How to Control a Swarm's Shape

Shiva Shahrokhi and Aaron T. Becker

***Abstract*— Micro- and nanorobots can be built in large numbers, but generating independent control inputs for each robot is prohibitively difficult. Instead, micro- and nanorobots are often controlled by a global field. In our previous work, inspired by human operators, we investigated controllers that use only the mean and variance of a robot swarm. We proved that the mean position is controllable, then provided conditions under which variance is controllable. We next derived automatic controllers for these and a hybrid, hysteresis-based switching control to regulate the first two moments of the robot distribution. In this work, we considered is if we had some maps like Fig. 1, how we can control covariance of the robots. We use friction to break the symmetry of the robots and make the covariance desired.**

I. Introduction

A. Using global control of a swarm

Micro- and nano robots are often controlled by global-broadcast signal. Some examples are Mag-Mite microrobots with different resonant frequencies controlled by a global magnetic field [1], using a common input to control large populations of simple robots with at least one obstacle [2], light-driven nanocars, which synthetic molecules actuated by a specific wavelength of light [3], scratch-drive microrobots, actuated and controlled by a DC voltage signal from a substrate[4], [5], When human controls large number of robots, lots of limitations appears, like the number of robots that a human can control simultaneously. To remove this limitation, using global input to all the robots would be sufficient.

B. Human user studies

Our previous work used an online game named "SwarmControl.net" with human user interfaces for controlling large number of simulated robots [6]. It is done this way because previous studies on real robots were not scalable because of the cost of the robots. Humans could complete the tasks designed in this platform like pushing a block from somewhere to a goal position which is shown in Fig. 2. This platform enabled us to have large number of human players all over the world to play.

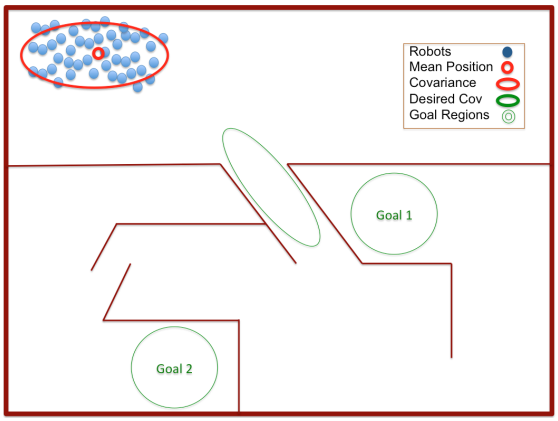
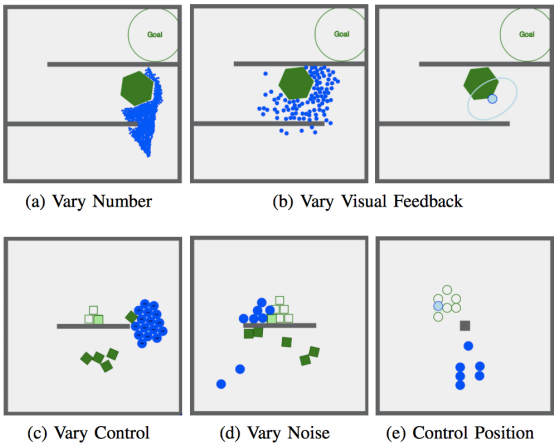


Fig. 1. We want to be able to go through the map which has corridors with different angles to reach the goals. Green ellipse is our first target covariance ellipse. The red covariance ellipse is our current covariance ellipse. We use friction to make it.

C. Automatic controllers

In our previous work for block pushing task, one surprising result was that humans with less information played better. Inspired by this work, we proved that we can control mean and variance of the swarm and with just knowing mean and variance, we could have auto controllers which could complete the block pushing task but with some limitations. Our algorithm could not handle difficult maps, because the swarm might fell apart and it could get stuck in some corners. One problem is that if there is just a narrow corridor and we



(a) Vary Number (b) Vary Visual Feedback

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

Fig. 2. Screenshots from the five online experiments controlling multi-robot systems with limited, global control[6].

should pass that corridor, we can not control the shape of a swarm to enable them to pass through that. Therefore, it is not enough to control mean and variance of the swarm, and if we could control covariance of the swarm in respect to x and y axis, controlling the shape of the swarm in any angle would be possible. So our next goal is controlling covariance of the swarm in a 2D plane.

II. Related Work

A. How to control mean

In our previous work, we proved that we can easily control mean position of the swarm. We control mean position with a PD controller that uses the mean position and mean velocity. Our control input is the global force applied to each robot:

$$u_x = K_p(x_{goal} - \bar{x}) + K_d(0 - \bar{v}_x)$$
$$u_y = K_p(y_{goal} - \bar{y}) + K_d(0 - \bar{v}_y) \quad (1)$$

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

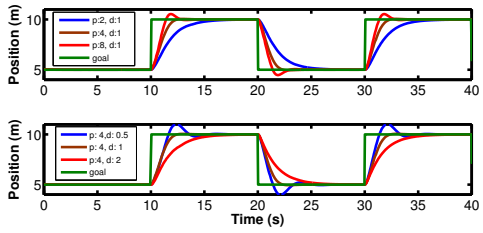


Fig. 3. Tuning proportional (K_p, top) and derivative (K_d, bottom) gain values in (1) improves performance. These plots show convergence with 100 robots.

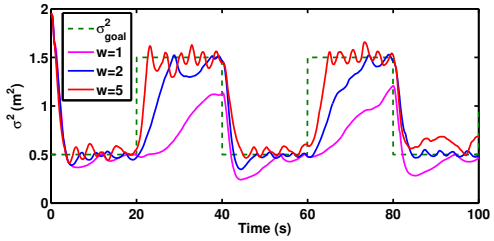


Fig. 4. Increased noise results in more responsive variance control because stronger Brownian noise causes a faster increase of variance.

B. How to control variance

For variance control we move away from the wall and wait to increase variance because Brownian noise naturally disperses the swarm in such a way that the variance increases linearly. If faster dispersion is needed, the swarm can be pushed through obstacles such as a diffraction grating or Pachinko board.

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

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

In a slight abuse of notation we call the gain scaling the variance error K_i because the variance integrates over time. Eq. 2 assumes the nearest wall is to the left of the robot at $x = 0$, and chooses a reference goal position that in steady-state would

have the correct variance:

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

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

III. Theory

A. How to control covariance

For controlling covariance, we can use friction. When the swarm move across the wall, friction force will apply to the robots that are going across the wall, and so they will have less velocity than the free ones. With this difference we have broken the symmetry and can control the covariance. As we can see from Eq. 4, total force for going forward will change.

$$\begin{aligned} F_f &= \mu_f N \\ N &= F cos\theta \\ F_{forward} &= F sin\theta - F_f \end{aligned} \tag{4}$$

where F is our control input. As shown in Fig. 5, the total force for going forward is less than a free robot. So we can reduce speed of some robots with friction.

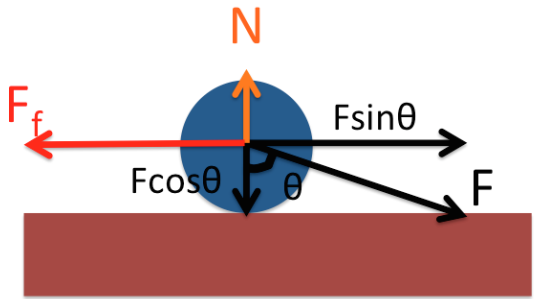


Fig. 5. When the robot is near the wall, the overall force for going forward is less than a free robot because of the friction.

IV. Simulation

Our simulations use a Javascript port of Box2D, a popular 2D physics engine with support for rigid-body dynamics and fixed-timestep simulation.

A. Controlling Covariance

In Algorithm 1 we put the robots near the wall in order to have some of them sliding wall, and some of them free. We keep y variance less than the optimal variance and while keeping it, we slide robots from one end point of the wall to the middle of the wall. By this moving, we will grow the angle of the covariance ellipse. At the moment that we reach the desired angle with some errors, we go to the target point.

Algorithm 1 Covariance Control

Require: Knowledge of swarm mean $[\bar{x}, \bar{y}]$, the locations of the rectangular boundary $\{x_{min}, x_{max}, y_{min}, y_{max}\}$, and the target mean position $[x_{target}, y_{target}]$, error threshold ϵ, radius of the robots r and number of robots n.

1: Compute the min σ_x^2 and σ_y^2 by $\sigma^2_{optimal}(n, r) = nr^2\frac{\sqrt{3}}{2\pi}$
2: $y_{goal} = y_{min} + \sigma_{optimal}$ **and** $x_{goal} = x_{max}/2$
3: **loop**
4: Compute σ_x^2, σ_y^2 **and** cov_{xy}
5: $angle_{cov} = tan^{-1}\{2 * cov_{xy}/\sigma_x^2 - \sigma_y^2\}$
6: **if** $\sigma_y^2 >= \sigma^2_{max}$ **then**
7: $y_{goal} \leftarrow y_{min}$
8: **else if** $counter\%2 = 1$ **and** $\sigma_y^2 < \sigma^2_{min}$
9: $x_{goal} \leftarrow x_{min}$
10: **if** $counter\%2 = 0$ **and** $\sigma_y^2 < \sigma^2_{min}$
11: $x_{goal} \leftarrow x_{max}/2$
12: **end if**
13: **if** $angle_{cov} < abs(angle_{desired}) + \epsilon$ **and** $\sigma_y^2 < \sigma^2_{min}$ **then**
14: $y_{goal} = y_{target}$
15: $x_{goal} = x_{target}$
16: **end if**
17: **end loop**

V. Conclusion and Future Work

We have implemented our preliminary controller but it does not work properly all the times because when it tries to come to the target goal, covariance may change some times. Future work should focus on solving this problem. And also implement these

algorithms on real robots to determine its robustness in real environment.

REFERENCES

[1] D. Frutiger, B. Kratochvil, K. Vollmers, and B. J. Nelson, "Magmites - wireless resonant magnetic microrobots," in *IEEE Int. Conf. Rob. Aut.*, Pasadena, CA, May 2008.

[2] 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.

[3] P.-T. Chiang, J. Mielke, J. Godoy, J. M. Guerrero, L. B. Alemany, C. J. Villagómez, A. Saywell, L. Grill, and J. M. Tour, "Toward a light-driven motorized nanocar: Synthesis and initial imaging of single molecules," *ACS Nano*, vol. 6, no. 1, pp. 592–597, Nov. 2012.

[4] B. Donald, C. Levey, C. McGray, I. Paprotny, and D. Rus, "An untethered, electrostatic, globally controllable MEMS micro-robot," *J. of MEMS*, vol. 15, no. 1, pp. 1–15, Feb. 2006.

[5] B. Donald, C. Levey, and I. Paprotny, "Planar microassembly by parallel actuation of MEMS microrobots," *J. of MEMS*, vol. 17, no. 4, pp. 789–808, Aug. 2008.

[6] A. Becker, C. Ertel, and J. McLurkin, "Crowdsourcing swarm manipulation experiments: A massive online user study with large swarms of simple robots," in *IEEE International Conference on Robotics and Automation*, 2014.