Steering a Swarm Using Global Inputs And Swarm Statistics

Shiva Shahrokhi, Chris Ertel, Aaron T. Becker*

May 19, 2016

Abstract

Micro- and nanorobotics have 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 to generating controlled motion. We want to use large swarms of robots to perform manipulation tasks; unfortunately, human-swarm interaction studies as conducted today are limited in sample size, are difficult to reproduce, and are prone to hardware failures. We present an alternative.

This paper first examines the perils, pitfalls, and possibilities we discovered by launching SwarmControl.net, an online game where players steer swarms of up to 500 robots to complete manipulation challenges. We record statistics from thousands of players, and use the game to explore aspects of large-population robot control. We present the game framework as a new, open-source tool for large-scale user experiments. One surprising result was that humans completed a block-pushing task *faster* when provided with only the mean and variance of the robot swarm than with full-state feedback. Inspired by human operators, this paper next investigates controllers that use only the mean and variance of a robot swarm. We prove that the mean position is controllable, then provide conditions under which variance is controllable. We next derive automatic controllers for these and a hybrid, hysteresis-based switching control to regulate the first two moments of the robot distribution. Finally, we employ these controllers as primitives for a block-pushing task and implement all the automatic controllers on 101 kilobots, with our vision system.

*S. Shahrokhi and A. Becker are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, 77004 USA sshahrokhi2@uh.edu,atbecker@uh.edu

1 Introduction

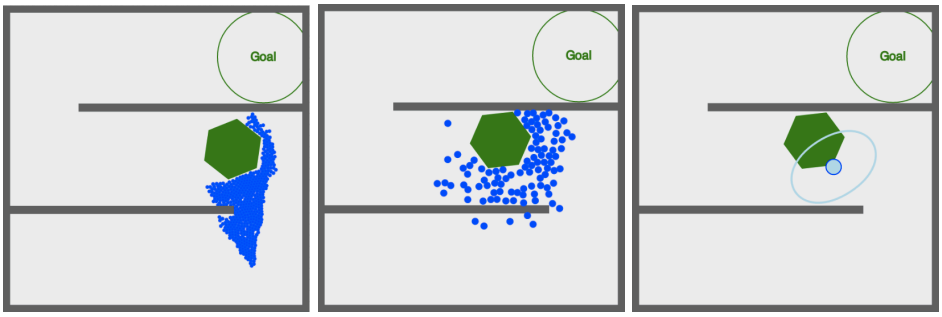
Large populations of micro- and nanorobots are being produced in laboratories around the world, with diverse potential applications in drug delivery and construction Peyer et al. (2013); Shirai et al. (2005); Chiang et al. (2011). These activities require robots that behave intelligently. Limited computation and communication rules out autonomous operation or direct control over individual units; instead we must rely on global control signals broadcast to the entire robot population. It is not always practical to gather pose information on individual robots for feedback control; the robots might be difficult or impossible to sense individually due to their size and location. However, it is often possible to sense global properties of the group, such as mean position and density. Finally, many promising applications will require direct human control, but user interfaces to thousands—or millions—of robots is a daunting human-swarm interaction (HSI) challenge.

The goal of this work is to provide a tool for investigating HSI methods through statistically significant numbers of experiments. There is currently no comprehensive understanding of user interfaces for controlling multi-robot systems with massive populations. We are particularly motivated by the sharp constraints in micro- and nanorobotic systems. For example, full-state feedback with 10^6 robots leads to operator overload. Similarly, the user interaction required to individually control each robot scales linearly with robot population. Instead, user interaction is often constrained to modifying a global input. This input may be nonstandard, such as the attraction/repulsion field from a scanning tunneling microscope (STM) tip.

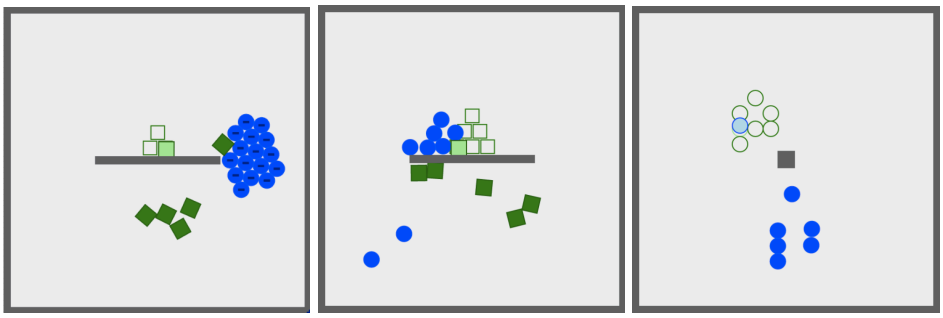
Our previous work with over a hundred hardware robots and thousands of simulated robots Becker et al. (2013) demonstrated that direct human control of large swarms is possible. Unfortunately, the logistical challenges of repeated experiments with over one hundred robots prevented large-scale tests.

Our goal was 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 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. 1. All code Ertel and Becker (2013), and experimental results are posted online.

Our experiments show that numerous simple robots responding to global control inputs are directly controllable by a human operator without special training, that the visual feedback of the swarm state should be very simple in



(a) Vary Number (b) Vary Visual Feedback



(c) Vary Control (d) Vary Noise (e) Control Position

Figure 1: Screenshots from our five online experiments controlling multi-robot systems with limited, global control. **(a)** Varying the number of robots from 1-500 **(b)** Comparing 4 levels of visual feedback **(c)** Comparing 3 control architectures **(d)** Varying noise from 0 to 200% of control authority **(e)** Controlling the position of 1 to 10 robots. See video overview at Shahrokhi and Becker (2015b)

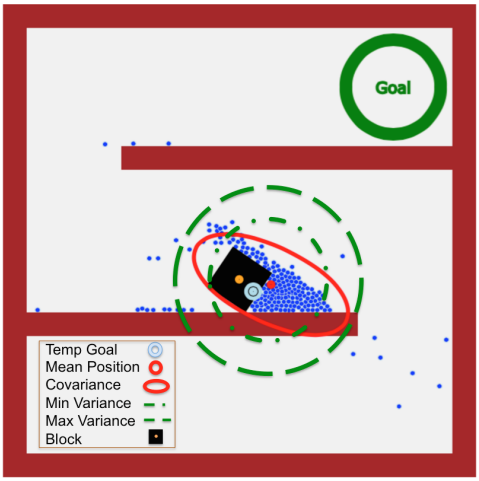


Figure 2: A swarm of robots, all controlled by a uniform force field, can be effectively controlled by a hybrid controller that knows only the first and second moments of the robot distribution. Here a swarm of simple robots (blue discs) pushes a black block toward the goal. See video attachment Shahrokhi and Becker (2015b).

order to increase task performance, and that humans perform swarm-object manipulation faster using attractive control schemes than repulsive control schemes.

Inspired by our online experiments, because it is not always possible to gather pose information on each robot for feedback control and robots might be difficult or impossible to sense individually due to their size and location. for example, micro-robots are smaller than the minimum resolution of a clinical MRI-scanner Martel et al. (2014), it is often possible to sense global properties of the group, such as mean position and variance. To make progress in automatic control with global inputs, this paper presents swarm manipulation controllers requiring only mean and variance measurements of the robot's positions. These controllers are used as primitives to perform a block-pushing task illustrated in Fig. 21.

Our paper is organized as follows. After a discussion of related work in Section 2,we will prove that mean and variance of the robots are controllable in Section 3, then we describe our experimental methods for an online human-user experiment in Section 4. We report the results of our online experiments in Section 4.2, and illustrate auto controllers that does the same thing in Section 5. We conclude with our implementation of the auto con-

trollers on our hardware robots and complete an object manipulation task in Section 7.

This paper is the synthesis of two preliminary conference papers, the first covering first few months of SwarmControl.net experiments, and the second with simulations of object manipulation. THis paper presents the final results from SwarmCOntorl.net. All hardware validation experiments with 100+ kilobots are new for this paper.

2 Related Work

2.1 Human-Swarm Interaction

Olson and Wood studied human *fanout*, the number of robots a single human user could control Olsen Jr and Wood (2004). 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, after which there were diminishing returns. They hypothesize that the location of this plateau is highly dependent on the underlying task, and our work indicated there are some tasks without plateaus. Their research investigated robots with 3 levels of autonomy. We use robots without autonomy, corresponding with their first-level robots.

Squire, Trafton, and Parasuraman designed experiments showing that user-interface design had a high impact on the task effectiveness and the number of robots that could be controlled simultaneously in a multi-robot task Squire et al. (2006).

A number of user studies compare methods for controlling large swarms of simulated robots, for example Bashyal and Venayagamoorthy (2008); Kolling et al. (2012); de la Croix and Egerstedt (2012). 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 de la Croix and Egerstedt (2012) was limited to a pool of 18 participants, Bashyal and Venayagamoorthy (2008) 5, and Kolling et al. (2012) 32. Using an online testing environment, we conduct similar studies but with much larger sample sizes.

2.2 Global-control of micro- and nanorobots

Small robots have been constructed 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 Donald et al. (2006, 2008); magnetic structures with different cross-sections that could be independently steered Floyd et al. (2011); Diller et al. (2013); *MagMite* microrobots with different resonant frequencies and a global magnetic field Frutiger et al. (2008); and magnetically controlled nanoscale helical screws constructed to stop movement at different cutoff frequencies of a global magnetic field Tottori et al. (2012); Peyer et al. (2013).

Similarly, our previous work Becker and Bretl (2012); Becker et al. (2012) focused on exploiting inhomogeneity between robots. These control algorithms theoretically apply to any number of robots—even robotic continuums—but in practice process noise cancels the differentiating effects of inhomogeneity for more than tens of robots. We desire control algorithms that extend to many thousands of robots.

2.3 Three challenges for massive manipulation

While it is now possible to create many micro- and nanorobots, there remain challenges in control, sensing, and computation.

2.3.1 Control—global inputs

Many micro- and nanorobotic systems Tottori et al. (2012); Shirai et al. (2005); Chiang et al. (2011); Donald et al. (2006, 2008); Takahashi et al. (2006); Floyd et al. (2011); Diller et al. (2013); Frutiger et al. (2008); Peyer et al. (2013) rely on global inputs, where each robot receives an exact copy of the control signal. Our experiments follow this global model.

2.3.2 Sensing—large populations

Parallel control of n differential-drive robots in a plane requires $3n$ state variables. Even holonomic robots require $2n$ state variables. Numerous methods exist for measuring this state in micro- and nanorobotics. 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 are increased at the nanoscale 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.

2.3.3 Computation—calculating the control law

In our previous work the controllers required at best a summation over all the robot states Becker et al. (2012) and at worst a matrix inversion Becker and Bretl (2012). 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.

2.4 Block-pushing and Compliant Manipulation

Unlike *caging* manipulation, where robots form a rigid arrangement around an object Sudsang et al. (2002); Fink et al. (2007), 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. Lynch (1999), 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 Armani et al. (2006); Becker et al. (2009).

Our n-robot system with 2 control inputs and $4n$ states is inherently under-actuated, and superficially bears resemblance to compliant, under-actuated manipulators Odhner et al. (2014); Deimel and Brock (2014). Like these manipulators, the swarm conforms to the object to be manipulated. However our swarm lacks the restoring force provided by flexures in Odhner et al. (2014) and the silicone in Deimel and Brock (2014). Our swarm tends to disperse itself, so we require artificial forces, such as the variance control primitives in Section 3.4, to regroup the swarm.

3 Theory

3.1 Models

Consider holonomic robots that move in the 2D plane. We want to control position and velocity of the robots. First, assume a noiseless system containing one robot with mass m. Our inputs are global forces $[u_x, u_y]$. 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:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t) \\ \mathbf{y}(t) &= C\mathbf{x}(t) + D\mathbf{u}(t)\end{aligned} \tag{1}$$

and our state space representation as:

$$\begin{bmatrix}\dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0\end{bmatrix} \begin{bmatrix}x_1 \\ x_2 \\ x_3 \\ x_4\end{bmatrix} + \begin{bmatrix}0 & 0 \\ \frac{1}{m} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m}\end{bmatrix} [u_x, u_y] \tag{2}$$

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

$$\mathcal{C} = [B, AB, A^2B, ..., A^{n-1}B]. \tag{3}$$

$$\text{Here } \mathcal{C} = \left[\begin{array}{cc|cc|cc|cc} 0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right], \tag{4}$$

and thus all four states are controllable.

3.2 Independent control with multiple robots is impossible

A single robot is fully controllable, but what happens with n robots? For holonomic robots, 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 robots to be controlled in the x axis, there are $2n$ states: n positions and n velocities. Our state-space representation is:

$$\begin{bmatrix}\dot{x}_{1,1} \\ \dot{x}_{2.1} \\ \vdots \\ \dot{x}_{1,n} \\ \dot{x}_{2,n}\end{bmatrix} = \begin{bmatrix}0 & 1 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 0\end{bmatrix} \begin{bmatrix}x_{1,1} \\ x_{2,1} \\ \vdots \\ x_{1,n} \\ x_{2,n}\end{bmatrix} + \begin{bmatrix}0 \\ 1 \\ \vdots \\ 0 \\ 1\end{bmatrix} u_x \tag{5}$$

However, just as with one robot, we can only control two states because $\mathcal{C}$ has rank two:

$$\mathcal{C} = \left[\begin{array}{c|c|c|c|c} 0 & 1 & 0 & 0 & \\ 1 & 0 & 0 & 0 & \\ \vdots & \vdots & \vdots & \vdots & , \ldots \\ 0 & 1 & 0 & 0 & \\ 1 & 0 & 0 & 0 & \end{array}\right] \tag{6}$$

3.3 Controlling Mean Position

This means *any* number of robots controlled by a global command have just two controllable states in each axis. We cannot arbitrarily control the position of two or more robots, but what states are controllable? One option is to control the position of the j^{th} robot. To find a potentially more useful option, we create the following reduced order system that represents the average x position and x velocity of the n robots:

$$\begin{bmatrix} \dot{\bar{x}}_1 \\ \dot{\bar{x}}_2 \end{bmatrix} = \frac{1}{n} \begin{bmatrix} 0 & 1 & \ldots & 0 & 1 \\ 0 & 0 & \ldots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{1,1} \\ x_{2,1} \\ \vdots \\ x_{1,n} \\ x_{2,n} \end{bmatrix}$$
$$+ \frac{1}{n} \begin{bmatrix} 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_x \tag{7}$$

Thus:

$$\begin{bmatrix} \dot{\bar{x}}_1 \\ \dot{\bar{x}}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_x \tag{8}$$

We again analyze $\mathcal{C}$:

$$\mathcal{C} = \left[\begin{array}{c|c} 0 & 1 \\ 1 & 0 \end{array}\right] \tag{9}$$

This matrix has rank two, and thus the average position and average velocity are controllable.

Due to symmetry of the control input, only the mean position and mean velocity are controllable. However, there are several techniques for breaking symmetry, for example by allowing independent noise sources Becker et al. (2014), or by using obstacles Becker et al. (2013).

3.4 Controlling the variance of many robots

The variance, σ_x^2, σ_y^2, of the swarm's position is computed:

$$\begin{aligned} \overline{x}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} x_{1,i}, \qquad \sigma_x^2(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_{1,i}-\overline{x})^2, \\ \overline{y}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} x_{3,i}, \qquad \sigma_y^2(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_{3,i}-\overline{y})^2. \end{aligned} \tag{10}$$

Controlling the variance requires being able to increase and decrease the variance. We will list a sufficient condition for each. Both conditions are readily found at the micro and nanoscale. Real systems, especially at the micro scale, are affected by unmodelled dynamics. These unmodelled dynamics are dominated by Brownian noise. To model this (1) must be modified as follows:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t) + W\boldsymbol{\varepsilon}(t) \\ \mathbf{y}(t) &= C\mathbf{x}(t) + D\mathbf{u}(t) \end{aligned} \tag{11}$$

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

$$\dot{\sigma_x^2}(\mathbf{x}(t), \mathbf{u}(t) = 0) = W\boldsymbol{\varepsilon} \tag{12}$$

If robots 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,

$$[\sigma_x^2, \sigma_y^2] = \frac{1}{12}[(\ell_x - 2r)^2, (\ell_y - 2r)^2]. \tag{13}$$

A flat obstacle can be used to decrease variance. Pushing a group of dispersed robots against a flat obstacle will decrease their variance until the minimum-variance (maximum density) packing is reached. For large n, Graham and Sloan showed that the minimum-variance packing $\sigma^2_{optimal}(n, r)$ for n circles with radius r is $\approx \frac{\sqrt{3}}{\pi}(nr)^2 \approx 0.55(nr)^2$ Graham and Sloane (1990).

We will prove the origin is globally asymptotically stabilizable by using a control-Lyapunov function Lyapunov, translated and edited by A.T. Fuller (1992). A suitable Lyapunov function is squared variance error:

$$\begin{aligned} V(t, \mathbf{x}) &= \frac{1}{2}(\sigma^2(\mathbf{x}) - \sigma^2_{goal})^2 \\ \dot{V}(t, \mathbf{x}) &= (\sigma^2(\mathbf{x}) - \sigma^2_{goal})\dot{\sigma^2}(\mathbf{x}) \end{aligned} \tag{14}$$

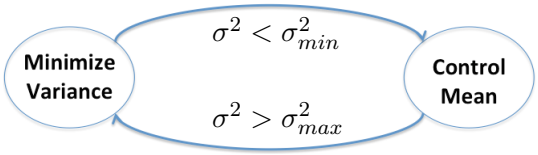


Figure 3: Two states for controlling the mean and variance of a robot swarm.

We note here that $V(t, \mathbf{x})$ is positive definite and radially unbounded, and $V(t, \mathbf{x}) \equiv 0$ only at $\sigma^2(\mathbf{x}) = \sigma^2_{goal}$. To make $\dot{V}(t, \mathbf{x})$ negative semi-definite, we choose

$$u(t) = \begin{cases} \text{move to wall} & \text{if } \sigma^2(\mathbf{x}) > \sigma^2_{goal} \\ \text{move from wall} & \text{if } \sigma^2(\mathbf{x}) \leq \sigma^2_{goal}. \end{cases} \tag{15}$$

For such a $u(t)$,

$$\dot{\sigma^2}(\mathbf{x}) = \begin{cases} \text{negative} & \text{if } \sigma^2(\mathbf{x}) > \max(\sigma^2_{goal}, \sigma^2_{optimal}(n, r)) \\ W\epsilon & \text{if } \sigma^2(\mathbf{x}) \leq \sigma^2_{goal}, \end{cases} \tag{16}$$

and thus $\dot{V}(t, \mathbf{x})$ is negative definite and the variance is globally asymptotically stabilizable.

3.5 Controlling both mean and variance of many robots

The mean and variance of the swarm cannot be controlled simultaneously, however if the dispersion due to Brownian motion is much less than the maximum controlled speed, we can adopt a hybrid, hysteresis-based controller to regulate the mean and variance shown in Alg. 1. Such a controller normally controls the mean position according to (19), but switches to minimizing variance if the variance exceeds some σ^2_{max}. The variance is lowered to less than σ^2_{min}, and the system returns to controlling the mean position. This is a standard technique for dealing with control objectives that evolve at different rates Sadraddini and Belta (2015); Kloetzer and Belta (2007), and the hysteresis avoids rapid switching between control modes. The process is illustrated in Fig. 3.

A key challenge is to select proper values for σ^2_{min} and σ^2_{max}. The optimal packing variance is $\sigma^2_{optimal}(n, r) = \frac{\sqrt{3}}{\pi} n r^2$. The random packings generated by pushing our robots into corners are suboptimal, so we choose

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 $[x_{target}, y_{target}]$.

$flag_x \leftarrow$ **false**, $flag_y \leftarrow$ **false**
$x_{goal} \leftarrow x_{target}$, $y_{goal} \leftarrow y_{target}$
loop
    if $\sigma_x^2 > \sigma_{max}^2$ **then**
        $x_{goal} \leftarrow x_{min}$
        $flag_x \leftarrow$ **true**
    else if $flag_x$ **and** $\sigma_x^2 < \sigma_{min}^2$
        $x_{goal} \leftarrow x_{target}$
        $flag_x \leftarrow$ **false**
    end if
    if $\sigma_y^2 > \sigma_{max}^2$ **then**
        $y_{goal} \leftarrow y_{min}$
        $flag_y \leftarrow$ **true**
    else if $flag_y$ **and** $\sigma_y^2 < \sigma_{min}^2$
        $y_{goal} \leftarrow y_{target}$
        $flag_y \leftarrow$ **false**
    end if
    Apply (19) to move toward $[x_{goal}, y_{goal}]$
end loop

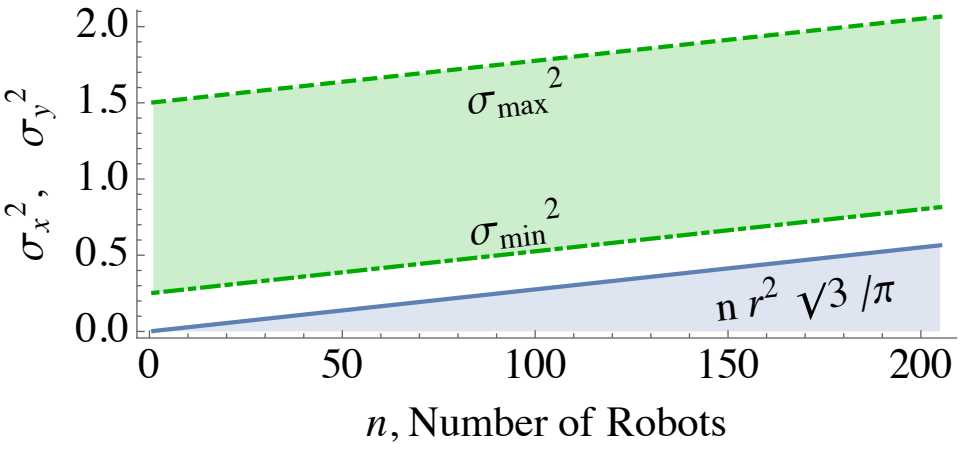


Figure 4: The switching conditions for variance control are set as a function of n, and designed to be larger than the optimal packing density. The above plot uses robot radius $r = 1/10$.

the conservative values shown in Fig. 4:

$$\sigma_{min}^2 = 2.5r + \sigma_{optimal}^2(n, r)$$
$$\sigma_{max}^2 = 15r + \sigma_{optimal}^2(n, r). \tag{17}$$

4 Online Experiment

4.1 Methods

We have developed a flexible testing framework for online human-swarm interaction studies. There are two halves to our framework: the server backend and the client-side (in-browser) frontend. The server backend is responsible for tabulating results, serving webpages containing the frontend code, and for issuing unique identifiers to each experiment participant. The in-browser frontend is responsible for running an experiment—that is to say, accepting user input, updating the state of the robot swarm, and ultimately evaluating task completion.

Overview A participant visits the site, initiating a communication between their browser and our server. The web server generates a unique identifier for the participant and sends it along with the landing page to the participant—this identifier is stored as a browser cookie and will be sent

along with all results the participant generates. The participant's browser prompts for confirmation of the terms-of-service and offers a menu of experiments.

Once the participant selects an experiment, their browser makes a new request to the server to load the experiment's webpage. The server sends scaffold HTML describing the layout of the page and a script block containing the experiment. The script runs the experiment and, upon a successful completion, posts the experiment data to the server. The participant is then given the option of playing again or trying a different experiment.

A participant may view all of the experimental data we have gathered; this information is available as either a webpage, a JSON file, or a comma-separated value file.

Backend The server backend is written in Ruby, using the Ruby-on-Rails (abbreviated *Rails*) web development framework. Ruby is a dynamically-typed object-oriented scripting language with a strong emphasis on programmer ergonomics and metaprogramming support. It is well-suited for the creation of domain-specific languages for a variety of tasks, as exemplified by the Rails framework. Our backend serves assets (images, scripts, stylesheets, and so forth) to participants, selects the correct script to send to perform a particular experiment, and stores results.

Each result is a database record containing the experiment name, the participant identifier, the duration of the experiment (time to completion), the number of robots involved, the detailed mode information of the experiment, and the user agent string of the browser running the experiment (which identifies the type of browser used by the participant). Rails automates the process of creating the relevant database-object bindings, and thus we spent little time creating or modifying the result records, allowing us to rapidly adapt the server to our needs–for example, adding tracking of the user agent and experiment mode both took less than five minutes of work on the server side.

The experiment script file to be sent to the client is chosen with the uniform resource identifier (URI) for the experiment webpage; this done, the server will render the page requested by the participant and insert the script for the selected experiment. The Rails framework has a great deal of support for optimizing and compacting (*minifying*) Javascript files.

Frontend The client frontend which runs in the participant's browser is written in Javascript, a dynamically-typed prototype-oriented scripting lan-

guage with some functional programming support. We make heavy use of the Underscore framework (a functional programming toolkit for Javascript) as well as the Javascript port of Box2D (a popular 2D physics engine with good support for rigid-body dynamics and fixed-timestep simulation). Our frontend also includes helper libraries for drawing robots, handling user input, and drawing graphs.

Our framework uses a base task to represent the lifecycle of an experiment—**instantiation, simulation, evaluation**, and **submission**. A particular experiment inherits from this prototype but overrides particular methods and adds its own variables for bookkeeping; this allows new or modified tasks to be created rapidly with minimal boilerplate code.

Our robots are disc-shaped, non-holonomic, and confined to the 2D plane. The control input u consists of a single bounded force vector that is applied to each robot, $|u| \leq u_{max}$. We include a linear ramp for this force value that starts at zero and increases to the maximum value in one second; this allows participants to do fine control of the robots by tapping the arrow keys.

$$\dot{x}_i = u_x, \qquad \dot{y}_i = u_y. \tag{18}$$

During the **instantiation** phase, an experiment sets up the web page elements with help text and other information, and creates the obstacles, robots, and workpieces that will be present during the experiment. It will also randomly select which mode to run in, if applicable.

The **simulation** phase is the time at which all of the robots are moved according to user input and given a chance to interact with each other and the environment. The simulation phase then draws the current state of the experiment to the canvas of the webpage.

The **evaluation** phase is when the experiment's completion criteria are applied to the current experiment state: are the robots in the goal zone, are the workpieces in the correct place, and so forth. If the criteria are not met, the experiment loops back into the simulation phase; if they are met, then the experiment proceeds to result submission.

The **submission** phase is when the results of the experiment are combined with other user data, such as the browser user agent string, and submitted to the server for collection.

4.1.1 Human subjects

Because our study involved recording data from human subjects, it required IRB approval before we could legally save user data (IRB #14-012E).

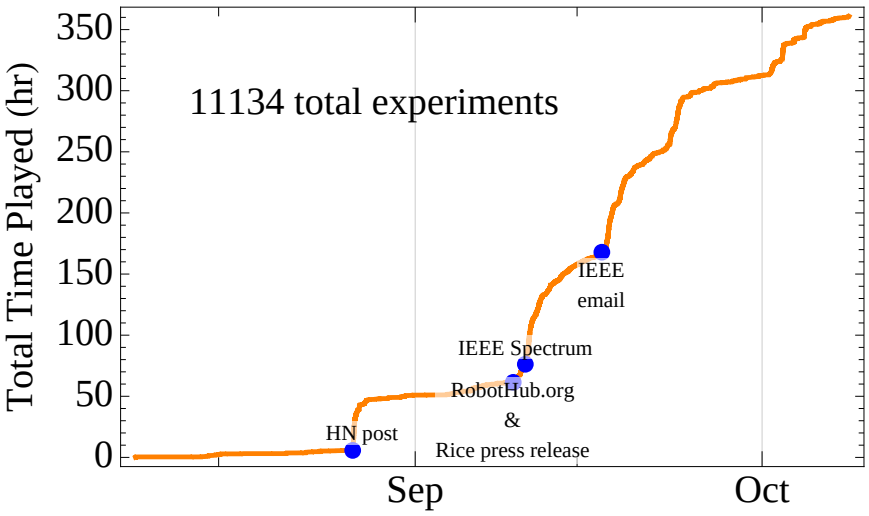


Figure 5: Cumulative time played for completed tests.

Subjects were recruited using a combination of social network effects and coordinated news posts. We asked our friends and colleagues to send links to our site to their friends via their preferred social networks, generally Twitter, Facebook, Google+, and through email. Additionally, we posted our site to several news aggregators in hopes that it would be seen and visited. Our first such posting was to Hacker News, an aggregator run by the Y-Combinator accelerator company; this posting resulted in our first thousand trials. A second posting was made to Reddit, but did not seem to cause much traffic. A third posting was made to the Robohub.org site. The traffic generated by these postings is shown in Fig. 5.

Concurrently, we contacted our university's *News and Media Relations Team*. They sent a writer and photographer to our lab, worked with us to draft a press release, and publicized with news outlets and alumni. Most universities have a media team, and this is a valuable no-cost resource to gain publicity.

4.1.2 Experimental costs

We've spent approximately one hundred dollars USD provisioning and running this experiment. Hosting is provided by Heroku, using a single web instance costing around $40/month, with additional monitoring services bring-

ing that up to $50/month. In the event of increased demand/participant traffic, we can provision another server to take up the load. We purchased our domain name from Namecheap.com for $11.66 a year, giving our site a short, easy to pronounce handle.

Given the large number of experiment sessions run (over 11,000 at the time of this writing), we see a per-experiment cost of less than three cents.

4.1.3 Instrumentation

When conducting an online experiment, it is very helpful to gather data about both the experiment infrastructure and the participants. For the backend, we use a service called Airbrake to monitor the 'health' of the Rails server, getting emails in the events of any errors occurring or suspicious activity. We also use another service called New Relic to provide monitoring and analytics on the server traffic, giving coarse statistics about site visitation, page load time, and other indicators of how our backend is performing.

For the frontend, we use Google Analytics to track user behavior. This tool allows us to see country of origin for users, time spent on the site, relative percent of people who look past the landing page ('bounce rate'), and user agent information (type of browser, type of device, etc.).

4.2 Web Massive Results

We designed five experiments to investigate human control of large robotic swarms for manipulation tasks. Screenshots of each experiment are shown in Fig. 1. Each experiment examined the effects of varying a single parameter: population of robots for manipulation, four levels of visual feedback, three control architectures, different levels of Brownian noise, and position control with 1 to 10 robots. The users could choose which experiment to try, and then our architecture randomly assigned a particular parameter value for each trial. We recorded the completion time and the participant ID for each successful trial. As Fig. 6 shows, one-third of all participants played only a single game. Still, many played multiple games, and their decreasing completion times demonstrates their skills improved.

Varying Number Transport of goods and materials between points is at the heart of all engineering and construction in real-world systems. This experiment varied from 1 to 500 the population of robots used to transport an object. We kept the total area, maximum robot speed, and total net force

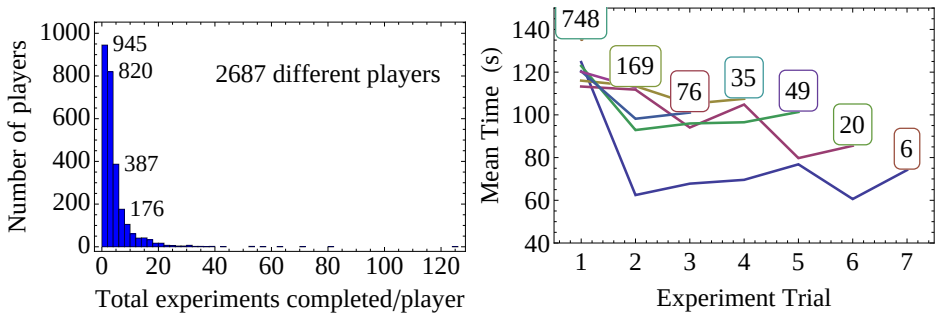


Figure 6: (Left) The total number of games played per player drops off exponentially. (Right) We are able to show that players skill improves as they retry tests using data from *Varying Number*

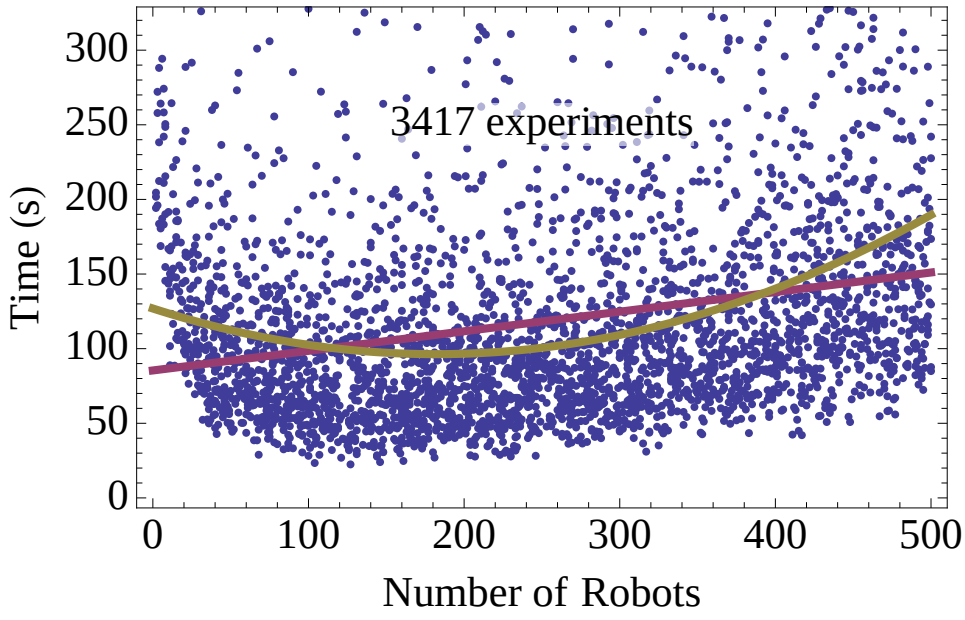


Figure 7: Data from *Varying Number* using robots to push an object through a maze to a goal location. The data indicates that this task has an optimal number of robots, perhaps due to the relative sizes of the robots, obstacles, and object. Best-fit linear and quadratic lines are overlaid for comparison.

the swarm could produce constant. The robots pushed a large hexagonal object through an 'S'-shaped maze. Our hypothesis was that participants would complete the task faster with more robots. The results, shown in Fig. 7, do not support our hypothesis, indicating rather that there is a local minima around 130 robots.

Varying Control Ultimately, we want to use swarms of robots to build things. This experiment compared different control architectures modeled after real-world devices.

We compared attractive and repulsive control with the global control used for the other experiments. The attractive and repulsive controllers were loosely modeled after scanning tunneling microscopes (STM), but also apply to magnetic manipulation Khalil et al. (2013) and biological models Goodrich et al. (2012). STMs can be used to arrange atoms and make small assemblies Avouris (1995). An STM tip is charged with electrical potential, and used to repel like-charged or to attract differently-charged molecules. In contrast, the global controller uses a uniform field (perhaps formed by parallel lines of differently-charged conductors) to pull molecules in the same direction. The experiment challenged players to assemble a three-block pyramid with a swarm of 16 robots.

The results were conclusive, as shown in Fig. **??**: attractive control was the fastest, followed by global control, with repulsive control a distant last. The median time using repulsive control was four times longer than with attractive control.

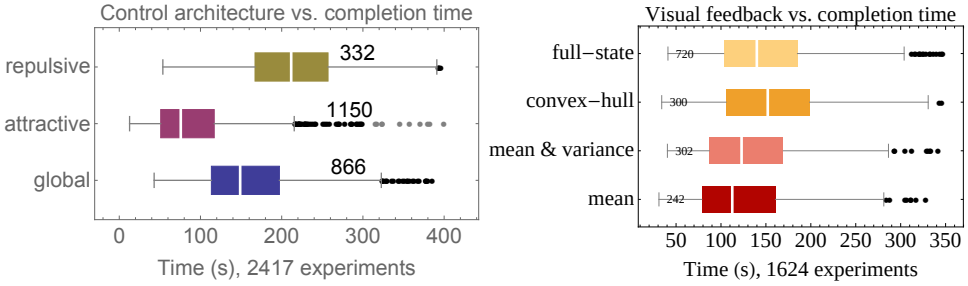


Figure 8: Left: Attractive control resulted in the shortest completion time and repulsive the longest for building a three-block pyramid. Right: Completion-time results for the four levels of visual feedback shown in Fig. 9. Surprisingly, players perform better with limited feedback–subjects with only the mean + variance outperformed all others.

Varying Visualization Sensing is expensive, especially on the nanoscale. To see nanocars Chiang et al. (2011), scientists fasten molecules that fluoresce light when activated by a strong light source. Unfortunately, multiple exposures can destroy these molecules, a process called *photobleaching*. Photobleaching can be minimized by lowering the excitation light intensity, but this increases the probability of missed detections Cazes (2005). This experiment explores manipulation with varying amounts of sensing information: **full-state** sensing provides the most information by showing the position of all robots; **convex-hull** draws a convex hull around the outermost robots; **mean** provides the average position of the population; and **mean + variance** adds a confidence ellipse. Fig. 9 shows screenshots of the same robot swarm with each type of visual feedback. Full-state requires $2n$ data points for n robots. Convex-hull requires at worst $2n$, but usually a smaller number. Mean requires two, and variance three, data points. Mean and mean + variance are convenient even with millions of robots. Our hypothesis predicted a steady decay in performance as the amount of visual feedback decreased.

To our surprise, our experiment indicates the opposite: players with just the mean completed the task faster than those with full-state feedback. As Fig. **??** shows, the levels of feedback arranged by increasing completion time are [mean + variance, mean, full-state, convex-hull]. Anecdotal evidence from beta-testers who played the game suggests that tracking 100 robots is overwhelming—similar to schooling phenomenons that confuse predators—while working with just the mean + variance is like using a "spongy" manipulator. Our beta-testers found convex-hull feedback confusing and irritating. A single robot left behind an obstacle will stretch the entire hull, obscuring the majority of the swarm.

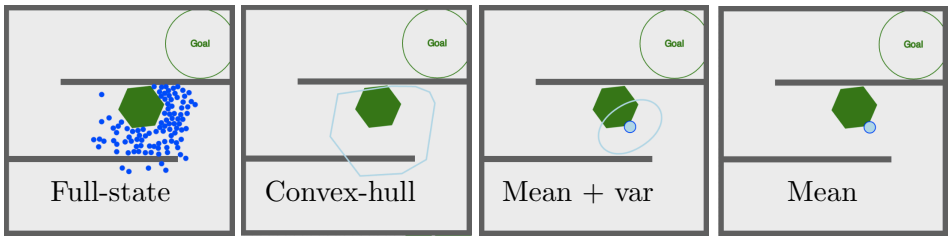


Figure 9: Screenshots from task *Vary Visualization*. This experiment challenges players to quickly steer 100 robots (blue discs) to push an object (green hexagon) into a goal region. We record the completion time and other statistics.

4.3 Varying Noise

Real-world microrobots and nanorobots are affected by turbulence caused by random collisions with molecules. The effect of these collisions is called Brownian motion.

This experiment varied the strength of these disturbances to study how noise affects human control of large swarms. Noise was applied at every timestep 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, ψ_i are uniformly IID, with $m_i \in [0, M]$ and $\psi_i \in [0, 2\pi]$, where M is 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 robots move erratically. Disproving our hypothesis, the results in Fig. **??** show only a 40% increase in completion time for the maximum noise.

4.4 Position Control

This experiment examined how completion time scales with the number of robots n. Using a single square obstacle, users arranged $n \in [1, 10]$ robots into a specified goal pattern. The goal pattern formed a block 'A' with 10

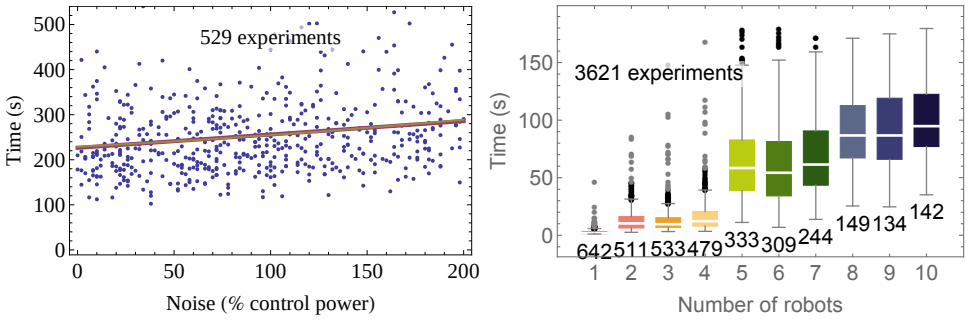


Figure 10: Left: Varying the noise from 0 to 200% of the maximum control input resulted in only a small increase in completion time. Right: Increasing the number of robots resulted in longer completion times. For more than 4 robots the goal pattern contained a void, which may have caused the longer completion times.

robots, and lesser numbers of robots used a subset of these goal positions. Our hypothesis was that completion time would increase linearly with the number of robots, as with our position control algorithm in Becker et al. (2013). Our results roughly corroborate this, as shown in Fig. **??**. Though the number of robots presented to game players is uniformly distributed, larger n are more difficult, and the number of successful experiments drops steadily as n increases.

Note there is a bifurcation between n=4 and n=5 robots. For $n \in [1, 4]$ the goal patterns are not hollow, but starting at n=5 they are. A better experiment design would randomly place the goal positions. Initially we tried this, but our beta-testers strongly disliked trying to arrange robots in random patterns.

5 Simulation of Control Laws

Our simulations use a Javascript port of Box2D, a popular 2D physics engine with support for rigid-body dynamics and fixed-time step simulation Catto (2010). All experiments ran on a Chrome web browser on a 2.6 GHz Macbook. All code is available at Shahrokhi and Becker (2015a).

5.1 Controlling the mean position

We control mean position with a PD controller that uses the mean position and mean velocity. Our control input is the global force applied to each robot:

$$\begin{aligned} u_x &= K_p(x_{goal} - \bar{x}) + K_d(0 - \bar{v}_x) \\ u_y &= K_p(y_{goal} - \bar{y}) + K_d(0 - \bar{v}_y) \end{aligned} \tag{19}$$

here K_p is the proportional gain, and K_d is the derivative gain. We performed a parameter sweep to identify the best values. Representative experiments are shown in Fig. 11. 100 robots were used and the maximum speed was 3 meters per second. As shown in Fig. 11, we can achieve an overshoot of 1% and a rise time of 1.52 s with $K_p = 4$, and $K_d = 1$.

5.2 Controlling the variance

For variance control we use the control law discussed in Section 3.4. Moving away from the wall and waiting is sufficient to increase variance because Brownian noise naturally disperses the swarm in such a way that the variance

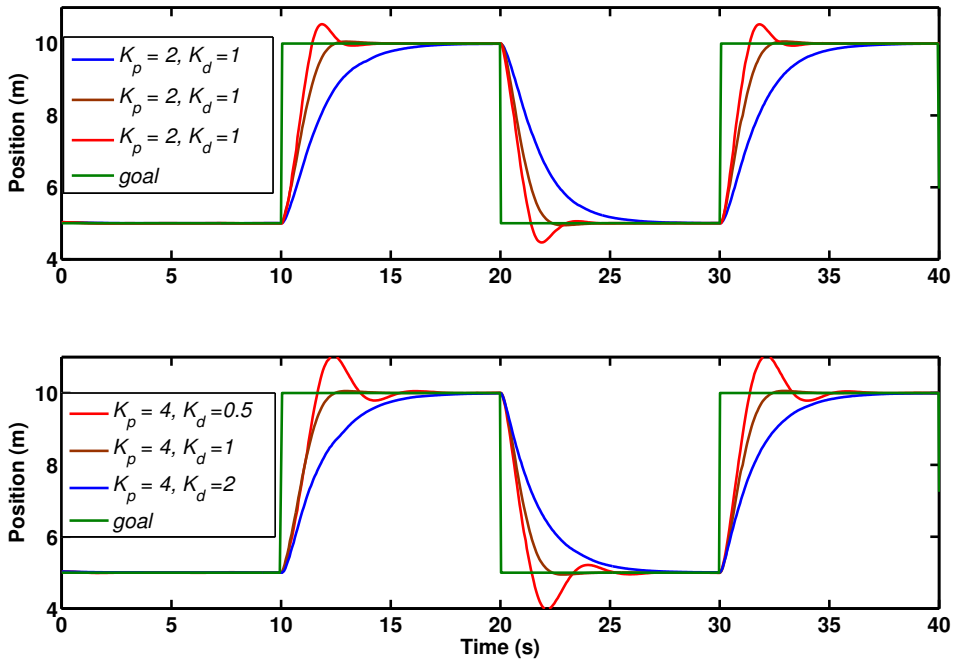


Figure 11: Tuning proportional (K_p, top) and derivative (K_d, bottom) gain values in (19) improves performance with $n = 100$ robots.

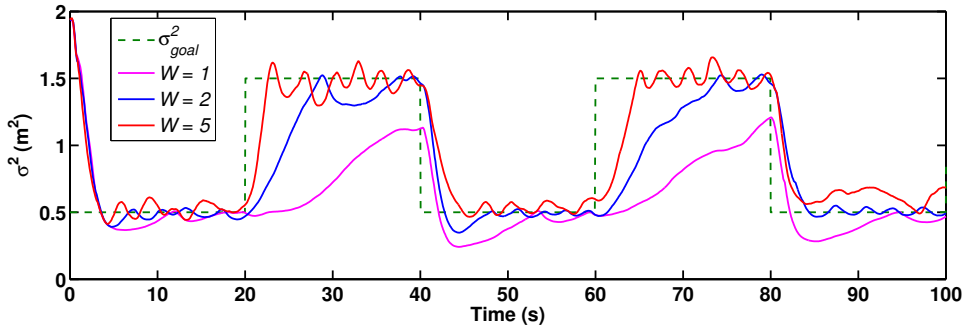


Figure 12: Increased noise results in more responsive variance control because stronger Brownian noise causes a faster increase of variance.

increases linearly Einstein (1956). If faster dispersion is needed, the swarm can be pushed through obstacles such as a diffraction grating or Pachinko board Becker et al. (2013).

The variance control law to regulate the variance to σ^2_{ref} has three gains:

$$\begin{aligned} u_x &= K_p(x_{goal}(\sigma^2_{ref}) - \bar{x}) - K_d\bar{v}_x + K_i(\sigma^2_{ref} - \sigma^2_x) \\ u_y &= K_p(y_{goal}(\sigma^2_{ref}) - \bar{y}) - K_d\bar{v}_y + K_i(\sigma^2_{ref} - \sigma^2_y). \end{aligned} \tag{20}$$

In a slight abuse of notation we call the gain scaling the variance error K_i because the variance, if unregulated, integrates over time. Eq. (20) assumes the nearest wall is to the left of the robot at $x = 0$, and chooses a reference goal position that in steady-state would have the correct variance according to (13):

$$x_{goal}(\sigma^2_{ref}) = r + \sqrt{3\sigma^2_{ref}} \tag{21}$$

If another wall is closer, the signs of $[K_p, K_i]$ are inverted, and the location x_{goal} is translated. Results are shown in Fig. 12, with $K_{p,i,d} = [4, 1, 1]$.

5.3 Hybrid Control of mean and variance

Fig. 13 shows a simulation run of the hybrid controller in Alg. 1 with 100 robots in a square workspace containing no internal obstacles.

6 Block-Pushing Results

This section analyzes a *block-pushing* task attempted by both our hybrid, hysteresis-based controller and by human users.

6.1 Human-Controlled Block-Pushing

As we saw in previous section, players with just the mean completed the task faster than those with full-state feedback. As Fig. **??** shows, the levels of feedback arranged by increasing completion time are [mean, mean + variance, full-state, convex-hull]. Interviews with beta-testers suggests that tracking 100 robots was overwhelming—similar to schooling phenomenons that confuse predators—while working with just the mean + variance was like using a "spongy" manipulator. Convex-hull feedback was confusing and irritating because a single robot left behind an obstacle would distort the entire hull, obscuring the information about the majority of the swarm.

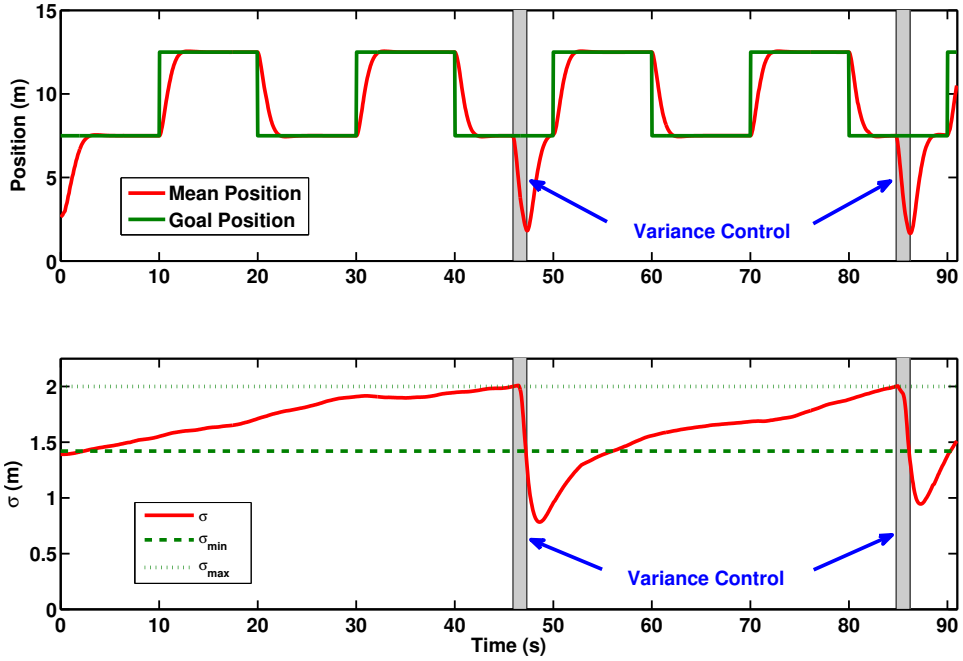


Figure 13: Simulation result with 100 robots under hybrid control Alg. 1, which controls both the mean position (top) and variance (bottom). For ease of analysis, only x position and variance are shown.

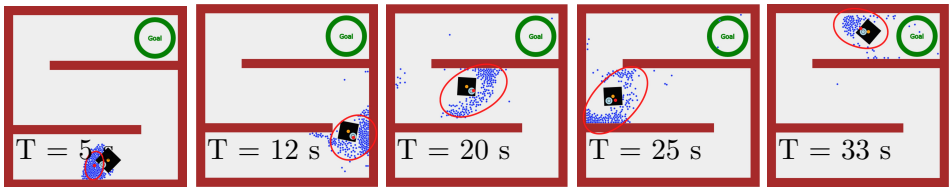


Figure 14: Snapshots showing the block-pushing experiment with 200 robots under automatic control. See the video attachment for an animation Shahrokhi and Becker (2015b).

6.2 Automated Block-Pushing

Fig. 14 shows snapshots during an execution of this algorithm. To solve this block-pushing task, we discretized the environment. On this discretized grid we used breadth-first search to determine $\mathbf{M}$, the shortest distance from any grid cell to the goal, and generated a gradient map $\nabla\mathbf{M}$ toward the goal as shown in Fig. 15. The block's center of mass is at $\mathbf{b}$ and has radius r_b. Three constants are needed, where $k_1 > k_2 > 1$ and $1 > k_2 > 0$. All experiments used $[k_1, k_2, k_3] = [2.5, 1.5, 0.1]$. The robots were directed to assemble behind the block at $\mathbf{b} - k_2 r_b \nabla\mathbf{M}(\mathbf{b})$, then move to $\mathbf{b} - k_3 r_b \nabla\mathbf{M}(\mathbf{b})$ to push the block toward the goal location. We use the hybrid hysteresis-based controller in Alg. 1 to track the desired position, while maintaining sufficient robot density to move a block by switching to minimize variance whenever variance exceeds a set limit. The minimize variance control law (20) is slightly modified to choose the nearest corner further from the goal than $\mathbf{b}$ with an obstacle-free straight-line path to $\mathbf{b}$. The control algorithm for block-pushing is listed in Alg. 2. Experimental results are summarized in Fig. **??**. Although larger populations of robots can apply more force, minimizing the variance requires more time with larger populations and dominates task completion time.

Algorithm 2 is an imperfect solution and has a failure mode if the robot swarm becomes multi-modal with modes separated by an obstacle, as shown in Fig. 17. In this case, moving toward a corner will never reduce the variance below σ^2_{min}.

The first challenge is to identify when the distribution has become multi-modal. Measuring just the mean and variance is insufficient to determine if a distribution is no longer unimodal, but if the swarm is being directed to a corner, and the variance does not reduce below σ^2_{min}, the swarm has become separated. In this case, we must either manipulate with a partial swarm, or run a gathering algorithm. For the 'S'-shaped workspace in this

Algorithm 2 Block-pushing controller for a robotic swarm.

Require: Knowledge of swarm mean $[\bar{x}, \bar{y}]$, variance $[\sigma_x^2, \sigma_y^2]$, moveable block's center of mass $\mathbf{b}$, map of the environment, and the locations of all convex corners $\mathbf{C}$

Require: Robot distribution is unimodal

Require: Obstacle-free, straight-line path from swarm to moveable block

Compute $\mathbf{M}$, the distance to goal, with breadth-first search
Compute the gradient, $\nabla\mathbf{M}$
$\mathbf{C} \leftarrow \text{sort}(\mathbf{C})$ according to $-\mathbf{M}$
while $\mathbf{b}$ is not in goal region **do**
    $\sigma^2 \leftarrow \max(\sigma_x, \sigma_y)$
    if $\sigma^2 > \sigma_{max}^2$ **then**
        while $\sigma^2 > \sigma_{min}^2$ **do**
            $\mathbf{c}_i \leftarrow$ the nearest corner in $\mathbf{C}$ to $[\bar{x}, \bar{y}]$
            $[x_{goal}, y_{goal}] \leftarrow \mathbf{c}_i$
            if $\mathbf{M}(\mathbf{b}) > \mathbf{M}(\mathbf{c}_i)$ **then**
                $[x_{goal}, y_{goal}] \leftarrow \mathbf{c}_{i-1}$
                Apply (19) to move toward $[x_{goal}, y_{goal}]$
            end if
        end while
    else
        if distance$(\mathbf{b}, [x_{goal}, y_{goal}]) > k_1 r_b$ **then**
            $r_p \leftarrow k_2 r_b$ ▷ guarded move
        else
            $r_p \leftarrow k_3 r_b$ ▷ pushing move
        end if
        $[x_{goal}, y_{goal}] \leftarrow \mathbf{b} - r_p \nabla\mathbf{M}(\mathbf{b})$
    end if
    Apply (19) to move toward $[x_{goal}, y_{goal}]$
end while

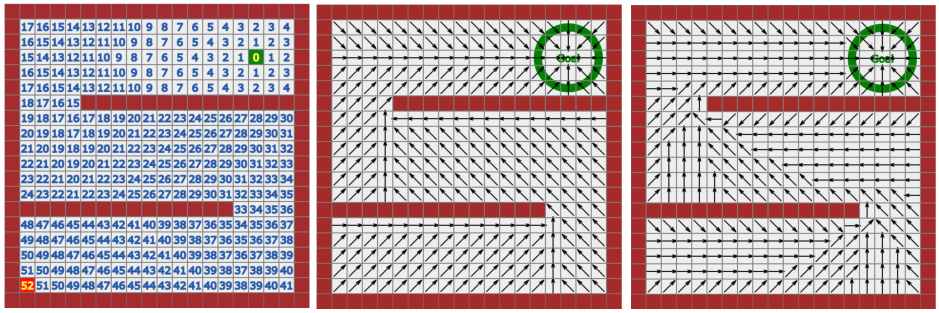


Figure 15: The BFS algorithm finds the shortest path for the moveable block (left), which is used to compute gradient vectors (middle). Using policy iterations enables encoding penalties for being near obstacles (right).

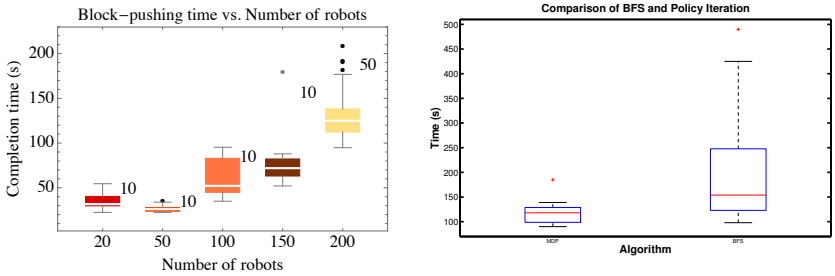


Figure 16: Completion-time results using the automatic controller from Alg. 2 for different numbers of robots. Each bar is labelled with the number of trials.

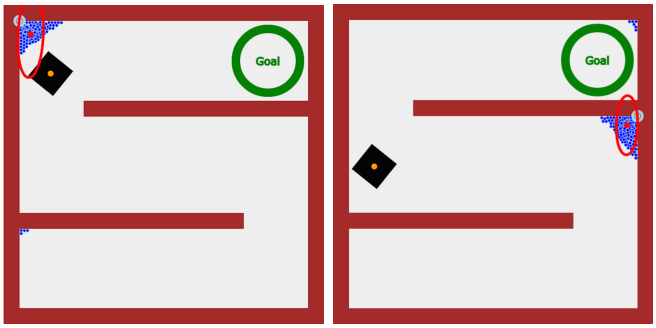


Figure 17: Algorithm 2 fails when some robots are separated by the maze and the swarm can not achieve $\sigma^2 < \sigma^2_{min}$. These failures occured during 14% of trials.

study, an open-loop input that commands the swarm to move in succession {LEFT, DOWN, RIGHT, DOWN} will move the swarm to the bottom right corner. This is not true for all obstacle fields. In a 'T'-shaped workspace, it is not possible to find an open-loop input that will move the entire swarm to the bottom of the 'T'.

Using only the mean and variance may be overly restrictive. Many heuristics using high-order moments have been developed to test if a distribution is multimodal Haldane (1951). Often the sensor data itself, though it may not resolve individual robots, will indicate multi-modality. For instance CCD images reveal clusters of bacteria, and MRI scans show agglomerations of particles Stuber et al. (2007). This data can be fitted with k-means or expectation maximization algorithms, and manipulation could be performed with the nearest swarm of sufficient size.

7 Object Manipulation With Hardware Robots

7.1 Environmental Setup

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 environment, and data collection. The kilobot **??** is a low-cost robot designed for testing collective algorithms with large numbers of robots. It is available commercially or as an open source platform **?**. 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

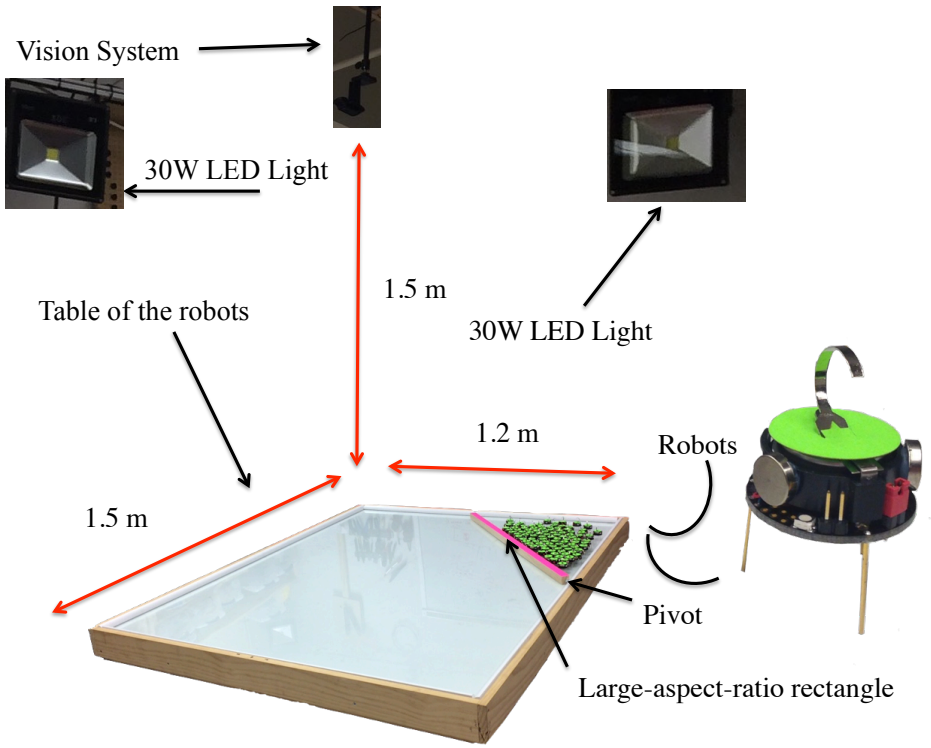


Figure 18: Hardware platform: table with 1.5×1.2 m workspace, surrounded by eight remotely triggered 30W LED floodlights, with an overhead machine vision system.

to 1 cm/s. Each robot has one ambient light sensor that is used to implement *phototaxis*, moving towards a light source. In these experiments as shown in Fig. 18, we used n=100 kilobots, a 1.5 m×1.2 m whiteboard as the workspace, and eight 30W LED floodlights arranged 1.5 m above the plane of the table at the $\{N, NE, E, SE, S, SW, W, NW\}$ vertices of a 6 m square centered on the workspace. The lights were controlled using an Arduino Uno board connected to an 8 relay shield board. Above the table, an overhead machine vision system tracks the position of the swarm. Laser-cut patterns for our neon green fiducial markers and our MATLAB tracking code are available at our github repository **?**.

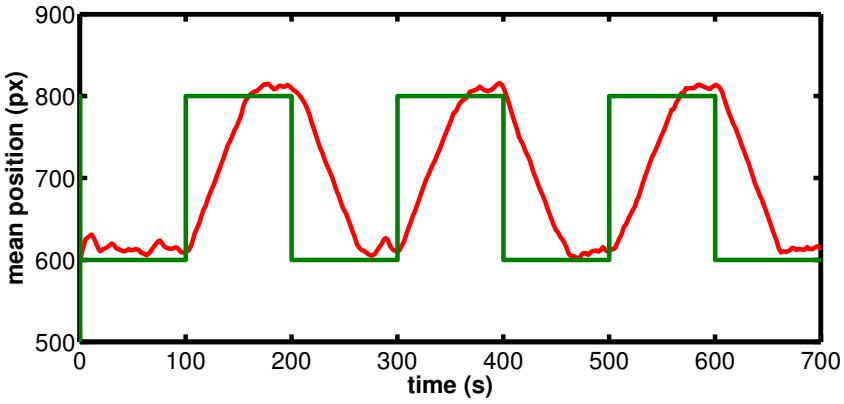


Figure 19: Mean Control plot with kilobots.

7.2 Mean Control With Real Robots

7.3 Automated Object Manipulation

References

Michael D. Armani, Satej V. Chaudhary, Roland Probst, and Benjamin Shapiro. Using feedback control of microflows to independently steer multiple particles. *Journal of Microelectromechanical systems*, 15(4), August 2006.

Phaedon Avouris. Manipulation of matter at the atomic and molecular levels. *Accounts of chemical research*, 28(3):95–102, 1995.

Shishir Bashyal and Ganesh Kumar Venayagamoorthy. Human swarm interaction for radiation source search and localization. In *Swarm Intelligence Symposium (SIS)*, pages 1–8. IEEE, 2008.

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*, pages 781 –786, oct. 2009. doi: 10.1109/IROS.2009.5354427. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5354427`.

Aaron Becker and Timothy Bretl. Approximate steering of a unicycle under

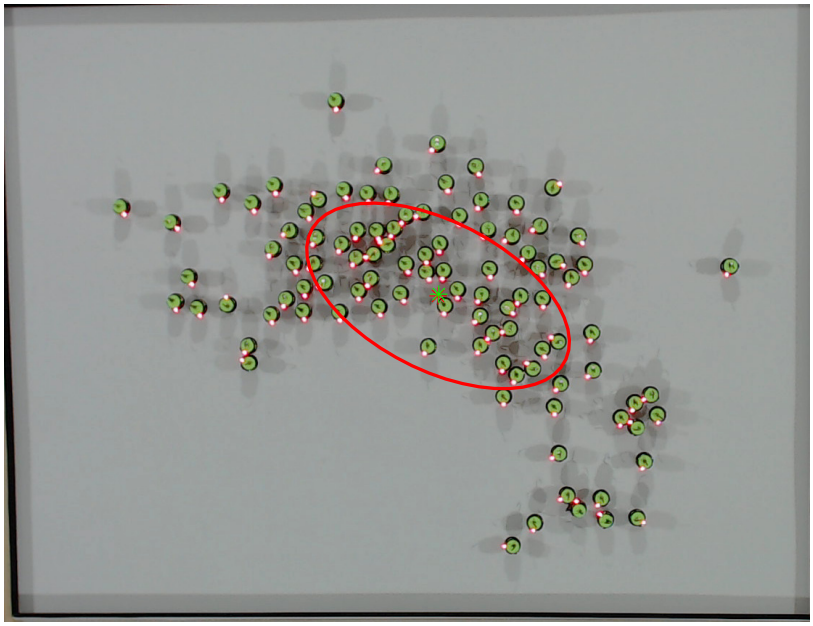


Figure 20: Mean Control experiment with kilobots.

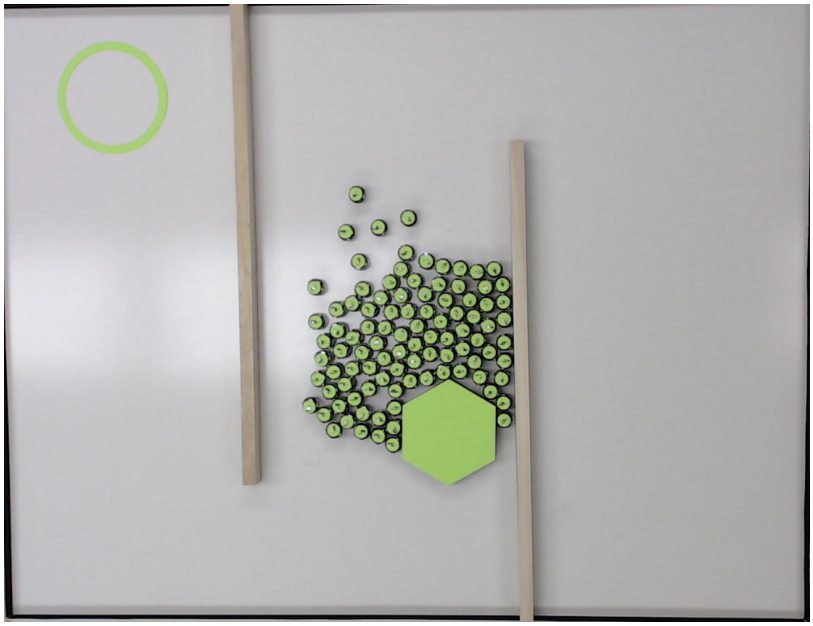


Figure 21: A swarm of robots, all controlled by a uniform force field, can be effectively controlled by a hybrid controller that knows only the first and second moments of the robot distribution. Here is a mockup of a swarm of hardware robots(kilobots) that pushes a green block toward the goal. See video attachment Shahrokhi and Becker (2015b).

bounded model perturbation using ensemble control. *IEEE Trans. Robot.*, 28(3):580–591, June 2012.

Aaron Becker, Cem Onyuksel, and Timothy Bretl. Feedback control of many differential-drive robots with uniform control inputs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2012.

Aaron Becker, Golnaz Habibi, Justin Werfel, Michael 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)*, pages 520–527, November 2013.

Aaron Becker, Cem Onyuksel, Timothy Bretl, and James McLurkin. Controlling many differential-drive robots with uniform control inputs. *The international journal of Robotics Research*, 33(13):1626–1644, 2014.

Erin Catto. User manual, Box2D: A 2D physics engine for games, http://www.box2d.org, 2010. URL `http://www.box2d.org`.

Jack Cazes. *Encyclopedia of Chromatography*, volume 2. Taylor & Francis Group, New York, second edition, 2005. ISBN 0824727878.

Pinn-Tsong Chiang, Johannes Mielke, Jazmin Godoy, Jason M. Guerrero, Lawrence B. Alemany, Carlos J. Villagómez, Alex Saywell, Leonhard Grill, and James M. Tour. Toward a light-driven motorized nanocar: Synthesis and initial imaging of single molecules. *ACS Nano*, 6(1):592–597, February 2011. doi: 10.1021/nn203969b.

Jean-Pierre de la Croix and Magnus Egerstedt. Controllability characterizations of leader-based swarm interactions. In *2012 AAAI Fall Symposium Series*, 2012.

Raphael Deimel and Oliver Brock. A novel type of compliant, underactuated robotic hand for dexterous grasping. *Robotics: Science and Systems, Berkeley, CA*, pages 1687–1692, 2014.

Eric Diller, Joshua Giltinan, and Metin Sitti. Independent control of multiple magnetic microrobots in three dimensions. *The International Journal of Robotics Research*, 32(5):614–631, 2013. URL `http://ijr.sagepub.com/content/32/5/614.abstract`.

B.R. Donald, C.G. Levey, C.D. McGray, I. Paprotny, and D. Rus. An untethered, electrostatic, globally controllable MEMS micro-robot. *J. of MEMS*, 15(1):1–15, February 2006. ISSN 1057-7157.

B.R. Donald, C.G. Levey, and I. Paprotny. Planar microassembly by parallel actuation of MEMS microrobots. *J. of MEMS*, 17(4):789–808, August 2008.

Albert Einstein. *Investigations on the Theory of the Brownian Movement*. Courier Corporation, 1956.

Chris Ertel and Aaron Becker. Swarmcontrol git repository, September 2013. URL `https://github.com/crertel/swarmmanipulate.git`.

J. Fink, N. Michael, and V. Kumar. Composition of vector fields for multi-robot manipulation via caging. In *Robotics Science and Systems*, Atlanta, GA, June 2007.

Steven Floyd, Eric Diller, Chytra Pawashe, and Metin Sitti. Control methodologies for a heterogeneous group of untethered magnetic micro-robots. *Int. J. Robot. Res.*, 30(13):1553–1565, November 2011.

Dominic Frutiger, Bradley Kratochvil, Karl Vollmers, and Bradley J. Nelson. Magmites - wireless resonant magnetic microrobots. In *IEEE Int. Conf. Rob. Aut.*, Pasadena, CA, May 2008.

Michael A Goodrich, Sean Kerman, Brian Pendleton, and PB Sujit. What types of interactions do bio-inspired robot swarms and flocks afford a human? In *Robotics: Science and Systems*, 2012.

Ronald L. Graham and Neil JA Sloane. Penny-packing and two-dimensional codes. *Discrete & Computational Geometry*, 5(1):1–11, 1990.

JBS Haldane. Simple tests for bimodality and bitangentiality. *Annals of eugenics*, 16(1):359–364, 1951.

Islam S. M. Khalil, Frank van den Brink, Ozlem Sardan Sukas, and Sarthak Misra. Microassembly using a cluster of paramagnetic microparticles. In *IEEE International Conference on Robotics and Automation*, pages 5507–5512, Karlsruhe, Germany, May 2013.

Marius Kloetzer and Calin Belta. Temporal logic planning and control of robotic swarms by hierarchical abstractions. *Robotics, IEEE Transactions on*, 23(2):320–330, 2007.

Andreas Kolling, Steven Nunnally, and Michael Lewis. Towards human control of robot swarms. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 89–96. ACM, 2012.

Aleksandr Mikhailovich Lyapunov, translated and edited by A.T. Fuller. *The General Problem of the Stability of Motion*. Tayor & Francis, London, 1992.

K.M. Lynch. Locally controllable manipulation by stable pushing. *IEEE Trans. Robot. Autom.*, 15(2):318–327, April 1999.

Sylvain Martel, Samira Taherkhani, Maryam Tabrizian, Mahmood Mohammadi, Dominic de Lanauze, and Ouajdi Felfoul. Computer 3d controlled bacterial transports and aggregations of microbial adhered nanocomponents. *Journal of Micro-Bio Robotics*, 9(1-2):23–28, 2014.

Lael U Odhner, Leif P Jentoft, Mark R Claffee, Nicholas Corson, Yaroslav Tenzer, Raymond R Ma, Martin Buehler, Robert Kohout, Robert D Howe, and Aaron M Dollar. A compliant, underactuated hand for robust manipulation. *The International Journal of Robotics Research*, 33(5):736–752, 2014.

Dan R. Olsen Jr and Stephen Bart Wood. Fan-out: Measuring human control of multiple robots. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 231–238, Vienna, Austria, April 2004.

Kathrin E. Peyer, Li Zhang, and Bradley J. Nelson. Bio-inspired magnetic swimming microrobots for biomedical applications. *Nanoscale*, 2013.

Sadra Sadraddini and Calin Belta. Swarm manipulation with a uniform magnetic field. In *unpublished*, 2015.

Shiva Shahrokhi and Aaron T. Becker. BlockPushingIROS2015, https://github.com/aabecker/swarmcontrolsandbox/blob/master/ examplecontrollers/blockpushingiros2015.html, July 2015a.

Shiva Shahrokhi and Aaron T. Becker. Stochastic swarm control, https://youtu.be/tcej-9e6-4o, October 2015b. URL `https://youtu.be/tCej-9e6-4o`.

Yasuhiro Shirai, Andrew J. Osgood, Yuming Zhao, Kevin F. Kelly, and James M. Tour. Directional control in thermally driven single-molecule

nanocars. *Nano Letters*, 5(11):2330–2334, February 2005. doi: 10.1021/nl051915k.

Peter Squire, Greg Trafton, and Raja Parasuraman. Human control of multiple unmanned vehicles: effects of interface type on execution and task switching times. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, HRI '06, pages 26–32, New York, NY, USA, 2006. ACM. ISBN 1-59593-294-1. doi: 10.1145/1121241.1121248. URL `http://doi.acm.org/10.1145/1121241.1121248`.

Matthias Stuber, Wesley D Gilson, Michael Schär, Dorota A Kedziorek, Lawrence V Hofmann, Saurabh Shah, Evert-Jan Vonken, Jeff WM Bulte, and Dara L Kraitchman. Positive contrast visualization of iron oxide-labeled stem cells using inversion-recovery with on-resonant water suppression (iron). *Magnetic Resonance in Medicine*, 58(5):1072–1077, 2007.

A. Sudsang, F. Rothganger, and J. Ponce. Motion planning for disc-shaped robots pushing a polygonal object in the plane. *IEEE Trans. Robot. Autom.*, 18(4):550–562, August 2002.

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.*, pages 1408–1413, May 2006. doi: 10.1109/ROBOT.2006.1641906. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1641906&isnumber=34383`.

S. Tottori, L. Zhang, F. Qiu, K.K. Krawczyk, A. Franco-Obregón, and B. J. Nelson. Magnetic helical micromachines: Fabrication, controlled swimming, and cargo transport. *Advanced Materials*, 24(811), 2012.