Application of Game-Theoretic Methods for Multi-Robot Coordination

Samet Güler and Jeff S. Shamma

Abstract— **One approach to multi-robot coordination is to represent each robot as a player in a game. A system planner designs both the incentives (utility function) for the robot and the learning rule that specifies how the robot adapts to its environment, namely the behaviors of other robots. In an ideal setting, the game design can be such that a desired collective behavior can emerge through local interactions. While such approaches come with supporting theoretical guarantees, the analytical setting is typically an idealization of a more realistic environment. This paper investigates how such game theoretic algorithms can be implemented in non-idealized settings in order to enable application to real-time distributed multi-robot coordination.**

I. Introduction

Game theory has been used to envision interactions between strategic decision making entities. Besides its conventional descriptive role, game theory has also been utilized as a prescriptive method for the design and analysis of multi-agent systems where the agents typically represent programmable machines rather than humans. The main goal in this thrust has been to design resilient networks with desired asymptotic behaviors.

Game theory methods have been applied to several network problems, including wireless communications, sensor coverage, task assignment, and smart grids. The majority of the past work on game theory applications formulated the problems as myopic adjustment processes and focused on proving theoretical guarantees for "ideal" agents with perfect sensing, communication, and computation capabilities. For instance, the authors in [1], [2] studied the coverage problem for agents that do not possess any uncertainty in their sensing and actuation mechanisms. Accordingly, the promising functionalities of the game theory methods have been demonstrated on ideal application scenarios with simulations.

Game theory methods offer significant advantages for multi-robot applications. The learning algorithms such as log-linear learning (LLL) and Metropolis-Hastings (MH) algorithms are designed in such a way that each entity makes decisions based on its knowledge about the environment. Remarkably, this structure leads to entirely distributed algorithms and can be directly encoded in robots. Furthermore, the distributed learning algorithms are computationally efficient, which is a desirable feature for small-size aerial robots with strict requirements on the onboard hardware.

Although game theory remains a powerful tool in the design and analysis of multi-robot systems, it poses several challenges when it comes to real-time implementation. When transitioning from theory to implementation, several factors need to be considered to take full advantage of these algorithms. We list three such factors as follows.

Incomplete sensory information. Robotic sensing mechanisms provide the perception of only a limited portion of the environment through uncertain measurements. Accordingly, a robot cannot acquire the complete sensory information necessary for utility calculation, which entails to having either a centralized multi-robot system or extensive data exchange between robots.

Asynchrony. Most game theoretical learning algorithms imply that the individual robots make decisions periodically at specified time instants of a common clock. However, such a synchronization may not be guaranteed in multi-robot systems.

Sensing and actuation uncertainty. Due to their uncertain characteristics, sensor measurements are usually represented by probabilistic models. For instance, most commercial GPS sensors provide position data of a drone with up to three-meter error. Similarly, actuators execute motion commands with uncertainty.

In this paper, we focus on the objective of aerial coverage by multiple drones. A truly distributed multi-robot coverage implementation requires to utilize the robots' onboard capabilities solely. Therefore, we investigate whether game theory methods are implementable on real-time multi-robot systems which exploit their onboard capabilities solely in a distributed manner. We propose remedies for the issues above and analyze the performance of standard learning algorithms in a realistic simulation environment.

Although various works have studied the coverage problem recently, the literature lacks demonstration of an entirely distributed multi-robot coverage application. References [3], [4] posed the coverage problem as a state-based potential game with a utility function conditioned on the balance between robots' consumed energy and total covered area. The authors in [5] considered the case of unknown sensor locations and formulated the problem as a game. However, none of these works addressed the implementation details listed above. On the other hand, several multi-robot coverage applications were performed with heuristic approaches to find optimal routes for maximum coverage [6], [7]. The indoor implementations in [8], [9] took full advantage of a motion capture system from which the robots acquired the real-time positions; however, such a positioning system does not have an equivalent outdoors. We explore multi-drone coverage frameworks that can operate both indoors and outdoors.

Samet Güler is with Abdullah Gül University, Kayseri, Turkey. Jeff S. Shamma is with King Abdullah University of Science and Technology, Thuwal, Saudi Arabia. samet.guler@kaust.edu.sa

II. Problem Formulation

We consider the objective of covering a bounded area by multiple drones with onboard vision capabilities. We define the mission space as a finite set of sectors $\mathcal{Q} \subset \mathbb{R}^2$. Some regions in $\mathcal{Q}$ may include more features of interest, or have higher density, than the rest of the area. Let $V(q,t) : \mathcal{Q} \times [0,\infty) \to \mathbb{R}_+$ denote the unknown *value function* representing the density of the sensory information at location $q \in \mathcal{Q}$ at time t, satisfying

$$V(q,t) \geq 0 \ \forall q \in \mathcal{Q}, \quad \sum_{q \in \mathcal{Q}} V(q,t) = 1. \tag{1}$$

Consider also a set of N drones with positions $x_i \in \mathbb{R}^3$, $i \in \{1,\dots,N\}$ and with the following specifications:

S1. *Action.* The drone can move to given 3D coordinates.
S2. *Computation.* The drone performs all computation on an onboard device, independent of a ground entity.
S3. *Communication.* The drone can transmit/receive position data to/from other robots in its communication neighborhood.
S4. *Perception.* The drone has a downward facing camera.

We assume that all drones fly at a constant altitude h, i.e., $x_i = \left[\bar{x}_i^\top, h\right]^\top$ where $\bar{x}_i \in \mathbb{R}^2$ is the Cartesian coordinates of drone i. Let $X = \left[\bar{x}_1^\top, \dots, \bar{x}_N^\top\right]^\top \in \mathbb{R}^{2N}$, and denote the total area perceived by the cameras at time t by $\mathcal{C}(t) \in \mathbb{R}^2$. We define the coverage function as

$$\phi(X(t)) = \sum_{q \in \mathcal{Q}} f(X(t),q) V(q,t), \tag{2}$$

where $f(X(t),q) = 1$ if q is covered by $\mathcal{C}(t)$ at time t and $f(X(t),q) = 0$ otherwise. We define the aerial coverage objective as follows.

Problem 1: Consider a team of N drones with specifications S1-S4. Design a motion control algorithm for the drones so that $\phi(X(t))$ is maximized for all $t \in [0,\infty)$.

III. Aerial Coverage Game

We formulate Problem 1 as a potential game, propose distributed learning algorithms to meet the objective, and discuss the real-time implementation issues. The design process consists of two steps: (i) Game formulation, i.e., determining players, actions, and utilities; and (ii) choosing the learning algorithm and the utility calculation method.

Let $\mathcal{G}_{\text{cov}} = \{\mathcal{P}, \mathcal{A}, \mathcal{U}\}$ denote the distributed coverage game with the player set $\mathcal{P}$, the action set $\mathcal{A}$, and the utility set $\mathcal{U}$. The drones constitute the player set $\mathcal{P}$. We set the Cartesian coordinates of a robot as its action, i.e., $a_i(t) = \bar{x}_i(t)$. Since a drone cannot reach any location in $\mathcal{Q}$ at a given time step, we adapt the *constrained action sets* introduced in [1], [2] to define $\mathcal{A}$. Particularly, the action set of a player $\mathcal{P}_i$ at time t is comprised of nine possible motion primitives around its previous action $a_i(t-1)$, defined by the set

$$\mathcal{S}_i = \{\text{UL}, \text{U}, \text{UR}, \text{L}, \text{S}, \text{R}, \text{BL}, \text{B}, \text{BR}\}, \tag{3}$$

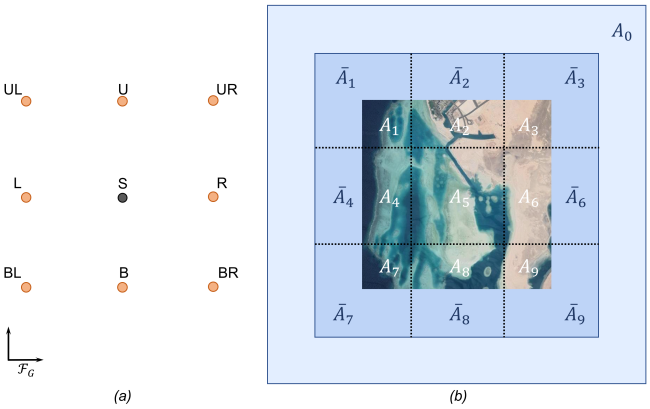


Fig. 1: (a) A drone's nine possible motion primitives with respect to a global frame $\mathcal{F}_\text{G}$. (b) Cell partitions of a drone's camera view. The inner image represents the original camera view and is divided into nine partitions (A_i). The cells shaded with dark blue ($\bar{A}_i$) denote the "predicted image". The light blue region (A_0) denotes the "optimism image".

where U, B, L, R, and S denoting up, bottom, left, right, and stay, respectively, with respect to an arbitrary coordinate frame (Fig. 1a). In other words, a robot is allowed either to move to its immediate neighbor locations or to stay at its current location.

We design the utilities based on sensory information retrieved from the drones' onboard cameras. The utility-based learning algorithms require calculating the utilities for the neighbor locations corresponding to the actions in $\mathcal{S}_i$. A drone is assumed to cover its bounded neighborhood with its downward facing camera. Thus, the algorithm must "predict" the utilities of the unseen areas which lie outside of the field-of-view (FOV) of the robot's camera. Accordingly, assuming that the robots operate at a common altitude, we partition the image retrieved from robot i's camera into nine sub-areas A_i $(i = 1,\dots,9)$ (Fig. 1b). Let the density of sub-area A_i $(i = 1,\dots,9)$, calculated by a vision algorithm, be denoted by d_i. Furthermore, we assume the immediate neighbor $\bar{A}_i$ of a sub-area A_i, which robot i cannot see, has the same density with its counterpart A_i, i.e., $\bar{d}_i = d_i$ $(i = 1,\dots,9)$, $i \neq 5$ (Fig. 1b). Finally, let d_γ denote the density of the outermost area in Fig. 1b. Although robot i does not have any clue about d_γ, the utilities for the actions in the set $\mathcal{S}_i$ depend on this density. Remarkably, the design parameter d_γ determines whether a robot should be optimistic about the unseen area. Moreover, we assume that each robot is informed with the positions of its two-hop neighbor robots through a communication channel. With this setup, all robots can calculate the utilities of their candidate locations determined by the set $\mathcal{S}$ in a distributed manner, addressing the first issue in Section I.

The utility-based learning algorithms require that all robots follow a discretized time scheme. For instance, the LLL and MH algorithms impose that, at each time step, one robot is selected randomly and allowed to alter its action. However, clock synchronization in real multi-robot implementations cost additional time and effort, and a perfect synchronization may not be guaranteed. Therefore, we propose to use Algo-

Algorithm 1 Time scheme for $\mathcal{P}_i$

Require: $t_{\text{clock}}, t_{\text{i}}, \tau_{\min}, \tau_{\max}$
Ensure: a_i
1: **if** $t_{\text{clock}} \geq t_{\text{i}}$ **then**
2: Choose a_i
3: Set $t_{\text{i}} \in [t_{\text{clock}} + \tau_{\min}, t_{\text{clock}} + \tau_{\max}]$
4: **else**
5: Idle
6: **end if**

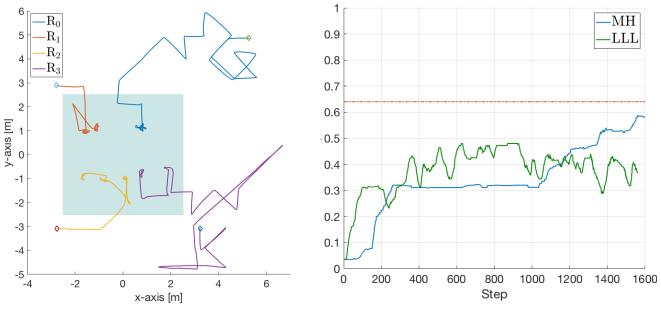
rithm 1 as the decision time scheme where each robot uses the time scheme of its computational unit to make decisions.

IV. Results

The LLL and MH algorithms assure that if all robots obey a specified time scheme of a common clock and calculate their utilities correctly, the robots spend most of the operation time in regions where ϕ in (2) is maximum [2]. In Section III, our game design included Algorithm 1 and the modified utility calculation method based on camera images. Thus, the synchronous clock and perfect sensing conditions of the original results are violated, and the asymptotic guarantees of the LLL and MH algorithms are no longer valid.

To test the performance of the LLL and MH learning algorithms with this modified framework, we simulated the multi-robot coverage game $\mathcal{G}_{\text{cov}}$ in Gazebo with Robot Operating System (ROS). The setup included four drones equipped with downward looking cameras that operated on a flat terrain part of which is set as the dense area (Fig. 2a). We used the method defined in Section III to calculate the marginal contribution utilities. Notably, the drones had separate clocks as in real-world experiments. We used $\tau_{\min} = 0.5$s, $\tau_{\max} = 1$s, and $d_\gamma = \{0, 1\}$.

Drones have highly nonlinear motion models, and their actuation mechanisms include uncertainties. Thus, drones cannot hover at a given coordinate perfectly, and steering drones on a lattice structured plane was not a suitable practice. We addressed this issue by introducing thresholds in the Cartesian coordinates such that once a drone enters a particular neighborhood of a commanded location, it is commanded to stop. Although this method prevented us from taking advantage of a lattice structured plane, we handled the actuation uncertainties efficiently.



(a) (b)

Fig. 2: (a) The drones' paths with the MH algorithm. The dense area is shown in blue, and the diamonds denote the initial locations of the drones; (b) The global welfare with the MH and LLL algorithms. The maximum achievable coverage by four drones is represented with the red dashed line.

We observed that both the LLL and MH algorithms performed well in simulations by steering the drones into the dense area (Fig. 2b). For instance, the MH algorithm test in Fig. 2a demonstrates that drones 1 and 4 reached the dense area after exploring a part of the plane even if they did not "see" the dense area at the beginning of the operation.

V. Conclusion

Game theory literature provides theoretical guarantees for several learning algorithms applied to idealized multi-agent systems. We have investigated whether modifications of the standard learning algorithms can be utilized to implement truly distributed, real-time multi-drone applications. Toward our ultimate goal of realizing game theory algorithms in real-time multi-drone experiments, we have tested the performance of two learning algorithms on a realistic simulation environment.

References

[1] J. R. Marden, G. Arslan, and J. S. Shamma, "Cooperative control and potential games," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 6, pp. 1393–1407, 2009.

[2] J. R. Marden and J. S. Shamma, "Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation," *Games and Economic Behavior*, vol. 75, no. 2, pp. 788–808, 2012.

[3] S. Rahili, J. Lu, W. Ren, and U. M. Al-Saggaf, "Distributed coverage control of mobile sensor networks in unknown environment using game theory: Algorithms and experiments," *IEEE Transactions on Mobile Computing*, vol. 17, no. 6, pp. 1303–1313, 2018.

[4] E. Paraskevas, D. Maity, and J. S. Baras, "Distributed energy-aware mobile sensor coverage: A game theoretic approach," in *2016 American Control Conference (ACC)*, 2016, pp. 6259–6264.

[5] M. Varposhti, V. Hakami, and M. Dehghan, "Distributed coverage in mobile sensor networks without location information," *Autonomous Robots*, 2019.

[6] C. Dornhege, A. Kleiner, A. Hertle, and A. Kolling, "Multirobot coverage search in three dimensions," *Journal of Field Robotics*, vol. 33, no. 4, pp. 537–558, 2016.

[7] J. Jin and L. Tang, "Coverage path planning on three-dimensional terrain for arable farming," *Journal of Field Robotics*, vol. 28, no. 3, pp. 424–440, 2011.

[8] M. Schwager, B. J. Julian, and D. Rus, "Optimal coverage for multiple hovering robots with downward facing cameras," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3515–3522.

[9] Z. Wang, R. Spica, and M. Schwager, "Game theoretic motion planning for multi-robot racing," in *Distributed Autonomous Robotic Systems*, N. Correll, M. Schwager, and M. Otte, Eds. Cham: Springer International Publishing, 2019, pp. 225–238.