaces, square and disk, with two arbitrary starting and goal configurations.

The contour plots in Fig. 4.20 left show the length of the path for two different

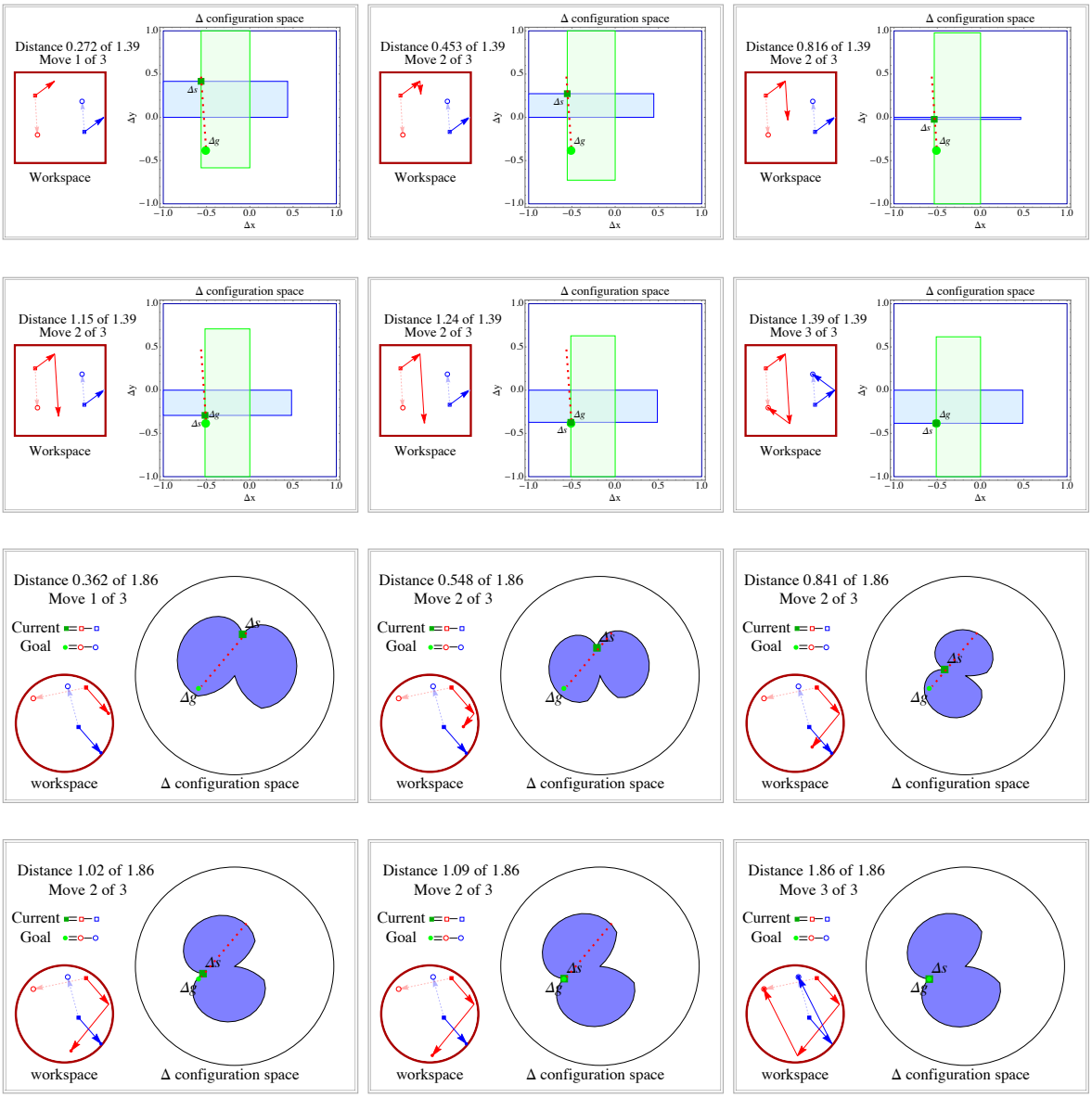


Figure 4.19: Frames from reconfiguring two particles. Top six images show a polygonal workspace and the corresponding Δ configuration space with its 2-move reachable sets. Bottom six images show a disk-shaped workspace and the corresponding Δ configuration space with its 2-move reachable sets.

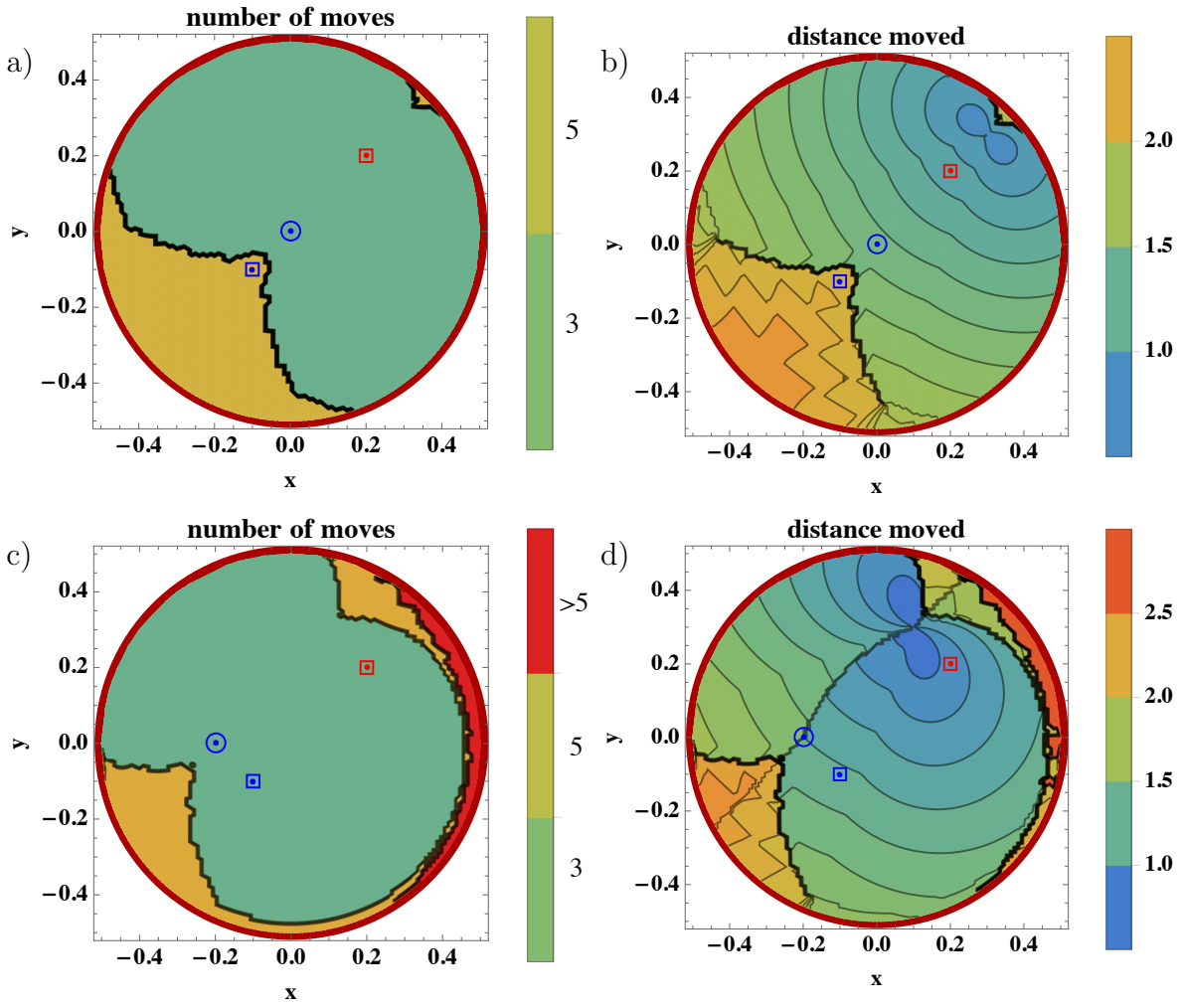


Figure 4.20: Plots show performance with one goal on the boundary.

settings. Top row considers $\{s_1, s_2, g_1\} = \{(0.2, 0.2), (-0.1, -0.1), (0, 0)\}$ and bottom row considers $\{s_1, s_2, g_1\} = \{(0.2, 0.2), (-0.1, -0.1), (-0.2, 0)\}$ each in a workspace with $r = 0.5$, and g_2 ranging over all the workspace. Fig. 4.20 right shows the number of moves and left shows the total distance of the path. This plot shows the nonlinear nature of the path planning. When the goal is in the middle of the workspace, a symmetry in the path length is expected as the top row shows. The bottom row shows a shift in the goal position which breaks the symmetry of the path length in the workspace.

The worst-case occurs when the ending points are at antipodes along the boundary (π angular distance). This can never be achieved but can be asymptotically approached

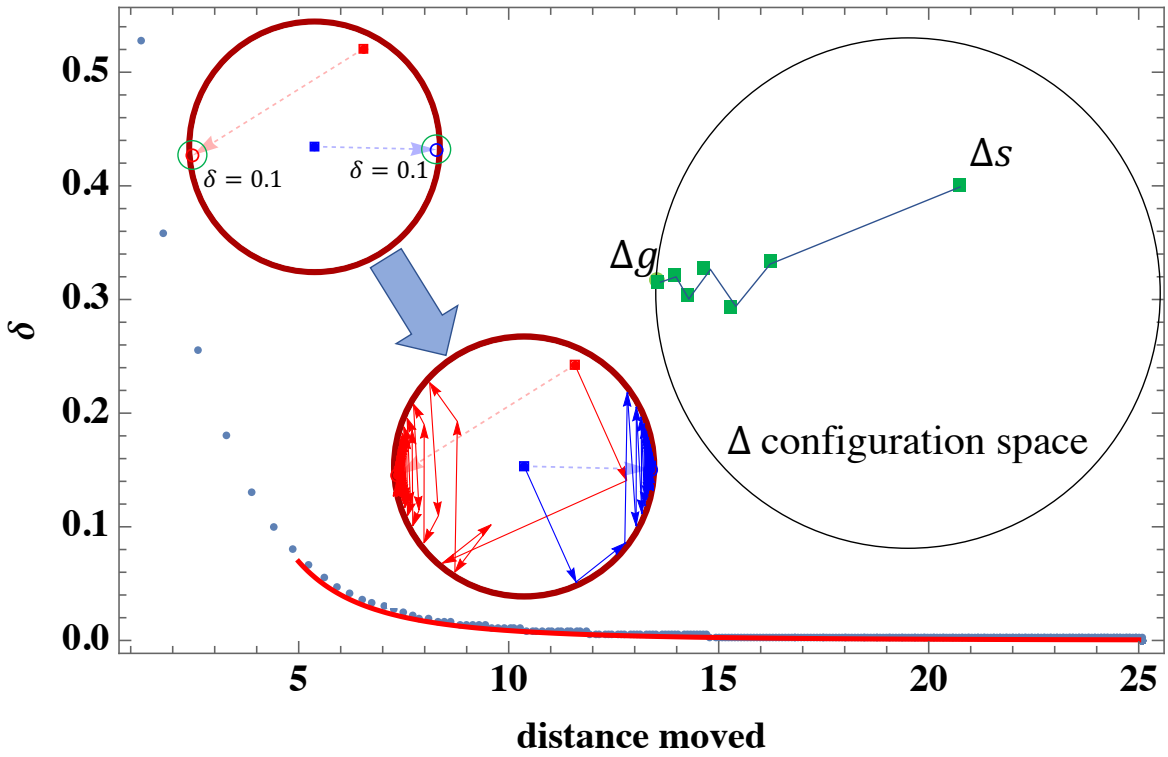


Figure 4.21: The worst-case path length occurs when particles must swap antipodes. This can never be achieved but can be asymptotically approached. Plot shows decreasing error as the number of moves grows.

as shown in Fig. 4.21. Figure 4.22 shows the same concepts in a square workspace. Figure 4.22 top and middle row considers the particles for three arbitrary starting and goal positions for the particles. Thus far, this paper has considered the particles to be unique. If particles are interchangeable, the path lengths often decrease. The bottom row of Fig. 4.22 considers interchangeable particles with the same configuration as the middle row with unique particles. The worst-case path lengths decrease by 33%, 60%, and 30% for the three test cases shown.

4.8 Experimental results

To demonstrate Alg. 3 experimentally, we performed several tests. Each used the same magnetic setup shown in Fig. 4.1. Two different intestine models were employed, the first a 3D-printed cross-section representation of a small intestine, and the second a

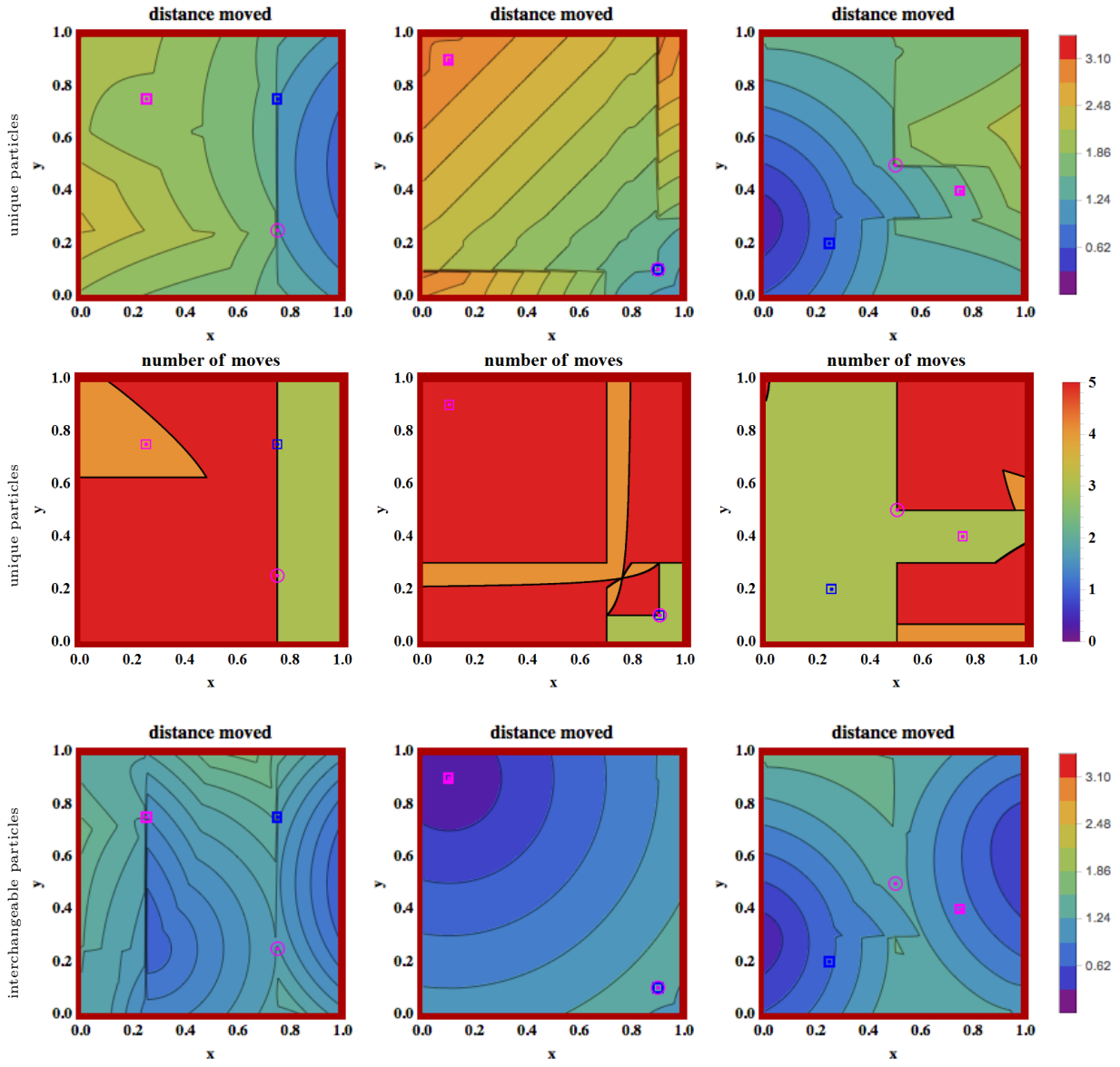


Figure 4.22: Starting positions of particles 1 and 2 and goal position of particle 2 are fixed, and $\epsilon = 0.001$. The top row of contour plots show the distance if particle 1's goal position is varied in x and y. The middle row shows the number of moves required for the same configurations. The bottom row shows the same configuration but when the particles are interchangeable.

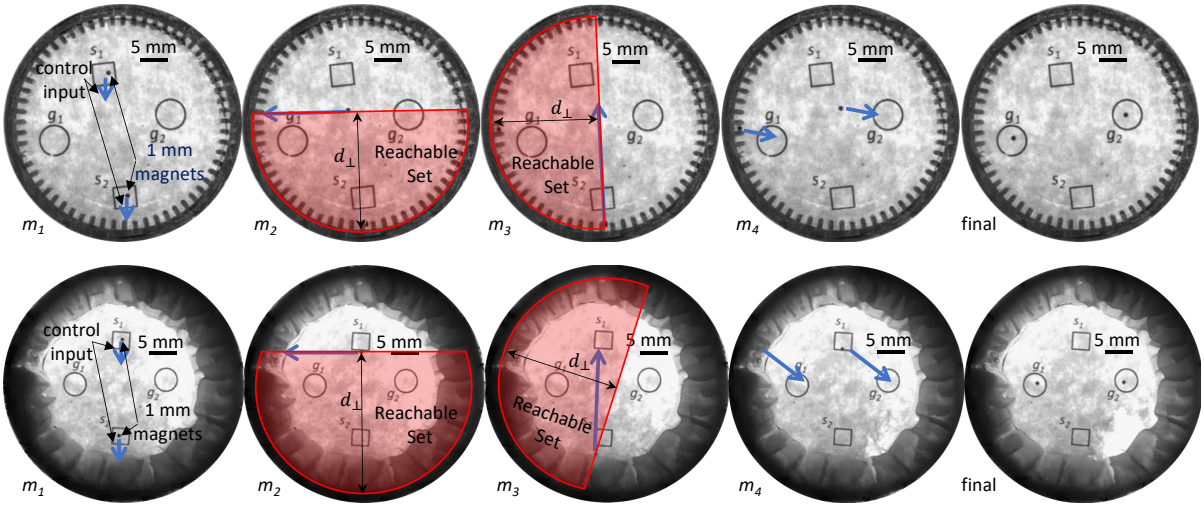


Figure 4.23: Frames showing particle positions before and after control inputs. Top row: small intestine phantom. Bottom row: cow stomach tissue.

cross-section of a bovine stomach.

4.8.1 Magnetic manipulation setup

The magnetic manipulation system has two pairs of electromagnetic coils, each with iron cores at their centers, and arranged orthogonal to each other. The iron core at the center of each coil concentrates the magnetic field towards the workspace. An Arduino and four SyRen regenerative motor drivers were used for control inputs to the coils. Finally, a FOculus F0134SB 659 x 494 pixel camera was attached to the top of the system, focusing on the workspace which was backlit by a 15 W LED light strip.

To obtain experimental data, the test samples (the phantom intestine model and the bovine cross section) were placed in laser-cut acrylic discs and then immersed in corn syrup. Corn syrup was used to increase the viscosity to 12000 cP for the experiments. Spherical 1 mm magnets (supermagnetman #SP0100-50) were used as our particles. Our experimental setup did not perfectly implement the system dynamics in (4.1). In particular, the magnetic field in this setup is only approximately uniform. The magnetic force varies in both magnitude and orientation. As shown in the video attachment, this

non-uniformity causes the particle closer to the coil to move faster than the other particle. This phenomenon makes it easier to increase particle separation than to decrease separation, but this can be compensated because boundary collisions easily decrease the separation. Also, magnetic forces are not exactly parallel, but point toward the center of the activated coil. Algorithm 3 still works despite these non-uniformities, but sometimes requires additional iterations.

4.8.2 Intestine phantom model

The intestine phantom model was used first and was made to mimic the geometry of an intestine and its villi. The model consists of a circular ring with an outer diameter of 50 mm, an inner diameter of 46 mm, and 60 2 mm long protrusions on its inner surface cut out of 6 mm thick acrylic to model the geometry of intestinal villi. Figure 4.23 top row shows an experiment. Starting and ending positions were printed beneath the workspace on transparency film. Our algorithm successfully delivered the particles to goal positions in 10 out of 10 trials.

4.8.3 Bovine stomach cross-section

Strips of cow stomach approximately 5 mm thick were cut and sewn to acrylic cylinder and then glued to an acrylic substrate using cyanoacrylate (superglue). This assembly was then filled with corn syrup. The experiment is shown in Fig. 4.23 bottom row. Our algorithm successfully delivered the particles to goal positions in 5 out of 5 trials. A video showing one trial of this experiment is available in the supplementary materials.

4.9 Conclusion

This chapter presented techniques for controlling the positions of two particles using uniform inputs and non-slip boundary contacts. The chapter provided algorithms for precise position control. The algorithms relied on calculating reachable sets in a 2D Δ configuration space. Extending Alg. 3 to 3D was straightforward, but increased the complexity. Hardware experiments illustrated the algorithms in ex vivo and in artificial workspaces that mimic the geometry of biological tissue.

This chapter assumed friction was sufficient to completely stop particles in contact with the boundary. The algorithms would require retooling to handle small friction coefficients.

Chapter 5

Conclusion

The small size of micro and nano particles makes individual control and autonomy challenging, so currently these particles are steered by global control inputs such as magnetic fields or chemical gradients.

Inspired by the human players in SwarmControl.net, this dissertation designed controllers and controllability results using only the mean and variance of a particle swarm. We developed a hysteresis-based controller to regulate the position and variance of a swarm. We designed a controller for object manipulation using value iteration for path planning, regions for outlier rejection, and potential fields for minimizing moving the object backwards. All automatic controllers were implemented using 100 kilobots steered by the direction of a global light source. These experiments culminated in an object manipulation task in a workspace with obstacles. This dissertation also presented techniques for controlling the orientation of an object by manipulating it using a swarm of simple robots with global inputs. It provided algorithms for precise orientation control, as well as demonstrations of orientation control. This dissertation finally presented techniques for controlling the positions of two particles using uniform inputs and non-slip boundary contacts. It provided algorithms for precise position control. The algorithms relied on calculating reachable sets in a 2D Δ configuration space. Extending Alg. 3 to 3D was straightforward, but increased the complexity. Hardware experiments illustrated the algorithms in ex vivo and in artificial workspaces that mimic the geometry of biological tissue.

5.1 Future work

Today there are more robots than ever before. This dissertation provides foundational algorithms and techniques for steering swarms, object manipulation, and addressing obstacle fields, but there are many opportunities to extend the work. Our future goal is to perform assembly using particle swarms to manipulate and attach components. Future efforts should be directed toward examining the effects of Brownian noise, pose control for multiple-part assembly, trajectory prediction, and manipulation in crowded workspaces. We assumed friction was sufficient to completely stop particles in contact with the boundary. The algorithms would require retooling to handle small friction coefficients.

Topics of interest include control with nonuniform flow such as fluid in an artery, gradient control fields like that of an MRI, competitive playing, multi-modal control, flexible workspaces, optimal-control, and targeted drug delivery in a vascular network. In out experiments, size of particles were always in millimeter scale or bigger. Future work should consider smaller particles and apply the proposed methods inside a fluid in an MRI machine. That would open up possibilities for non-invasive surgeries and drug delivery.

References

[1] K. E. Peyer, L. Zhang, and B. J. Nelson, "Bio-inspired magnetic swimming microrobots for biomedical applications," *Nanoscale*, 2013.

[2] 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.

[3] 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.

[4] B. J. Nelson, I. K. Kaliakatsos, and J. J. Abbott, "Microrobots for minimally invasive medicine," *Annual review of biomedical engineering*, vol. 12, pp. 55–85, 2010.

[5] S. Martel, S. Taherkhani, M. Tabrizian, M. Mohammadi, D. de Lanauze, and O. Felfoul, "Computer 3d controlled bacterial transports and aggregations of microbial adhered nano-components," *Journal of Micro-Bio Robotics*, vol. 9, no. 1-2, pp. 23–28, 2014.

[6] S. Shahrokhi and A. T. Becker, "Stochastic swarm control with global inputs," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sep. 2015, pp. 421–427.

[7] A. Becker, G. Habibi, J. Werfel, M. Rubenstein, and J. McLurkin, "Massive uniform manipulation," in *IEEE International Conference on Intelligent Robots and Systems*, Nov. 2013.

[8] K. Sugawara, N. Correll, and D. Reishus, "Object transportation by granular convection using swarm robots," in *Distributed autonomous robotic systems*. Springer, 2014, pp. 135–147.

[9] A. Becker, G. Habibi, J. Werfel, M. Rubenstein, and J. McLurkin, "Massive uniform manipulation: Controlling large populations of simple robots with a common input signal," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2013, pp. 520–527.

[10] S. Shahrokhi, L. Lin, C. Ertel, M. Wan, and A. T. Becker, "Steering a swarm of particles using global inputs and swarm statistics," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 207–219, 2018.

[11] D. Milutinovic and P. Lima, "Modeling and optimal centralized control of a large-size robotic population," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1280–1285, 2006.

[12] A. Prorok, N. Correll, and A. Martinoli, "Multi-level spatial modeling for stochastic distributed robotic systems," *International Journal of Robotics Research*, vol. 30, no. 5, pp. 574–589, 2011.

[13] N. Demir and B. Açıkmeşe, "Probabilistic density control for swarm of decentralized on-off agents with safety constraints," in *American Control Conference (ACC), 2015*. IEEE, 2015, pp. 5238–5244.

[14] 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.

[15] 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.

[16] S. Floyd, E. Diller, C. Pawashe, and M. Sitti, "Control methodologies for a heterogeneous group of untethered magnetic micro-robots," *Int. J. Robot. Res.*, vol. 30, no. 13, pp. 1553–1565, Nov. 2011.

[17] E. Diller, J. Giltinan, and M. Sitti, "Independent control of multiple magnetic microrobots in three dimensions," *The International Journal of Robotics Research*, vol. 32, no. 5, pp. 614–631, 2013. [Online]. Available: http://ijr.sagepub.com/content/32/5/614.abstract

[18] D. Frutiger, B. Kratochvil, K. Vollmers, and B. J. Nelson, "Magmites - wireless resonant magnetic microrobots," in *IEEE Int. Conf. Rob. Aut.*, Pasadena, CA, May 2008.

[19] S. Tottori, L. Zhang, F. Qiu, K. Krawczyk, A. Franco-Obregón, and B. J. Nelson, "Magnetic helical micromachines: Fabrication, controlled swimming, and cargo transport," *Advanced Materials*, vol. 24, no. 811, 2012.

[20] 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, Jun. 2012.

[21] 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)*, Oct. 2012.

[22] T. Bretl, "Control of many agents using few instructions." in *Robotics: Science and Systems*, 2007.

[23] A. Becker, C. Onyuksel, T. Bretl, and J. McLurkin, "Controlling many differential-drive robots with uniform control inputs," *Int. J. Robot. Res.*, vol. 33, no. 13, pp. 1626–1644, 2014.

[24] K. Takahashi, K. Hashimoto, N. Ogawa, and H. Oku, "Organized motion control of a lot of microorganisms using visual feedback," in *IEEE Int. Conf. Rob. Aut.*, May 2006, pp. 1408–1413. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1641906&isnumber=34383

[25] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[26] P. Abbeel, A. Coates, and A. Y. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1608–1639, 2010.

[27] N. Ayanian, A. Spielberg, M. Arbesfeld, J. Strauss, and D. Rus, "Controlling a team of robots with a single input," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1755–1762.

[28] D. R. Olsen Jr and S. B. Wood, "Fan-out: Measuring human control of multiple robots," in *SIGCHI Conference on Human Factors in Computing Systems*, Vienna, Austria, Apr. 2004, pp. 231–238.

[29] S. Bashyal and G. K. Venayagamoorthy, "Human swarm interaction for radiation source search and localization," in *Swarm Intelligence Symposium (SIS)*. IEEE, 2008, pp. 1–8.

[30] J.-P. de la Croix and M. Egerstedt, "Controllability characterizations of leader-based swarm interactions," in *2012 AAAI Fall Symposium Series*, 2012.

[31] A. Kolling, S. Nunnally, and M. Lewis, "Towards human control of robot swarms," in *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM, 2012, pp. 89–96.

[32] A. Sudsang, F. Rothganger, and J. Ponce, "Motion planning for disc-shaped robots pushing a polygonal object in the plane," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 550–562, Aug. 2002.

[33] J. Fink, M. Hsieh, and V. Kumar, "Multi-robot manipulation via caging in environments with obstacles," in *Robotics and Automation, ICRA IEEE International Conference on*, May 2008, pp. 1471–1476.

[34] K. Lynch, "Locally controllable manipulation by stable pushing," *IEEE Trans. Robot. Autom.*, vol. 15, no. 2, pp. 318–327, Apr. 1999.

[35] M. D. Armani, S. V. Chaudhary, R. Probst, and B. Shapiro, "Using feedback control of microflows to independently steer multiple particles," *Journal of Microelectromechanical systems*, vol. 15, no. 4, Aug. 2006.

[36] A. Becker, R. Sandheinrich, and T. Bretl, "Automated manipulation of spherical objects in three dimensions using a gimbaled air jet," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, oct. 2009, pp. 781 –786. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5354427

[37] L. U. Odhner, L. P. Jentoft, M. R. Claffee, N. Corson, Y. Tenzer, R. R. Ma, M. Buehler, R. Kohout, R. D. Howe, and A. M. Dollar, "A compliant, underactuated hand for robust manipulation," *The International Journal of Robotics Research*, vol. 33, no. 5, pp. 736–752, 2014.

[38] R. Deimel and O. Brock, "A novel type of compliant, underactuated robotic hand for dexterous grasping," *Robotics: Science and Systems, Berkeley, CA*, pp. 1687–1692, 2014.

[39] C. Ertel, S. Shahrokhi, and A. T. Becker, "SwarmControl.net git repository," Aug. 2016. [Online]. Available: https://github.com/RoboticSwarmControl/SwarmControlRedux.git

[40] S. Har-Peled, "On the expected complexity of random convex hulls," *arXiv preprint arXiv:1111.5340*, 2011.

[41] A. Einstein, *Investigations on the Theory of the Brownian Movement*. Courier Corporation, 1956.

[42] R. L. Graham and N. J. Sloane, "Penny-packing and two-dimensional codes," *Discrete & Computational Geometry*, vol. 5, no. 1, pp. 1–11, 1990.

[43] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *Robotics, IEEE Transactions on*, vol. 23, no. 2, pp. 320–330, 2007.

[44] E. Catto, "User manual, Box2D: A 2D physics engine for games, http://www.box2d.org," 2010.

[45] S. Shahrokhi, M. Wan, L. Lin, and A. T. Becker, "Steering Swarm Simulation, http:goo.gl/qsVqTU," Aug. 2016.

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

[47] A. T. Becker, "MDP robot grid-world example, https://www.mathworks.com/matlabcentral/fileexchange/49992-mdp-robot-grid-world-example," Mar. 2015.

[48] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. John Wiley & Sons, 2008.

[49] D. L. Christensen, S. A. Suresh, K. Hahm, and M. R. Cutkosky, "Let's all pull together: Principles for sharing large loads in microrobot teams," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1089–1096, 2016.

[50] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *IEEE Int. Conf. Rob. Aut.*, May 2012, pp. 3293–3298.

[51] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014.

[52] K-Team, "Kilobot, www.k-team.com," 2015.

[53] S. Shahrokhi, L. Lin, M. Wan, and A. T. Becker, "Kilobot Swarm Control using Matlab + Arduino, http://mathworks.com/matlabcentral/fileexchange/58690," Aug. 2016.

[54] S. Shahrokhi and A. T. Becker, "Object manipulation and position control using a swarm with global inputs," in *IEEE Conference on Automation Science and Engineering (CASE)*, Aug. 2016, pp. 1–6.

[55] J. Fink, N. Michael, and V. Kumar, "Composition of vector fields for multi-robot manipulation via caging," in *Robotics Science and Systems*, Atlanta, GA, Jun. 2007.

[56] K. M. Lynch, "Nonprehensile robotic manipulation: Controllability and planning," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, Mar. 1996.

[57] 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.

[58] J. D. Bernheisel and K. M. Lynch, "Stable transport of assemblies by pushing," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 740–750, Aug. 2006.

[59] S. Martel, "Magnetotactic bacteria for the manipulation and transport of micro-and nanometer-sized objects," *Micro-and Nanomanipulation Tools*, pp. 308–317, 2015.

[60] Y. Ou, D. H. Kim, P. Kim, M. J. Kim, and A. A. Julius, "Motion control of tetrahymena pyriformis cells with artificial magnetotaxis: model predictive control (MPC) approach," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 2492–2497.

[61] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *IEEE Trans. Robotics Automat.*, vol. 17, pp. 947–951, 2001.

[62] M. A. Hsieh, V. Kumar, and L. Chaimowicz, "Decentralized controllers for shape generation with robotic swarms," *Robotica*, vol. 26, no. 05, pp. 691–701, 2008.

[63] X. Yan, Q. Zhou, J. Yu, T. Xu, Y. Deng, T. Tang, Q. Feng, L. Bian, Y. Zhang, A. Ferreira, and L. Zhang, "Magnetite nanostructured porous hollow helical microswimmers for targeted delivery," *Advanced Functional Materials*, vol. 25, no. 33, pp. 5333–5342, 2015.

[64] D. L. Christensen, S. A. Suresh, K. Hahm, and M. R. Cutkosky, "Let's all pull together: Principles for sharing large loads in microrobot teams," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1089–1096, 2016.

[65] S. Shahrokhi and A. T. Becker. (2016, March). [Online]. Available: https://github.com/aabecker/SwarmControlSandbox/blob/master/TorqueControl/Torque.html

[66] ——. (2016, March). [Online]. Available: https://github.com/aabecker/SwarmControlSandbox/blob/master/TorqueControl/Orientation.html

[67] S. Chowdhury, W. Jing, and D. J. Cappelleri, "Controlling multiple microrobots: recent progress and future challenges," *Journal of Micro-Bio Robotics*, vol. 10, no. 1-4, pp. 1–11, 2015.

[68] A. L. Bertozzi, T. Kolokolnikov, H. Sun, D. Uminsky, and J. Von Brecht, "Ring patterns and their bifurcations in a nonlocal model of biological swarms," *Communications in Mathematical Sciences*, vol. 13, no. 4, pp. 955–985, 2015.

[69] B. R. Donald, C. G. Levey, I. Paprotny, and D. Rus, "Planning and control for microassembly of structures composed of stress-engineered mems microrobots," *The International Journal of Robotics Research*, vol. 32, no. 2, pp. 218–246, 2013.

[70] T. Bretl, "Control of many agents using few instructions," in *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007, pp. 1–8.

[71] Z. Nosrati, N. Li, F. Michaud, S. Ranamukhaarachchi, S. Karagiozov, G. Soulez, S. Martel, K. Saatchi, and U. O. Hfeli, "Development of a coflowing device for the size-controlled preparation of magnetic-polymeric microspheres as embolization agents in magnetic resonance navigation technology," *ACS Biomaterials Science & Engineering*, vol. 4, no. 3, pp. 1092–1102, 2018.

[72] F. Lamiraux and L. E. Kavraki, "Positioning of symmetric and non-symmetric parts using radial and constant fields: Computation of all equilibrium configurations," *International Journal of Robotics Research*, vol. 20, no. 8, pp. 635–659, 2001.

[73] T. H. Vose, P. Umbanhowar, and K. M. Lynch, "Sliding manipulation of rigid bodies on a controlled 6-dof plate," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 819–838, 2012.

[74] S. Salmanipour and E. Diller, "Eight-degrees-of-freedom remote actuation of small magnetic mechanisms," in *IEEE International Conference on Robotics and Automation*, 2018.

[75] A. Denasi and S. Misra, "Independent and leader follower control for two magnetic micro-agents," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 218–225, Jan 2018.

[76] E. Diller, J. Giltinan, G. Z. Lum, Z. Ye, and M. Sitti, "Six-degree-of-freedom magnetic actuation for wireless microrobotics," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 114–128, 2016.

[77] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[78] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[79] E. M. Purcell, "Life at low Reynolds number," *American Journal of Physics*, vol. 45, no. 1, pp. 3–11, 1977. [Online]. Available: http://dx.doi.org/10.1119/1.10903

[80] S. Shahrokhi, A. Mahadev, and A. T. Becker, "Algorithms for shaping a particle swarm with a shared input by exploiting non-slip wall contacts," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, 2017.

[81] J. Shi and A. T. Becker, "Shortest path between two points in the unit disk reflecting off the circumference," Sep. 2017. [Online]. Available: http://demonstrations.wolfram.com/ShortestPathBetweenTwoPointsInTheUnitDiskReflectingOffTheCir/

[82] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE transactions on computers*, no. 2, pp. 108–120, 1983.

[83] H. Zhao and A. T. Becker, "Distribution of a robot swarm in a square under gravity, wolfram demonstrations project," Jan. 2016. [Online]. Available: http://demonstrations.wolfram.com/DistributionOfARobotSwarmInASquareUnderGravity/

[84] S. Shahrokhi, A. Mahadev, and A. T. Becker, "Algorithms for shaping a particle swarm with a shared control input using boundary interaction," *ArXiv e-prints*, Feb. 2017. [Online]. Available: http://arxiv.org/abs/1609.01830

[85] H. Zhao and A. T. Becker, "Distribution of a swarm of robots in a circular workplace under gravity, wolfram demonstrations project," Feb. 2016. [Online]. Available: http://demonstrations.wolfram.com/DistributionOfASwarmOfRobotsInACircularWorkplaceUnderGravity/

[86] P. J. Pritchard, *Fox and McDonald's Introduction to Fluid Mechanics, 8th Edition*. John Wiley and sons inc., 2011.