Exploiting Non-Slip Wall Contacts to Position Two Particles Using a Shared Input

Shiva Shahrokhi, Jingang Shi, Benedict Isichei, and Aaron T. Becker

***Abstract*— Steered particles offer a method for targeted therapy, interventions, and drug delivery in regions inaccessible by large robots. Magnetic actuation has the benefits of requiring no tethers, being able to operate from a distance, and in some cases allows imaging for feedback (e.g. MRI). Given three orthogonal magnetic fields, steering one particle in 3D is trivial. Adding additional particles to steer makes the system underactuated because there are more states than control inputs. The walls of in vivo and artificial environments often have surface roughness such that the particles do not move unless actuation pulls them away from the wall. In previous works, we showed that the individual 2D position of two particles is controllable in a square workspace with non-slip wall contact. However, in vivo environments are usually not square. This work extends the previous work to convex workspaces and 3D positioning. This paper also implements the algorithms using a hardware setup inspired by intestine anatomy.**

I. Introduction

Particle swarms propelled by a uniform field, where each particle receives the same control input, are common in applied mathematics, biology, and computer graphics.

The small size of these robots makes it difficult to perform onboard computation. Instead, these robots are often controlled by a broadcast signal. The tiny robots themselves are often just rigid bodies, and it may be more accurate to define the *system*, consisting of particles, a uniform control field, and sensing, as the robot. Such systems are severely underactuated, having 2 degrees of freedom in the shared control input, but $2n$ degrees of freedom for the particle swarm. Techniques are needed that can handle this underactuation. In previous work, we showed that the 2D position of each particle in such a swarm is controllable if the workspace contains a single obstacle the size of one particle.

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

The paper is arranged as follows. After a review of recent related work in Sec. II, Sec. III-A introduces a model for boundary interaction. We provide a shortest-path algorithm to arbitrarily position two robots in Sec. IV. Sec. V describes implementations of the algorithms in simulation and Sec. VI describes hardware experiments, as shown in Fig. 1. We end with directions for future research in Sec. VII.

*This work was supported by the National Science Foundation under Grant No. [IIS-1553063] and [IIS-1619278].

Authors are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204 USA {sshahrokhi2,atbecker}@uh.edu

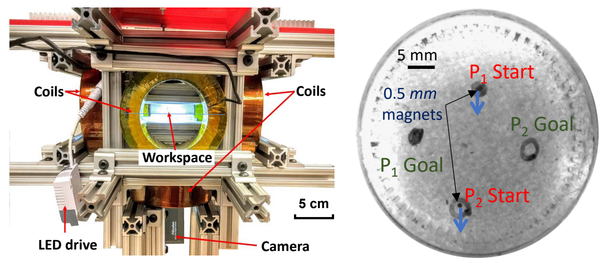

Fig. 1. Positioning particles that receive the same control inputs, but cannot move while a control input pushes them into a boundary.

This paper is elaboration of preliminary work in a conference paper [1] which consider only square workspaces. This work extends the analysis to convex workspaces and 3D positioning. This paper also implements the algorithms using a hardware setup inspired by intestine anatomy.

II. 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 [2], and the gradient-based decentralized controllers using control-Lyapunov functions in [3]. 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 [4]–[6] lack onboard computation.

Instead, this paper focuses on centralized techniques that apply the same control input to both robots. Precision control requires breaking the symmetry caused by the uniform input. Symmetry can be broken using particles that respond differently to the uniform control signal, either through agent-agent reactions, see work modeling biological swarms [7], or engineered inhomogeneity [8]–[10]. This work assumes a uniform control with homogenous particles, as in [11]. The techniques in this paper are inspired by 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 [12]. [13] demonstrated a collection of 2D force fields generated by six degree-of-freedom vibration inputs to a rigid plate. These force fields, including shear forces, could be used as a set of primitives for motion control to steer the formation of multiple objects. However unlike the uniform control model in this paper, their control was multi-modal and position-dependent.

This paper develops control algorithms using uniform control fields, such as the magnetic resonance navigation [14].Much recent work has focused on exploiting inhomogeneities in the magnetic field to control multiple micro particles using gradient-based pulling [15], [16]. Unfortunately, using large-scale external magnetic fields makes it challenging to independently control more than one microrobot unless the distance between the electromagnetic coils is at the same length scales as the robot workspace [17], [15].

In contrast to methods that exploit inhomogeneities in the magnetic field to control multiple micro particles, e.g. [16], who used four magnetic coils in close proximity to the workspace to achieve trajectory control of two microspheres, this paper uses the same control input requiring only a coil in each direction to position the particles. Systems like this one are poorly suited for PRM and RRT*-type methods [18] because if during a movement a collision occurs, that movement is irreversible.

III. Theory

A. Boundary Interaction Model

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

If the i^{th} particle has position $\mathbf{x}_i(t)$ and velocity $\dot{\mathbf{x}}_i(t)$, we assume the following system model:

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

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

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

These system dynamics represent particle swarms in low-Reynolds number environments, where viscosity dominates inertial forces and so velocity is proportional to input force [19]. In this regime, the input force command $\mathbf{u}(t)$ controls the velocity of the robots. The same model can be generalized to particles moved by fluid flow where the vector direction of fluid flow $\mathbf{u}(t)$ controls the velocity of particles, or for a swarm of robots that move at a constant speed in a direction specified by a uniform input $\mathbf{u}(t)$ [20]. As in our model, fluid flowing in a pipe has zero velocity along the boundary. Similar mechanical systems exist at larger scales, e.g. all tumblers of a combination look move uniformly unless obstructed by an obstacle. Our control problem is to design the control inputs $\mathbf{u}(t)$ to make all n particles achieve a task.

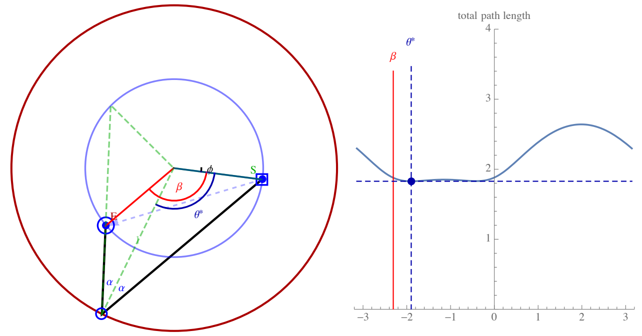

Fig. 2. The shortest path between two points (blue square) to (blue ellipse) in the unit disk that intersects the circumference. The path length as a function of intersection point, $(\cos\theta, \sin\theta)$ is shown at right.

B. Shortest Path

The shortest path between two points in the unit disk that reflects off the circumference is composed of two straight line segments shown in Fig. 2. The problem can be simplified by choosing the coordinate system carefully. We define the x axis along the position of the starting point: $S = (s, 0)$, and define the point of intersection by the angle θ from the x axis $P = (\cos\theta, \sin\theta)$, and the final point by a radius e and angle β, $E = e(\cos\beta, \sin\beta)$. Then define a symmetry point about S of line OP named T. Then the length of the two line segments is

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

which is minimized by choosing an appropriate θ value. This equation can be simplified to

$$\sqrt{1 + e^2 - 2e\cos(\beta - \theta)} + \sqrt{1 + s^2 - 2s\cos\theta}. \tag{3}$$

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

Simplifying this system of equations results in: $s = e\csc\theta(s\sin(2\theta-\beta) + \sin(\beta-\theta))$. Solving this last equation results in a quartic solution that has a closed-form solution with four roots, each of which can be either a clockwise or a counterclockwise rotation θ, depending on the sign of β, with $-\pi <= \beta <= \pi$. We evaluate each and select the solution that results in the shortest length path. Note that the optimal path satisfies the law of reflection off the unit circle, with angle of incidence equal to angle of reflection.

IV. Position Control of Two Robots Using Boundary Interaction

Alg. 1 uses non-slip contacts with walls to arbitrarily position two robots in a convex workspace. In our previous work we used a rectangular workspace. We use the same idea here but we modify the algorithm to handle any convex workspace, then we expand how the algorithm works in circular workspace.

A. Convex Polygonal Workspaces

Fig. 3 shows different workspaces and their representative Δ configuration spaces. Alg. 2 shows how to compute reachable set for any convex polygonal workspace. Consider one robot touching each vertex of the workspace as shown in Fig.4. For each pair of vertices, compute where the other robot will be if the first robot goes to that vertex. If the final position of the second robot is still inside the workspace, then its position is one of the vertices of the reachable set. If the point is not inside the polygon, then the intersection of the line that point and the second robot's position with the polygon is one vertex of the reachable set. Compute the distance to all the vertices of the workspace from this point. By subtracting the relative distance of the robots, then all the vertices of one reachable set are found. Doing this for all the vertices of the workspace will give us all the reachable sets.

B. Circular Workspaces

Assume two robots are initialized at s_1 and s_2 with corresponding goal destinations g_1 and g_2. Denote the current positions of the robots p_1 and p_2 and the current distance between the robots is d. Values $.x$ and $.y$ denote the x and y coordinates, i.e., $p_1.x$ and $p_1.y$ denote the x and y locations of p_1. The algorithm assigns a uniform control input at every instance. The goal is to move the particles to the goal positions using a shared control input. We do this by first moving them to the correct relative position and then translating the particles to the goal. The first step minimizes $||\Delta g - \Delta p|| = ||(g_2 - g_1) - (p_2 - p_1)||$.

We define a Δ configuration space as a circular shape that considers all possible Δps. We also show the starting and ending relative distance as Δs and Δe in Δ configuration space in Fig. 5. Reachable set is the part of Δ configuration space where if one robot touches a wall in a specific location, the other robot can make the required relative distance without causing touching robot to move. To compute reachable set for circular workspace, first we considered all passible hitting point locations in the workspace. The set of boundary points that a robot can touch before the other robot touches is an arc of angle $2(\pi - \frac{\arcsin d}{r})$, where $d = |s_1 - s_2|$

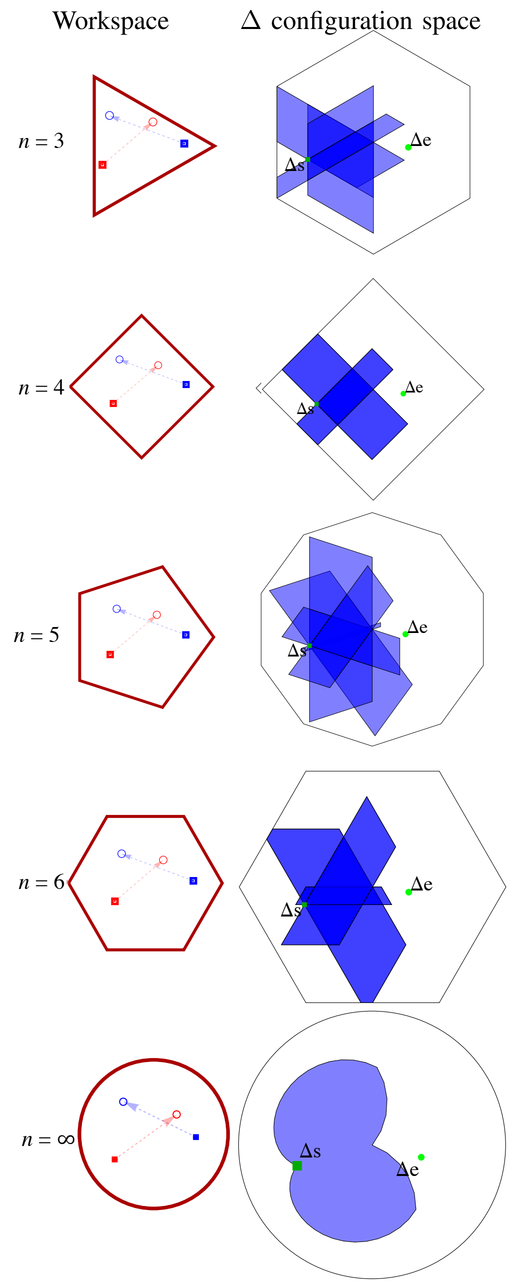

Fig. 3. Workspace and Δ configuration spaces for different polygonal workspaces and their representative Δ configuration spaces and reachable sets. As the number of sides in the polygon increases, the total area of the Δ configuration space is four times of the workspace.

and r is the radius of the circle. We define the angle between two particles as $\theta = \arctan(\frac{p_1.x - p_2.x}{p_1.y - p_2.y})$.

Expanding a path means either moving directly to the goal, or pushing one robot to a wall and adjusting the relative position of the other robot. As soon as the goal is reached, the algorithm returns this path.

There are infinite reachable sets, parameterized by first contact location ψ, as shown in Fig. 5.

$$\psi \in \left[\theta + \frac{\sin^{-1} d}{2r} - \frac{\pi}{2}, \theta - \frac{\sin^{-1} d}{2r} + \frac{\pi}{2}\right] \quad (4)$$

γ is half the angle of the arc that the reachable set's chord

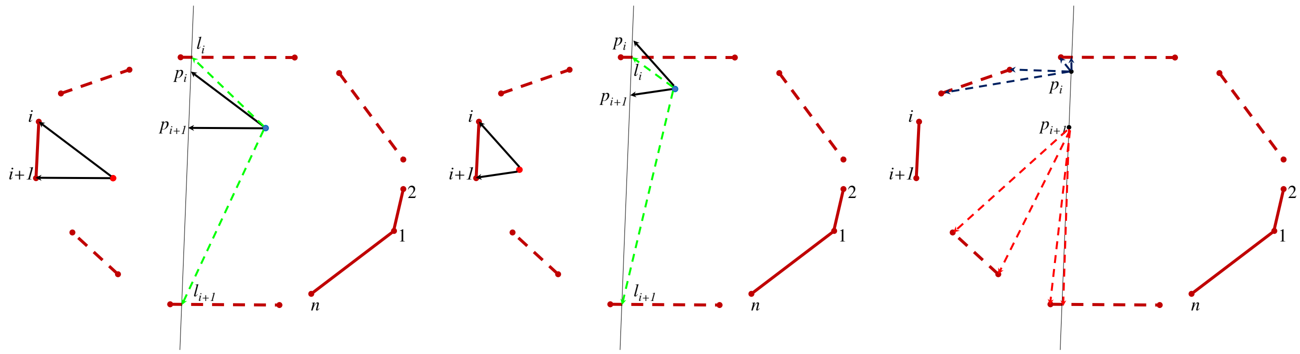

Fig. 4. Steps to generate the reachable set when one particle collides with edge $i, i+1$ of a convex polygonal workspace.

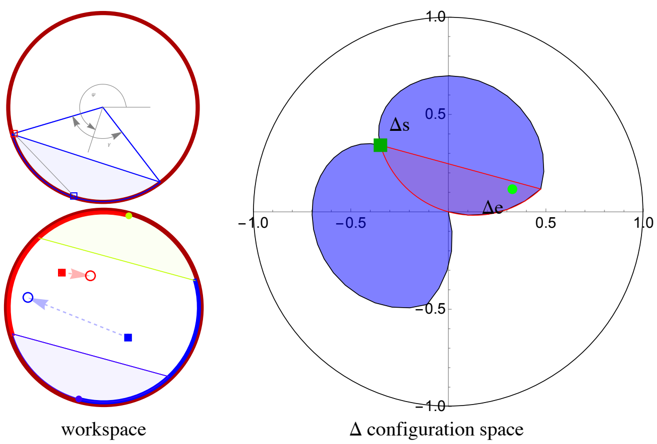

Fig. 5. Left: the set of points where the red robot is the first to contact the boundary are drawn with a red arc. The set of points where the blue robot is the first to contact the boundary are drawn with a blue arc. The possible points for the blue and pink particles to touch the boundary is shown in blue and pink arcs. Right: When the blue particle is touching a wall (blue square) the other particle (pink square) can go anywhere in the reachable set (blue region).

has shown in Fig. 5 and is calculated by:

$$p_\psi = r[\cos(\psi), \sin(\psi)] \tag{5}$$

$$d_\perp = 2\,|(s_1.p_\psi - s_2.p_\psi)| \tag{6}$$

$$\gamma = \cos^{-1}\left(1 - \frac{d_\perp}{r}\right) \tag{7}$$

Reachable sets with π difference in ψ value are equivalent in the Δ configuration space, so we can plan in this space and choose to immobilize the particle closest to a wall.

The equation for the four lines outlining the reachable set can be found as follows:

$$\begin{aligned}
l_1 &= r\Big((\cos\psi_{\min} - \cos(\gamma_c + \psi_{\min})) \\
&\quad + (\sin\psi_{\min} - \sin(\gamma_c + \psi_{\min}))\Big) && 0 < \gamma_c < \gamma_{\psi_{\min}}, \\
l_2 &= r\Big((\cos\psi_{\max} - \cos(\gamma_c + \psi_{\max})) \\
&\quad + (\sin\psi_{\max} - \sin(\gamma_c + \psi_{\max}))\Big) && \gamma_{\psi_{\max}} < \gamma_c < 0, \\
l_3 &= r\Big((\cos\psi - \cos(\psi + \gamma_c)) \\
&\quad (\sin\psi - \sin(\psi + \gamma_c))\Big) && \psi_{\min} < \psi < \psi_{\max}, \\
l_4 &= r\Big((\cos\psi - \cos(\psi - \gamma_c)) \\
&\quad (\sin\psi - \sin(\psi - \gamma_c))\Big) && \psi_{\min} < \psi < \psi_{\max}.
\end{aligned} \tag{8}$$

We combine these boundaries to make the reachable set. The algorithm first makes the reachable set, and then checks if the goal relative position is in the reachable set. If it is not in the reachable set, the closest point on the reachable set from Δg would be our current goal. We need to find a ψ that would enable us to reach to the required relative goal distance. To do so, we first check ψ_{min} and ψ_{max}. (9) shows if any point p, is in the region made by the robots if the touching robot has the angle ψ.

$$\begin{aligned}
&(p.x - c.x)^2 + (p.y - c.y)^2 > r^2 \\
&(p_x - c_x)\cos\psi + (p.y - c.y)\sin\psi \leq -r.\cos\gamma
\end{aligned} \tag{9}$$

Here c is coordinate of the circle and r is the radius of the workspace. If Δg is not in the region made by $\psi_{\min}$ or $\psi_{\max}$, we draw a line from Δg and the current relative position, Δs. This line is a chord of the circle and we find the ψ that makes this region. This equation finds this ψ:

$$\psi = \tan^{-1}(\Delta p - \Delta g). \tag{10}$$

Now that ψ is found, we move the particles to make the current goal. If current goal is our final goal, we go to the final goal position. If it is not our final goal, we continue to set the closest point on the reachable set to our current goal until we reach the goal.

C. 3D workspaces: Cylinders and Prisms

Extending path planning to 3D is possible only if the two particles do not initially have the same x and y positions. For ease of analysis, we assume the workspace boundaries extend in the $\pm z$ direction to form either right cylinders or right prisms. If the 3D projection is at a different angle, redefine the 2D workspace as a region perpendicular to the projection. First, we move the closest particle to the boundary, which prevents its z coordinate from changing. We

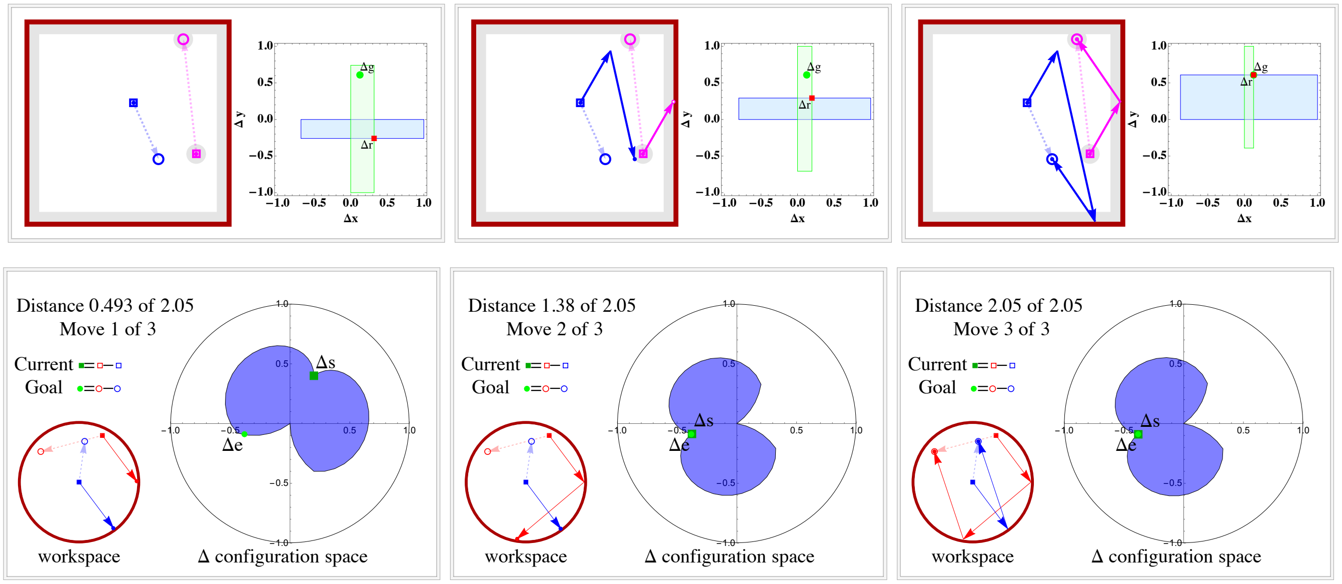

Fig. 6. Top row shows a polygonal workspace with its reachable sets. Bottom row, left circle shows the workspace. Right shows the Δ configuration space and the reachable set that is shown in red is representative of the point we need to go to get to the goal relative distance in one move.

Algorithm 1 2-PARTICLEPATHPLANNING(s_1, s_2, g_1, g_2, P)

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

$(p_1, p_2) \leftarrow (s_1, s_2)$
moves $\leftarrow \{\}$
$R \leftarrow \{p_1, p_2, g_1, g_2, \text{moves}\}$ ▷ R contains the current robot positions, the goal positions, and the move sequence
$\Delta p \leftarrow p_2 - p_1$
$\Delta g \leftarrow g_2 - g_1$
while $|\Delta g| < |\Delta p| + \epsilon$ **do**
  R_s = Compute reachable set ▷ use Alg. 2 or Alg. 3
  $\Delta g_c \leftarrow$nearest point in R_s to Δg
  add move to hit the wall
  add move to reach Δg_c
  Add the required moves to moves for the particles to achieve Δg_c
end while
moves $\leftarrow$ moves $+ g_2 - p_2$ ▷ translate to goal
return moves

Algorithm 2 POLYGONALWORKSPACE(s_1, s_2, g_1, g_2, P)

Require: knowledge of starting (s_1, s_2) and goal (g_1, g_2) positions of two robots. P is a list of the vertices of a convex polygon. RSet contains all the reachable polygons.

for i in P **do**
  $p_i \leftarrow s_1 + s_2 - i$
  $p_{i+1} \leftarrow s_1 + s_2 - (i + 1)$
  $L \leftarrow$ line with (p_{11}, p_{12})
  $l_i, l_{i+1} \leftarrow$ intersections of L and polygon
  if $p_i \notin$ polygon **then**
    $p_i \leftarrow l_i$
  end if
  $v_i \leftarrow$ All vertices $\in [l_i, l_{i+1}]$
  $d_i \leftarrow$ All vectors from p_i to $[l_i, i] \cup p_{i+1}$ to $[i+1, l_{i+1}]$
  RSet $\leftarrow$ RSet + polygon $(s_2 - (s_1 + d_i))$
end for
Return RSet

next apply actuation in either the $\pm z$ direction to achieve the desired Δz. Then the particles are actuated away from the boundary and to the appropriate z positions. Path planning continues using Alg. 1 to position the particles to the desired x and y positions. As an example, consider Fig. 7 which shows a cylindrical workspace. The blue particle starts in the blue disk and the red particle starts in the red disk. The two candidate shortest-length paths that touch the wall are shown with parallel arrows. Each arrow will cause one of the particles to touch the wall, enabling the other robot to move freely in the z axis to achieve the required relative position. This can be extended to other 3D workspaces if the workspace can be locally approximated as a 3D prism or cylinder. Other workspaces may be better handled by other path planners, such as [21], which used collisions with protrusions of the workspace to rearrange particles.

V. SIMULATION

A. Position Control of Two Robots

Algorithm 1 was implemented in Mathematica using point robots (radius = 0).

The contour plots in Fig. 8 left shows the length of the path for given s_1, s_2, g_1 with g_2 ranging over all the workspace. Fig. 8 right shows the total distance of the path. This plot clearly shows the nonlinear nature of the path planning.

Algorithm 3 CIRCULARWORKSPACE(s_1, s_2, g_1, g_2)

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

1: $\theta = \arctan(\frac{p_1.x - p_2.x}{p_1.y - p_2.y})$
2: Calculate p_ψ ▷ use (5)
3: Calculate γ ▷ use (7)
4: Calculate l_1, l_2, l_3, l_4 ▷ use (8)
5: polygon $\leftarrow$ Join(l_1, l_2, l_3, l_4)
6: return polygon

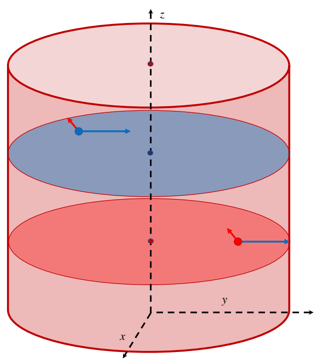

Fig. 7. Extending the algorithm to position the particles in 3D.

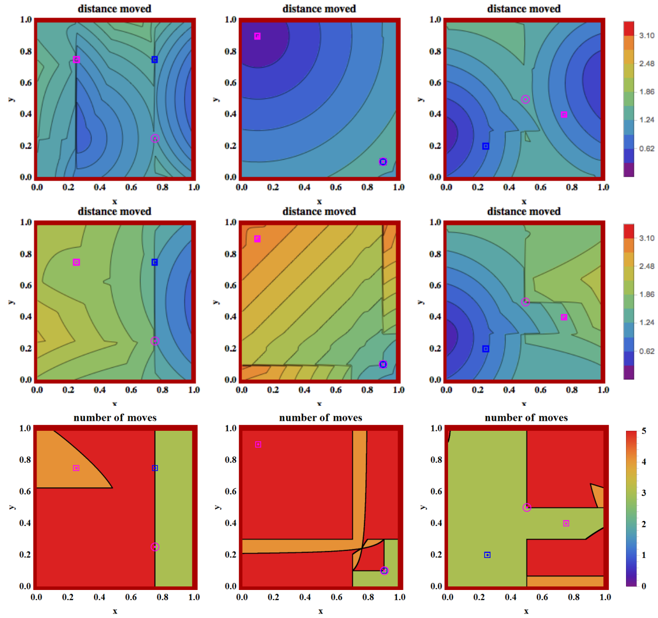

Fig. 10. Starting positions of robots 1 and 2 and goal position of robot 2 are fixed, and $\epsilon = 0.001$. The top row of contour plots show the distance if robot 1's goal position is varied in x and y. The bottom row shows the number of moves required for the same configurations.

The hardest point to achieve is the when the goals have π difference and are very close to the boundary.

The plots in Fig. 9 show the exponentially increasing number of moves and distance when the accuracy of reaching to the goal (δ) is getting to zero when the goal positions have π difference with each on the boundaries.

VI. EXPERIMENTAL RESULTS

To demonstrate Alg. 1 experimentally, we performed several tests. Each used the same magnetic setup. Two different intestine models were employed, the first a 3D-printed cross-section representation of a small intestine, and the second bovine small intestine.

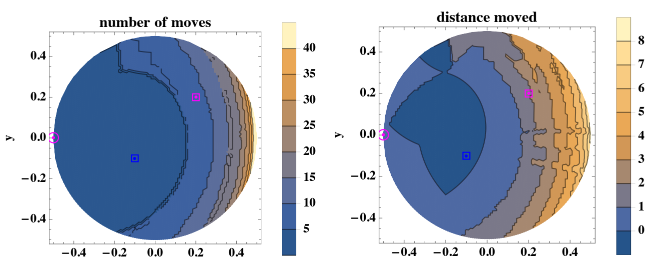

Fig. 8. Plots showing the algorithm with one goal on the boundary.

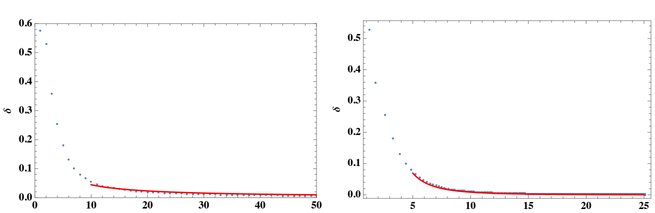

Fig. 9. Plots showing decreasing error when the number of moves grows.

A. Magnetic Manipulation Setup

The magnetic manipulation system consists of two pairs of electromagnetic coils that consist of iron cores at their centers, and arranged orthogonal to each other. The iron core at the center of each coil served to concentrate the magnetic field towards the workspace. To set up the magnetic manipulation system for this project, an Arduino and four SyRen regenerative motor drivers were used for control inputs to the coils. Finally, a Basler ace black and white camera was attached to the top of the system focusing on the back-lit workspace.

To obtain experimental data, the test samples which comprised of the Phantom intestine model and the bovine cross section, were placed in laser cut acrylic discs and then immersed in fluid. Over the course of experimentation, corn syrup was used to have the best viscosity for the experiments. The velocities of particles immersed in it were dampened enough to control their movements with ease. Spherical $0.5mm$ magnets were used as our particles.

todo: image of the magnetic setup with scale bar

B. Intestine Phantom Model

The Intestine Phantom Model was used as the first test field of the project and was made to mimic an intestine and its villi. The model consists of a circular ring with an outer diameter of $50mm$, an inner diameter of 46mm, and a thickness of 2mm created using a 3D printer and Fused filament fabrication. The model had some $2mm$ long protrusions on its inner surface to mimic the effects of intestinal villi on the target particles.

C. Bovine Intestine Cross-section

This phase of the project involved the use of beef intestines. Strips of intestine about 5mm thick were cut and placed in Neutrally buffered formalin for 24 hours for fixation. After fixation, each sample was transferred to 70% ethanol for storage. For the experiments, a slice of fixed intestine was attached to the acrylic disc with cyanoacrylate (superglue) and then submerged in corn syrup. A drawback of fixing the tissue samples before experimentation is that they tended to shrivel and dry up a few minutes after being removed from the 70% ethanol.

VII. Conclusion and Future Work

This paper presented techniques for controlling the position of a swarm of robots using uniform inputs and interaction with boundary friction forces. The paper provided algorithms for precise position control, as well as robust and efficient covariance control. Extending algorithms 1 to 3D is straightforward but increases the complexity. Additionally, this paper assumed friction was sufficient to completely stop particles in contact with the boundary. The algorithms require retooling to handle small friction coefficients. The algorithms assumed a rectangular workspace. This is a reasonable assumption for artificial environments, but in vivo environments are curved. A best-first-search program could still work, but it cannot take advantage of the 4-fold rotational symmetry as in a rectangular environment. Future efforts should be directed toward improving the technology and tailoring it to specific robot applications.

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

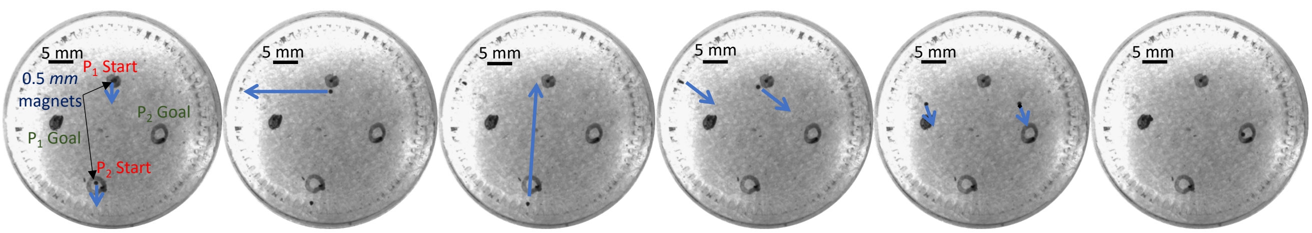

Fig. 11. Positioning particles that receive the same control inputs, but cannot move while a control input pushes them into a boundary.