Algorithms For Shaping a Particle Swarm With a Shared Control Input Using Boundary Interaction

Shiva Shahrokhi, Arun Mahadev, and Aaron T. Becker

***Abstract*—Consider a swarm of particles controlled by global inputs. This paper presents algorithms for shaping such swarms in 2D using boundary walls. The range of configurations created by conforming a swarm to a boundary wall is limited. We describe the set of stable configurations of a swarm in two canonical workspaces, a circle and a square. To increase the diversity of configurations, we add boundary interaction to our model. We provide algorithms using friction with walls to place two robots at arbitrary locations in a rectangular workspace. Next, we extend this algorithm to place n agents at desired locations. We conclude with efficient techniques to control the covariance of a swarm not possible without wall-friction. Simulations and hardware implementations with 100 robots validate these results.**

These methods may have particular relevance for current micro- and nano-robots controlled by global inputs.

I. Introduction

Particle swarms steered by a global force are common in applied mathematics, biology, and computer graphics.

$$[\dot{x}_i, \dot{y}_i]^\top = [u_x, u_y]^\top, \qquad i \in [1, n] \tag{1}$$

The control problem is to design $u_x(t), u_y(t)$ to make all n particles achieve a task. As a current example, micro- and nano-robots can be manufactured in large numbers, see Chowdhury et al. [6], Martel et al. [16], Kim et al. [12], Donald et al. [7], Ghosh and Fischer [9], Ou et al. [18] or Qiu and Nelson [19]. Someday large swarms of robots will be remotely guided ex vivo to assemble structures in parallel and through the human body, to cure disease, heal tissue, and prevent infection. For each application, large numbers of micro robots are required to deliver sufficient payloads, but the small size of these robots makes it difficult to perform onboard computation. Instead, these robots are often controlled by a global, broadcast signal. These applications require techniques to shape the swarm that can reliably exploit large populations despite high under-actuation.

Even without obstacles or boundaries, the mean position of the swarm in (1) is controllable. By adding rectangular boundary walls, some higher-order moments such as the swarm's position variance orthogonal to the boundary walls (σ_x and σ_y for a workspace with axis-aligned walls) are also controllable [23]. A limitation is that global control can only compress a swarm orthogonal to obstacles. However, navigating through narrow passages often requires control of the variance and the covariance.

The paper is arranged as follows. §II-A provides analytical position control results in two canonical workspaces with frictionless walls. These results are limited in the set of shapes that can be generated. To extend the range of possible shapes, §II-B introduces wall friction to the system model. We prove that two orthogonal boundaries with high friction are sufficient to arbitrarily position two robots in §III-A, and §III-B extends this to prove a rectangular workspace with high-friction boundaries can position a swarm of n robots arbitrarily within a subset of the workspace. §IV describes implementations of both position control algorithms in simulation and §V describes experiments with a hardware setup and up to 100 robots, as shown in Fig. 1. After a review of recent related work §VI, we end with directions for future research §VII.

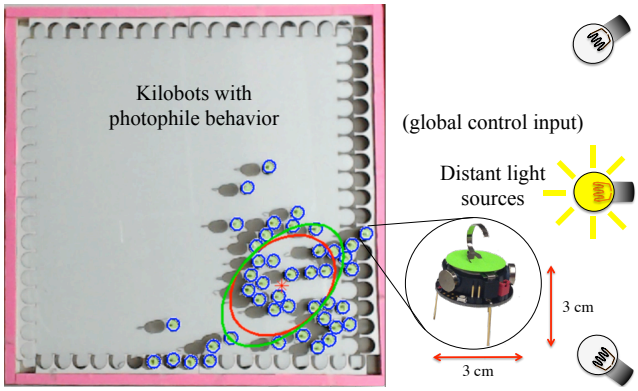

Fig. 1. Swarm of kilobots programmed to move toward the brightest light source as explained in §V. The current covariance ellipse and mean are shown in red, the desired covariance is shown in green. Navigating a swarm using global inputs is challenging because each member receives the same control inputs. This paper focuses on using boundary walls and wall friction to break the symmetry caused by the global input and control the shape of a swarm.

II. Theory

A. Using Boundaries: Fluid Settling In a Tank

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.

a) Square workplace: This section examines the mean $(\bar{x}, \bar{y})$, covariance $(\sigma_x^2, \sigma_y^2, \sigma_{xy})$, and correlation ρ_{xy} of a very large swarm of robots as they move inside a square workplace under the influence of gravity pointing in the direction β. The swarm is large, but the robots are small in comparison, and together occupy a constant area A. Under a global input such as gravity, they flow like water, moving to a side of the workplace and forming a polygonal shape, as shown in Fig. 2.

The range for the global input angle β is [0,2π). In this range, the swarm assumes eight different polygonal shapes. The shapes alternate between triangles and trapezoids when

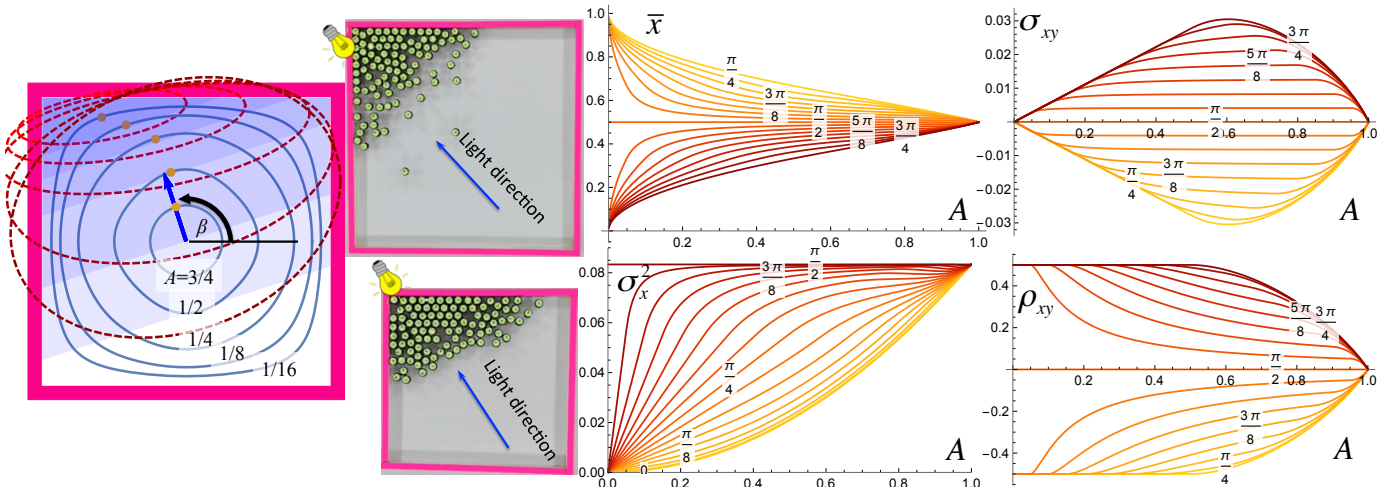

Fig. 2. Pushing the swarm against a square boundary wall allows limited control of the shape of the swarm, as a function of swarm area A and the commanded movement direction β. Left plot shows locus of possible mean positions for five values of A. The locus morphs from a square to a circle as A increases. The covariance ellipse for each A is shown with a dashed line. Center shows two corresponding arrangements of kilobots. At right is $\bar{x}(A), \sigma_{xy}(A), \sigma_x^2(A)$, and $\rho(A)$ for a range of β values. See online interactive demonstration at [29].

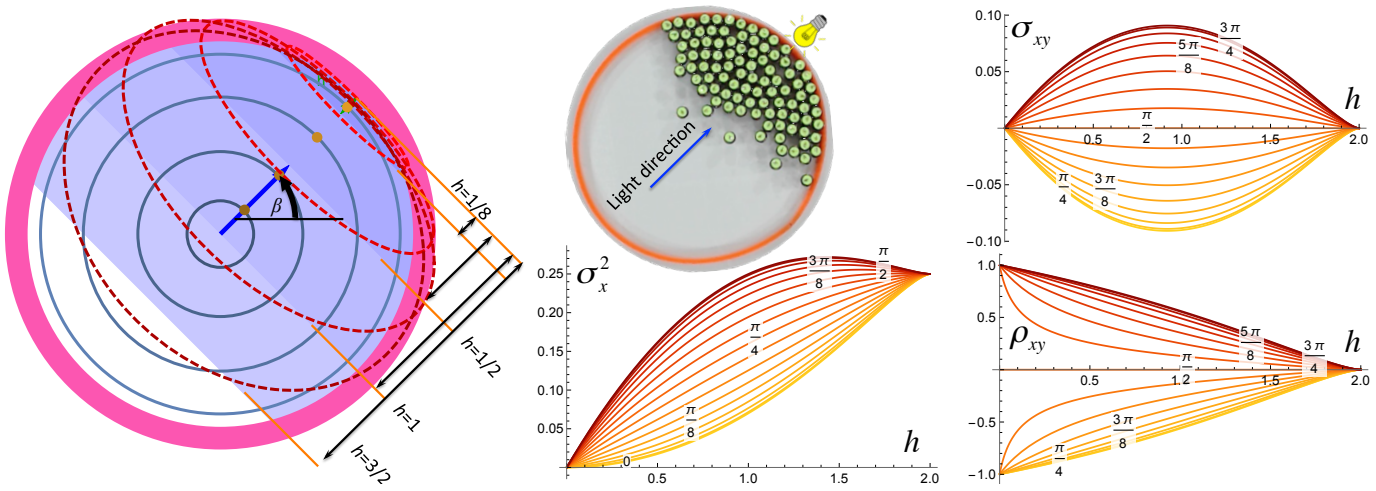

Fig. 3. Pushing the swarm against a circular boundary wall allows limited control of the shape of the swarm, as a function of the fill level h and the commanded movement direction β. Left plot shows locus of possible mean positions for four values of h. The locus of possible mean positions are concentric circles. See online interactive demonstration at [28].

the area $A<1/2$, and alternate between squares with one corner removed and trapezoids when $A>1/2$.

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

$$\bar{x} = \frac{\iint_R x\,dx\,dy}{A}, \qquad \bar{y} = \frac{\iint_R y\,dx\,dy}{A} \tag{2}$$

$$\sigma_x^2 = \frac{\iint_R (x-\bar{x})^2\,dx\,dy}{A}, \sigma_y^2 = \frac{\iint_R (y-\bar{y})^2\,dx\,dy}{A} \tag{3}$$

$$\sigma_{xy} = \frac{\iint_R (x-\bar{x}_x)\,(y-\bar{y})\,dx\,dy}{A}, \rho_{xy} = \frac{\sigma_x^2}{\sigma_x\sigma_y} \tag{4}$$

The region of integration R is the polygon containing the swarm. If the force angle is β, the mean when the swarm is in the lower-left corner is:

$$\bar{x}(A,\beta) = \frac{\int_0^{\sqrt{2}\sqrt{-A\tan(\beta)}} \left(\int_0^{\sqrt{2}\sqrt{-A\cot(\beta)}+x\cot(\beta)} x\,dy \right) dx}{A}$$

$$= \frac{1}{3}\sqrt{2}\sqrt{A\tan(\beta)} \tag{5}$$

$$\bar{y}(A,\beta) = \frac{\int_0^{\sqrt{2}\sqrt{-A\tan(\beta)}} \left(\int_0^{\sqrt{2}\sqrt{-A\cot(\beta)}+x\cot(\beta)} y\,dy \right) dx}{A}$$

$$= \frac{1}{3}\sqrt{2}\sqrt{A\cot(\beta)} \tag{6}$$

The full equations are included in the appendix, and are summarized in Fig. 2. A few highlights are that the correlation is maximized when the swarm is in a triangular shape, and

is $\pm 1/2$. 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.

b) Circular workplace: Though rectangular boundaries are common in artificial workspaces, biological workspaces are usually rounded. Similar calculations can be computed 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

$$A(h) = \arccos(1-h) - (1-h)\sqrt{(2-h)h}. \tag{7}$$

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:

$$r(h) = \frac{2(-(h-2)h)^{3/2}}{3\left(\sqrt{-(h-2)h}(h-1) + \arccos(1-h)\right)} \tag{8}$$

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 summarized in Fig. 3.

B. Using Boundaries: Friction and Boundary Layers

Global inputs move a swarm uniformly. Controlling covariance 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. Directly controlling covariance by pushing the swarm into a boundary requires changes to the boundary. An obstacle in the lower-right corner is enough to generate positive covariance. Generating both positive and negative covariance requires additional obstacles. Requiring special obstacle configuration also makes covariance control dependent on the local environment. Instead of pushing our robots directly into a wall, this paper examines an oblique approach, by 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 negative friction force that opposes movement along the boundary. This causes robots along the boundary to slow down compared to 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 $|\theta| < \pi/2$. The resulting net force on the robot, $F_{forward}$, is aligned with the wall and given by

$$F_{forward} = F\sin(\theta) - F_f$$
$$\text{where } F_f = \begin{cases} \mu_f N, & \mu_f N < F\sin(\theta) \\ F\sin(\theta), & \text{else} \end{cases} \tag{9}$$
$$\text{and } N = F\cos(\theta)$$

Fig. 4 shows the resultant forces on two robots when one is touching a wall. As illustrated, both experiences different net forces although each receives the same inputs. 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, if the control input is parallel or into the wall, the touching robot will not move. There are many alternate models of friction that also break control symmetry. Fig. 4c shows fluid flow along a boundary. Fluid in the free-flow region moves uniformly, but flow decreases to zero in the boundary layer.

$$u(y) = u_0[1 - \frac{(y-h)^2}{h^2} = u_0\frac{y}{h}[2 - \frac{y}{h}] \tag{10}$$

The next section shows how a system with friction model (9) and two orthogonal walls can arbitrarily position two robots.

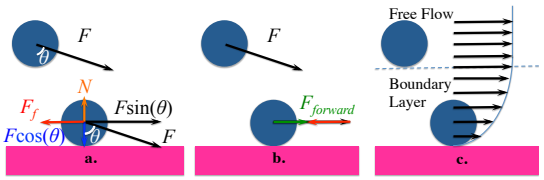

Fig. 4. (a,b) Wall friction reduces the force for going forward $F_{forward}$ on a robot near a wall, but not for a free robot. (c) velocity of a fluid reduces to zero at the boundary.

III. Algorithms

A. Position Control of 2 Robots Using Wall Friction

This section describes Alg. 1, which uses wall-friction to arbitrarily position two robots in a rectangular workspace. This algorithm introduces concepts that will be used for multi-robot positioning. It only requires collisions with two orthogonal walls, in this case, the bottom and left walls. Fig. 5 shows a Mathematica implementation of the algorithm, and is useful as a visual reference for the following description.

Assume two robots are initialized at s_1 and s_2 with corresponding goal destinations e_1 and e_2. Denote the current positions of the robots r_1 and r_2. Subscripts $_x$ and $_y$ denote the x and y coordinates, i.e., s_{1x} and s_{1y} denote the x and y locations of s_1. The algorithm assigns a global control input at every instance. The goal is to adjust $\Delta r_x = r_{2x} - r_{1x}$ from $\Delta s_x = s_{2x} - s_{1x}$ to $\Delta e_x = e_{2x} - e_{1x}$ and adjust $\Delta r_y = r_{2y} - r_{1y}$ from $\Delta s_y = s_{2y} - s_{1y}$ to $\Delta e_y = e_{2y} - e_{1y}$ using a shared global control input. This algorithm exploits the position-dependent friction model (9).

Our algorithm solves the positioning problem in two steps: First, $|\Delta r_x - \Delta e_x|$ is reduced to zero while Δr_y is kept constant in Alg. 2. Second $|\Delta r_y - \Delta e_y|$ is reduced to zero while Δr_x is kept constant.

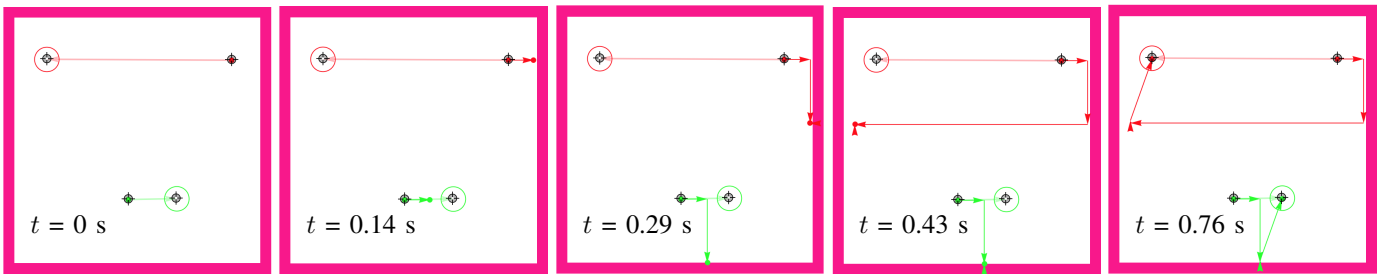

Fig. 5. Frames from an implementation of Alg. 1: two robot positioning using walls with infinite friction. The algorithm only requires friction along the bottom and left walls. Robot initial positions are shown by a crosshair, and final positions by a circled crosshair. Dashed lines show the shortest route if robots could be controlled independently. Solid arrows show path given by Alg. 1. Online demonstration and source code at [22].

Algorithm 1 WallFrictionArrange2Robots(s_1, s_2, e_1, e_2, L)

Require: starting (s_1, s_2) and ending (e_1, e_2) positions of two robots. $(0, 0)$ is bottom corner, s_1 is rightmost robot, L is length of the walls. Current position of the robots are (r_1, r_2).

1: (r_1, r_2) = GenerateDesired$x$-spacing(s_1, s_2, e_1, e_2, L)
2: GenerateDesired$y$-spacing(r_1, r_2, e_1, e_2, L)

Algorithm 2 GenerateDesired$x$-spacing(s_1, s_2, e_1, e_2, L)

Require: Knowledge of starting (s_1, s_2) and ending (e_1, e_2) positions of two robots. $(0, 0)$ is bottom corner, s_1 is topmost robot, L is length of the walls. Current robot positions are (r_1, r_2).

Ensure: $r_{1y} - r_{2y} \equiv s_{1y} - s_{2y}$

1: $\epsilon \leftarrow$ small number
2: $\Delta s_x \leftarrow s_{1x} - s_{2x}$
3: $\Delta e_x \leftarrow e_{1x} - e_{2x}$
4: $r_1 \leftarrow s_1$, $r_2 \leftarrow s_2$
5: **if** $\Delta e_x < 0$ **then**
6: $\quad m \leftarrow (L - \epsilon - \max(r_{1x}, r_{2x}), 0)$ ▷ Move to right wall
7: **else**
8: $\quad m \leftarrow (\epsilon - \min(r_{1x}, r_{2x}), 0)$ ▷ Move to left wall
9: **end if**
10: $m \leftarrow m + (0, -\min(r_{1y}, r_{2y}))$ ▷ Move to bottom
11: $r_1 \leftarrow r_1 + m$, $r_2 \leftarrow r_2 + m$ ▷ Apply move
12: **if** $\Delta e_x - (r_{1x} - r_{2x}) > 0$ **then**
13: $\quad m \leftarrow (\min(|\Delta e_x - \Delta s_x|, L - r_{1x}), 0)$ ▷ Move right
14: **else**
15: $\quad m \leftarrow (-\min(|\Delta e_x - \Delta s_x|, r_{1x}), 0)$ ▷ Move left
16: **end if**
17: $m \leftarrow m + (0, \epsilon)$ ▷ Move up
18: $r_1 \leftarrow r_1 + m$, $r_2 \leftarrow r_2 + m$ ▷ Apply move
19: $\Delta r_x = r_{1x} - r_{2x}$
20: **if** $\Delta r_x \equiv \Delta e_x$ **then**
21: $\quad$ **return** (r_1, r_2)
22: **else**
23: $\quad$ **return** GenerateDesired$x$-spacing(r_1, r_2, e_1, e_2, L)
24: **end if**

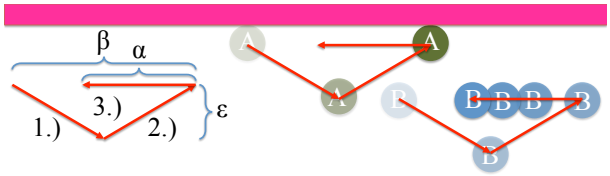

Fig. 6. A DriftMove$(\alpha, \beta, \epsilon)$ to the right repeats a triangular movement sequence $\{(\beta/2, -\epsilon), (\beta/2, \epsilon), (-\alpha, 0)\}$. Robot A touching a top wall moves right β units, while robots not touching the top move right $\beta - \alpha$.

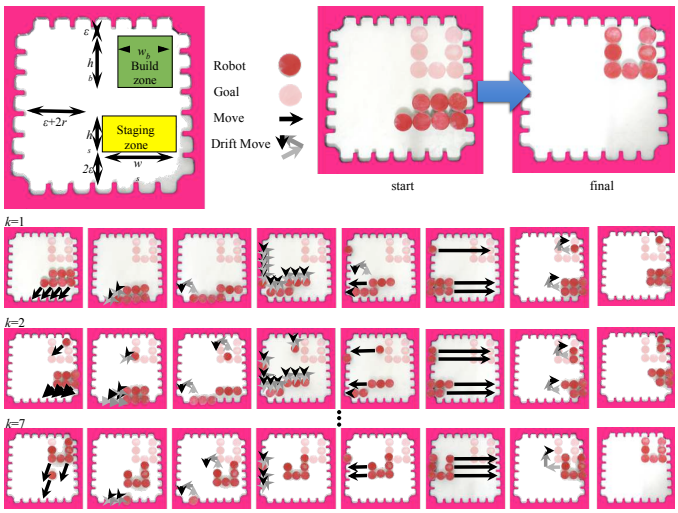

Fig. 7. Illustration of Alg. 3, n robot position control using wall friction.

B. Position Control of n Robots Using Wall Friction

Alg. 1 can be extended to control the position of n robots using wall friction under several constraints. The solution described here is an iterative procedure with n loops. The kth loop moves the kth robot from a *staging zone* to the desired position in a *build zone*. All robots move according to the global input, but due to wall friction, at the end the kth 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. 7 for a schematic of the build and staging zones.

Assume an open workspace with four axis-aligned walls with infinite friction. 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. Furthermore, there must be at least ϵ space above the build zone, ϵ below the staging zone, and $\epsilon + 2r$ to the left of the build and staging zone, where r is the radius of a robot. The minimum workspace is then $(\epsilon + 2r + \max(w_f, w_s), 2\epsilon + h_s, h_f)$.

The n robot position control algorithm relies on a $\mathrm{DriftMove}(\alpha, \beta, \epsilon)$ control input, shown in Fig. 6. A drift move consists of repeating a triangular movement sequence $\{(\beta/2, -\epsilon), (\beta/2, \epsilon), (-\alpha, 0)\}$. The robot touching a top wall moves right β units, while robots not touching the top move right $\beta - \alpha$.

Let $(0, 0)$ be the lower left corner of the workspace, p_k the x, y position of the kth robot, and f_k the final x, y position of the kth robot. Label the robots in the staging zone from left-to-right and top-to-bottom, and the f_k configurations right-to-left and top-to-bottom as shown in Fig. 7.

Algorithm 3 PositionControl$n$RobotsUsingWallFriction(k)

```
move( −ϵ, r − p_{k,y})
while p_{k,x} > r do
    DriftMove(ϵ, min(p_{k,x} − r, ϵ), ϵ) left
end while
m ← ceil((f_{k,y} − r)/ϵ)
β ← (f_{k,y} − r)/m
α ← β − (r − p_{k,y} − ϵ)/m
for m iterations do
    DriftMove(α, β, ϵ) up
end for
move(r + ϵ − f_{k,x}, 0)
move(f_{k,x} − r, 0)
```

Alg. 3 proceeds as follows: First, the robots are moved left away from the right wall, and down so robot k touches the bottom wall. Second, a set of $\mathrm{DriftMove}()$s are executed that move robot k to the left wall with no net movement of the other robots. Third, a set of $\mathrm{DriftMove}()$s are executed that move 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.

C. Controlling Covariance Using Wall Friction

Assume an open workspace with infinite boundary friction. Goal variances and covariance are $(\sigma^2_{goalx}, \sigma^2_{goaly}, \sigma_{goalxy}$ and mean, variances and covariance of the swarm are $(\bar{x}, \bar{y}, \sigma^2_x, \sigma^2_y, \sigma_{xy})$. For our experiments, $c_1 = 0.1$.

1) swarm is pushed into the left wall until $\sigma^2_x < c_1\sigma^2_{goalx}$.
2) swarm's mean position is moved to the center of the workspace
3) swarm is pushed into the bottom wall until $\sigma^2_y \leq \sigma^2_{goalt}$.
4) if $\sigma_{goalxy} > 0$ swarm slides right until $\sigma_{xy} \geq \sigma_{goalxy}$ else swarm slides left until $\sigma_{xy} \leq \sigma_{goalxy}$
5) swarm's mean position is moved to the center of the workspace

IV. Simulation

Two simulations were implemented using wall-friction for position control. The first controls the position of two robots, the second controls the position of n robots. All code is available online at Zhao and Becker [28, 29].

Two additional simulations were performed using wall-friction to control variance and covariance. The first is an open-loop algorithm that demonstrates the effect of varying friction levels. The second uses a closed-loop controller to achieve desired variance and covariance values.

A. Position Control of Two Robots

Algorithms 1, 2, were implemented in Mathematica using point robots (radius = 0). Fig. 5 shows this algorithm for two configurations. Robot initial positions are shown by a crosshair, and final positions by a circled crosshair. Dashed lines show the shortest route if robots could be controlled independently. The path given by Alg. 1 is shown with solid arrows. Each row has five snapshots taken every quarter second. For the sake of brevity axis-aligned moves were replaced with oblique moves that combine two moves simultaneously. Δr_x is adjusted to Δe_x in the second snapshot at $t = 0.25$. The following frames adjust Δr_y to Δe_y. Δr_y is corrected by $t = 0.75$. Finally, the algorithm moves the robots to their corresponding destinations.

In the worse case, adjusting both Δr_x and Δr_y requires two iterations. Two iterations of Alg. 2 are only required if $|\Delta e_x - \Delta s_x| > L$. Similarly, two iterations are only required if $|\Delta e_y - \Delta s_y| > L$. An online interactive demonstration and source code of the algorithm are available at [22].

B. Position Control of n Robots

Alg. 3 was simulated in MATLAB using square block robots with unity width. Code is available at [14]. Simulation results are shown in Fig. 8 for arrangements with an increasing number of robots, n= [8, 46, 130, 390, 862]. The distance moved grows quadratically with the number of robots n. A best-fit line $210n^2 + 1200n - 10,000$ is overlaid by the data..

In Fig. 8, the amount of clearance is $\epsilon = 1$. Control performance is sensitive to the desired clearance. As ϵ increases, the total distance decreases asymptotically, as shown in Fig. 9, because the robots have more room to maneuver and fewer $\mathrm{DriftMoves}$ are required.

C. Efficient Control of Covariance

Random disturbances impair the performance of Alg. 1 and Alg. 3. Still, we are able to control covariance of the swarm. This section demonstrates simulations of controlling covariance of the swarm. These simulations use the 2D physics engine Box2D, by Catto [5]. 144 disc-shaped robots were controlled by an open-loop control input as illustrated in Fig. 10. All robots had the same initial conditions, but in four tests the boundary friction was $F_f = \{0, 1/3F, 2/3F, F\}$.

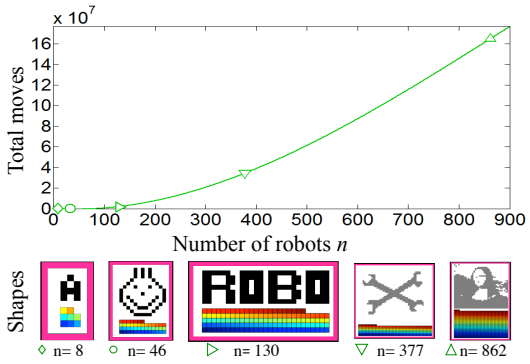

Fig. 8. The required number of moves under Alg. 3 using wall-friction to rearrange n square-shaped robots. See hardware implementation and simulation at [14].

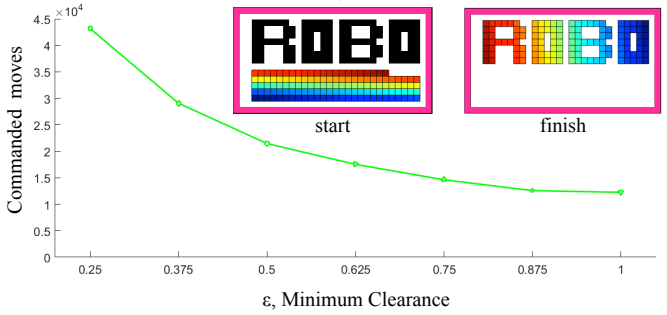

Fig. 9. Control performance is sensitive to the desired clearance ϵ. As ϵ increases, the total distance decreases asymptotically.

Without friction, covariance has minimal variation. As friction increases, the covariance can be manipulated to greater degrees.

144 disc-shaped robots were also controlled by a closed-loop controller using the procedure in §III-C. Fig. 11 illustrates that covariance and variances in x and y axis were controlled

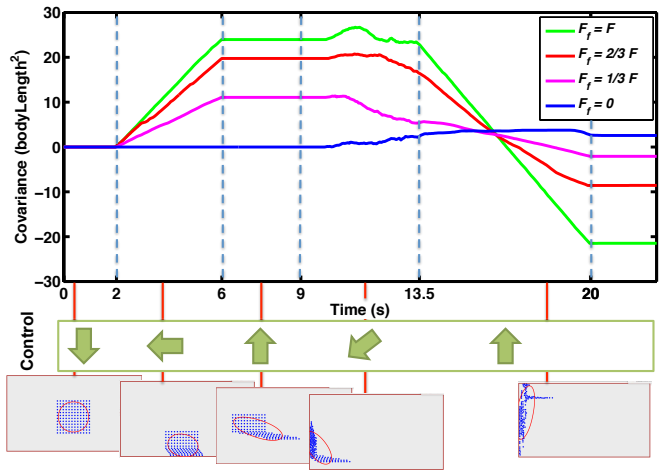

Fig. 10. Open-loop simulation with 144 disc robots and varying levels of boundary friction under the same initial conditions. Without friction, covariance is unchangeable. As friction increases, the covariance can be manipulated to greater degrees.

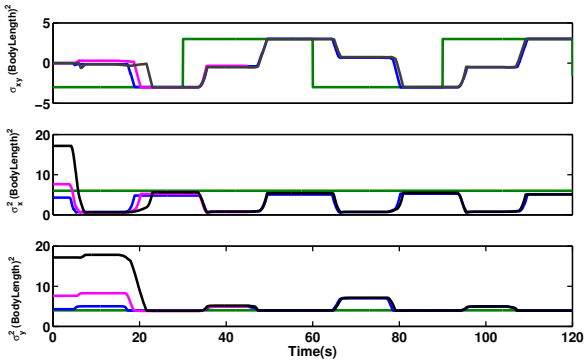

Fig. 11. Closed-loop simulation with 144 disc robots and three sets of initial conditions. The algorithm tracks goal variance and covariance values (green). The goal covariance switches sign every 30 s.

from a set of initial conditions.

V. EXPERIMENT

Our experiments are on centimeter-scale hardware systems called *kilobots*. These 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, from Rubenstein et al. [20, 21] is a low-cost robot designed for testing collective algorithms with large numbers of robots. It is available as an open source platform or commercially from K-Team [11]. 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. In these experiments as shown in Fig. 13, we used n=100 kilobots, a 1 m×1 m whiteboard as the workspace, four 30W and four 50W 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. Above the table, an overhead machine vision system tracks the position of the swarm.

A. Hardware Experiment: Position Control of Two Robots

The walls of the hardware platform have almost infinite friction, due to a laser-cut, zigzag border. When a kilobot is steered into the zigzag border, they pin themselves to the wall unless the global input directs them away from the wall. This wall friction is sufficient to enable independent control of two kilobots, as shown in Fig. 12.

B. Hardware Experiment: Position Control of n Robots

The hardware setup has a bounded platform, magnetic sliders, and a magnetic guide board. Designs for each are available at [**?**]. The pink boundary is toothed with a white free space, as shown in Fig 7. Only discrete, 1 cm moves in the x and y directions are used. The goal configuration highlighted in the top right corner represents a 'U' made of seven sliders. The dark red configuration is the current position of the sliders. Due to the discretized movements allowed by

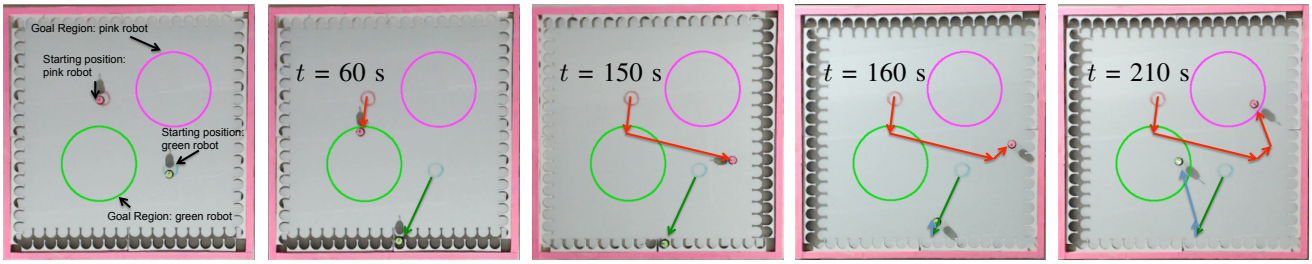

Fig. 12. Position control of two kilobots (Alg. 2) steered to corresponding colored circle. Boundary walls have nearly infinite friction, so the green robot is stopped by the wall from $t = 60$s until the commanded input is directed away form the wall at $t = 150$s, while the pink robot in free-space is unhindered.

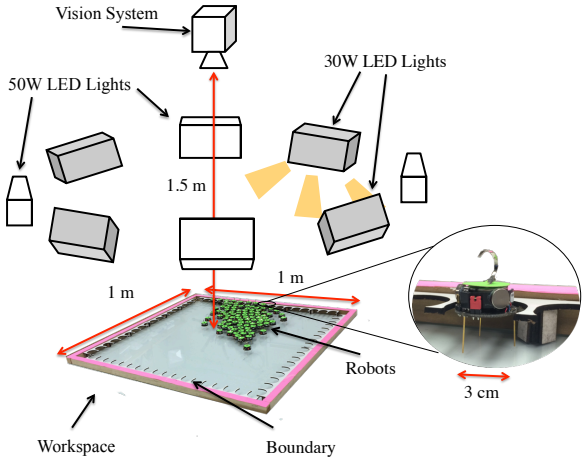

Fig. 13. Hardware platform: table with 1×1 m workspace, surrounded by eight triggered LED floodlights, with an overhead machine vision system.

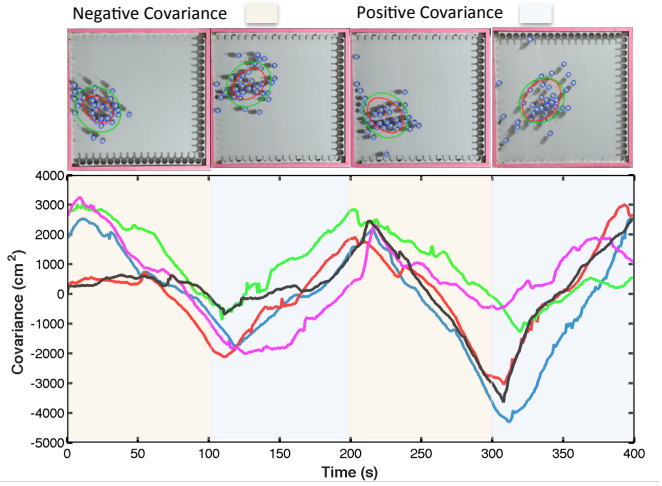

Fig. 14. Hardware demonstration steering 100 kilobot robots to desired covariance. The goal covariance is negative in first 100 seconds and is positive in the next 100 seconds. The actual covariance is shown in different trials. Frames above the plot show output from machine vision system and an overlaid covariance ellipse.

the boundary, drift moves follow a 1 cm square. Free robots return to their start positions but robots on the boundary to move laterally, generating a net sliding motion.

Fig. 7 follows the motion of the sliders through iterations k=1, 2 and 7. All robots receive the same control inputs, but boundary interactions breaks the control symmetry. Robots reach their respective goal position in a first-in, first-out arrangement beginning with the bottom-left robot from the staging zone occupying the top-right position of the build zone.

C. Hardware Experiment: Control of Covariance

To demonstrate covariance control n=100 robots were placed on the workspace and manually steered with lights, using friction with the boundary walls to vary the covariance from -4000 to 3000 cm^2. The resulting covariance is plotted in Fig. 14, along with snapshots of the swarm.

VI. 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 Egerstedt and Hu [8], and the gradient-based decentralized controllers using control-Lyapunov functions in Hsieh et al. [10]. 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 Martel [15], Yan et al. [27] and Chowdhury et al. [6], lack onboard computation.

Instead, this paper focuses on centralized techniques that apply the same control input to each member of the swarm. Precision control requires breaking the symmetry caused by the global input. Symmetry can be broken using agents that respond differently to the global control, either through agent-agent reactions, see work modeling biological swarms Bertozzi et al. [3], or engineered inhomogeneity Bretl [4], Donald et al. [7], Becker et al. [2]. This work assumes a uniform control (1) with homogenous agents, as in Becker et al. [1]. The techniques in this paper are inspired by fluid-flow techniques and artificial force-fields.

Fluid-flow: Shear forces are unaligned forces that push one part of a body in one direction, and another part of the body in the opposite direction. These are common in fluid flow along boundaries. Most introductory fluid dynamics textbooks provide models, for example, see Munson et al. [17]. Similarly, a swarm of robots under global control pushed

along a boundary will experience shear forces. This is a position-dependent force, and so can be exploited to control the configuration or shape of the swarm. Physics-based swarm simulations used these forces to disperse a swarm's spatial position for coverage in Spears et al. [24].

Artificial Force-fields: Much research has focused on generating non-uniform artificial force-fields that can be used to rearrange passive components. Applications have included techniques to design shear forces for sensorless manipulation of a single object by Lamiraux and Kavraki [13]. Vose et al. [25, 26] demonstrated a collection of 2D force fields generated by 6DOF 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, but required multi-modal, position-dependent control.

VII. Conclusion and Future Work

This paper presented techniques for controlling the shape of a swarm of robots using global inputs and interaction with boundary friction forces. The paper provided algorithms for precise position control, as well as demonstrations of efficient covariance control. Extending algorithms 2 and 1 to 3D is straightforward but increases the complexity. Future efforts should be directed toward improving the technology and tailoring it to specific robot applications.

With regard to technological advances, this includes designing controllers that efficiently regulate σ_{xy}, perhaps using Lyapunov-inspired controllers as in Kim et al. [12]. Additionally, this paper assumed that wall friction was nearly infinite. The algorithms require retooling to handle small μ_f friction coefficients. It may be possible to rank controllability as a function of friction. In hardware, the wall friction can be varied by laser-cutting boundary walls with different of profiles.

Acknowledgments

This work was supported by the National Science Foundation under Grant No. [IIS-1553063].

References

[1] 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.

[2] Aaron Becker, Cem Onyuksel, Timothy Bretl, and James McLurkin. Controlling many differential-drive robots with uniform control inputs. *Int. J. Robot. Res.*, 33(13):1626–1644, 2014.

[3] Andrea L Bertozzi, Theodore Kolokolnikov, Hui Sun, David Uminsky, and James Von Brecht. Ring patterns and their bifurcations in a nonlocal model of biological swarms. *Communications in Mathematical Sciences*, 13(4), 2015.

[4] Timothy Bretl. Control of many agents using few instructions. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

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

[6] Sagar Chowdhury, Wuming Jing, and David J. Cappelleri. Controlling multiple microrobots: recent progress and future challenges. *Journal of Micro-Bio Robotics*, 10(1-4):1–11, 2015.

[7] Bruce R Donald, Christopher G Levey, Igor Paprotny, and Daniela Rus. Planning and control for microassembly of structures composed of stress-engineered mems microrobots. *The International Journal of Robotics Research*, 32(2):218–246, 2013.

[8] Magnus Egerstedt and Xiaoming Hu. Formation constrained multi-agent control. *IEEE Trans. Robotics Automat.*, 17:947–951, 2001.

[9] Ambarish Ghosh and Peer Fischer. Controlled propulsion of artificial magnetic nanostructured propellers. *Nano Letters*, 9(6):2243–2245, 2009.

[10] M Ani Hsieh, Vijay Kumar, and Luiz Chaimowicz. Decentralized controllers for shape generation with robotic swarms. *Robotica*, 26(05): 691–701, 2008.

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

[12] Paul Seung Soo Kim, Aaron Becker, Yan Ou, Anak Agung Julius, and Min Jun Kim. Imparting magnetic dipole heterogeneity to internalized iron oxide nanoparticles for microorganism swarm control. *Journal of Nanoparticle Research*, 17(3):1–15, 2015.

[13] 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*, 20(8):635–659, 2001.

[14] Arun Viswanathan Mahadev and Aaron T. Becker. "Arranging a robot swarm with global inputs and wall friction [discrete]." MATLAB Central File Exchange, December 2015. URL https://www.mathworks.com/matlabcentral/fileexchange/54526.

[15] Sylvain Martel. Magnetotactic bacteria for the manipulation and transport of micro-and nanometer-sized objects. *Micro-and Nanomanipulation Tools*, 2015.

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

[17] Bruce R. Munson, Alric P. Rothmayer, Theodore H. Okiishi, and Wade W. Huebsch. *Fundamentals of Fluid Mechanics*. Wiley, 7th edition, 2012.

[18] Yan Ou, Dal Hyung Kim, Paul Kim, Min Jun Kim, and A. Agung Julius. Motion control of magnetized tetrahymena pyriformis cells by magnetic field with model predictive control. *Int. J. Rob. Res.*, 32(1):129–139, January 2013.

[19] Famin Qiu and Bradley J Nelson. Magnetic helical micro-and nanorobots: Toward their biomedical applications. *Engineering*, 1(1): 21–26, 2015.

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

[21] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345 (6198):795–799, 2014.

[22] Shiva Shahrokhi and Aaron T. Becker. "Moving Two Particles with Shared Control Inputs Using Wall Friction", Wolfram Demonstrations Project, November 2015. URL http://demonstrations.wolfram.com/MovingTwoParticlesWithSharedControlInputsUsingWallFriction/.

[23] Shiva Shahrokhi and Aaron T. Becker. Stochastic swarm control with global inputs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page tbd, September 2015.

[24] Diana Spears, Wesley Kerr, and William Spears. Physics-based robot swarms for coverage problems. *The international journal of intelligent control and systems*, 11(3), 2006.

[25] T.H. Vose, P. Umbanhowar, and K.M. Lynch. Friction-induced velocity fields for point parts sliding on a rigid oscillated plate. *The International Journal of Robotics Research*, 28(8):1020–1039, 2009.

[26] Thomas H Vose, Paul Umbanhowar, and Kevin M Lynch. Sliding manipulation of rigid bodies on a controlled 6-dof plate. *The International Journal of Robotics Research*, 31(7):819–838, 2012.

[27] Xiaohui Yan, Qi Zhou, Jiangfan Yu, Tiantian Xu, Yan Deng, Tao Tang, Qian Feng, Liming Bian, Yan Zhang, Antoine Ferreira, and Li Zhang. Magnetite nanostructured porous hollow helical microswimmers for targeted delivery. *Advanced Functional Materials*, 25(33):5333–5342, 2015. ISSN 1616-3028.

[28] Haoran Zhao and Aaron T. Becker. "distribution of a swarm of robots in a circular workplace under gravity", wolfram demonstrations project, February 2016. URL http://demonstrations.wolfram.com/DistributionOfASwarmOfRobotsInACircularWorkplaceUnderGravity/.

[29] Haoran Zhao and Aaron T. Becker. "distribution of a robot swarm in a square under gravity", wolfram demonstrations project, January 2016. URL http://demonstrations.wolfram.com/DistributionOfARobotSwarmInASquareUnderGravity/.