Brownian Cargo Capture in Mazes via Intelligent Colloidal Microrobot Swarms

Kun Xu, Yuguang Yang,* and Bo Li*

Intelligent microrobot systems at the microscopic scale provide enormous opportunities for emerging biomedical and environmental applications. Herein, a multiagent stochastic feedback control framework to control colloidal microrobot swarms for capturing Brownian cargo particles in complex environments such as mazes is proposed. The decision-making module in the control framework consists of the adaptive generation of target sites surrounding the cargo, optimal target assignment, and approximate motion planning. The stochastic trajectories of robot swarms are efficiently navigated toward their exclusively assigned target around the cargo particle and enable the cargo to be captured. The capture strategy realized by the control framework is robust, adaptive, and flexible in that it accommodates diverse local geometries in the vicinity of a cargo, swarm, and maze sizes and is able to spontaneously split the workforce to catch multiple Brownian cargo particles via multitasking. The present intelligent robot swarm enabled by the multiagent control offers a path to realize complex functions at the microscopic scale in a resilient and flexible manner.

1. Introduction

Developing intelligent robotic systems at the microscopic scale is one essential goal at the interfaces of artificial intelligence, robots, and nanotechnology. Over the past decade, active colloidal particles, which can extract energy from their surroundings to propel themselves, have drawn considerable attention because of their fascinating underlying nonequilibrium physics, as well as their technological potential as intelligent systems. These colloidal robots are being prototyped for a wide range of emerging applications, including cargo transport and gene delivery, environmental remediation, assembly assistance, and others.

K. Xu, Y. Yang, B. Li
Institute of Biomechanics and Medical Engineering
Applied Mechanics Laboratory
Department of Engineering Mechanics
Tsinghua University
Beijing 100084, China
E-mail: yyang60@jhu.edu; libome@tsinghua.edu.cn

© 2021 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution, and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202100115

Despite the various successful demonstrations from exploiting a single colloidal robot, there has been a considerable hurdle on developing a macroscale robot system that is capable of accomplishing complex tasks like its modern macroscale robot counterpart. By counting on a sophisticated integration of energy supply, sensing, decision-making, and actuators with multiple degrees of freedom, modern macroscale robots or intelligent machines are able to singlehandedly achieve complex functions, including automated fabrication, driverless package delivery, etc.; yet, the integration of those elements into a micrometer-sized colloidal robot remains a tremendous challenge. In addition, Brownian motion and other external disturbance (e.g., sensing error) can produce significant adversarial impact on the locomotion and controllability of robots operating at the microscopic scale, thus making single-agent microrobot systems rather fragile.

An appealing route to overcome the incompetence and fragility of single-agent microrobot systems is to develop collaborative multiagent microrobot systems, which can accomplish more sophisticated tasks beyond individuals in a more robust and resilient fashion. Successful examples of multiagent systems can be found in biology, where ant swarms conduct efficient food source search and transport in human society, where individuals form companies and groups to achieve bigger economic, societal, and political goals, and in macrorobotics, where robot swarms are engaged in distributed search and rescue on a large landscape. Additional robustness of multiagent systems comes from the fact that the system can typically still function if only a handful of agents fail. There have been two major paradigms to realize a multiagent microrobot system, or a colloidal robot swarm, at the microscopic scale. One paradigm involves harnessing a global external control and carefully designing local interactions between individuals to achieve desired configurations. Another paradigm involves implementing stochastic multiagent control and a local external actuation mechanism to manipulate a group of individually addressable robots. Given the limited design space for local interactions, the first paradigm in general comes with the fundamental limit on the level of emerging complexity. The latter paradigm provides more degrees of freedom to achieve broader design objectives yet presents challenges for developing multiagent control strategies that are efficient, robust, and scalable. Following the latter paradigm, we have developed an intelligent colloidal microrobot system with...
centralized controlled robots (> 30) to collaborate on microscopic cargo capture and transport in free space.\textsuperscript{[28]} Here, we report an algorithmic control framework that can orchestrate a colloidal microrobot swarm as an intelligent multiagent system, which can collaboratively capture one or multiple Brownian cargo particles within an environment with obstacles and dead-ends, specifically within a maze. Our control method strategically steers each robot to conduct different navigation subtasks that eventually contribute to the capture of cargo particles. The intelligent multiagent system enabled by our algorithm is resilient, the robot swarm can function as a whole, withstanding disturbances in the environment, and flexible, the swarm can accommodate a broad range of settings, including varying maze geometries, swarm sizes, or the number of cargo particles. Ultimately, our study is a first step toward achieving sophisticated, intelligent functions in a complex, adversarial environment at the microscopic scale via controlling a colloidal microrobot system.

2. Results and Discussion

2.1. Multiagent Control Framework

Our intelligent robot swarm consists of self-propelled colloidal active particles that are capable of converting external energy into directed propulsive movement. Their dominant dynamics with control over propulsion strength but not propulsive direction is given in Equation (3) (see Experimental Section), which models the contributions of Brownian rotation, Brownian translation, and forces arising from robot–robot, robot–cargo, and robot–obstacle interactions.

Controlling the robot swarm to capture Brownian cargo particles is achieved by a feedback control system consisting of the following elements (Figure 1B): a) sensing the system state, including swarm robots’ positions and orientations and cargo position; b) generating target sites surrounding the cargo;
c) assigning each robot to track a unique target; d) calculating the optimal self-propulsion speeds to steer each robot toward their assigned targets; and e) actuating each robot’s propulsion strength according to the calculated speed and returning to (a). The experimental implementation can be realized via particle tracking in the sensing step and light activation on a light array (e.g., a liquid crystal display screen surface\(^{29}\)) in the actuation step.

Our main focus here is the decision-making component, that is, elements (b), (c), and (d) of the control system, which are also summarized in Algorithm 1. To capture a Brownian cargo particle, we steer multiple robots toward target sites surrounding the cargo and cause the cargo to be captured. As the cargo undergoes Brownian motion, the surrounding target sites are dynamically constructed around the cargo at every control time step. In free space, the surrounding target sites are configured to be a symmetric, hexagonally closed-packed lattice (Figure 1D); in mazes, we adapt the lattice-like target sites to the maze geometries in the vicinity of cargo, including when the cargo is near a wall (Figure 1E) and when the cargo is at a corner (Figure 1F). The construction of target sites in both free space and mazes can be realized by a single algorithm utilizing a graph-based breadth–width-search method (see code sample in Supporting Information). Although here we assume equal size between cargo particles and robots, the dynamic target generation algorithm can accommodate different cargo sizes.

The assignment step aims to optimally assign each robot a unique target such that the sum of travel distances between all robots and their targets is minimized. The optimal assignment is a time-dependent assignment and is mathematically given by

\[
g^* = \arg \min_{g \in G} \sum_{i \in I_A} D(r_{R,i}(t), r_{G,i}(t)) \quad (1)
\]

where \(r_{R,i}(t)\) and \(r_{G,i}(t)\) are swarm robot and target position vectors indexed by the set \(I_R = [1, 2, \ldots, N]\) and \(D(r_{R,i}(t), r_{G,j}(t))\) is the shortest geometrical path distance from the robot position \(r_{R,i}(t)\) to its candidate target position \(r_{G,j}(t)\). \(D\) is approximated via a landmark Dijkstra algorithm (Algorithm 2) to enable on-the-fly computation in the simulation. The assignment of each individual robot to track its target is specified by the assignment function, \(g(i) \in I_R\), which denotes the target index that the swarm robot \(i\) is assigned to track at time step \(t\). The optimal assignment \(g^*\) is obtained by optimizing all possible assignments \(G\) using the Hungarian algorithm for combinatorial optimization\(^{,30}\). The optimal assignment decision specified by \(g^*\) coordinates individual robots to optimize their movement for a common group-level goal—capturing the cargo. To minimize the total travel distance of the group, robots might not necessarily be assigned to track their nearest targets.

After assigning self-propelled particles to targets around the cargo, the optimal propulsion speed for each particle is determined as part of path planning. The landmark Dijkstra algorithm (Algorithm 2) not only provides an estimation of the shortest path length, but also specifies a series of intermediate targets that guide the robot toward the ultimate target site surrounding the cargo. When the distance between the robot and the ultimate target position is large or there is blockage between them, the nearest intermediate target is used as a temporary target to determine the optimal propulsion speed. In contrast, the ultimate target is used as the temporary target if the robot is sufficiently close to the ultimate target without blockage.

Denoting the temporary target for robot \(i\) as \(r_{A,i}\) we actuate the robot self-propulsion speed based on

\[
\nu^*_{R,i} = \begin{cases} 
\min \left( \frac{d_i}{3C_r}, v_{\text{max}} \right), & d_i > 0 \\
0, & d_i \leq 0 
\end{cases} \quad (2)
\]

Algorithm 1. Cargo capture.

1. Initialize cargo particle positions and robot swarm positions and orientations.
2. Initialize step counter \(k = 0\).
3. Initialize assignment interval \(M\). \(M = 10000\) for free space and \(M = 500000\) for mazes.
4. while True do
5. Sense all robots’ positions and orientations and cargo’s position.
6. Reconstruct the target sites around the cargo using Breadth-First-Search Algorithm (See Supporting Information Code Sample).
7. if \(k \mod M = 0\) then
8. Compute \(D(r_{R,i}(t), r_{G,j}(t))\) between all robots and all targets using the landmark Dijkstra algorithm [Algorithm 2]. We can assemble all pair distances to a distance matrix.
9. Update the optimal target assignment from the distance matrix using Hungarian algorithm.
10. end
11. Plan the shortest path between each robot and its assigned target and determine the temporary targets [Algorithm 2].
12. Calculate the optimal self-propulsions for all robots using Eq. (2).
13. Update \(k = k + 1\).
14. end

Algorithm 2. Landmark Dijkstra.

Result \(D(r_{A,i}, r_{T,j})\) and the temporary guiding target \(r_{A,i}\)

1. Initialize distance threshold \(\zeta = 100\)
2. if \(|r_{A,i} - r_{T,j}| < \zeta\) and there is no blockage in between then
3. \(D(r_{A,i}, r_{T,j}) = |r_{A,i} - r_{T,j}|\)
4. \(r_{A,i} = r_{T,j}\)
5. else
6. Find the starting landmark \(A\) as starting point closest to robot \(i\) without blockage between.
7. Find the destination landmark \(B\) as destination closest to target \(j\) without blockage between.
8. Calculate the shortest path between landmark \(A\) and destination landmark \(B\) through the Dijkstra’s Algorithm. \(D(r_{A,i}, r_{T,j})\) is shortest path length result.
9. \(r_{A,i}\) is the landmark next to landmark \(A\) in the shortest path.
10. end
where $d_i = (r_{A,i} - r_{T,i}) \cdot n_i$ is the projection of the displacement vector (from robot to its temporary target) onto its orientation vector $n_i$.

The formula given in Equation (2) was used previously in single robot navigation strategies\[16,28\] which aims to minimize the expected distance between each self-propelled particle and its assigned target position at each control update time. Intuitively, when the robot orients toward the target, self-propulsion strength proportional to the distance is applied, up to a given maximum-allowed propulsion speed $v_{\text{max}}$; otherwise, no propulsion is used and the robot just waits for Brownian rotation to sample favorable orientations (Figure 1C). Although the speed control policy in Equation (2) does not take into account the dynamic obstacles formed by other robots in the capture process, the time-dependent assignment has been shown to help avoid dynamic obstacles via target reassignment\[26\] Equation (2) might be subject to change when considering additional physics that is not captured here (e.g., hydrodynamics\[31\], robots with nonspherical shape\[1,2\]), but our general feedback control framework remains effective.

### 2.2. Cargo Capture in Free Space and Mazes

We first demonstrate the effectiveness of the cargo-capture algorithm (Algorithm 1) through steering a robot swarm of 36 robots to capture a single Brownian cargo particle in both free space and increasingly complicated mazes (Figure 2A–D). Figure 2A (animated in Movie S1, Supporting Information) shows a representative trajectory of the robot and the trajectory of the cargo particle during one typical cargo-capture-simulated realization in free space, where 36 robots were initially 50a away from the cargo. The capture process can be roughly divided into two stages. In the first stage, which we will call the “fast-approaching” stage, the robot swarm rapidly moves toward the target sites surrounding the cargo, causing the cargo to be loosely caged by the swarm; in the second stage, which we will call the “slow-adjusting” stage, the robots carefully adjusted their position to achieve a “tight” capture of the cargo. Eventually, most robots arrived at their intended target sites and a stable hexagonal lattice was formed around the cargo. Notably, at the end of the first stage, when the cargo was only loosely caged by swarming robots, we observed enhanced random motion of the cargo.
compared with when there were few robots around. The enhanced random motion is attributed to the imbalanced collisions from surrounding swarming robots, which is analogous to the fact that imbalanced molecular collision causes Brownian motion. The enhanced random motion also assists the cargo in escaping the capture and prevents the cargo from being captured by robots of smaller swarm size (e.g., fewer than 18 robots), as we observed previously.\[26\]

Figure 2B and Movie S2, Supporting Information, show the trajectory in a typical cargo-capture-simulated realization in a square maze (width of 82a). Robots are initially placed at random locations inside the maze and a cargo particle is placed at the center of the maze. Similar to the free space cargo-capture case, the cargo-capture process can be also roughly divided into two stages. In the first fast-approaching stage, the robots are steered to the cargo by circumventing blocking walls and dead-ends in the maze. Particularly, the robots follow the global shortest path, computed by the landmark Dijkstra algorithm (Algorithm 2), leading to their assigned targets. In the second slow-adjusting stage, where the robots fine tune their positions to tightly cage the cargo, the cargo particle and its surrounding robot swarm randomly move inside the maze. Depending on if the cargo is far away from any wall, near a planar wall, or near a corner, the target sites around the cargo are also adjusted dynamically to accommodate the local maze geometry. As a result, at the end of the second stage, the cargo and its surrounding robot swarm can end up forming clusters of different shapes, depending on the final location of the cargo and its local maze geometry (Figure S2, Supporting Information). The algorithm can be readily applied to conduct cargo capture in larger mazes of size 164a and 246a (Figure 2C,D and Movie S2, Supporting Information). Robots are all successfully steered to capture cargo particles despite the increasing maze size, maze complexity, and total travel distance.

In addition, from our repeated simulated experiments, successful cargo capture is achieved in every simulation, indicating the reliability of our method. As shown in Figure S2, Supporting Information, we also observe that cargo is most often captured in a corner and less often captured against a flat wall; it is rather rare when cargo is captured in empty regions in a maze. This is consistent with the expectation that the confinement from corners and walls prevents the cargo from escaping and assists robots in forming stable caging structures surrounding the cargo.

2.3. Analysis of Cargo Capture in a Maze

Given the stochastic nature of the cargo-capture process, we can also study the evolution of robot swarm positions by averaging trajectories from repeated simulated experiments. In Figure 3A, we visualize the evolution of robot swarm positions toward a spatially fixed cargo particle at a corner via spatial–temporal density distribution \(\rho(r, t)\). At the beginning of the cargo-capture process (10 s), most robots are concentrated at their initial positions. At 20 and 30 s, we observe a general
trend that the position density moves toward the cargo position. At 50 s, nearly all robots are concentrated around the cargo, forming a compact cluster at a corner.

The key decision-making component to achieve collaborative cargo capture ultimately boils down to when to turn on self-propulsion and when to turn it off for each robot. This self-propulsion decision-making of the algorithm can be visualized by self-propulsion velocity field \( v(r) \) by averaging the self-propulsion velocity from multiple realizations (Figure 3B). The velocity field shows that, in general, self-propulsion speeds enable robots to escape dead-ends, circumventing blocking walls, and follow the global shortest paths leading toward the cargo. The self-propulsion velocity field can be also interpreted as the decision maps: the optimal propulsion at each spatial location to get to the cargo.

Another quantitative characterization of the cargo-capture process is to measure the shortest path distances \( L \) between robots and their assigned targets with respect to elapsed time. As shown in Figure 3C, from the onset of cargo capture in free space, we observe that the average distance \( \langle L \rangle \) rapidly decreases to 5a within 20 s, corresponding to the first fast-approaching stage. The second slow-adjusting stage is roughly between 20 and 80 s, where the average distance \( \langle L \rangle \) further decreases to nearly 0. For mazes of different sizes, we observed qualitatively similar decreasing trends in \( \langle L \rangle \). Furthermore, larger mazes have longer periods of the fast-approaching stage as the robots are initially farther away from the cargo. Moreover, there are significant variations in individual distance decrements as the Brownian rotation randomizes the directed propulsion directions.\(^{[15]}\)

To quantify the efficiency of cargo-capture process, we can further measure the speed as \( \langle \dot{L} \rangle \) decreases. This not only reveals how fast robot swarms approach their targets but also indicates if the robot system operates efficiently. We focus on the initial regime of the plot \( \langle L \rangle \sim t \), which corresponds to the onset of the fast-approaching stage. If we assume that the robot system operates at its optimal efficiency, then all the robots with favorable propulsion direction are in full propulsion speed to follow their shortest path toward their targets, whereas all the robots with unfavorable propulsion direction turn off their propulsion. In free space, the initial average distance’s decreasing speed can be estimated by the expected propulsion speed from Equation (2) given by \( \langle \dot{L} \rangle = \int_{-\pi/2}^{\pi/2} \frac{1}{2} v_{\text{max}} \cos \theta \, d\theta = \frac{1}{2} v_{\text{max}} \), where \( \theta \) is the angle between the robot orientation vector and the displacement vector from robot to the temporary target. Assuming that half the time a robot is in its favorable orientation, then, \( \theta \) is integrated between \(-\pi/2\) and \(\pi/2\), corresponding to the case where the target is in front of the robot, such that the robot propulsion direction is favorable. The factor \( 1/2\pi \) is the probability density of \( \theta \) taking values between 0 and \( 2\pi \) because the robot’s orientation is arbitrary. In mazes, we refine the earlier estimation by multiplying a factor less than 1 to account for the fact that a robot self-propelling in a maze has chances of being obstructed by obstacles (see Supporting Information). A reasonable agreement (within 30% error) between the theoretically estimated optimal speed versus the actually measured speed is shown in Figure S3, Supporting Information, indicating that the stochastic trajectories of robot swarms are efficiently navigated toward their targets.

### 2.4. Multiple Cargo Capture via Multitasking

We have demonstrated that robot swarms under algorithmic control can capture a single Brownian cargo particle in both free space and mazes. The algorithm also readily applies to capturing multiple Brownian cargo particles simultaneously by intelligently dividing all robots into the same number of worker groups as the number of cargo particles. Capturing multiple cargo particles simultaneously, or more generally, multitasking, is one inherent advantage of multirobot systems over single-robot systems. In a multiple cargo-capture scenario, the high-level control flow in Figure 1B remains unchanged and the specific differences are the following: surrounding target sites are constructed around “each and every” cargo particle; robots are steered toward these target sites and the cargo is caged just like the single cargo-capture scenario.

Figure 4 and Movie S3–S4, Supporting Information, show multiple cargo-capture results in both free space and a medium-sized maze. Initially, 108 robots are equally divided into three groups (each group has 36 robots) and are assigned to capture three cargo particles (Figure 4A,B). The division and assignment are ensured by our assignment algorithm to minimize the total travel length, which again implies that our algorithm encodes collaboration for both the single cargo-capture task and multicargo-capture task. As the assignment strategy is to minimize the sum of all robots’ travel path length as a whole, a few robots are assigned to a cargo particle far away instead of the nearest one. Finally, all robots can arrive at their assigned targets and successfully capture all cargo particles (Figure 4C,D).

Because cargo particles undergo random motion, two cargo particles can have chances of getting rather closer to each other. When swarm robots are trying to cage the two cargo particles, the two separate robot–cargo clusters may eventually merge into a big cluster surrounding the two cargo particles (Figure S4, Supporting Information). The formation of stable merged clusters demonstrates the robustness of our algorithm when two nearby clusters interfere with each other.

### 2.5. Impact of Geometrical Confinement on Cargo Capture

So far we have considered cargo capture within mazes of different sizes and have demonstrated that our algorithm can generalize across maze size, swarm size, and the number of cargo particles to be captured. Now we examine the impact of geometrical confinement, specifically the width of maze passageway, on the cargo-capture process. Under the circumstances of strong confinement, we also analyze the impact of robot shortage on cargo-capture processes. Being capable of working in environments with a wide variety of geometrical features and confinements is essential to enable an intelligent microrobot system that can accomplish tasks in a broad range of applications, including cargo delivery in hard-to-reach tissues, diverse vessel structures, and porous media with cavities of varying sizes.

Here we consider mazes with the same size but with three different passageway widths (8a, 10a, and 14a) (Figure 5). Here, the maze passageway widths are controlled by modifying the width of the walls inside the maze. Swarms consisting of different number of robots (54, 72, and 108) are controlled to
capture three Brownian cargo particles. Equivalently, each cargo will be chased and captured by 18, 24 and 36 robots. In all cases, despite the confinement effects in different mazes, multiple cargo particles are all successfully captured by the swarm. For a maze with passageway width 14a (Figure 5A), cargo particles can be surrounded by the robot swarm against either a one-sided wall or a corner. Because the passageway width is wide enough, the eventual robot–cargo cluster can only be in contact with one wall or one corner. When we reduce the maze passageway width to 10a (Figure 5B) and 8a (Figure 5C), the final robot–cargo cluster shape is adapted to the strong confinement of the two walls, leading to the formation of an oval-shaped cluster sandwiched by the two walls. Note that when we reduce the passageway width to 8a, in which case maximally three robots can fit between two walls, the maze geometry poses strong confinement and affects the shape of the formed robot–cargo cluster.

In terms of cargo-capture capacity, it is interesting to know what would be the failing point if we continuously reduce the number of robots used to conduct cargo capture. Knowing the cargo-capture capacity can help determine the minimum resource allocation when we assign a robot swarm to accomplish multiple tasks at the same time. To identify the capacity in the cargo-capture task, we reduce the number of robots further down to 6 in the maze with passageway width 8 and examine at what point cargo cannot be captured. First, swarms of 18 robots can successfully capture cargo particles both in a corner and near a wall (Figure 5C); for swarms of 12 robots, cargo capture is only achieved at a corner and fails in free space or near a single plane wall (Figure S5, Supporting Information). As for swarms consisting of only 6 robots, cargo cannot be captured even at the corner (Figure S5, Supporting Information). In short, 12 robots and 18 robots are sufficient to cage a Brownian cargo at a corner and near a wall, respectively, whereas in free space, at least 36 robots are needed.[26] It is also demonstrated that our algorithms can smartly leverage maze geometrical features when there is a shortage of robots. Robot–wall interaction is another critical factor that influences the capture capacity when there is a shortage of robots. In this study, we adopt the assumption that robots and walls interact via electrostatic repulsion, which is most commonly observed in experiments.[32] Having attractive robot–wall interaction can be expected to further increase the system capacity.

Figure 4. Multiple cargo capture via multitasking. A,B) Initial positions and their assignments of a robot swarm (108 robots) to capture three Brownian cargo particles in A) free space and B) a maze of size 164a × 164a. Hollow circles denote targets’ positions, solid circles denote robots’ initial positions, and stars denote cargo particles. Three different colors denote the split of the whole swarm into three subgroups to capture different cargo particles. C,D) Final positions of robots and cargo particles after cargo particles are captured in C) free space and D) a maze. Hollow circles denote robots’ initial positions, solid circles denote robots’ final positions, and a representative trajectory (distinguished by different colors) of a robot is shown for each cargo.
3. Conclusion and Outlook

We have presented a multiagent feedback control algorithm that can efficiently control a swarm of self-propelled colloidal micro-robots to collaboratively capture Brownian cargo particles in a 2D maze. Virtual targets sites are constructed around cargo in a hexagonal-closely packed pattern and the construction adapts to local geometries in the vicinity of the cargo. By navigating individual robots to target sites surrounding cargo particles, cargo particles are caged and captured. Our algorithm is generalized across maze size, maze geometrical confinement, number of cargo particles, and robots and is resilient to the shortage of robots.

Our scheme is a flexible and extensible framework consisting of separate modules responsible for target generation, assignment, path planning, and optimal speed computation. To extend the current scheme to colloidal robot systems with different robot dynamics and physics (e.g., hydrodynamics and biological barriers), we can work out the right equation to substitute Equation (2) for optimal speed computation. The assignment framework (as in Equation (1)) can be used with other cost functions (i.e., replacing D) to incorporate different control objectives and constraints, which ultimately broaden the applications of a robot swarm in a wide variety of adversarial environments beyond the mazes considered here, such as flow fields and blood vessels and gastrointestinal tracts. Additional enhancements to current work include integrating neural network controller collaboration in unknown environments, utilizing the state-of-the-art multiagent deep reinforcement learning for more sophisticated collaboration strategies between colloidal robots and adopting decentralized control schemes.

Figure 5. Impact of maze geometrical confinement and swarm size on cargo capture. Three robot swarms with each made up of 18, 24, and 36 robots (left column, middle column, and right column) are controlled to capture three cargo particles in mazes with passageway widths w of 14a, 10a, and 8a (top row, middle row, and bottom row).
to achieve scalable intelligent systems consisting of thousands of robots.\[43\]

4. Experimental Section

**Brownian Dynamic Simulations.** Let \( r_{B,i}, \theta_i, i = 1, \ldots, N_B \) be the positions and orientations of \( N_B \) self-propelled robots making up the swarm. Let \( r_{O,j}, i = 1, \ldots, N_O \) be the positions of \( N_O \) cargo particles. The equation of motion for self-propelled robots, accounting for Brownian translation and rotation, and robot–robot, robot–obstacle, robot–cargo, and cargo–obstacle interactions, is given by

\[
\begin{align*}
\dot{r}_i(t+\Delta t) &= r_i(t) + \frac{D_i}{kT} F_i \Delta t + \Delta r_{R,i}^B + \nu_{R,i}(t) \eta_i \Delta t \\
\dot{\theta}_i(t+\Delta t) &= \theta_i(t) + \Delta \phi_i^B
\end{align*}
\]  

(3)

where \( D_i \) and \( D_\theta \) are the translational and rotational diffusivities for spherical robots, \( \nu_{R,i} \) is propulsion speed as the control input, and \( \eta_i = (\cos(\theta_i), \sin(\theta_i)) \) is the robot orientation. Further, \( kT \) is thermal energy, \( \Delta t \) is the integration time step, \( \Delta r_{R,i}^B \) denotes zero-mean Brownian translational movements with diagonal variance matrix of \( 2D_i \Delta t, \) and \( \Delta \phi_i^B \) denotes zero-mean Brownian rotation movement with variance of \( 2D_\theta \Delta t. \)

The interactive force, \( F_i, \) acting on the spherical robot \( i, \) is computed from the negative gradients of scalar potentials as

\[
F_i = -\left( \sum_{j \neq i} \nabla U^F(r_{ij}) + \sum_{j \neq i} \nabla U^B(r_{ij}) + \nabla \sum_{j \neq i} U^O(r_{ij}) \right)
\]

(4)

where the pairwise interactive potentials include robot–cargo interaction \( U^F, \) robot–robot interaction \( U^B, \) and robot–obstacle interaction \( U^O. \) The robot–robot interactions are modeled as the sum of electrostatic repulsion and depletion attraction \( a_{ij}^{\text{DE}} \)

\[
U^B(r_{ij}) = B \exp\left[-\kappa\left(r_{ij}^2 - 2a\right)\right] + \Delta \Pi V_{ex}(r_{ij})
\]

(5)

where \( B \) contains material property constants, \( \kappa^{-1} \) is Debye length, \( a \) is robot radius, \( r_{ij} = ||r_i - r_j|| \) is particle pair separation, and \( V_{ex} \) is excluded volume between spheres given by

\[
V_{ex}(r) = \frac{4\pi}{3}(a + L)^3 \left[ 1 - \frac{3}{4}\left( \frac{r}{a + L} \right) + \frac{1}{16}\left( \frac{r}{a + L} \right)^4 \right]
\]

(6)

\( \Delta \Pi \) is osmotic pressure, which is adjusted to ensure that the resulting interaction has around 5\( kT \) attraction throughout the work. This level of attraction was found to stabilize the caging structure via an induced crystallization mechanism without affecting the ability of robots to move around.\[24\] For simplicity, the robot–cargo and robot–obstacle pair potential was given by electrostatic repulsion, same as the first term in the robot–robot pair potential.

Simulation parameters are shown in **Table 1**, including a) spherical robot radii, b) electrostatic prefactor, c) Debye length, d) depletant radius, e) osmotic pressure, f) translational diffusivity, g) rotational diffusivity, h) integration time step, i) maximum propulsion speed, and j) control update time.

**Approximate Shortest Path:** We have developed a landmark Dijkstra algorithm (Algorithm 2) to estimate and plan the shortest path between robots and targets in a maze. The calculation is conducted on the fly along with the simulation. We first discretized the space into grids and placed landmarks on the grid points unless the grid points overlapped with obstacles. We then construct a graph based on the landmarks, in which nodes are landmarks and each node is connected with its neighboring landmark nodes (at most eight). The neighboring landmark nodes might be fewer than eight if there are obstacles nearby. The graph’s edges are assigned weights equal to the Euclidean distance between nodes. Using Dijkstra’s algorithm on this weighted graph, we can obtain the shortest path and the path length between any two nodes (landmarks). Given a robot and a target, we define starting landmark \( A \) as the nearest admissible (not blocked by obstacles) to the robot and destination landmark \( B \) as the nearest admissible to the target. Then, the shortest path between the robot and the target is approximated by the shortest path between landmark \( A \) and landmark \( B \) using the Dijkstra algorithm on the constructed landmark graph. Also note that the landmark next to the landmark \( A \) in the shortest path serves as the temporary target to guide the robot toward its target. Practically, this approximate landmark Dijkstra algorithm is only used when the distance between the robot and its target landmark is greater than a threshold (10\( s \)); otherwise, the robot is directed to propel straight toward the target position.

**Acknowledgements**

K.X. and Y.Y. contributed equally to this work. Support from the National Natural Science Foundation of China (grant nos. 11922207, 11961131005, and 11921002) is acknowledged. The Supporting Information of this article can be found here: https://authorea.com/doi/full/10.22541/au.162557471.16734880/v1.

**Conflict of Interest**

The authors declare no conflict of interest.

**Data Availability Statement**

Data available on request from the authors.

**Keywords**

artificial intelligence, colloidal microrobot swarms, multiagents, multitasking

**Table 1. Parameters in simulation.**

| Parameter | Equation | Value | Parameter | Equation | Value |
|-----------|----------|-------|-----------|----------|-------|
| \( a(\text{nm})^2 \) | (5) | 1000 | \( D_i(\text{m}^2/\text{s}) \) | (3) | 2.145e - 13 |
| \( B(\text{a} kT) \) | (5) | 2.29 | \( D_i(\text{rad}^2/\text{s}) \) | (3) | 0.161 |
| \( \kappa^{-1}(\text{nm})^2 \) | (5) | 50 | \( \Delta \Pi(\text{ms}) \) | (3) | 0.05 |
| \( L^d \) | (6) | 200 | \( \nu_{ex}(\text{m/s}) \) | (2),(3) | 5e - 6 |
| \( \Delta \pi(\text{kT/nm}^2) \) | (5) | 5.8 | \( \Delta \Pi(\text{s}) \) | (1),(2) | 0.1 |

**References**

[1] S. Sanchez, L. Soler, J. Katuri, Angew. Chem. Int. Ed. 2015, 54, 1414.
[2] J. Li, I. Rozen, J. Wang, ACS Nano 2016, 10, 5619.
[3] E. Fodor, M. C. Marchetti, Phys. A 2018, 504, 106.
[4] C. Nardini, E. Fodor, E. Tjhung, F. Van Wijland, J. Tailleur, M. E. Cates, Phys. Rev. X 2017, 7, 021007.
[5] C. Bechinger, R. Di Leonardo, H. Löwen, C. Reichhardt, G. Volpe, G. Volpe, Rev. Mod. Phys. 2016, 88, 045006.
[6] T. Bäuerle, R. C. Löffler, C. Bechinger, Nat. Commun. 2020, 11, 2547.
