A Correlated Random Walk Model to Rapidly Approximate Hitting Time Distributions in Multi-Robot Systems

Yi Zhang, Daniel Boley, John Harwell, and Maria Gini

Computer Science and Engineering, University of Minnesota, Minneapolis MN 55455, USA, {zhan2854, boley, harwe006, gini}@umn.edu

Abstract. Multi-robot systems are frequently used for tasks involving searching, so it is important to be able to estimate the searching time. Yet, simulation approaches and real-world experiments to determine searching time can be cumbersome and even impractical. In this work, we propose a correlated-random-walk based model to efficiently approximate hitting time distributions of multi-robot systems in large arenas. We verified the computational results by using ARGoS, a physics-based simulator. We found that the Gamma distribution can provide a good fit to the hitting time distributions of random walkers.

Keywords: multi-robot systems, correlated random walk, hitting time distribution

1 Introduction

Operators can release multiple robots to search for a target within a predefined area. We denote this process as "searching" because robots don't know the target location. Robot searching has many real-world applications (e.g. foraging, rescuing, providing supplies, etc.) [14, 19]. In this paper, we define the searching time as the time required by at least one of the robots to first reach the target. For many tasks, especially the time-sensitive ones, an estimate of searching time is often necessary. Furthermore, having an estimated searching time can help in the design of multi-robot systems that are more efficient at searching. Experimental investigation involving real robots (see [10] for an example) is often costly and can be even impractical; simulation (e.g. ARGoS [26]) can also be time-consuming, especially when the searched region is large. In this work, we aim to develop a computation model that can rapidly approximate the searching time.

Hitting time (HT) or first-passage time is the average time to first visit a node while doing a random walk (RW) from another node in a given network [21, 22]. Many results for HT exist. In particular, the average time to reach a given node can be computed based on the fundamental matrix associated with the probability transition matrix of the network [13], by treating the target node as an absorbing state. Boley et al. [5] showed how to obtain average HTs from any node to any node in a directed graph at once using the assymetric graph Laplacian, and later showed one way to obtain this rapidly using sparse matrix methods [4]. In the HT literature, there are also papers about HT higher moments [17, 9], upper bounds [6], and distributions [20]. In addition, there are studies on the speed up of random walker search when there are multiple searchers [1, 11, 25] and random walker collisions [2, 12].

To the best of our knowledge, none of the methods cited above has been applied directly to multi-robot systems or swarm robots. Thus, we aim to leverage some of these results to develop a computation model that can rapidly and accurately approximate the HT in the multi-robot searching process. Due to the forward persistence of robot motion, Jeong et al. [16] used correlated random walk (CRW, i.e. RW where the state transition depends on both the location and orientation of the walker [24]) to approximate robot movements in their network-based model. This is the approach used here.

Our work makes three main contributions to the field of multi-robot systems and stochastic process. First, our work is a pioneer attempt to apply HT results to analyze multi-robot searching processes. Second, we developed a computation model that can efficiently approximate the HT distributions for multi-robot searching processes, and we verified the computation results with the ARGoS simulator. Third, we found that the Gamma distribution can provide a good fit to CRW HT distributions.

The rest of the paper is organized as the follows. Section 2 describes the problem setting. Section 3 provides the Correlated Random Walk (CRW) computation results. Section 4 compares the computation results with the ARGoS robot simulation results. Section 5 provides a discussion and concludes the paper.

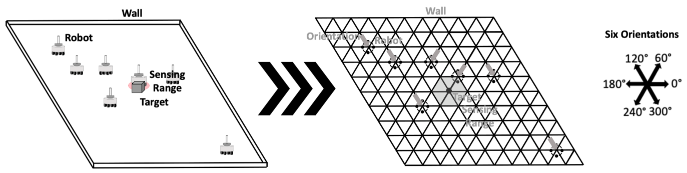


Fig. 1. Problem settings. Arena shape is a parallelogram with sides of equal length. Target location is the center. Left: a multi-robot searching example. Middle: the corresponding representation in our computation model (discretization by equilateral triangles). Right: six orientations of the robot.

2 Problem Settings

The problem settings are shown in Fig. 1 with details in the caption. We used parallelogram for arena shape because it simplifies the construction of the probability transition matrix. We discretized the arena using equilateral triangles

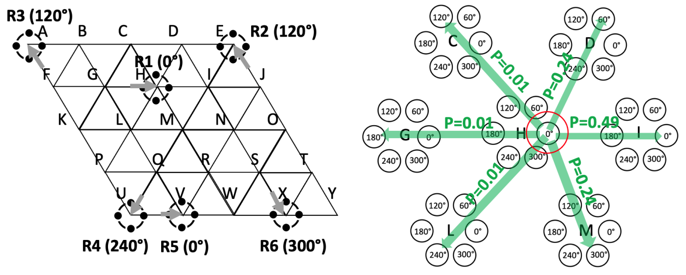
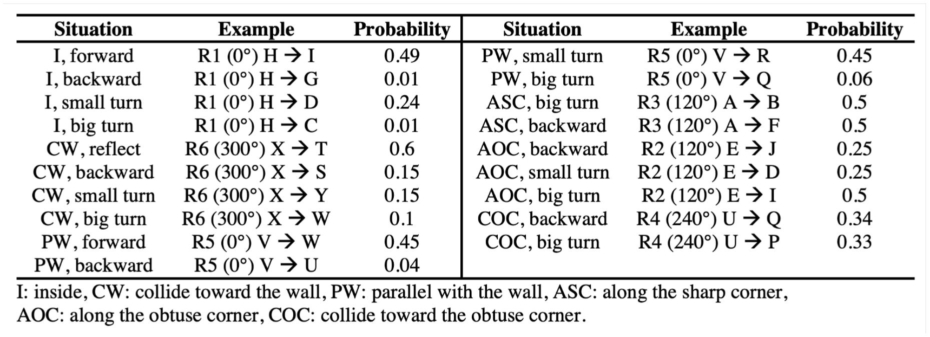


Fig. 2. CRW state transition details. Left: six possible situations a robot could encounter in an arena (from R1 to R6). Right: state transitions (green arrows) associated with robot R1 from the left. Table 1 lists the transition probabilities.

Table 1. Transition probabilities for different situations encountered by robots.



Situation	Example	Probability	Situation	Example	Probability
I, forward	R1 (0°) H → I	0.49	PW, small turn	R5 (0°) V → R	0.45
I, backward	R1 (0°) H → G	0.01	PW, big turn	R5 (0°) V → Q	0.06
I, small turn	R1 (0°) H → D	0.24	ASC, big turn	R3 (120°) A → B	0.5
I, big turn	R1 (0°) H → C	0.01	ASC, backward	R3 (120°) A → F	0.5
CW, reflect	R6 (300°) X → T	0.6	AOC, backward	R2 (120°) E → J	0.25
CW, backward	R6 (300°) X → S	0.15	AOC, small turn	R2 (120°) E → D	0.25
CW, small turn	R6 (300°) X → Y	0.15	AOC, big turn	R2 (120°) E → I	0.5
CW, big turn	R6 (300°) X → W	0.1	COC, backward	R4 (240°) U → Q	0.34
PW, forward	R5 (0°) V → W	0.45	COC, big turn	R4 (240°) U → P	0.33
PW, backward	R5 (0°) V → U	0.04			

I: inside, CW: collide toward the wall, PW: parallel with the wall, ASC: along the sharp corner,
AOC: along the obtuse corner, COC: collide toward the obtuse corner.

instead of squares because the former approach leads to six possible orientations for robots and thus increases flexibility, while keeping all link lengths the same. When a robot is within a certain distance from the target (sensing range), it can sense the target and pick it up. We model the robot motion as a correlated random walk (CRW), implemented as an ordinary random walk with states encoding both robot position and orientation from the previous state. The nodes in the network are the states (position, orientation). The edges are state transitions, each weighted by the transition probability. At each position of the discretized arena, a robot has six possible orientations (Fig 1 right).

Fig 2 (right side) provides an example of the state transitions in CRW associated with the robot R1 on the left. Currently, R1 is in state (H, 0°) (circled in red). R1 can move forward to reach position I with orientation 0°. Hence the new state of R1 would be (I, 0°). R1 can also make a small turn (60°) to reach D (if turning left) or M (if turning right). The new states would be (D, 60°) or (M, 300°) respectively. The robot can also make a big turn (120°, right or left) or go backward. If near the wall, the robot might also reflect. Since the robot tends to move forward and resists making large turns, the state transition probabilities are set to heavily favor forward motion, as listed in Table 1. Note that each robot could encounter six situations: interior within the arena (R1), move along an obtuse corner (R2), move along a sharp corner (R3), collide into an obtuse corner (R4), move parallel to a wall (R5), or collide into a wall (R6).

Transition probabilities associated with these situations are shown in Table 1, which contains examples for each situation. In the column "Example," "R1 (0°) H → I" denotes the state transition of R1 from position H with orientation 0° to location I (Fig 2). The remaining examples follow the same convention. Note that we used a transition probability setting known to reduce HT, but other probability settings can also be used. After specifying the probabilities, we constructed a probability transition matrix **P** that can be easily scaled for arenas of any size. Note that we use bold symbols to denote vectors or matrices in this paper. An $N \times N$ arena leads to $6(N+1)^2$ states, and the associated adjacency matrix will be $6(N+1)^2 \times 6(N+1)^2$.

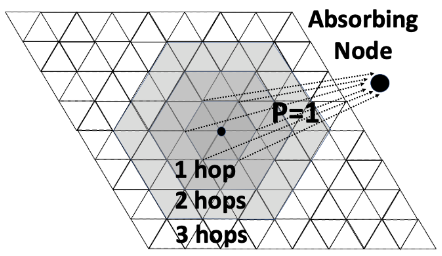


Fig. 3. Robot sensing ranges (e.g. 1-hop, 2-hop and 3-hop).

Our modelling approach of robot sensing ranges is shown in Fig 3. When robots sense the target, they will reach the target following the shortest path. To model this behavior, we add an extra absorbing node, and we adjust the network so that whenever the robot/random walker hits the sensing range boundary (shaded region), it will be absorbed by the extra absorbing node with a probability of 1. The connectivity shown in Fig 3 corresponds to the case with 1-hop sensing range. This method will always be valid if the robot starts outside the sensing range.

3 Correlated Random Walk Computations

Computation of HT mean and variance. Using the transition matrix **P** associated with this network, the HT mean $\mathbf{HT}_\mu$ and variance $\mathbf{HT}_{\sigma^2}$ from states outside the sensing range to the absorbing state can be determined. Following [13], we partition **P** to get **Q**, the probability transition matrix corresponding to the non-absorbing states, via:

$$\mathbf{P} = \begin{bmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{1}$$

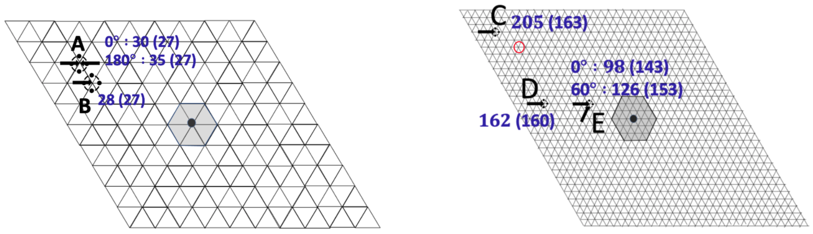


Fig. 4. Computed HT mean & standard deviation for a single robot starting from various initial positions. Blue numbers are $\mu(\sigma)$, on a 10×10 arena (left) and a 30×30 arena (right).

$\mathbf{R}$ is associated with the absorbing state(s) and $\mathbf{I}$ is the identity matrix. The fundamental matrix $\mathbf{N}$ of $\mathbf{P}$ is [13]:

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1} \tag{2}$$

$\mathbf{HT}_\mu$ can be determined by:

$$\mathbf{HT}_u = \mathbf{Nc} \tag{3}$$

where $\mathbf{c}$ is a vector of all 1s. After rearranging, we get:

$$(\mathbf{I} - \mathbf{Q})\mathbf{HT}_\mu = \mathbf{c} \tag{4}$$

$\mathbf{HT}_\mu$ can be efficiently solved for by iterative methods such as Restarted GMRES [4, 23], even if $\mathbf{I} - \mathbf{Q}$ is extremely large (e.g. $100,000 \times 100,000$). $\mathbf{HT}_{\sigma^2}$ can be calculated by [17]:

$$\mathbf{HT}_{\sigma^2} = (2\mathbf{N} - \mathbf{I})\mathbf{HT}_\mu - \mathbf{HT}_\mu^2, \tag{5}$$

where $\mathbf{HT}_\mu^2$ means elementwise squaring. After rearranging, we get:

$$(\mathbf{I} - \mathbf{Q})\mathbf{HT}_{\sigma^2} = 2\mathbf{HT}_\mu - (\mathbf{I} - \mathbf{Q})\mathbf{HT}_\mu - (\mathbf{I} - \mathbf{Q})\mathbf{HT}_\mu^2 \tag{6}$$

$\mathbf{HT}_{\sigma^2}$ can be computed similarly as before, and subsequently the HT standard deviation $\mathbf{HT}_\sigma$ can be obtained.

Some computation results are shown in Fig 4. All computations were done in Python using the sparse matrix format.

Single robot HT distribution. We used CRW simulations to determine HT distributions. Specifically, we started a random walker on the constructed network and let it move (i.e. state transition) with the probabilities specified in $\mathbf{P}$. We recorded HTs in 300 random trials to get a distribution.

The CRW simulation allows us to obtain the HT distribution. However, it is still time-consuming. We observed that all the obtained CRW HT distributions resemble the shape of a Gamma probability density function (PDF) $F(x)$:

$$F(x) = \frac{\beta^u}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \text{ with } \alpha = \frac{(\text{mean}(x))^2}{\text{var}(x)}, \beta = \frac{\text{var}(x)}{\text{mean}(x)}, \tag{7}$$

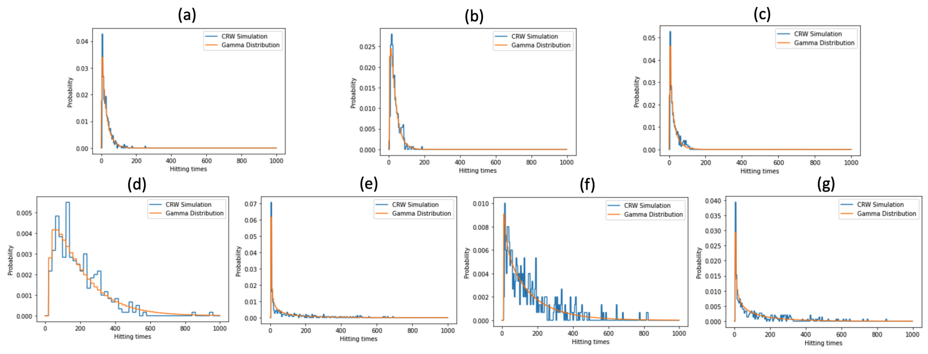


Fig. 5. Single robot HT distributions obtained by CRW simulation and Gamma distribution. (a) to (g) correspond to the initial robot states of (A, 0°), (A, 180°), (B, 0°), (C, 0°), (E, 0°), (D, 0°), and (E, 60°) shown in Fig 4, respectively.

where x is the random variable ($HT - HT_{\min}$ in this case), and α, β are the shape and rate parameters written in terms of the mean and variance of x. The hitting time cannot be less than the shortest path length $HT_{\min}$, hence we shift the Gamma distribution by this amount. Thus, based on Eq. 7, we can get single robot HT distributions using HT_{σ^2} (or HT_{σ}) and HT_{μ} by leveraging the Gamma distribution. Note that the Gamma PDF assigns positive probabilities to HTs smaller than the minimum HT ($HT_{\min}$) required for the robot to reach the target. To address this issue, we subtract $HT_{\min}$ from HT_{μ} to get HT_{μ^*}, and we fit the gamma PDF by HT_{μ^*} and HT_{σ}. Last, we shift the entire Gamma distribution to the right by $HT_{\min}$.

In Fig 5, we show the single robot HT distributions obtained by CRW simulation and Gamma distribution corresponding to 7 initial robot states in Fig 4. All settings (e.g. arena size and robot locations) are shown in Fig 4. The Gamma distribution parameters were determined from the mean and variance using Eq. 7 obtained using Eq. 4 and Eq. 6. Note that the HT distributions from CRW simulation are discrete, yet the Gamma distribution is continuous. To enable the comparison of distributions, the discrete distribution is converted to its continuous counterpart by summing discrete probabilities in small intervals, and then dividing the sums by the interval size. The same change is applied in the sequel to discrete distributions from CRW or ARGoS simulations. Clearly, a Gamma distribution can closely approximate HT distributions for a single robot/random walker in the computation model.

Multi-robot HT distribution. Next, we determined the multi-robot HT distributions using the computation model. First, we obtained the cumulative probability density function (CDF) $C\left(HT_{1\,robot}\right)$ corresponding to the PDF $F\left(HT_{1\,robot}\right)$ obtained with the Gamma distribution. Then, based on each value of $C\left(HT_{1\,robot}\right)$ (we denote a single value as $\dot{C}\left(HT_{1\,robot}\right)$), we can determine the cumulative probability for at least one robot out of n robots to reach the target within certain HT $\dot{C}\left(HT_{n\,robots}\right)$ by:

$$\dot{C}\left(HT_{n\,robots}\right) = 1 - \left(1 - \dot{C}\left(HT_{1\,robot}\right)\right)^{n} \tag{8}$$

Note that this equation relies on the assumption that the robots/random walkers are independent. This holds in our computation model. Strictly speaking, Eq. 8 also requires that all robots/random walkers have the same initial state. However, Eq. 8 is still a good approximation when different initial robot states are similar and lead to similar HT distributions. Note that Eq. 8 can be easily extended to the case when different robots start with different states. At this point, the computation model is able to approximate HT distributions for multiple robots using the computed HT_μ and HT_σ for a single robot. We verified this approach using the CRW simulation of multiple robots (Fig 6). The agreement between the results corroborates our computation approach.

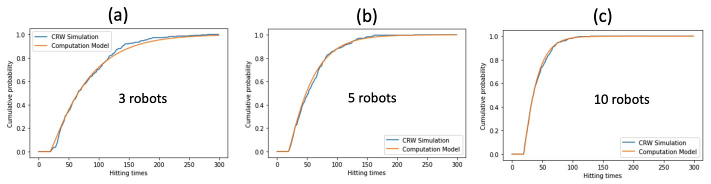


Fig. 6. Multi-robot HT CDF distributions determined by CRW simulation and computation. Arena 30×30, sensing range 3-hop, starting location of the robots circled in red in Fig 4 with initial orientation $0°$.

4 Verification by ARGoS

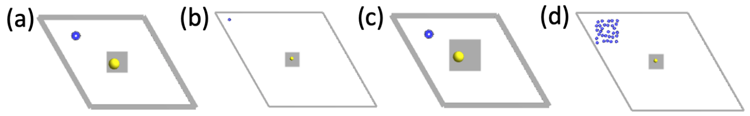


Fig. 7. Examples of ARGoS settings. (a) Small arena, sensing range 1, robot position point A (Fig 4) (b) Big arena, sensing range 3, robot position point C (Fig 4) (c) Sensing range 3. (d) 30 robots in the upper left corner.

Next, we use the ARGoS simulator to verify the computation model. Example screenshots of ARGoS simulation are in Fig 7. In ARGoS, we set the robot motion parameters so that it takes approximately 10 and 30 time steps respectively for the robot to travel the distances corresponding to the wall lengths of small and big arenas. The robot has a probability of 0.5 to move forward or making 60° turns (0.25 for turning upward or downward). Larger turns are impractical and not allowed. Note that we assigned very small probabilities (e.g. 0.01) for the random walker to make larger turns in the computation model mainly to ensure the network connectivity. To mimic the robot sensing range, we place the nest in the center of the arena. After the simulation starts, the time step when at least one robot first enters the nest is recorded as the HT. To mimic larger sensing ranges, we simply increase the nest size. The starting locations and orientations of the robots are set to match the computation settings as well. Note that multiple robots always start within a confined region located on the upper-left side of the arena with orientation 0°. These initial states are similar, and we verified that they lead to similar single-robot HT distributions. Note that the selection of the confined region and initial state is arbitrary as long as it is kept similar in computation and simulation.

Comparisons between ARGoS simulation and computation results for HT distributions of a single robot with different initial states are shown in Fig 8. HT distributions with different sensing ranges from two methods are in Fig 9. HT CDFs corresponding to different numbers of robots are in Fig 10.

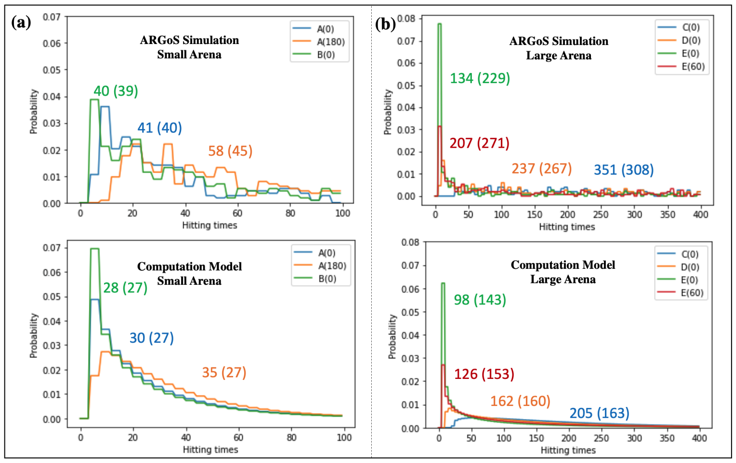


Fig. 8. Comparisons between CRW computation and ARGoS simulation results for single-robot HT distributions. (a) 10 × 10 arena (b) 30 × 30 arena. Robot starting states in the legend. For example, A(0) means the initial state of position A (Fig 4) and 0° orientation. Value pairs "mean (standard deviation)" from the computation model and ARGoS are next to the curves with the same colors.

Improved computation model. As shown in Figs 8, 9 and 10, the computation model and ARGoS simulation yield similar HT distributions under all circumstances we investigated. The computation model correctly predicts the ranking of HTs of a single robot with different initial states (Fig 8) and the

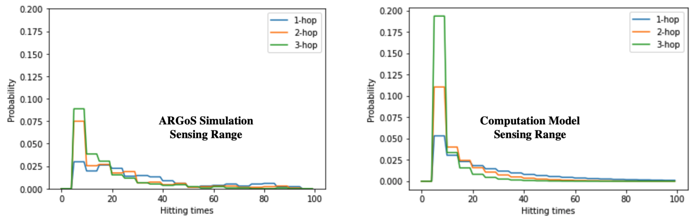


Fig. 9. ARGoS vs CRW: HT distribution for a single robot on a 10×10 arena with various sensing ranges, starting from point A.

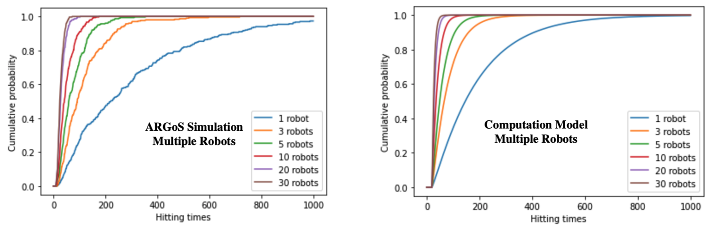


Fig. 10. ARGoS vs CRW: cumulative HT distributions for 30×30 arena, sensing range 3, with various number of robots.

speed up of the searching process as the robot sensing range (Fig. 9) and the number of robots increase (Fig. 10). Yet, we noticed that the value pairs of HT_μ and HT_σ obtained from ARGoS are on average 50% larger than the computed ones (Fig. 8). This led us to propose an improved computation model.

The aforementioned discrepancy is mostly due to the extra step associated with turning 60° in ARGoS simulation. By contrast, a random walker can turn instantly. To address it, we simply scale up the computed HT_μ and HT_σ by 50%. The rationale is the following. The robot has a probability of 0.5 to move forward or make 60° turns. Thus, for every 2 steps in the computation model, roughly 1 step is moving forward, and 1 step is turning 60°. However, practically, turning 60° requires 2 steps. Thus, every hitting time from the computation model should be increased by 50%. Subsequently, the computed HT_μ and HT_σ should be increased by 50%. We applied this change to the computation model and redrew Fig 10, and we obtained Fig 11.

As shown in Fig 11, the HT distributions from two approaches agree better, especially when the robot starts far away from the target (e.g. point C and D in Fig 4). Since our computation model works well on arenas of two different sizes, we tested its applicability to larger arenas. In the application of our improved computation model we aim to answer the following question: *In a* 150×150 *arena, how many robots (sensing range 1, initial position near arena upper left corner, initial orientation* 0°*) are required to have a probability of 0.8 that at least one robot will reach the target in the arena center within 500 time steps.*

This problem is related to many real-world tasks (e.g. foraging, rescuing, explosive removal). To solve it, first, we constructed the network associated with the robot movement in a 150×150 arena. This resulted in a $136,806 \times 136,806$

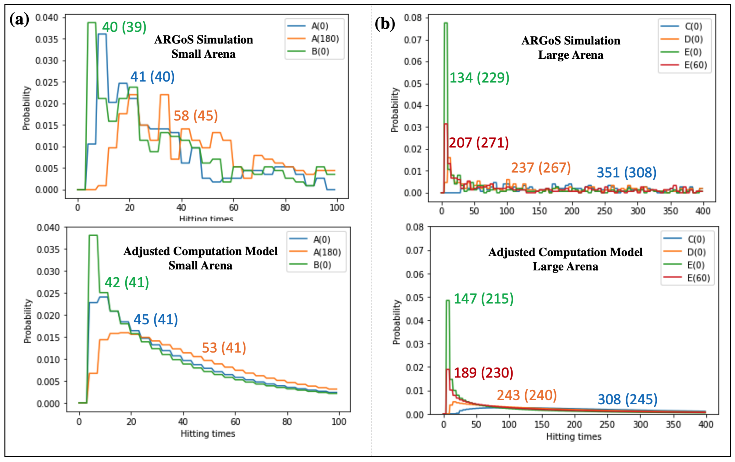


Fig. 11. Comparisons between ARGoS and the improved computation model.

sparse matrix. We solved Eqs. 4 and 6 with restarted GMRES (20 restart iterations)[4, 23] within 3 minutes using the CPU from Google Colab. Then, we can get HT_μ and HT_σ of a single robot with almost every initial state. We are only interested in one initial state (upper left corner, $0°$), and its associated HT_μ and HT_σ are 13,356 and 12,021 respectively. We scaled up these values by 50% and determined that the robot takes a minimum of 140 steps the target. Based on these values, we fitted the Gamma CDF for a single robot, and then applied Eq. 8 to determine the multi-robot HT CDFs. We varied the number of robots (Fig 12), and found that 200 robots are needed to have a probability of 0.8 that at least one robot will reach the target within 500 steps. ARGoS may take hours or days to answer the same question, and testing with real robots will be even more challenging.

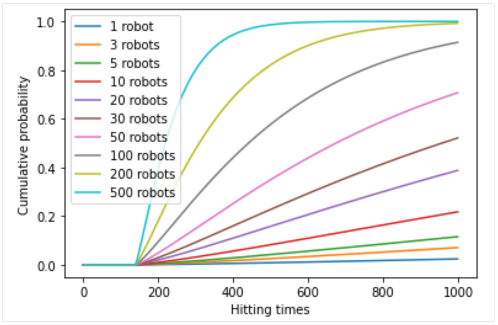


Fig. 12. Computed HT CDFs of multi-robot systems in a 150×150 arena.

5 Discussion and Conclusions

Searching time is often a quantity of interest, especially when tasks are time-sensitive. In addition, being able to rapidly determine Hitting Times (HTs) can facilitate the design of multi-robot systems. However, experiments with real robots are often impractical, and detailed simulation is time-consuming.

In this paper, we propose a Correlated Random Walk (CRW) computation model that can efficiently approximate the HT distributions of multi-robot systems in large arenas. We first tessellate a large arena with equilateral triangles to form a large network associated with robot movements. Based on the network, the single robot HT mean and standard deviation can be efficiently computed. We found and verified that the Gamma distribution can fit RW HT distributions. Furthermore, by leveraging a simple probability formula, we enabled the computation of multi-robot HT distributions based on the single robot HT distributions. Thus, our computation model can efficiently approximate multi-robot HT distributions. We verified the computation results by ARGoS simulation, which resembles real-world robot searching. We further improved the computation model by scaling up the computed mean and standard deviation of HT.

We found that the Gamma distribution can accurately approximate RW HT distributions when given only mean and standard deviation. This result itself is useful for stochastic processes, and can be used in fields involving stochastic processes (e.g. transportation, web, and animal modelling) [3, 8, 7, 15, 22].

Our model can approximate the ARGoS simulation results. Thus, we could approximate the optimal searching HT and associated parameter settings (e.g. transition probabilities) by performing a grid-search over the entire parameter space to determine the minimal HT. There are some sources of inaccuracy in our model. For example, our model ignores interactions between robots (e.g., avoidance, collisions), assuming robots are pair-wise independent. The inter-robot communication and issues with search area overlapping, collision, and under-utilization of robots will be investigated in the future. In the future, we will also consider the situation when all robots are required to reach the target. Other future directions include adding multiple targets and obstacles to the model, considering robot malfunctions [18], and modelling flying robots in a 3-D arena.

In conclusion, we proposed an efficient computation model that can rapidly approximate HT distributions of multi-robot systems. The model can be readily applied to solve real-world problems. Meanwhile, this study opens a wide range of research directions for future work.

References

1. Alon, N., Avin, C., Koucký, M., Kozma, G., Lotker, Z., Tuttle, M.R.: Many random walks are faster than one. Combinatorics, Probability and Computing **20**(4) (2011) 481–502
2. Barlow, M.T., Peres, Y., Sousi, P.: Collisions of random walks. Annales de l'IHP Probabilités et statistiques **48**(4) (2012) 922–946

3. Besenczi, R., Bátfai, N., Jeszenszky, P., Major, R., Monori, F., Ispány, M.: Large-scale simulation of traffic flow using Markov model. PLOS One **16**(2) (2021)
4. Boley, D.: On fast computation of directed graph Laplacian pseudo-inverse. Linear Algebra and its Applications **623** (2021) 128–148
5. Boley, D., Ranjan, G., Zhang, Z.L.: Commute times for a directed graph using an asymmetric Laplacian. Linear Algebra and its Applications **435**(2) (2011) 224–242
6. Brightwell, G., Winkler, P.: Maximum hitting time for random walks on graphs. Random Structures & Algorithms **1**(3) (1990) 263–276
7. Codling, E.A., Bearon, R.N., Thorn, G.J.: Diffusion about the mean drift location in a biased random walk. Ecology **91**(10) (2010) 3106–3113
8. Codling, E.A., Plank, M.J., Benhamou, S.: Random walk models in biology. Journal of the Royal Society Interface **5**(25) (2008) 813–834
9. Dayar, T., Akar, N.: Computing moments of first passage times to a subset of states in Markov chains. SIAM J Matrix Anal **27**(2) (2005) 396–412
10. Dimidov, C., Oriolo, G., Trianni, V.: Random walks in swarm robotics: an experiment with kilobots. In: Intl Conf on Swarm Intel. Springer (2016) 185–196
11. Efremenko, K., Reingold, O.: How well do random walks parallelize? In Dinur, I., Jansen, K., J, N., Rolim, J., eds.: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Volume 5687. Springer (2009)
12. Gaudillière, A.: Collision probability for random trajectories in two dimensions. Stochastic processes and appl **119**(3) (2009) 775–810
13. Grinstead, C.M., Snell, J.L.: Introduction to Probability. Amer Math Soc. (1997)
14. Hamann, H.: Modeling swarm systems and formal design methods. In: Swarm Robotics: A Formal Approach. Springer (2018) 95–127
15. Hill, N., Häder, D.P.: A biased random walk model for the trajectories of swimming micro-organisms. J Theor Biology **186**(4) (1997) 503–526
16. Jeong, M., Harwell, J., Gini, M.: Analysis of exploration in swarm robotic systems. IAS-16 (2021)
17. Kemeny, J.G., Snell, J.L.: Finite Markov Chains. Springer (1976)
18. Khalastchi, E., Kalech, M.: Fault detection and diagnosis in multi-robot systems: A survey. Sensors **19**(18) (2019)
19. Lancaster, J.P., Gustafson, D.A.: Predicting the behavior of robotic swarms in search and tag tasks. Procedia Computer Science **20** (2013) 77–82
20. Lau, H.W., Szeto, K.Y.: Asymptotic analysis of first passage time in complex networks. EPL (Europhysics Letters) **90**(4) (2010) 40005
21. Lovász, L.: Random walks on graphs. Combinatorics **2**(1-46) (1993) 4
22. Masuda, N., Porter, M.A., Lambiotte, R.: Random walks and diffusion on networks. Physics reports **716** (2017) 1–58
23. Morgan, R.B.: GMRES with deflated restarting. SIAM J Sci Comput **24**(1) (2002) 20–37
24. Nain, R., Sen, K.: Transition probability matrices for correlated random walks. J Appl Prob **17**(1) (1980) 253–258
25. Patel, R., Carron, A., Bullo, F.: The hitting time of multiple random walks. SIAM J Matrix Anal **37**(3) (2016) 933–954
26. Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., et al.: ARGoS: a modular, multi-engine simulator for heterogeneous swarm robotics. In: IEEE/RSJ Intel Conf Intelligent Robots and Systems, IEEE (2011) 5027–5034