Supporting Information for "Brownian Cargo Capture in Mazes via Intelligent Colloidal Microrobot Swarms"

Kun Xu[1], Yuguang Yang[1], and Bo Li[1]

[1]Tsinghua University

July 6, 2021

Abstract

This supporting information includes supplemental figures, movies, additional results, and the pseudocode of breadth-first-search target generation algorithm.

Corresponding author Email: yyang60@jhu.edu (Y.Y.) libome@tsinghua.edu (B.L.)

1 List of Supplemental Figures

2 Supplemental Movies

- Movie S1. Single cargo capture in free space.
- Movie S2. Single cargo capture in mazes of three different sizes.
- Movie S3. Multiple cargo capture in free space and a maze of size 164a×164a.
- Movie S4. Multiple cargo capture in a maze (passageway width 8a) with different numbers of robots.

3 Supplemental Methods and Results

3.1 The landmark Dijkstra algorithm in a maze

Our strategic control of a robot swarm to capture a cargo particle in a maze relies on the landmark Dijkstra algorithm to approximate the shortest path between robots and targets (as described in the main text Algorithm 2). In short, the algorithm first discretizes a 2D space to get grids and landmarks. Landmarks are then treated as graph nodes, and connections between adjacent nodes are treated as graph edges whose weight equals Euclidean distance between nodes. By applying the Dijkstra Algorithm on this weighted graph, the shortest path between arbitrary nodes/landmarks can be computed, which will be further used to approximate the shortest path between robots and targets. In Fig. S1, we show the shortest path generated from the algorithm, where the blue end refers to the starting point and the yellow end refers to the destination.

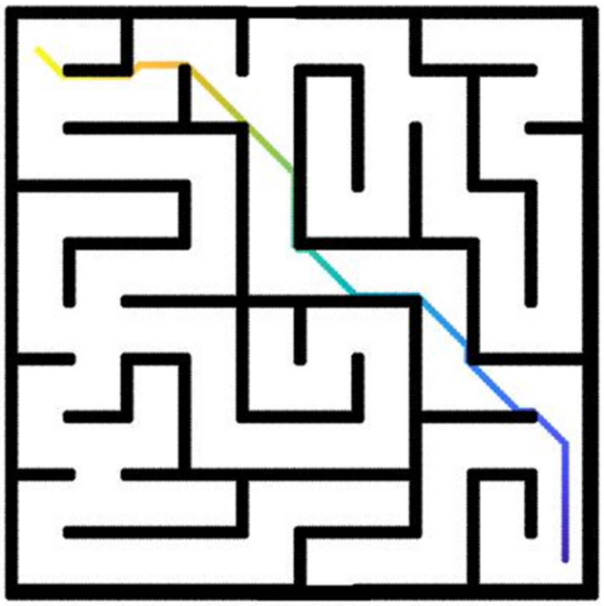

Figure 1: The shortest path generated from landmark Dijkstra algorithm in a maze from the start point (blue) to the destination (yellow).

3.2 Additional cargo capture results

Besides the representative final capture configuration shown in the main text Fig. 2, here we report additional results of a swarm (36 robots) capturing a single Brownian cargo particle in a medium-sized maze $164a \times 164a$ [Fig. S2] . Because the cargo and swarm undergo Brownian motion, the formed clusters can end up at different locations in the maze.

3.3 Initial navigation speed analysis

As described in the main text, we performed the initial navigation speed (0-0.4s) for the results in main text Fig. 3. 50 independent simulations were carried out in each maze with different initial conditions. In free space, the initial optimal navigation speed can be estimated to be $\langle v_{T,f} \rangle = \frac{1}{\pi} v_{\max}$ as described in the main text. Fig. S3A shows a good agreement between theoretical predicted mean speed and actual measured average speed from simulation experiments.

The same analysis in a maze needs to account for the case that robots may be obstructed by obstacles and cannot contribute to the reduction of path distance to the target. Hence we apply a positive factor less than 1 as the correction to $\langle v_{T,f} \rangle$, leading to $\langle v_{T,m} \rangle = \langle v_{T,f} \rangle \times f_{\text{obs}}$. Here we propose an approximation $f_{\text{obs}} = \frac{S_{\text{obs}}}{8}$, where 8 denotes the 8 evenly spaced directions in a 2D plane and $S_{\text{obs}} \in \{1, 2, ..., 8\}$ is the number of directions (out of these 8 directions) it can go without hitting obstacles. S_{obs} can be computed

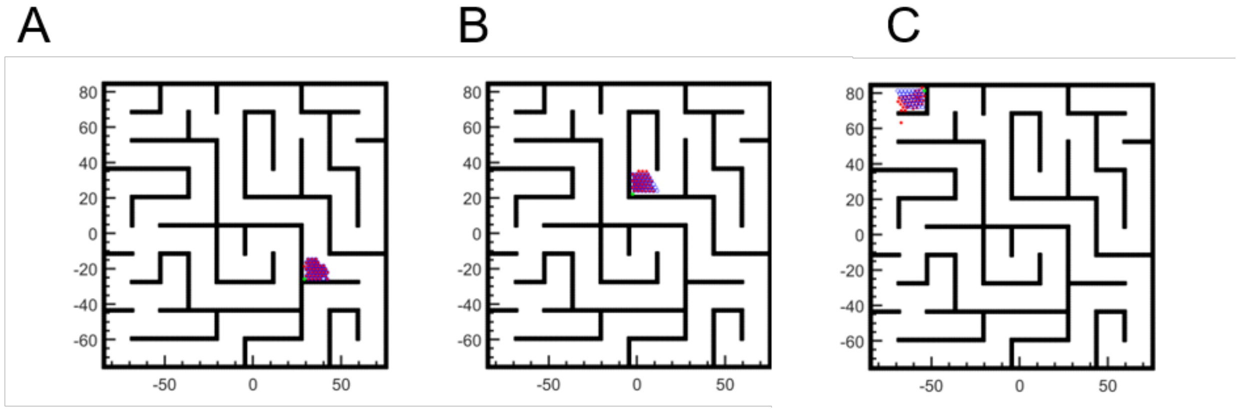

Figure 2: Ending configurations from different simulation runs of a swarm (N=36) capturing a single Brownian cargo in a 164a × 164a maze.

from the robot's position, orientation, and presence of obstacles near the robot. Intutively, f_{obs} approximately characterizes the probability of the robot *not* being obstructed at its current position. In Fig. S3B-D, we observed an reasonable agreement (up to 30% relative error) between the theoreical prediction and the averaged speeds from simualted experiments.

3.4 Algorithmic robustness analysis

In multiple cargo capture experiments, two cargo particles can get rather closer to each other due to Brownian motion. As swarm robots are trying to cage the two cargo particles, two separate clusters may eventually merge into a big cluster surrounding the two cargo particles. Fig. S4 shows the observations from experiments in free space and in a maze. Note that the formation of stably merged clusters indeed demonstrates the robustness of our algorithm when two nearby clusters are interfering each other.

3.5 Cargo capture with a shortage of robots

Here we show additional experimental results to support the investigation of cargo capture failure caused by a shortage of robots. In Fig. S5, we show that: for a swarm of 6 robots, cargo cannot be captured; for a swarm of 12 robots, cargo capture is only achieved at a corner but both fails in free space and near a single plane wall.

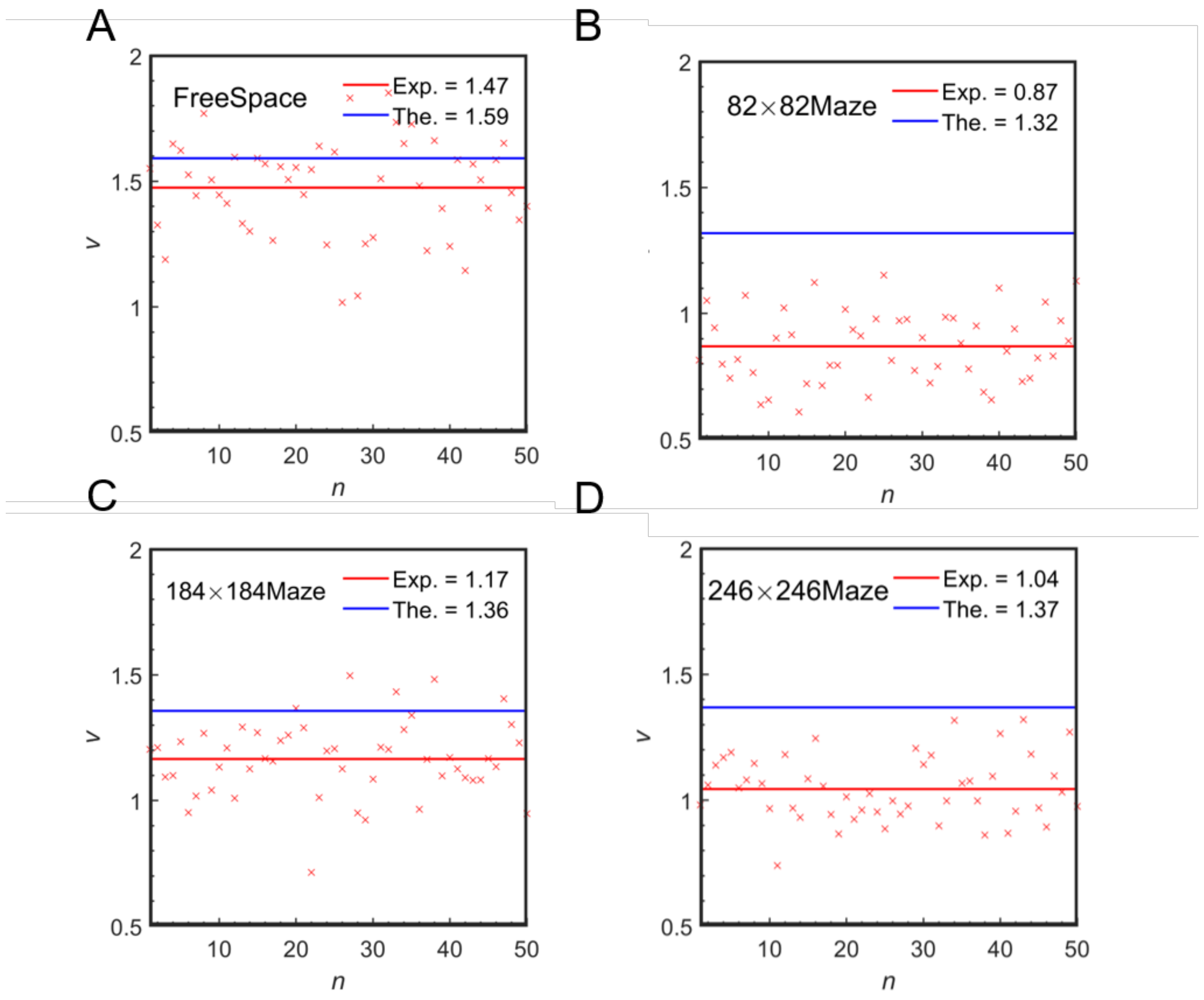

Figure 3: Initial navigation speed analysis(0-0.4s), the red crosses refer to the experiment points, red lines refer to statistical average initial travel speed, blue lines show the probabilistic expectation. (A) In free space. (B) In 82a×82a maze. (C) In 164a×164a maze. (D) In 246a×246a maze.

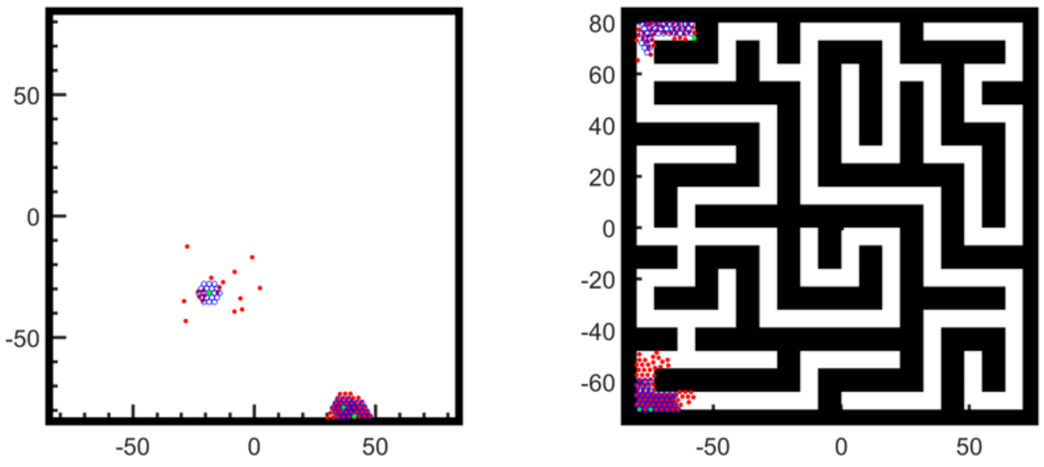

Figure 4: Final robot-cargo cluster configurations indicate algorithmic robustness to multi-cargo merging scenarios in free space (left) and in a maze (right). Blue solid circles are cargo particles. Red solid circles are robots. Hollow blue circles are generated target sites around the cargo particles.

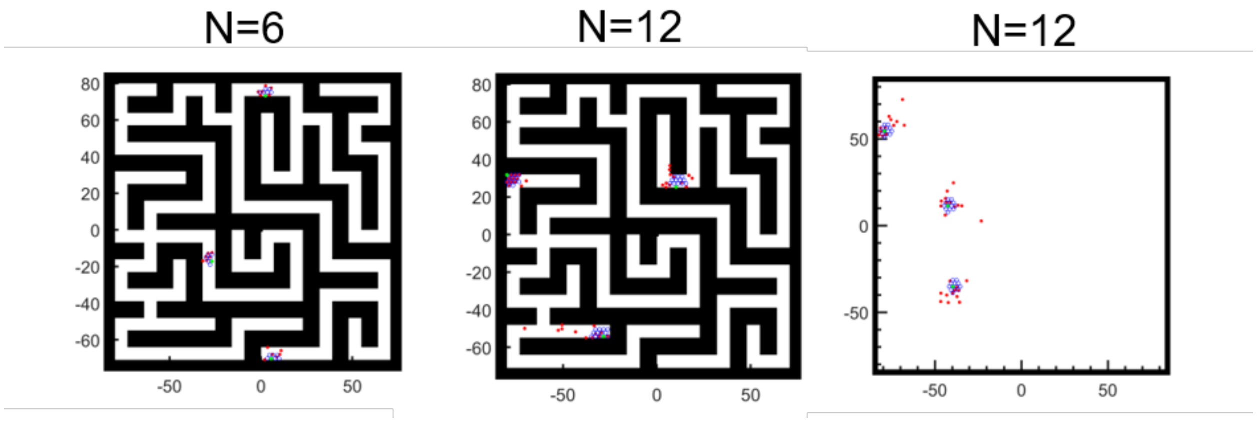

Figure 5: Additional experimental results to support the investigation of cargo capture failure caused by a shortage of robots.

Code Sample

This part shows the pseudocode of Breadth-First-Search algorithm for our dynamic targets generation (in C++ style).

```
class DynamicTargetsGeneration{
function:
read_candidate_targets;
generate_targets;

data:
_candidate_targets_list;
_target_neighbor_matrix;
};



function read_candidate_targets ( )
{
    let the center be the origin;
    generate a large number of candidate targets in hexagonal-close-packed structure;
    _candidate_taregts_list[0] <- origin;
    store candidate targets' id and positions relative to the origin into _candidate_targets_list in th

    for each candidate target:
        search its at most 6 nearest neighbors;
        store neighbor's id into target_neighbor_matrix;
    }
}


function generate_targets ( cargo_position , total_target_num ) {
    working_queue: a queue used to store working candidate targets;
    used_set: a set used to store used candidate targets;
    result_targets_list: stores allowed targets to be output;

    initialize working_queue.push(0);
    initialize used_set.insert(0);

    while ( length( result_targets_list ) < total_target_num ) {
        if ( working_queue is empty ) { ERROR "Not Enough Targets Found!" };
        working_target <- working_queue.front();
        working_queue.pop();

        if ( working_queue.front() != 0 ){
            result_targets_list.pushback( working_target );
        }

        for each neighbor of working_target{
            if ( this neighbor target does not exist in used_set ) {
                positon <- cargo_position + neighbor_position;
                check if this neighbor target overlap with any obstacle;
                if ( not overlap ) {
                    working_queue.push( this neighbor );
                    used_set.insert( this neighbor );
                }
            }
        }
    }

    return result_targets_list;
}
```

Supplementary Movie S1

Figure 6: Movie S1. Single cargo capture in free space.

Supplementary Movie S2

Figure 7: Movie S2. Single cargo capture in mazes of three different sizes.

Supplementary Movie S3

Figure 8: Movie S3. Multiple cargo capture in free space and a maze of size 164a×164a.

Supplementary Movie S4

Figure 9: Movie S4. Multiple cargo capture in a maze (passageway width 8a) with different numbers of robots.

References