Controlling Torque of an Object Using a Swarm With Global Inputs

Shiva Shahrokhi and Aaron T. Becker

Abstract— **Micro and nano robots are suited for targeted drug delivery and micro scale manufacturing because they are small enough to go toward the passageways of the body. But due to their small size robots cannot contain onboard processing for autonomy nor onboard power. Instead they are controlled by external signal such as a magnetic field. However, because each robot can only provide a small amount of force or transport a small amount of material, we need large swarms of robots, all controlled by the same external field. This work presents controllers and algorithms for steering such an under-actuated swarm. In our previous work, we showed that mean and variance of the swarm is controllable. We also showed that with controlling mean and variance we are able to manipulate objects, and for a challenging environments showed that covariance of the swarm is controllable. But one of the remaining challenges was controlling torque of the object. In some environments, we have narrow ways that we may need to not only control covariance of the swarm, but also control torque of the object to be able to pass those ways. This work first proves that torque control is possible, then presents algorithms to do the task. We conclude by showing the experiments with 100 real robots when they are manipulating a rectangle and discuss the future challenges.**

I. Introduction

Micro- and nano-robots can be manufactured in large numbers. Our vision is for large swarms of robots 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. The biggest barrier to this vision is a lack of control techniques that can reliably exploit large populations despite incredible under-actuation.

In previous work, we proved 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 (σ_x and σ_y 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 swarm's position covariance to be able to navigate the swarm through the workspaces with narrow corridors, but there is still another challenge remained. When the task is object manipulating, we need to control torque of the object also to successfully pass all the ways. This paper first discusses the ways that we can control torque of an object in Section III. Then it introduces algorithms for torque control in Section IV. We show that algorithms work in Section **??** on 100 kilobots.

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`

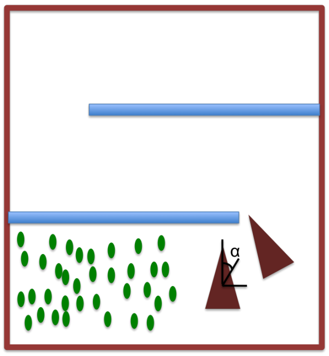

Fig. 1. How can we control torque of the object with a swarm of robots and a shared input?

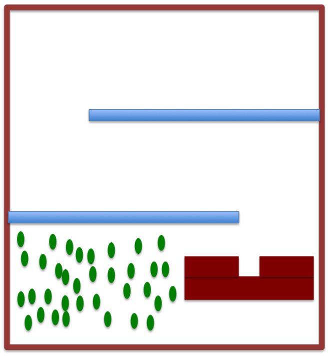

Fig. 2. How can we control torque of the object with a swarm of robots and a shared input?

II. Related Work

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. control-Lyapunov functions gradient based decentralized 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 [**?**].

A. Using shear forces to shape a set of particles

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 shear forces are common in fluid flow along boundaries, as described in introductory fluid dynamics textbooks [**?**]. 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 have used these forces to disperse a swarm's spatial position for accomplishing coverage tasks [**?**].

More research has focused on generating artificial force-fields. Applications have included techniques to design shear forces to a single object for sensorless manipulation [**?**]. Vose et al. demonstrated a collection of 2D force fields generated by 6DOF vibration inputs to a rigid plate [**?**], [**?**]. This collection of force fields, including shear forces, could be used as a set of primitives for motion control for steering the formation of multiple objects.

III. Theory

A. Controlling Torque

One of the features of an object is its orientation. When a swarm is manipulating an non-symmetric object, in narrow corridors orientation matters. We can control orientation by controlling torque of the object. Equation 1 shows the parameters that we should control.

$$\tau = F \times d \qquad (1)$$

where d

IV. Simulation

Algorithms **??**, **??**, **??**, were implemented in Mathematica using point robots (radius = 0). All code is available at a public github repository [**?**]. Figs 3 and 4 show the examples of the implementation of our algorithm. In both of these figures we have denoted the starting points and the destinations by small circles. However, destination points are surrounded by larger circles so as to be distinct from starting points.

In each of these figures we have five snapshots of the running of our algorithm taken every quarter second. For the sake of brevity we have replaced straight moves (e.g. upward, downward, etc) with oblique moves that shows a combination of two moves simultaneously (e.g. left and down together).

As we can see, in Fig. 3 Δr_x is adjusted to Δe_x in the second snapshot, i.e., at $t = t_1$ where $t_1 < 0.25$. The rest of the steps in this figure is dedicated to adjusting the Δr_y to Δe_y. As it is clear from Fig. 3, Δr_y is also adjusted at $t = t_2$ where $0.75 < t_2 < 1$. Finally, once Δr_x and Δr_y are adjusted, the algorithm gives a global input both of the robots so as to move them toward their corresponding destinations. This is happening in the time interval of $(t_2, 1]$.

Similarly, in Fig. 4 we can see that the Δr_x is adjusted in the third snapshot, i.e., at $t = t_3$ where $0.25 < t_3 < 0.5$ and Δr_y is adjusted in the last snapshot at $t = t_4$ where $0.75 < t_4 < 1$. The final positioning steps are happening in the time interval of $(t_4, 1]$.

As we pointed out earlier, adjusting each of Δr_x and Δr_y needs two iterations in the worst case. In other words, both of the Alg. **??** and Alg. **??** are executed two times in the worst case in positioning process of the robots. It is easy to see that we need two iterations of Alg. **??** only if $|\Delta e_x - \Delta s_x| > L$. Similarly we need two iterations of Alg. **??** only if $|\Delta e_y - \Delta s_y| > L$.

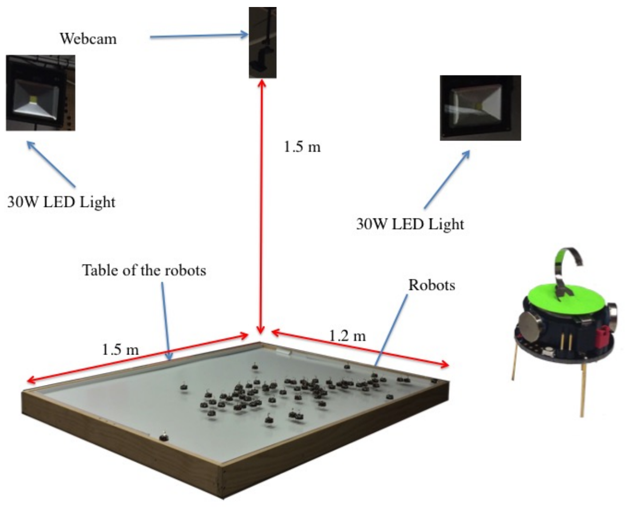

Fig. 5. Hardware platform: table with 1.5×1.2 m workspace, surrounded by four remotely triggered 30W LED floodlights, with an overhead machine vision system.

V. Experiment

A. Hardware system

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 [**?**], [**?**] is a low-cost robot designed for testing collective algorithms with large numbers of robots. It is available commercially or as an open source platform [**?**]. Each robot is approximately 3 cm in diameter, 3 cm tall, and uses two vibration motors to move on a flat surface at speeds up 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. 5, we used n=64 kilobots, a 1.5 m×1.2 m whiteboard as the workspace, and four 30W LED floodlights arranged 1.5 m above the plane of the table at the $\{N, E, S, W\}$ vertices of a 6 m square centered on the workspace. The lights were controlled using an Arduino Uno board connected to an 8 relay shield board. At top of the table, an overhead machine vision system was added to track the position of the swarm. Laser-cut patterns for our neon green fiducial markers and our MATLAB tracking code are available at our github repository [**?**].

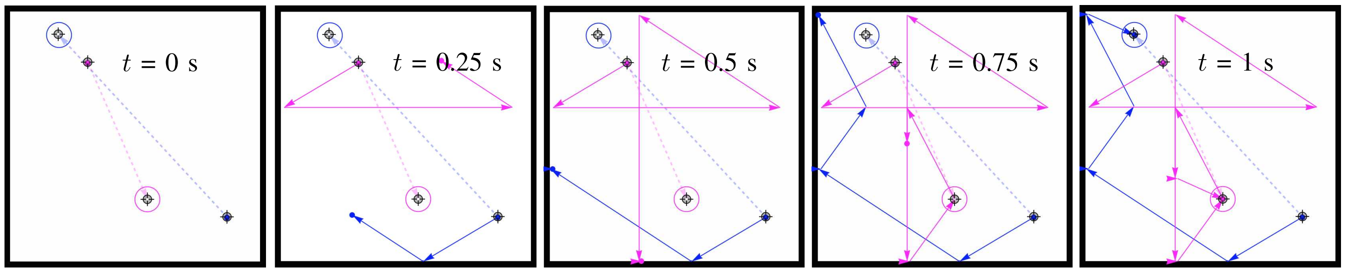

Fig. 3. Frames from an implementation of Alg. **??**: two robot positioning using walls with infinite friction. Robot initial positions are shown by a crosshair, and final positions by a circled crosshair. Dashed lines show the shortest route if robots could be controlled independently. The path given by Alg. **??** is shown with solid arrows.

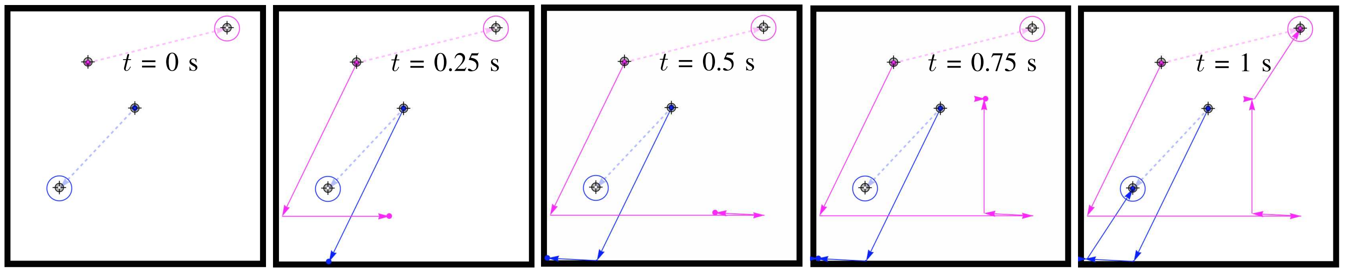

Fig. 4. Two robot positioning: switching positions using walls with infinite friction. Code available at [**?**].

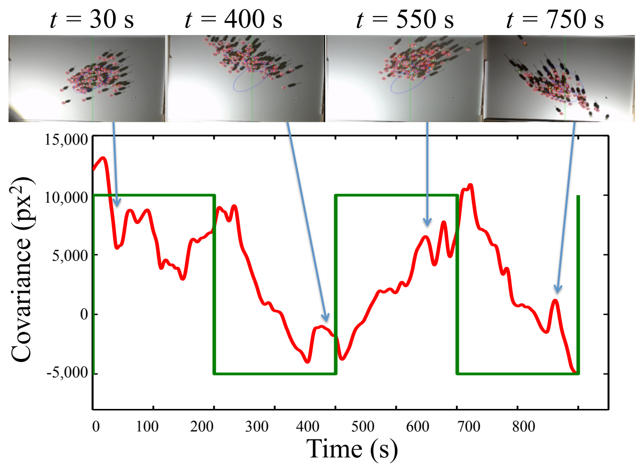

Fig. 7. Hardware demonstration steering 64 kilobot robots to desired covariance. Frames above the plot show output from machine vision system and an overlaid covariance ellipse.

The walls of the hardware platform have almost infinite friction, due to the three legged design of the kilobots. When a kilobot is steered into the wall, they pin themselves to the wall until the light changes direction and they begin turning in the other direction. This wall friction is sufficient to enable independent position control of two kilobots, as shown in Fig. 6.

To demonstrate covariance control $n = 64$ robots were placed on the workspace and manually steered with a single light source, using friction with the boundary walls to vary the covariance from -5000 to 10,000. The resulting covariance is plotted in Fig. 7, along with snapshots of the swarm.

VI. 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. 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 [**?**], [**?**]. 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, friction could be modified by outfitting the kilobots with a round skirt to avoid the almost infinite friction due to the triangular leg arrangement. The wall friction can be varied by laser-cutting boundary walls with different of profiles.

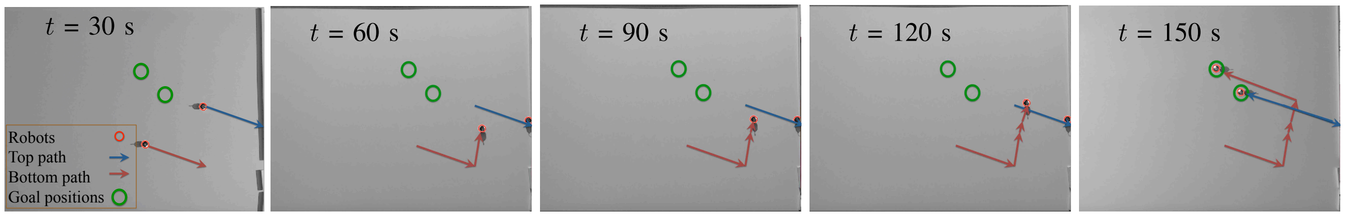

Fig. 6. Two robot positioning using the hardware setup and two kilobot robots. The walls have nearly infinite friction, as illustrated by this fig, the robot with the blue path that is stopped by the wall until the light changes orientation, while the orange robot in free-space is unhindered.