Object Manipulation and Position Control Using a Swarm With Global Inputs

Shiva Shahrokhi and Aaron T. Becker

***Abstract*— Manipulating objects with a swarm of simple robots with global control inputs is difficult because they are highly under-actuated, the number of robots in contact with object changes dynamically, the robots contacting each other changes dynamically, and because the robots disperse over time they must be recollected.**

This work presents controllers and algorithms for steering such an under-actuated swarm to manipulate objects. Previous work showed that mean and variance of the swarm is controllable. This was used to manipulate convex polygonal objects through a simple maze. A key remaining challenge is controlling the torque applied to an object. Torque control is necessary for manipulating objects as well as for aligning sensors, emitters, or redirecting an incoming signal. This work first proves that swarm torque control is possible, then presents algorithms to automate the task. The paper concludes with experimental results using 100 hardware robots to manipulate rectangles with large aspect ratios.

I. Introduction

Micro-robots can be manufactured in large numbers. Our vision is for large swarms of robots to be remotely guided 1) through the human body, to cure disease, heal tissue, and prevent infection and 2) ex vivo to assemble structures in parallel. 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. Future implementations require control techniques that can reliably exploit large populations despite significant under-actuation.

Previous work proves that the mean position of a swarm is controllable and that, with an obstacle, the swarm's position variance orthogonal to rectangular boundary walls is also controllable (these are σ_x^2 and σ_y^2 for a workspace with axis-aligned walls). The usefulness of these techniques was demonstrated by several automatic controllers. One controller steered a swarm of robots to push a larger block through a 2D maze [**?**]. We also showed methods to control a swarm's position covariance to enable the swarm to navigate through a workspaces with narrow corridors. However, object manipulation often also requires controlling torque on an object, for example changing the orientation of an object with a large aspect ratio to thread through narrow corridors. Torque control is also necessary for a variety of alignment tasks including retroreflectors, and targeted radiation therapy.

Accurate torque control is difficult. A swarm with n agents has $2n$ degrees of freedom. The swarm, when steered toward an object, begins interacting with the object at different times. The number of robots touching this object as a function of time is difficult to predict and often impossible to directly measure. Stochastic effects make long-term prediction challenging. Even when it is possible to predict which agents will hit the object first, as agents interact with the object, the swarm's configuration changes. The challenge is not only limited to swarm-object interaction, but also to swarm-swarm interactions when the swarm self-collides or is split into multiple components. As a result, the force the swarm will exert on the object is not easy to predict. This information is often hard to gather, because remote sensing of tiny particles is difficult. Particles may be smaller than the minimum resolution of MRI, PET, and ultrasound, but these sensing modalities can still return aggregate data. From this aggregate data, some statistics—including mean and

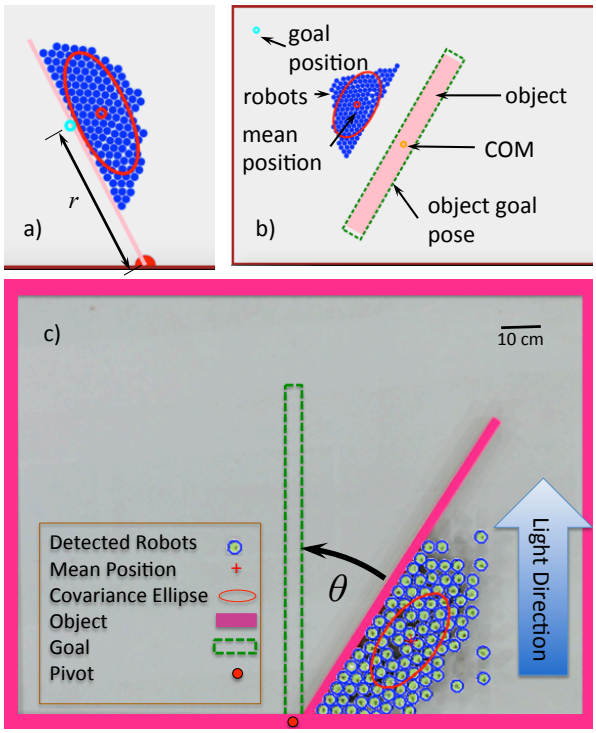

Fig. 1. Torque control of an object is essential for manipulating objects to their goal position when there are narrow passageways and for aligning sensors, emitters, or other objects. This paper provides feedback control laws to apply torques and forces using a highly under-actuated system where all robots are controlled globally by the same input. (a) Simulation of robots exerting torque on a hinged "door". (b) Simulation of swarm orientation manipulation. (c) 97 hardware robots applying torque to an object. These robots have light sensors and are programmed to move toward the brightest light in the room. This light is a shared control input that operates globally on the swarm. A full resolution video is available at: [**?**]

S. Shahrokhi and A. Becker are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204-4005 USA {sshahrokhi2, atbecker}@uh.edu

variance—are easy to obtain. Rather than design open-loop algorithms, this paper focuses on feedback control strategies using just two statistics of the swarm, the mean and variance of the swarm's position.

Fig. 1 illustrates the torque control using 97 kilobots, as well as two simulation snapshots of torque and orientation control.

II. Related Work

Unlike *caging* manipulation, where robots form a rigid arrangement around an object [**?**], [**?**], 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. [**?**], where forces must be applied along the center of mass of the moveable object.

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

While ferrous particles tend to clump in a magnetic field, the magnetotactic bacteria of [**?**], [**?**] are directed by the orientation of the magnetic field and do not suffer from magnetic aggregation.

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

III. Torque Control

The orientation of an object's major axis is important when a swarm is manipulating a non-symmetric object through narrow corridors. Orientation is controllable by applying torque to the object. To change the output torque τ in Eq. (1), we can choose the direction and magnitude of the force applied F, and the moment arm from the object's center of mass (COM) to the point of contact r.

$$\tau = F \times r \tag{1}$$

The swarm version of (1) is the summation of the forces contributed by individual robots.

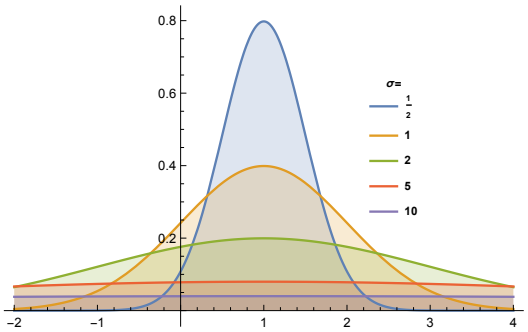

Fig. 2. PDF of Normal distributions with the same μ and different σs.

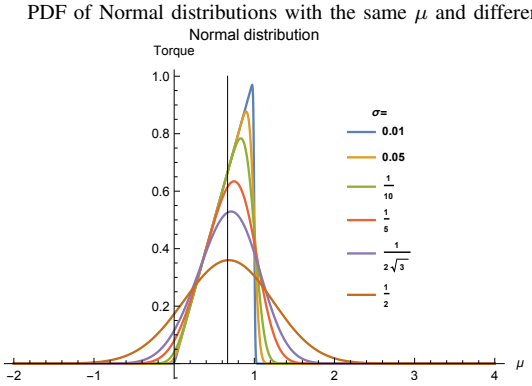

Fig. 3. Torque as a function of the mean position, μ which is the pushing location when the swarm is normally distributed with different standard deviations.

$$\tau_{total} = \sum_{i=1}^{n} \rho_i F_i \times (P_i - O) \tag{2}$$

$$F_{total} = \sum_{i=1}^{n} \rho_i F_i \tag{3}$$

Here F_i is the force that the ith robot applies. If all robots are identical and the control input is uniform, the force is equivalent for every robot and $F_i = F_c$. Not all robots are in contact with the object. ρ_i is an indicator variable: ρ_i is 1 if the robot is in direct contact with the object or touching a chain of robots where at least one robot is in contact with the object. Otherwise $\rho_i = 0$. The moment arm is the robot's position P_i to the object's COM $O = [O_x, O_y]^\top$. Consider a normal distribution,

$$P(x)_n = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{4}$$

$$\begin{aligned} \tau &= \int_0^1 \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} x\, dx \\ &= \frac{(-e^{-\frac{(-1+\mu)^2}{2\sigma^2}} + e^{-\frac{-\mu^2}{2\sigma^2}})\sigma}{\sqrt{2\pi}} \\ &\quad + \frac{1}{2}\mu(erf(\frac{1-\mu}{\sqrt{2}\sigma}) + erf(\frac{\mu}{\sqrt{2}\sigma})) \end{aligned} \tag{5}$$

$$\frac{d\tau}{d\mu} = -\frac{e^{-\frac{-1+\mu^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} + \frac{1}{2}(erf(\frac{1-\mu}{\sqrt{2}\sigma}) + erf(\frac{\mu}{\sqrt{2}\sigma})) \tag{6}$$

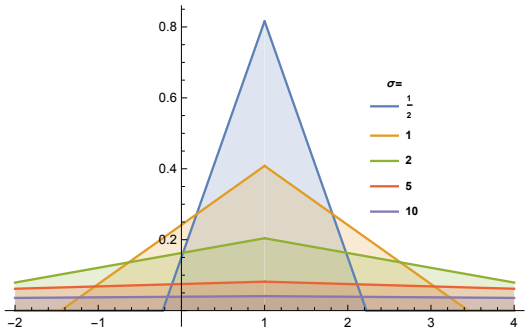

Fig. 4. PDF of Triangular distributions with the same μ and different σs.

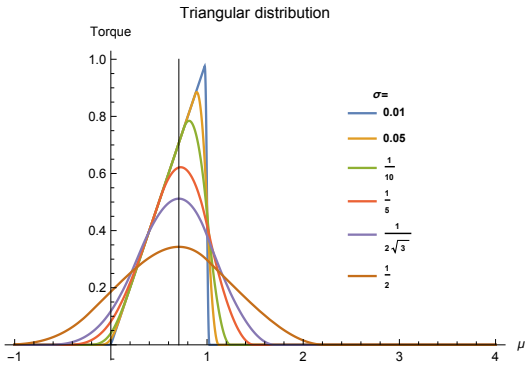

Fig. 5. Torque as a function of the mean position, μ which is the pushing location when the swarm is distributed by triangular distribution with different standard deviations.

$$P(x)_t = \begin{cases} \frac{x-\mu+\sqrt{6}\sigma}{6\sigma^2}, & \text{for } \mu - \sqrt{6}\sigma \le x \le \mu \\ \frac{-x+\mu+\sqrt{6}\sigma}{6\sigma^2}, & \text{for } \mu < x \le \mu + \sqrt{6}\sigma \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

$$\tau_t = \begin{cases} \mu, & \mu < 1 \& \sqrt{6}\mu + 6\sigma \\ \frac{2-3\mu+3\sqrt{6}\sigma}{36\sigma^2}, & (\mu > 1 \& \sqrt{6}\mu \le 6 \\ \frac{-2+3\mu-2\mu^3+3\sqrt{6}\sigma}{36\sigma^2}, & \mu < 1 \& \sqrt{6}\mu + 6\sigma \\ \frac{1}{2} - \frac{\sigma}{\sqrt{6}}, & \mu = 1 \& \sigma < \frac{1}{\sqrt{6}} \\ -\frac{2+\mu^3+3\sqrt{6}\mu^2\sigma+3\sqrt{6}\sigma(-1+2\sigma^2)-3\mu(1+6\sigma^2}{36\sigma^2} & \mu < 1 \& \sqrt{6}\mu + 6\sigma \\ \frac{2+\mu^3-3\sqrt{6}\mu^2\sigma+3\sqrt{6}\sigma(1-2\sigma^2)+3\mu(-1+6\sigma^2}{36\sigma^2} & \mu > 1 \& 0 < \sqrt{6}\mu - \\ \frac{1}{36}(18\mu - \frac{\mu^3}{\sigma^2} + \frac{3\sqrt{6}\mu^2}{\sigma} + 6\sqrt{6}\sigma) & \sqrt{6}\mu + 6\sigma < \sqrt{6} \& \sqrt{6}\mu \le 6\sigma \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$P(x)_u = \begin{cases} \frac{1}{2\sqrt{3}\sigma}, & \text{for } \mu - \sqrt{3}\sigma \le x \le \mu - \sqrt{3} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

IV. Simulation

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

$$\begin{aligned} u_x &= K_p(goal_x - \bar{x}) + K_d(0 - \bar{v}_x) \\ u_y &= K_p(goal_y - \bar{y}) + K_d(0 - \bar{v}_y) \end{aligned} \tag{10}$$

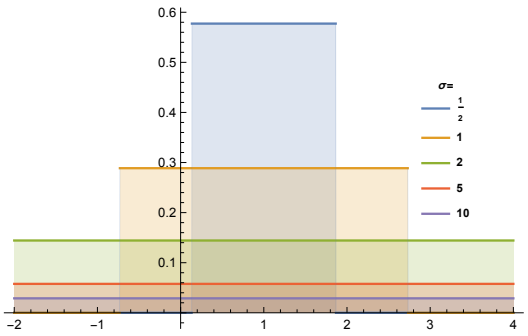

Fig. 6. The Uniform distribution with different standard deviations and the same mean.

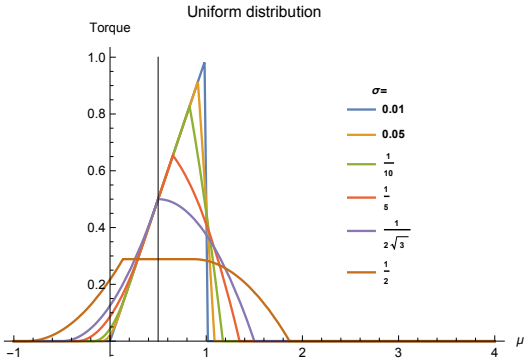

Fig. 7. Torque as a function of the mean position, μ which is the pushing location when the swarm is uniformly distributed with different standard deviations.

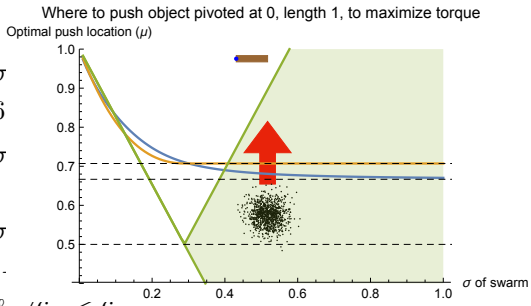

Fig. 8. Comparison for three different distributions for the best location to push the object.

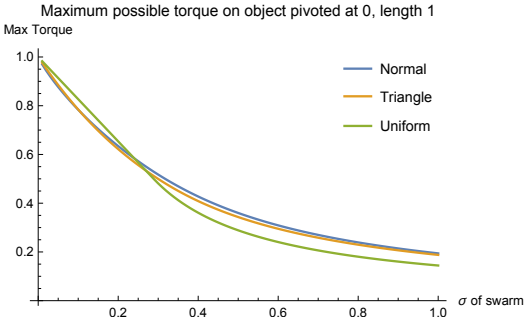

Fig. 9. Comparison for three different distributions for the maximum torque possible.

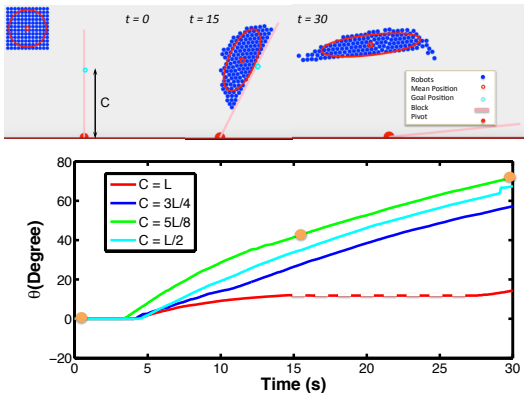

Fig. 10. Simulation results from a swarm applying force to a hinged door. The swarm mean is steered toward a point C units along the object from the pivot point. The red dashed line indicates the times that the swarm was in variance control mode. Simulation used 144 robots of diameter 0.2 m with a standard deviation of less than 1.5 m and an object length of 6 m.

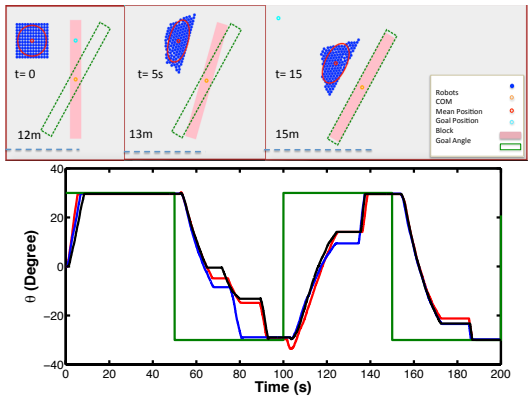

Fig. 11. Plot demonstrating orientation control of a rectangular object. The green line is the goal orientation. Other lines show different random starting position of the swarm. When the plot traces are constant the swarm is no longer pushing the object and instead is being regathered in a corner of the workspace until the variance is below a desired threshold.

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

a) Pure torque control: An object with a pivot point can rotate, but not translate. A door is an example. If there was only one robot touching the object, the robot should push at the point which maximizes the moment arm, at the extreme end of the object furthest from the pivot point. The optimal pushing location provides the maximum force, because it maximizes r in (1). However, given a swarm of robots, maximizing r is no longer the optimal solution. If the mean of the swarm hits the object at the extreme edge, half of the robots will miss the object and the swarm will be split. Because few robots remain, the force is significantly decreased and torque is not maximized. In our simulation, the swarm applies torque until the swarm's mean position is beyond the object. Then the swarm will regather in a corner before returning to torque control, a time-consuming task. The key parameter of interest for a hinged door of length L is C, the position along the door where the mean of the swarm will push. The goal position in (11) is set to:

$$\begin{aligned} goal_x &= O_x + C\cos(O_\theta) \\ goal_y &= O_y + C\sin(O_\theta), \end{aligned} \tag{11}$$

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

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

$$\begin{aligned} goal_x &= O_x + K_{orient}(O_\theta - goal_\theta)\cos(O_\theta) \\ goal_y &= O_y + K_{orient}(O_\theta - goal_\theta)\sin(O_\theta) \end{aligned} \tag{12}$$

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

Fig. 9 illustrates this controller with different starting positions. When the plot traces are constant the swarm is no longer pushing the object and instead is being regathered in a corner of the workspace. Code is available at [**?**].

c) Straight translation while regulating object orientation: When the total force is applied perpendicular to the object and in line with the center of mass, according to Eq. (1) there will be no torque. The following goal position for the mean position of the swarm regulates the object's orientation using Δ_θ for proportional feedback to determine where to apply force. $\Delta_\theta = goal_\theta - O_\theta$ is the difference between the goal angle and the current object angle. K_τ is a constant and is tuned manually to 10. (O_x, O_y) is the position of the object's COM.

$$\begin{aligned} goal_x &= O_x \\ goal_y &= K_\tau \Delta_\theta + O_y \end{aligned} \tag{13}$$

Fig. 10 shows how Δ_θ converges to zero with different initial configurations of the swarm. When the swarm is above or below the object it applies a torque to the object. Code is available at [**?**].

d) Line following with perpendicular orientation: This task designs a control law that, given an arbitrary line, will push an object's COM onto that line and regulate the object's orientation to be perpendicular to that line. Any line can be parameterized in the form $ax + by + c = 0$, which gives the locus of (x, y) points along the line. The point on this line nearest to the swarm COM (O_x, O_y) is P:

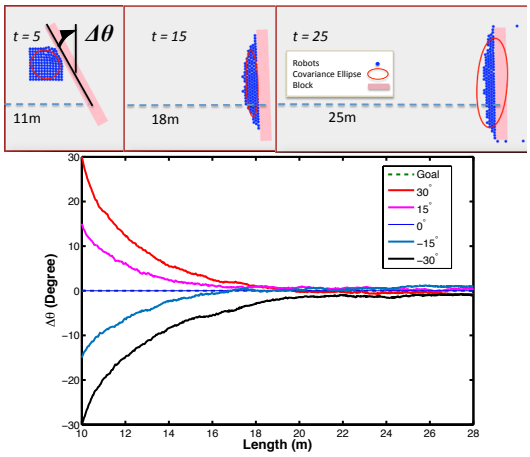

Fig. 12. In this task, the swarm pushed the object in the $+x$ direction while trying to regulate the orientation to $goal_\theta = 0°$. The swarm can push the object without changing its orientation only if it pushes along a line intersecting the COM of the object. A feedback control law regulates the

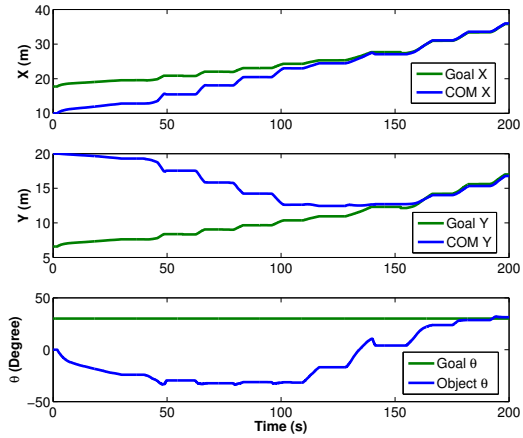

Fig. 13. Following an arbitrary line with perpendicular orientation. This control law centers the object to the line, while it regulates its orientation.

$$P_x = \frac{b(bO_x - aO_y) - ac}{a^2 + b^2}, \qquad (14)$$
$$P_y = \frac{a(-bO_x + aO_y) - bc}{a^2 + b^2}$$

The following control law regulates the goal position for the swarm's mean to push the object COM onto the line and regulates the object's angle.

$$goal_x = O_x + K_p(O_x - P_x) + K_\tau \frac{O_x - P_x}{||O_x - P_x||}\Delta_\theta$$
$$goal_y = O_y + K_p(O_y - P_y) + K_\tau \frac{O_y - P_y}{||O_y - P_y||}\Delta_\theta \qquad (15)$$

Fig. 11 shows the position and orientation over time while line following with perpendicular orientation. The goal position for the swarm is based on the nearest point from COM to the line. Code is available at [**?**].

e) Object pose control: This section presents two algorithms designed to control the *pose*, the position and orientation, of an object. Each works by first controlling the orientation and then regulating that orientation while translating the object.

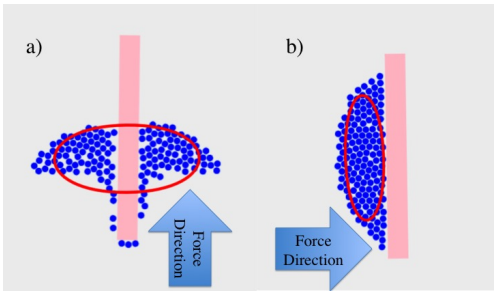

Fig. 14. a.) Pushing an object perpendicular to its minor axis. The swarm spreads around the object. b.) Applying force perpendicular to the object's long axis reduces the probability of splitting the swarm.

The naïve approach is represented in Alg. 1. It defines the coordinate frame such that $[goal_x, goal_y, goal_\theta] = [0, 0, 0]$. Then it cycles between regulating the angular error below some threshold T_θ, pushing the object along the major axis until position error perpendicular to the major axis is less than T_x, then pushing perpendicular to the minor axis until error perpendicular to the minor axis is less than T_y. This naïve approach works poorly on objects with large aspect ratios because the swarm flows past the object, as shown in Fig. 12. Instead Alg. 2 seeks to always push the object perpendicular to the major axis, preventing the swarm from flowing around.

Algorithm 1 PerpendicularPushesPoseControl

Require: $goal_x, goal_y, goal_\theta, O_x, O_y, O_\theta$.
1: Define coordinate frame such that $[goal_x, goal_y, goal_\theta]^\top = [0, 0, 0]^\top$
2: **while** $|O_x| > T_x \vee |O_y| > T_y \vee |O_\theta| > T_\theta$ **do**
3: **while** $|O_\theta| > T_\theta$ **do**
4: Orientation control §IV.a
5: **end while**
6: **while** $|O_x| > T_x$ **do**
7: Straight translation §IV.c along line $y = 0$
8: **end while**
9: **while** $|O_y| > T_y$ **do**
10: Straight translation §IV.c along line $x = 0$
11: **end while**
12: **end while**

Given a goal pose for the object, the algorithm constructs the line perpendicular to the long axis that intersects the goal pose. Alg. 2 first moves the object to this line using the control law from §IV.d, then pushes the object to the goal pose using §IV.d. A hysteresis-based variance control is used. Whenever the swarm variance is bigger than a maximum variance threshold, the swarm is steered to a corner to regather itself. This causes delays, but it usually prevents robots from flowing around the object. However, there is still some probability that a part of the swarm appears in front of the object, as shown in Fig. 13 where at $t \approx 170$ the orientation of the object is affected when the swarm is trying to regather because several robots were in front of the object.

Algorithm 2 PoseControl

Require: $goal_x, goal_y, goal_\theta, O_x, O_y, O_\theta, C < 1$.

$a = \cos(goal_\theta)$ ▷ Line Equation
$b = \sin(goal_\theta)$
$c = -a \cdot goal_x - b \cdot goal_y$
repeat
  $P_x = \frac{b(bO_x - aO_y) - a \cdot c}{a^2 + b^2}$ ▷ Nearest point on line
  $P_y = \frac{a(-bO_x + aO_y) - b \cdot c}{a^2 + b^2}$
  $\theta_t = goal_\theta + \pi/2$
  $a_t = \cos(\theta_t)$ ▷ Line Equation
  $b_t = \sin(\theta_t)$
  $c_t = -a_t \cdot P_x - b_t \cdot P_y$
  Line following with perpendicular orientation on line (a_t, b_t, c_t) §IV.d
  $d = \sqrt{(P_x - O_x)^2 + (P_y - O_y)^2}$
until $d < C$ ▷ Reaching near the line
Line following with perpendicular orientation on line (a, b, c) §IV.d

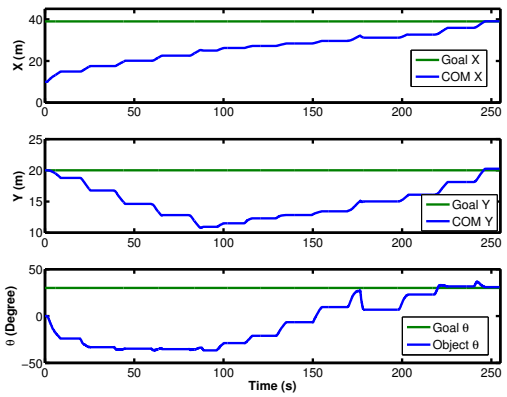

Fig. 15. Pose control using Alg. 2. The objects is delivered to a goal position and orientation. The parts that the position including orientation is not changing, the swarm is in variance control mode to avoid splitting as much as it is possible.

Fig. 14 shows screenshots of the simulation. Simulation code that runs natively in any modern web browser is available at [**?**].

V. Experiment

Our experiments are on centimeter-scale hardware systems called *kilobots*. These allow us to emulate a variety of dynamics, while enabling a high degree of control over robot function, the environment, and data collection. The kilobot, from [**?**], [**?**], 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 [**?**]. 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. 15, we used n=97 kilobots, a glass-covered 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

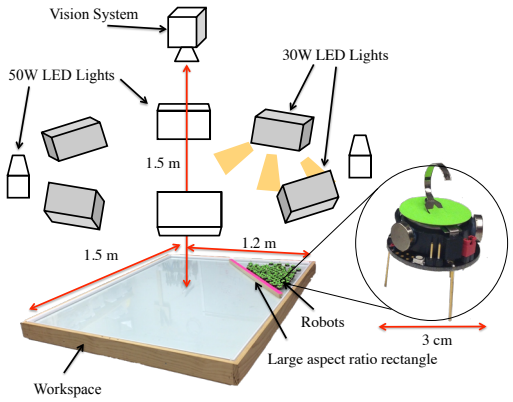

Fig. 17. Hardware platform: table with 1.5×1.2 m workspace, surrounded by eight remotely triggered 30W LED floodlights, with an overhead machine vision system.

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.

The experiments from Section IV.a were manually demonstrated using this physical swarm. Fig. 16 illustrates an experiment showing pure torque control with a swarm of robots. In this figure a large-aspect-ratio rectangle (91×2 cm, colored pink in the image) was hinged to one side of the table. Like a door, this object could only be moved around this pivot. Two trials were performed. In each trial the swarm was initialized in the lower right side of the table, and then commanded to push the object with the mean position of the swarm directed toward a point distance C from the pivot point. Data was recorded for 150 seconds. In the first trial, $C = L$, so the robots were commanded to push the door at the extreme edge of the door from the pivot. In the second trial $C = 1/2L$, and so the swarm pushed the object in the center of the rectangle. As discussed in Section IV, the robots spread when commanded to push the object at the extreme end, and half of the robots flowed past the end of the rectangle without engaging the rectangle. This illustrates a key difference between robotic swarms and a single pusher robot. The swarm exerts the most torque when (2) is maximized. (2) is maximized when the majority of the swarm engages the object. For this reason in Fig. 16, the trial in the second row of screenshots moves the door further than the trial in the first row.

VI. Conclusion and Future Work

This paper presented techniques for controlling the orientation of an object by manipulating it using a swarm of simple robots with global inputs. The paper provided algorithms for precise orientation control, as well as demonstrations of orientation control.

Future efforts should be directed toward optimizing torque control, examining the effects of Brownian noise, applying the techniques to hardware robots, pose control for multiple part assembly, and manipulation in a crowded workspace.

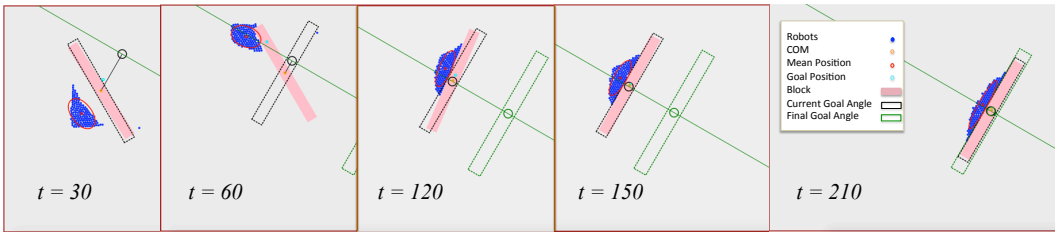

Fig. 16. Different stages for position control of a block while controlling its orientation. The first task (0–90 s) is to push the object COM to a line perpendicular to the goal. The second task (90–210 s) is to push the object along this perpendicular line while regulating the orientation.

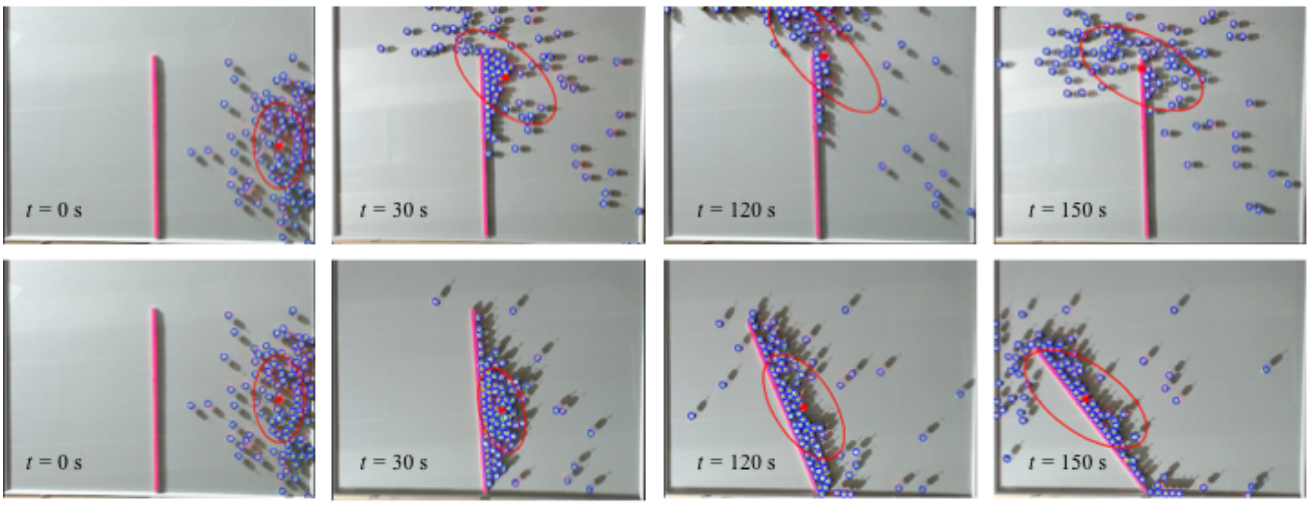

Fig. 18. Snapshots showing the effect of pushing a pivoted rectangular object at different distances from the pivot point. 97 robots were programmed to move toward the brightest light in the room, and controlled by choosing which of 8 lights were on at any given instant. The top row of snapshots illustrate the swarm pushing at the end of the object. In this case, the swarm flows past the object and the force decreases. The bottom row illustrates that when the swarm pushes at the middle of the object the force provided by the swarm remains constant. In this case the swarm does not flow past the object. See video attachment for recordings of these experiments at [**?**].

The control laws in this paper used only the mean and variance of the swarm. The control techniques may be optimized using high-order moments, or by stochastic modeling of the collisions between swarm members and the object.

VII. Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1553063. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.