Robotic Harvesting of a Moving Swarm Represented by a Markov Process

Shriya Bhatnagar† Steban Soto† Javier Garcia Aaron T. Becker

***Abstract*— This paper investigates motion planning for one or more robot(s) that attempt to harvest agents from a moving swarm. Generating motion paths that maximize the number of agents harvested differs from many traditional coverage problems because the agents move. This movement allows previously cleared areas to become recontaminated. We assume that the swarm agents prefer certain regions over others, and that we can represent the swarm by a Markov Process that encodes the agents' preferred regions and their speed of motion. We exploit this model to design and simulate robotic coverage paths that maximize the number of agents harvested by a fleet of robots in a given time budget.**

I. Introduction

Canonical robotic coverage seeks to navigate a robot such that the robot's *coverage footprint* passes over every point in the workspace. Coverage tasks have received significant attention from the robotics community with applications from search and rescue to painting [1]. Coverage problems with dynamic workspaces, such as having moving targets, changing or unknown environments, or coverage uncertainty are more challenging than variants with static workspaces and deterministic actions. This paper focuses on a form of time-varying coverage where a large population of moving agents are distributed in the workspace, and the robot harvests a fraction of all the agents within its coverage footprint. Furthermore, we assume that the agents' movements can be represented by a spatially discretized Markov model that is specified by a stationary distribution and a scalar diffusion coefficient. The model chosen moves the agents probabilistically, positioning this problem, as illustrated in Table I, between periodic coverage problems and pursuit-evasion problems.

This model enables encoding preferences swarm agents might have for certain locations. The coverage goal is to maximize the number of agents harvested in a given time budget. Areas previously covered by the robot may be recontaminated, as illustrated in Fig. 1.

This research has much in common with visibility-based coverage tasks. In these tasks, robots clear adversaries within their coverage footprint, and attempt to construct a series of movements that prevent adversaries from being able to enter previously cleared regions while enlarging the cleared area [7]–[10]. Such problems assume that adversaries are infinitely clever, so planners attempt to design a solution that guarantees coverage, or returns failure. In contrast, this paper assumes adversary motion is predictable in the aggregate, and represents adversaries by a Markov motion model. This paper is also related to the work of Pimenta et al., who devised a controller that minimizes the time required for coverage of an environment while also tracking moving targets [13]. Because the targets are moving, their coverage solution requires constant adjustment. Similarly, our coverage robots must continue to move due to the changing environment.

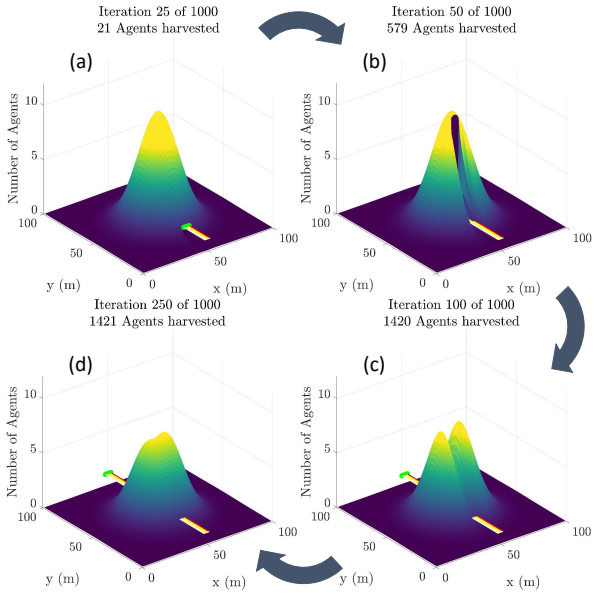

Fig. 1. Three successive snapshots of harvesting a moving swarm. Robots (represented as green points) pass through an agent distribution (represented by a density plot). Although many agents are harvested, the swarm reforms and contaminates the cleared regions.

Previous work in path planning for persistent sensing resulted in algorithms that allow a robot to modify a path by shifting a set of waypoints to focus on dynamic areas of interest [5]. The controllers are custom designed to find a locally optimal path given an unknown environment. While the current population of agents in each workspace cell corresponds to the areas of interest in [5], this paper compares a variety of controllers that either follow a consistent path or plan using greedy or heuristic policies.

Other previous research on coverage under uncertainty provided a "probably approximately correct" measure of coverage. This enabled a robot to generate policies that guarantee (to an arbitrary level of certainty) the coverage of a fraction of the free space [14]. Similarly, in this work, agent populations obey dynamics with large variance and we wish to design policies that reliably harvest the agents.

Optimization problems related to this research include

*This work was supported by the National Science Foundation under Grant No. [IIS-1553063] and [IIS-1619278].

† These authors contributed equally to this paper. Authors are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204 USA atbecker@uh.edu.

Time-Varying Coverage Problems		
Item count changes but no cell-to-cell movement	**Probabilistic motion models for many agents**	**Pathological cases–intelligent agents**
Lawn mowing/vacuuming [1], [2]	Killing mosquitoes/larvae [3]	Art gallery problem [4]
Persistent sensing [5], [6]	Commercial fishing/hunting	Pursuit/evasion problems [7]–[10]
Data ferry/sensor recharge [5]	Multiple pesticides applications [11], [12]	Coverage and tracking [13]

TABLE I

SPECTRUM OF ROBOTIC COVERAGE WITH TIME-VARYING CHARACTERISTICS. THIS PAPER FOCUSES ON THE MIDDLE COLUMN.

the traveling salesman problem (TSP) and the art gallery problem [4]. Both problems are proven to be nondeterministic polynomial-time hard (NP-hard) and typically rely on heuristics for a complete, but not optimal solution. This paper explores using heuristic-based algorithms to cover a swarm of moving agents.

This work is inspired by motion planning challenges involved in mosquito control. Coverage of mosquitoes is difficult because movement of both larva and adults causes recontamination of previously cleared regions. However, mosquito motion is at least partially predictable in the aggregate. Mosquitos exhibit preferences for certain regions. Recent research on mosquito control methods that incorporate robotics include the use of Unmanned Aerial Vehicles (UAV) with chemical larvicides [15], UAVs with controlled release of sterile adult male mosquitoes [16], and experiments in trajectory-planning for a UAV equipped with a mosquito-zapping electric screen [17]. Past work in mapping of mosquito larvae exploit satellite imagery and vision processing techniques to locate where mosquito larvae are most likely to be found [18].

Another application for this research is environmental tracking in marine environments. Tokekar et al., designed a robotic boat capable of covering a region to look for tagged invasive fish [19]. They introduced a coverage problem that requires a robot to cover only specific regions of a lake by assuming that fish have preferred regions over others. They assumed that these preferred regions were distinct and solved a TSP to minimize the time to move between the preferred regions. In contrast, we assume that agents can move probabilistically throughout the workspace, but have higher probabilities of moving toward preferred regions.

This paper is organized as follows: Markov processes for modeling are covered in Section II, and several controllers are presented in Section III. Simulation results are reported in Section IV.

II. MODELING

We assume that some agents prefer certain regions over others. In previous work, we represented these preferences by a 2D histogram where each grid cell represented the relative prevalence of agents in that cell and designed paths that maximized the number of agents harvested while respecting an energy budget [17], [20]. That model assumed agents were immobile. Representing agent mobility requires increasing the model complexity. One method to represent mobility is to directly model individual agents, as in [21], however this approach becomes unwieldy with large numbers of agents and required running a statistically significant number of trials to obtain representative data.

In this paper, we model aggregate behavior of the agent population as a whole. This eliminates the need for running many trials for each situation. We can then generalize the behavior of the agents' movement by representing it as a Markov process.

Algorithm 1 Markov Transition Matrix $P(w,k)$

w is the stationary distribution ($w_{i,j} \geq 0$, $\sum w = 1$) and $k \in [0, 1/4]$ is the dispersion parameter (fraction of swarm that leaves a cell in any direction).

$(\ell_w, \ell_h) \leftarrow \text{size}(w)$
$P \leftarrow$ zero matrix that is $(\ell_w \ell_h) \times (\ell_w \ell_h)$
for $r = 1$ to ℓ_h **do**
  for $c = 1$ to ℓ_w **do**
    $x \leftarrow \ell_w(c-1) + r$ ▷ current cell
    $e \leftarrow \ell_w c + r$ ▷ cell to east
    $s \leftarrow \ell_w(c-1) + r + 1$ ▷ cell to south
    if $c < \ell_w$ **then**

$$\begin{bmatrix} P_{e,x} \\ P_{x,e} \end{bmatrix} \leftarrow \begin{cases} \begin{bmatrix} k \\ k\frac{w(r,c)}{w(r,c+1)} \end{bmatrix} & w(r,c) < w(r,c+1) \\ \begin{bmatrix} k\frac{w(r,c+1)}{w(r,c)} \\ k \end{bmatrix} & \text{else} \end{cases}$$

    if $r < \ell_h$ **then**

$$\begin{bmatrix} P_{s,x} \\ P_{x,s} \end{bmatrix} \leftarrow \begin{cases} \begin{bmatrix} k \\ k\frac{w(r,c)}{w(r+1,c)} \end{bmatrix} & w(r,c) < w(r+1,c) \\ \begin{bmatrix} k\frac{w(r+1,c)}{w(r,c)} \\ k \end{bmatrix} & \text{else} \end{cases}$$

for $i = 1$ to $\ell_w \ell_h$ **do**
  $P_{i,i} = 1 - \sum_{j=1}^{\ell_w \ell_h} P_{i,j}$ ▷ columns sum to 1
return P

A. Uncontrolled System

This paper assumes agents are distributed in a 2D grid of size $\ell_w \times \ell_h$ and that the system state is represented at time t by $\mathbf{x}(t)$. $\mathbf{x}(t)$ is a 1D vector of ($\mathbf{x}(t) \in \mathbb{R}^{\ell_w \ell_h}$) that represents the agent population in each grid cell at time t. To propagate the moving swarm one time step, we use

$$\mathbf{x}(t+1) = P\mathbf{x}(t). \tag{1}$$

The system evolution matrix $P \in \mathbb{R}^{\ell_w \ell_h \times \ell_w \ell_h}$ assumes interaction only between grid cells that share edges, and is parameterized by the diffusion rate k. The transition matrix P is determined entirely by the stationary distribution w and the diffusion rate k, and is computed once in $\mathrm{O}(w^2)$ time by Alg. 1. If $\ell_w = \ell_h = \ell$, this produces a sparse $\ell^2 \times \ell^2$ matrix $P_{x,y}$, with only $\ell(5\ell - 4)$ non-zero entries. If all the w values are greater than zero and $k > 0$, the resulting Markov chain is aperiodic and irreducible. Disturbed distributions heal and return to their stationary distribution after some time. The evolution of such a system is shown in Fig. 1.

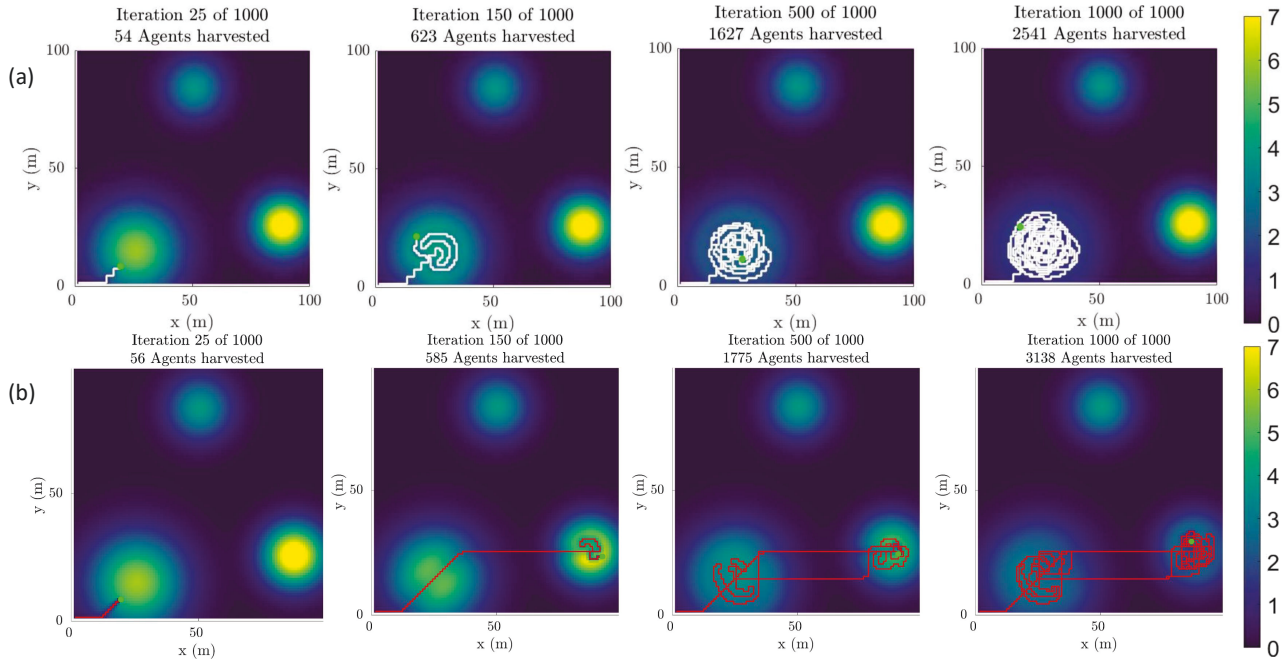

Fig. 2. Sequence of simulations showing the agent distribution as a density plot and the path of the robot in red. The agents' position evolves according to a Markov process, but the robot harvests $f_{\text{kill}} = 1$ fraction of the agents in the cell it is covering and the agents diffuse with parameter $k = 0.005$. (a) Greedy Controller (b) h-step horizen heuristic controller See simulation video at `https://youtu.be/u1OTBK5kq70`.

B. Controlled System

We assume that the robot's coverage footprint is the size of one grid cell, that all n_r robots are in different cells, and that each robot harvests $f_{\text{kill}} \in [0, 1]$ of the agents in its grid cell. f_{kill} is a parameter that represents the efficiency of the harvester. The following algorithm propagates the moving swarm one time step, under the actions of n_r robots:

$$
\begin{aligned}
&1: \ \mathbf{x}' \leftarrow \mathbf{x}(t) \\
&2: \ \textbf{for} \ \ j = 1 \text{ to } n_r \ \textbf{do} \\
&3: \qquad i = \text{position of robot } j \\
&4: \qquad \mathbf{x}'_i = (1 - f_{\text{kill}})\mathbf{x}'_i \\
&5: \ \mathbf{x}(t+1) = P\mathbf{x}'.
\end{aligned}
\tag{2}
$$

III. Controllers

This section compares several controllers used for single and multi-robot applications.

A. Controllers for single robot applications

The *boustrophedon* controller steers the robot back and forth from west to east, moving one row upwards each time it reaches a boundary. If the agents are evenly distributed and cannot move, this controller harvests the most agents since it covers the whole region without overlapping. When these assumptions are violated, it performs poorly.

The *random coverage* controller randomly commands the robot to move one grid cell east, south, west, or north. If this move is blocked by the boundary, a different direction is chosen. Since the random controller does not use any sensor measurements besides boundary detection, it performs poorly, but it does not get stuck in local minimums.

In contrast, a *greedy* controller with 1-step lookahead compares the utility of moving one grid cell east, south, west, north, or for staying still. The movement that results in the largest number of agents harvested is selected. This type of controller is an exploitation strategy that does little exploration. Similarly, a greedy controller with d-step lookahead computes the number of expected agents harvested by all possible 5^d movement sequences, and implements the first step of the sequence that harvests the largest number of agents.

All these controllers have a limited ability to predict into the future. In the simulation of Fig. 2.A, the Gaussian peaks are at least 70 units apart. Due to the curse of dimensionality, the brute-force greedy controller cannot predict far enough into the future to make crossing between the Gaussian hills competitive with staying in the initial distribution ($5^{70} \approx 8 \times 10^{48}$).

To address this, our final controller is the *h-step horizon heuristic*, which switches between exploration and exploitation strategies by simulating a small number of deep searches, as shown in Fig. 2.B. This controller is similar to model predictive control because each time step it simulates the expected number of agents harvested h steps into the future by following different strategies. Each strategy is simulated, and the strategy that harvest the most agents is selected. All strategies start by setting a goal destination. If the simulated robot reaches the destination within h steps, the robot uses its remaining time by obeying a d-step greedy controller. The null policy sets the current position as the goal destination, so the robot obeys a d-step greedy policy for all h iterations. The implementation in this paper compares the null policy and sending the robot to each of the Gaussian peaks. If there are α peaks, $\alpha + 1$ policies are compared. While moving to a destination, the robot makes locally

optimal choices if the movement requires both horizontal and vertical movements, choosing the option that harvests more agents.

There is a trade-off when choosing h because at each time step, the heuristic only simulates one switch of goal destination. If the horizon h is too short, the robot will not harvest many agents from a distant peak (and may not even reach the destination). If h is too long, the simulated robot will overexploit the destination. There is also a computation trade-off. To calculate h steps into the future, each of the h steps compares 5^d options and selects the best, and then (2) is evaluated which is dominated by the matrix multiplication $P\mathbf{x}$. The total computation is roughly $(\alpha+1)\cdot h\cdot 5^d$ multiplied by the cost of the matrix multiplication $P\mathbf{x}$. In practice we use $d = 1$ to make the simulation fast.

B. Controllers for multi-robot applications

The *forest boustrophedon* [22] controller moves the robots to starting positions that evenly divide the workspace, and then robots perform boustrophedon paths. This is shown in Fig. 3(A).

The *formation boustrophedon* controller performs coverage paths in echelon form with the robots moving in a diagonal formation. This can be seen in Fig. 3(B).

Both controllers in Fig. 3(A) and Fig. 3(B) do not use any sensor measurements. Adding sensing can improve performance. The *greedy depth* controller in Sec. III-A can be implemented on multiple robots to achieve higher amounts of harvesting. However, Fig. 3(C) shows an instance where the robots fail to discover a distribution peak in the north. They could be more effective if they were spread out based on the initial agent density map.

To implement such a controller, goal destinations are introduced to the greedy depth controller. We build a mixture of Gaussians model that is fit to $x(0)$ using an expectation maximization (EM) algorithm. This expectation maximization identifies τ_i, the variables that encode the relative probabilities of each Gaussian. The robots are distributed so that τ_i robots are sent to each Gaussian peak. Once the robots are positioned, they continue to perform their regular greedy algorithm. These initial goal positions can send robots to distant peaks that the regular greedy depth controller may be unable to reach. A resulting path is shown in Fig. 3(D). In this simulation, two robots are sent to the Gaussian peak at the top of the map that was missed by the greedy depth controller.

To combat the exponential cost of simulating n_r robots d steps into the future, our approach instead uses a priority-based system, where the $(i+1)^{\text{th}}$ robot computes its controller after simulating the harvesting performed by robots 1 to i.

IV. Simulation

For most simulations, we used a stationary distribution of agents described as the sum of three Gaussian distributions on an $\ell \times \ell$ grid with $\ell = 100$ and used World 1, defined with

- 1/2 of the agents distributed with mean $\ell(1/7, 1/4)$ and $(\sigma_x, \sigma_y) = \ell(1/8, 1/8)$,
- 1/3 of the agents distributed with mean $\ell(1/4, 7/8)$ and $(\sigma_x, \sigma_y) = \ell(1/12, 1/12)$, and
- 1/6 of the agents distributed with mean $\ell(1/6, 1/2)$ and $(\sigma_x, \sigma_y) = \ell(1/12, 1/12)$.

The robot was initialized in the southwest corner of the workspace, and the percentage of population harvested when a robot visits a cell was set to 100% ($f_{\text{kill}} = 1$). All simulations were performed in MATLAB with open source code[1].

a) Comparing controllers: Our first set of simulations compare six controllers with the same agent diffusion parameter, $k = 0.05$. The amount of harvested agents as a function of simulation steps are shown in Fig. 4(a). As expected, the random strategy performed the worst with 874 agents harvested. The boustrophedon pattern performed only slightly better, with 1103 agents harvested. The greedy controller exploits local information and kills approximately twice as many as boustrophedon. However, the greedy controller can not explore further than d steps ahead. Due to local maximas, having a deeper lookahead function did not always translate into a higher number of harvested agents. For the d-step greedy strategy, $d = \{1, 3, 5\}$ harvested $\{2583, 2583, 2580\}$ agents.

In contrast, the heuristic controller balances exploration and exploitation. This enables the heuristic controller to eventually outperform the greedy controller. However, due to the time spent on moving toward goal destinations, the greedy controller often outperforms the heuristic during the early stages of the simulation. Each time the heuristic controller travels to a new maxima, the harvesting rate flattens, which shows up as the five plateaus in Fig. 4(a). The heuristic with a 200-step horizon harvested 3138 agents, even though its greedy search had only a d=1 step lookahead.

b) Varying k, the agent diffusion rate: The second set of simulation experiments compared the effect of the dispersion parameter, which is the maximum fraction of agents in a cell moving in a cardinal direction ($k \in [0, 1/4]$). The results are shown in Fig. 4(b). If $k = 0$ the agents are stationary and the problem is equivalent to traditional coverage. As k increases, the distribution returns to the stationary distribution more quickly. With one notable exception, performance was inversely proportional to k. For the largest, $k = 0.25$, only 2478 agents were harvested. Diffusions of $k = \{0.1, 0.01, 0.005\}$ harvested $\{2495, 2689, 2801\}$ agents. However, for $k = 0$ only 702 agents were harvested, even though for the first 142 steps this strategy performed better than all others. The system reached a configuration where all the agents in every direction had been harvested and the limited 1-step lookahead was unable to plan a trajectory.

c) Comparing controllers for multiple robots: The third experiment compared four different controllers with multiple robots. For each controller the number of robots is $n_r = 10$, with $k = 0.05$. The amount of harvested agents as a

[1]Simulation: github.com/RoboticSwarmControl/2018mosquitoCoverage

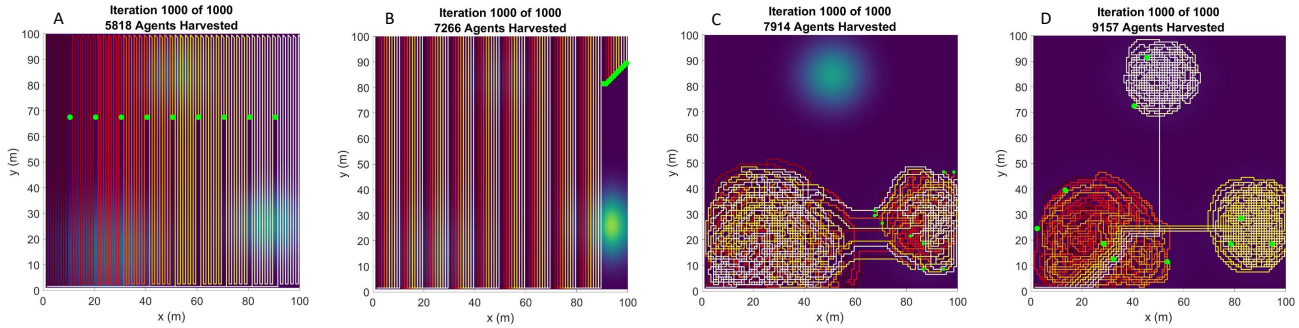

Fig. 3. Simulations using different multi-robot coverage algorithms with $n_r = 9$ robots and $N = 10,000$ agents over 1,000 simulation steps, with a $d = 3$ depth greedy search, $k = 0.05$, and $n_r = 10$. (A) Forest coverage boustrophedon path. (B) Formation boustrophedon path. (C) Greedy-Depth implementation. (D) Greedy-Depth search with initial goal destinations assigned based on population density.

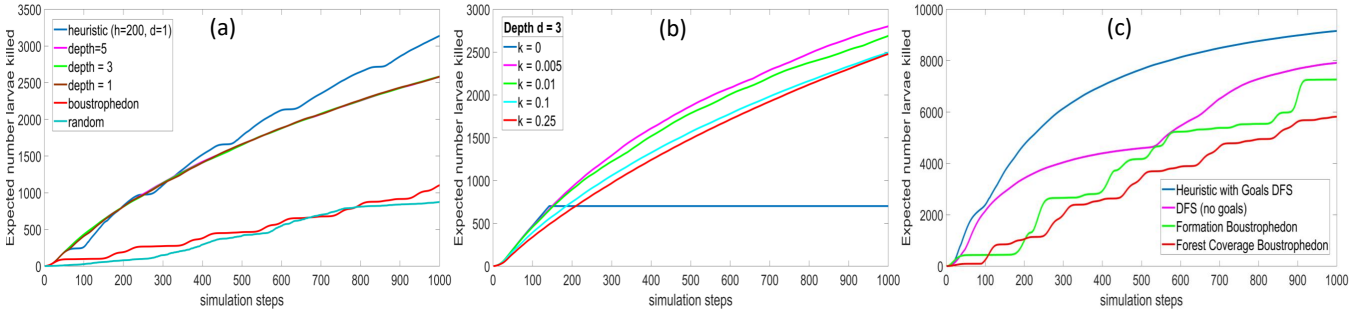

Fig. 4. (a) Simulations comparing the controllers presented in Section III with $n_r = 1$ robot, $N = 10,000$ agents, 1,000 simulation steps, and diffusion parameter $k = 0.05$. (b) Simulations varying agent dispersion parameter k, as described in Section IV, with $n_r = 1$ robot, $N = 10,000$ agents over 1,000 simulation steps, with a $d = 3$ depth greedy search. (c) Simulations varying the controllers as described in Section III, with $N = 10,000$ agents, 1,000 simulation steps, a $d = 3$ depth greedy search, $k = 0.05$, and $n_r = 10$. See simulation video.

function of simulation steps are shown in Fig. 4(c). The forest boustrophedon coverage performed the worst, with 5815 agents harvested. It had a spike in agents harvested at 125 steps, and briefly outperformed formation boustrophedon coverage. The formation boustrophedon coverage performed better, harvesting 7266 agents. The greedy depth search without goal destinations for n_r robots captured 7914 agents. The best performance, harvesting 9157, was earned by the $d = 3$ greedy depth search with robots assigned to goal destinations based on the agent population density.

d) Varying f_{kill} in two different worlds: The fourth experiment compared two different worlds while also varying the f_{kill} parameter from 0.1 to 1, with an increment of 0.1 between each run. World 1 is the same world used in all the other simulations. World 2 is a stationary distribution of agents described as the sum of four Gaussians with the following parameters: $\ell = 100$, with

- 1/6 of the agents distributed with mean $\ell(1, 1/2)$ and $(\sigma_x, \sigma_y) = \ell(1/8, 1/8)$,
- 1/3 of the agents distributed with mean $\ell(2/5, 8/9)$ and $(\sigma_x, \sigma_y) = \ell(1/12, 1/12)$,
- 1/6 of the agents distributed with mean $\ell(1/2, 1/2)$ and $(\sigma_x, \sigma_y) = \ell(1/12, 1/12)$,
- 1/3 of the agents distributed with mean $\ell(1/5, 1/10)$ and $(\sigma_x, \sigma_y) = \ell(1/8, 1/8)$.

In this experiment, the efficacy of the robots is put the test. When f_{kill} is set to 0 the robots completely fail at harvesting the agents. When f_{kill} is set to 1, the robots harvest all agents present in their cell. In Fig. 5.A, increasing f_{kill} increases the amount of agents harvested. After increasing f_{kill} above 0.7, the marginal utility decreases. However, in Fig. 5.B, in World 2, increasing f_{kill} did not always increase the amount of agents harvested. This could be the effect of resource starvation, where after a certain amount of simulation steps robots are not sent back to areas with recontaminated areas. This could be avoided by implementing the h-step horizon heuristic controller for multi-robot applications.

e) Marginal utility of additional robots: The fifth experiment compared the marginal effect of additional robots. The robots were programmed to follow a $d = 3$ depth greedy search once they had reached their assigned initial goal destination, with $k = 0.05$ and $n_r \in [1, 30]$. Robots beyond n_r = 10 have low impact on the number of agents harvested. Figure 6 fits this data with an exponential decay.

V. Conclusions

This paper presented an alternate coverage problem where the objects to be covered are moving agents that obey a Markov motion model and have a stationary distribution. We presented heuristic and greedy controllers that outperform standard coverage techniques for this problem. This research was motivated by current challenges in mosquito larvae control, but may have applications to commercial fishing, pesticide treatments, or steering a predator to maximize the number of prey harvested when the predator has a limited coverage footprint and the prey obey a Markov motion model. For future work, we plan on expanding the h-step horizon heuristic controller to multi-robot applications,

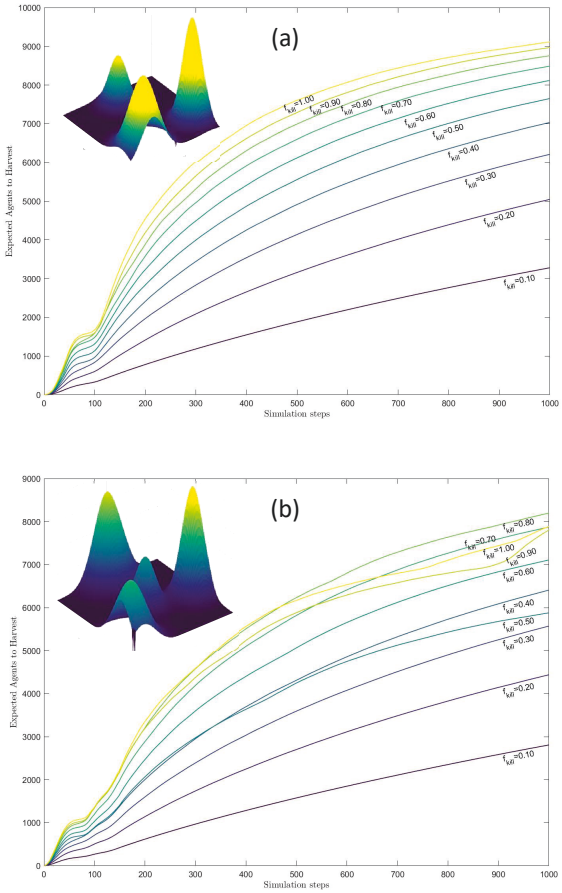

Fig. 5. Simulation parameters: $N = 10,000$ agents, 1,000 simulation steps, running Heuristic with Goals DFS, $k = 0.05$, and $n_r = 10$ (a) Simulation comparing the number of agents harvested as the kill rate parameter varies, $f_{\text{kill}} \in [0.1, 1]$, run in World 1. (b) Simulation comparing the number of agents harvested as the kill rate parameter varies, $f_{\text{kill}} \in [0.1, 1]$, run in World 2.

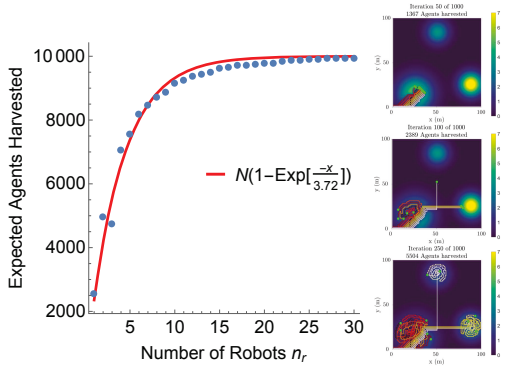

Fig. 6. Simulations varying the number of robots with $N = 10,000$ agents, 1000 steps, $d = 3$ depth greedy search, and diffusion parameter $k = 0.05$. Subplots show the heat map of the agents, and the robot paths (in multi-colors) after 50, 100, and 250 steps.

where every n number of simulations steps, it re-evaluates the expected return for redistributing the robots across the world. We also plan to enable the controller to avoid over-exploitation by choosing the amount of time it spends at a goal. Finally, we plan on verifying the results seen in the simulations through hardware experiments.

References

[1] Howie Choset. Coverage for robotics – a survey of recent results. *Ann Math Artif Intell*, 31(1):113–126, Oct 2001.

[2] J Colegrave and A Branch. A case study of autonomous household vacuum cleaner. *AIAA/NASA CIRFFSS*, 107, 1994.

[3] Sachit Butail, Nicholas Manoukis, Moussa Diallo, José M Ribeiro, Tovi Lehmann, and Derek A Paley. Reconstructing the flight kinematics of swarming and mating in wild mosquitoes. *Journal of The Royal Society Interface*, 9(75):2624–2638, 2012.

[4] D. Lee and A. Lin. Computational complexity of art gallery problems. *IEEE Trans Inf Theory*, 32(2):276–282, March 1986.

[5] D. E. Soltero, M. Schwager, and D. Rus. Generating informative paths for persistent sensing in unknown environments. In *IROS*, pages 2172–2179, Oct 2012.

[6] Stephen L Smith, Mac Schwager, and Daniela Rus. Persistent robotic tasks: Monitoring and sweeping in changing environments. *IEEE Transactions on Robotics*, 28(2):410–426, 2012.

[7] Timothy H. Chung, Geoffrey A. Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous Robots*, 31(4):299, Jul 2011.

[8] Andreas Kolling and Alexander Kleiner. Multi-UAV motion planning for guaranteed search. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 79–86. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[9] Alexander Kleiner and Andreas Kolling. Guaranteed search with large teams of unmanned aerial vehicles. In *ICRA*, pages 2977–2983. IEEE, 2013.

[10] Nicholas M Stiffler and Jason M O'Kane. Complete and optimal visibility-based pursuit-evasion. *The International Journal of Robotics Research*, 36(8):923–946, 2017.

[11] Philip J Sammons, Tomonari Furukawa, and Andrew Bulgin. Autonomous pesticide spraying robot for use in a greenhouse. In *Australian Conference on Robotics and Automation*, pages 1–9, 2005.

[12] Kyukwang Kim, Hwijoon Lim, Whimin Kim, Duckyu Choi, Sungwook Jung, and Hyun Myung. Collaborative UAV type robotic system for mosquito habitat puddle searching and larvicide spray. In *IROS*. IEEE, 2016.

[13] Luciano CA Pimenta, Mac Schwager, Quentin Lindsey, Vijay Kumar, Daniela Rus, Renato C Mesquita, and Guilherme AS Pereira. Simultaneous coverage and tracking (SCAT) of moving targets with robot networks. In *Algorithmic foundation of robotics VIII*, pages 85–99. Springer, 2009.

[14] Colin Das, Aaron Becker, and Timothy Bretl. Probably approximately correct coverage for robots with uncertainty. In *IROS*, pages 1160–1166. IEEE, 2011.

[15] John-Thones Amenyo, Daniel Phelps, Olajide Oladipo, Ekuoe Sewovoe, Sangeeta Jadoonanan, Sandeep Jadoonan, Tahseen Tabassum, Salim Gnabode, Taging D Sherpa, Michael Falzone, Abrar Hossain, and Aerren Kublal. Ultra-low cost, low-altitude, affordable and sustainable UAV multicopter drones for mosquito vector control in malaria disease management. In *IEEE Global Humanitarian Technology Conference*, pages 590–596. IEEE, 2014.

[16] Evan Ackerman. Drones distribute swarms of sterile mosquitoes to stop zika and other diseases. *IEEE Spectrum*, 2017.

[17] An Nguyen, Dominik Krupke, Mary Burbage, Shriya Bhatnagar, Sándor P Fekete, and Aaron T. Becker. Using a UAV for destructive surveys of mosquito population. In *ICRA*. IEEE, May 2018.

[18] Li Zou, Scott N. Miller, and Edward T. Schmidtmann. Mosquito larval habitat mapping using remote sensing and GIS: Implications of coalbed methane development and west nile virus. *Journal of Medical Entomology*, 43(5):1034–1041, 2006.

[19] P. Tokekar, E. Branson, J. Vander Hook, and V. Isler. Tracking aquatic invaders: Autonomous robots for monitoring invasive fish. *IEEE Robotics Automation Magazine*, 20(3):33–41, Sep. 2013.

[20] Aaron T Becker, Mustapha Debboun, Sándor P Fekete, Dominik Krupke, and An Nguyen. Zapping zika with a mosquito-managing drone: Computing optimal flight patterns with minimum turn cost. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 77. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[21] Mary Burbage. Maximizing swarm coverage: Hunting for members of a moving population. Master's thesis, University of Houston, Houston, TX, May 2017.

[22] Xiaoming Zheng, Sonal Jain, Sven Koenig, and David Kempe. Multi-robot forest coverage. In *IROS*, pages 3852–3857. IEEE, 2005.