Game-theoretic Utility Tree for Multi-Robot Cooperative Pursuit Strategy

Qin Yang*[a] and Ramviyas Parasuraman[a]
[a]Department of Computer Science, University of Georgia, Athens, GA, USA. *Speaker: *qy03103@uga.edu.*

Abstract

Underlying relationships among multiagent systems (MAS) in hazardous scenarios can be represented as game-theoretic models. In adversarial environments, the adversaries can be intentional or unintentional based on their needs and motivations. Agents will adopt suitable decision-making strategies to maximize their current needs and minimize their expected costs. This paper proposes and extends the new hierarchical network-based model, termed Game-theoretic Utility Tree (GUT), to arrive at a cooperative pursuit strategy to catch an evader in the Pursuit-Evasion game domain. We verify and demonstrate the performance of the proposed method using the Robotarium platform compared to the conventional constant bearing (CB) and pure pursuit (PP) strategies. The experiments demonstrated the effectiveness of the GUT, and the performances validated that the GUT could effectively organize cooperation strategies, helping the group with fewer advantages achieve higher performance.

1 Introduction

Although cooperative multiagent system (MAS)[1] decision-making is studied in many separate communities, such as evolutionary computation, complex systems, game theory, graph theory, and control theory, these problems are either be episodic or sequential [1]. Agents' actions or behavior are usually generated from a sequence of actions or policies, and the decision-making algorithms are evaluated based on policy optimality, search completeness, time complexity, and space complexity.[2]

From a game-theoretic control perspective, most of the research in game theory has been focused on single-stage games with fixed, known agent utilities [3], such as distributed control in communication [4] and task allocation [5]. Especially, recent MAS research domains focus on solving path planning problems for avoiding static or dynamical obstacles [6] and formation control [7] from the unintentional adversary perspective. For intentional adversaries, the "pursuit domain" [8, 9] primarily deals with how to guide one or a group of pursuers to catch one or a group of moving evaders [10]. Similarly, the robot soccer domain [11] deals with how one group of robots wins over another group of robots on a typical game element.

Furthermore, the *Pursuit-Evasion* problems have been studied using a wide variety of approaches and have many different instantiations that can be used to illustrate different MAS scenarios [12]. Recently, [10] presents optimal evader strategies to escape from a pursuer when the pursuers follow either a constant bearing (CB) or a pure pursuit (PP) strategy to capture the evader. Here, each evader is assigned to a set of pursuers based on the instantaneous positions of all the players. More specifically, in the case of a CB strategy, the bearing angle between a pursuer and the evader remains constant until the time of capture. But in the PP strategy, the velocity vector of the pursuer is aligned along the line of sight. From the perspective of multi-robot cooperative capture, [9] discussed an approximate solution that the pursuers minimize the safe-reachable area (the set of points where an evader can travel without being caught) of the evader. However, the literature does not address the problem of how to organize robots' group behaviors, presenting a more complex and efficient strategy to catch the evader. Therefore, we make the following contributions.

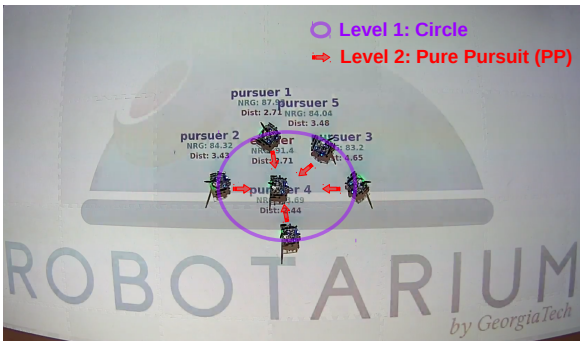


Figure 1 Illustration of five pursuers catching one evader in Pursuit-Evasion game through the game-theoretic utility tree (GUT) with circle formation (level 1) and PP tactic (level 2) in Robotarium.

Contributions

- Firstly, this paper presents a new hierarchical network-based model, termed Game-theoretic Utility

[1]In this paper, we use the terms agent and robot interchangeably.

[2]A policy is optimal if it has the highest utility/reward. A search algorithm is complete if it guarantees to return an optimal policy in a finite time when it exists. Time complexity quantifies the amount of time needed to search for a solution, while space complexity quantifies the amount of required computational memory [2].

Tree (GUT), to arrive at a cooperative pursuit strategy catching the evaders in the Pursuit-Evasion problem by creating a multi-level strategy decomposition.

- Secondly, we demonstrate the performance of the GUT in the real robot implementing the GUT algorithm in the Georgia Tech's **Robotarium** platform [13], which is an open access multi-robot testbed. We open source this implementation in GitHub[3].
- Finally, we verify the effectiveness of the GUT through simulations and real-robot experiments by comparing it to the conventional constant bearing (CB) and pure pursuit (PP) strategies. The results show that the GUT could effectively organize cooperation strategies, helping the group with fewer advantages achieve higher performance.

Fig. 1 demonstrates five pursuers catching one evader in the Pursuit-Evasion game through the game-theoretic utility tree (GUT) with circle formation (level 1) and PP tactic (level 2) in Robotarium.

2 Related Work

Recently, multiagent systems (MAS) working in adversarial environments like hazardous and disaster scenarios became an emergent topic in robotics, especially implementation in multi-robot systems (MRS). They have four main categories based on task performance: Adversarial Patrol [14]; Adversarial Coverage [6, 15]; Adversarial Formation [7]; and Adversarial Navigation [16]. For no artificial agents, they might be a natural force like wind, fire, rain, or obstacles. Many studies solve these problems by detecting opponents, path planning avoiding static or dynamical obstacles, formation control avoiding collisions, and so forth [7, 17]. More specifically, in the urban search and rescue (USAR) missions, artificial agents (like robots) need to face various unintentional adversaries, such as radiation, clutters, or natural forces such as wind and fire, adopting suitable plans and tactics to adapt it [18].

In the game-theoretic field, game-theoretic approaches are still a promising new direction to distributed control of MAS. Especially, the optimal control of MAS via game theory assumes a system-level object is given, and the utility functions for individual agents are designed to convert a multiagent system into a potential game [19]. Specially, [20] studied the dynamic non-cooperative game theory using distributed optimization, [21] investigated the MAS consensus, and [22] provided an algorithm for large-scale MAS optimization. Furthermore, [23] describes multiagent control problems using potential game theory architecture. Moreover, there are still open challenges in the area, such as designing utility functions, learning from global goals for potential game-based optimization of control systems, and converting the local optimization problem to an original optimization problem into a potential networked game [24].

[3] https://github.com/RickYang2016/Gut-Pursuit-Domain-Robotarium-ISR2022

On the other hand, cooperative decision-making among the agents is essential to address the threats posed by intentional physical adversaries or determine tradeoffs in tactical networks [25]. Current research mainly focuses on solving multiplayer *pursuit and evasion* game problem [8, 26], which primarily deals with how to guide one or a group of pursuers to catch one or a group of moving evaders. Recent works in this domain concentrate on optimal evasion strategies and task allocation [10] and predictive learning from agents' behaviors [27].

Furthermore, from the realistic and practical perspective, [28] design a new game domain called *Explore Domain*, which can analyze how to organize more complex relationships and behaviors in MAS cooperation, achieving given tasks with a higher success probability and lower costs in uncertain environments. In this domain, multiple explorer agents need to cooperatively execute assigned tasks while overcoming the adversarial opponent agents in the environment.

As we can see, organizing complex relationships through a hierarchical structure enabled by GUT will significantly help address the challenges in the multiagent pursuit-evasion problems in the literature.

3 Background and Preliminaries

This section provides essential background about *Utility Theory*, *Game Theory* and *Game-theoretic Utility Tree (GUT)*. We use the notations and the relative definitions from the corresponding papers when describing a specific method.

3.1 Utility Theory

The dominant approach to modeling an agent's interests or needs is *utility theory*. This theoretical approach aims to quantify an agent's degree of preference across a set of available alternatives and understand how these preferences change when an agent faces uncertainty about which alternative it will receive [29].

To describe the interactions between multiple utility-theoretic agents, we use the specific *utility function* to analyze their preferences and rational action. The utility function is a mapping from states of the world to real numbers, which are interpreted as measures of an agent's level of happiness (needs) [30] in the given states. If the agent is uncertain about its current state, the utility is defined as the *expected value* (Eq. (1)) of its utility function for the appropriate probability distribution over the states [29]. From the perspective of the connection between a decision-maker and its preference, a decision-maker would rather implement a more preferred alternative (act, course of action, strategy) than one that is less preferred [31].

$$\mathbb{E}[u(X)] = \sum_i u(x_i) \cdot P(x_i) \tag{1}$$

3.2 Game Theory

Game Theory is the science of strategy, which provides a theoretical framework to conceive social situations among

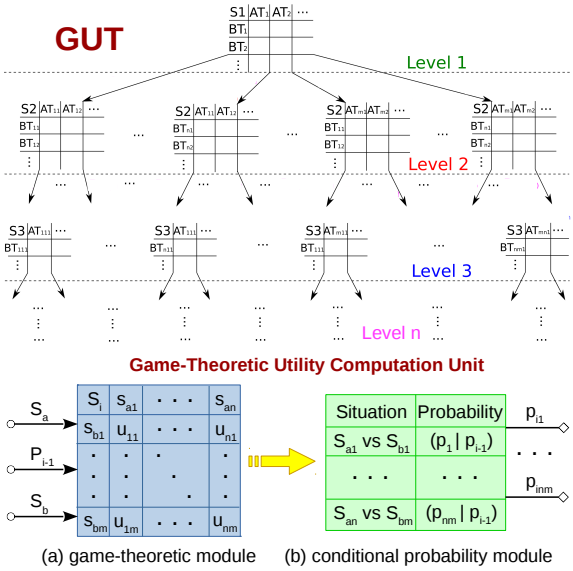


Figure 2 Structure of the Game-theoretic Utility Tree (GUT) for hierarchical decomposition of strategies.

competing players and produce optimal decision-making of independent and competing actors in a strategic setting [32]. In a non-cooperative game, players compete individually and try to raise their profits alone. Especially in the zero-sum games, their total value is constant and will not decrease or increase, which means that one player's profit is associated with another loss. In contrast, if different players form several coalitions trying to take advantage of their coalition, that game will be cooperative [33].

For the game's solutions, if players adopt a *Pure Strategy*, it will provide maximum profit or the best outcome. Therefore, it is regarded as the best strategy for every player of the game. On the other hand, in a *Mixed Strategy*, players execute different strategies with the possible outcome through a probability distribution over several actions.

Furthermore, *Nash Existence Theorem* is a theoretical framework that guarantees the existence of a set of mixed strategies for a finite, non-cooperative game of two or more players in which no player can improve its payoff by unilaterally changing strategy. It guarantees that every game has at least one Nash equilibrium [34], which means that every finite game has a *Pure Strategy Nash Equilibrium* or a *Mixed Strategy Nash Equilibrium*.

3.3 Game-theoretic Utility Tree (GUT)

Game-theoretic Utility Tree (GUT) [28] is a new network model for artificial intelligent agent decision-making or MAS achieving cooperative decisions in uncertain environments, especially in adversarial scenarios. It builds a hierarchical relationship between individual behaviors and interactive entities and helps agents represent more complex behaviors to adapt to various scenarios. Instead of computing an intricate big game involving all the strategies in a complex situation, GUT decomposes the big game (considering all strategy combinations in one game) into several simple dependent games with corresponding strategies representing them as a tree structure to describe the complex situation. It helps reduce the time complexity of finding an optimal strategy combination.

On the other hand, a polynomial-time algorithm for normal-form games is of little use if the normal form is too large for the computer to store, and in this case, the computer needs to operate directly on a more concise representation of the game [35]. By decomposing the big game into simple games and organizing them as a tree, the GUT also improves the space complexity in the entire computation process.

Specifically, *GUT* consists of *Game-theoretic Utility Computation Units* distributed in multiple levels by decomposing strategies, thereby significantly lowering the game-theoretic operations in the strategy space dimension. It combines the core principles of *Utility Theory* and *Game Theory*. Moreover, the payoff (utility) values in the *Game-theoretic Utility Computation Units* are calculated through the agent needs expectations organized hierarchically following the *Agent (robot) Needs Hierarchy* [30, 36]. As a comprehensive artificial intelligent architecture, *GUT* not only considers the data from perceiving the environments with different "senses" (e.g., vision and hearing) but also infer the world's conditional (or even causal) relations and corresponding uncertainty with the hierarchical network.

Fig. 2 outlines the structure of the *Game-theoretic Utility Tree (GUT)* and its computation units distributed in each level. First, the *game-theoretic module* (Fig. 2 (a)) calculates the nash equilibrium based on the utility values $(u_{11},...,u_{nm})$ of corresponding situations, $(p_1,...,p_{nm})$ presenting the probability of each situation. Then, through the *conditional probability (CP)* module (Fig. 2 (b)), the CP of each situation can be described as $(p_{i1},...,p_{inm})$, where $p_{inm} = (p_{nm}|p_{i-1})$, $i,n,m \in Z^+$.

Here, p_{i-1} and S_i present the probability of previous situation and current strategy in the game-theoretic payoff table; s_a, s_b and n, m represent their strategy space and size on both sides, respectively.

4 Methodology

As we discussed above, agents first decompose the specific group strategy into several independent sub-strategies based on the same category of individual low-level primitives or atomic operations in the GUT [30, 28]. Then, through calculating various Nash equilibrium based on different situation utility values in each level's *Game-theoretic Utility Computation Units*, agents can get optimal or suboptimal strategy sets tackling the current status according to *Nash Existence Theorem*.

From the task execution perspective, the *GUT* can be regarded as a *Task-Oriented Decision Tree*. It decomposes a big game into several sub-games with conditional dependence in each level, which organizes agents' strategies and presents more complex behaviors adapting to adversarial environments.

We build a two-level GUT decomposing high-level strategies into executable low-level actions for cooperative pur-

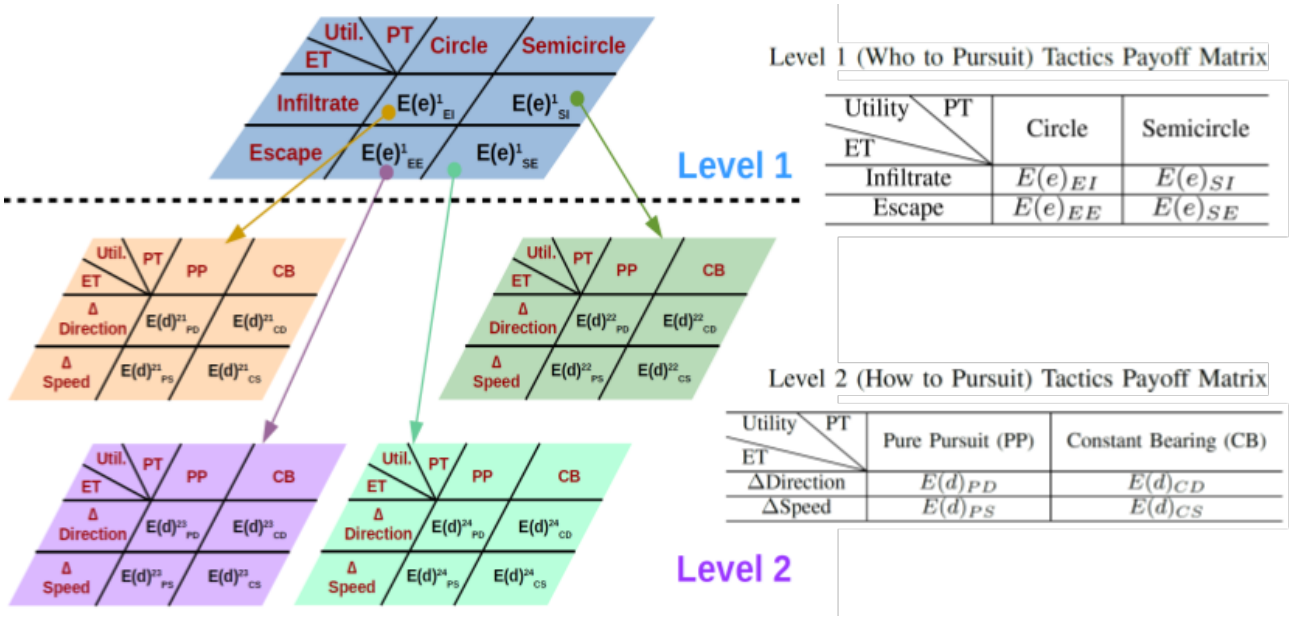


Figure 3 Illustration of two level game-theoretic utility tree (GUT) for multi-robot cooperative pursuit

Algorithm 1: Pursuer's Collective Strategy Using *GUT* Model in the *Pursuit-Evasion Game*.

Input: Pursuers' and Evaders' states. (d = average distance between Pursuers and Evaders)
Output: formation shape s; pursuit strategy t.

```
set state = "level one";
while |d| > β (β is the capture distance) do
    if state=="level one" then
        Compute the Nash Equilibrium;
        Get the most feasible formation shape s;
        state = "level two"
    else if state=="level two" And s != Null then
        Compute the Nash Equilibrium;
        Get the most feasible pursuit strategy t;
return s,t
```

suit strategy decisions in our scenarios. GUT decomposes a complex big game or scenario into conditionally dependent small games or simple situations and presents them as a tree structure. It combines with a new payoff measure based on agent needs [30] for real-time strategy games and provides a novel method to organize agents' group behaviors in the *Pursuit* domain. By calculating the *Nash equilibrium* in various games distributed in corresponding levels, we can theoretically guarantee to find an optimal strategy (tactics) trajectory in the GUT for the current situation based on *Nash Existence Theorem*. Fig. 3 illustrates the two-level GUT for multi-robot cooperative pursuit in the *Pursuit-Evasion* game.

In our pursuit-evasion game, we assumed that the energy and distance utility expectation follow the Eq. (2).

$$\begin{aligned} E(e_{cir/sem}, \alpha_{\Delta inf/\Delta esc}) &= \alpha_{\Delta inf/\Delta esc} d_{cir/sem} \\ E(d_{cb/pp}, \beta_{\Delta dir/\Delta spd}) &= \beta_{\Delta dir/\Delta spd} d_{cb/pp} \end{aligned} \tag{2}$$

Here, $d_{cir/sem}$ are the distances to the goal point using circle or semicircle methods. $\alpha_{\Delta inf/\Delta esc}$ are the coefficients of evader with the strategies of infiltrating or escaping, respectively. $d_{cb/pp}$ presents the current distance between pursuer and capture position, executing CB or PP tactic. $\beta_{\Delta dir/\Delta spd}$ describe the coefficients of evader implementing changing direction or speed strategy following normal distributions with different expectations correspondingly. E_e and E_d are the expected energy and distance utilities between the current position and goal point, respectively.

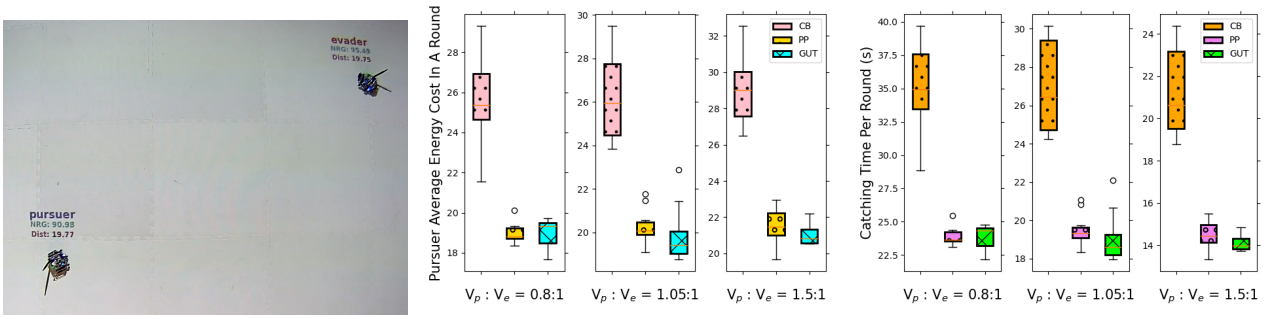
Alg. 1 presents the two-level GUT-based strategic decision-making in the "Pursuit Domain." Specifically, the first level defines the pursuer's high-level strategies, which are represented as *Circle* and *Semicircle* formation shapes of the pursuer team. Based on the first level decision, they need to decide the specific pursuit strategy in the second level. Here, we assume that pursuers have two basic tactics: constant bearing (CB) and pure pursuit (PP). Correspondingly, the evader also has two categories of strategies – *Infiltrate* and *Escape* in the first level, *Changing Direction* and *Changing Speed* in the second level – distributed at different levels, respectively.

5 Experiments and Evaluation

5.1 Experiment Setting

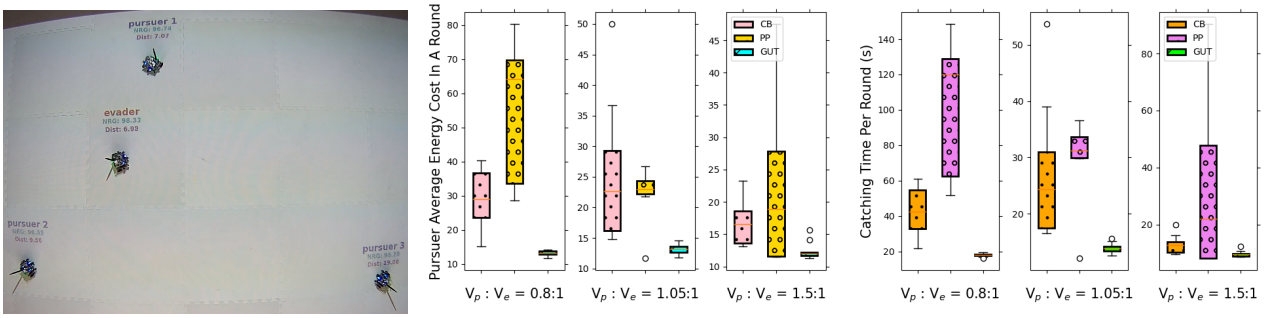
To demonstrate the GUT on the Pursuit-Evasion domain, we implement our method in the Robotarium [13] platform, a remote-accessible multi-robot experiment testbed that supports controlling up to 20 robots simultaneously on a 3.2m × 2.0m large rectangular area. Each robot has the dimensions 0.11 m × 0.1 m × 0.07 m in the testbed. The platform also provides a simulator helping users test their code, which can rapidly prototype their distributed control algorithms and receive feedback about their implementation feasibility before sending them to be executed by the robots on the Robotarium.

Our experiments design a pursuit-evasion game with several pursuers and one evader implementing different approaches to capture the evader: *constant bearing (CB)*, *pure pursuit (PP)*, and *GUT*. The uncontrolled evaders use



(a) 1 pursuers vs 1 evader (b) Pursuer Average Energy Cost in A Round (c) Time To Catch Evader Per Round

Figure 4 The Performance (1P vs 1E) of Robotarium Experiments with Different Speed Proportion in Pursuit-Evasion Domain.



(a) 3 pursuers vs 1 evader (b) Pursuer Average Energy Cost in A Round (c) Time To Catch Evader Per Round

Figure 5 The Performance (3P vs 1E) of Robotarium Experiments with Different Speed Proportion in Pursuit-Evasion Domain.

either infiltrate or escape approaches (to avoid getting captured by the pursuers), and they can change their direction or speed randomly (unexpectedly) during the game. The pursuers are controlled by our GUT algorithm, which uses a two-level *Pursuer and Evader Tactics Payoff Matrices* structure. The pursuers calculate the utilities based on the expected energy and distance between the current and goal points.

Especially, the Level one payoff describes the pursuer and evader, choosing which tactic captures the evader and searches the area correspondingly. Based on the work in [9], we design two kinds of high-level strategies in Level one – *Circle* and *Semicircle* – for the cooperative capture. In the *Circle* tactic, pursuers would center on the evader and surround it gradually until the evader can not move in any direction. In the *Semicircle* tactic, the pursuers encompass the evader in a semicircle, forcing it to a dead end. On the other hand, the evader will select the strategies – *Infiltrate* and *Escape* – based on the current situation correspondingly in Level one.

According to the Level one results, the Level two payoff will be calculated to execute the low-level implementation of the tactic. Here, we directly implement the methods of *CB* and *PP* [10] as the pursuers' tactics. The evader will execute *Changing Direction* (Δ Direction) or *Changing Speed* (Δ Speed) adapting to the current scenario. We can further extend these hierarchical levels depending on the application domain and possible low-level decomposition of strategies. Both the levels are presented in Fig. 3.

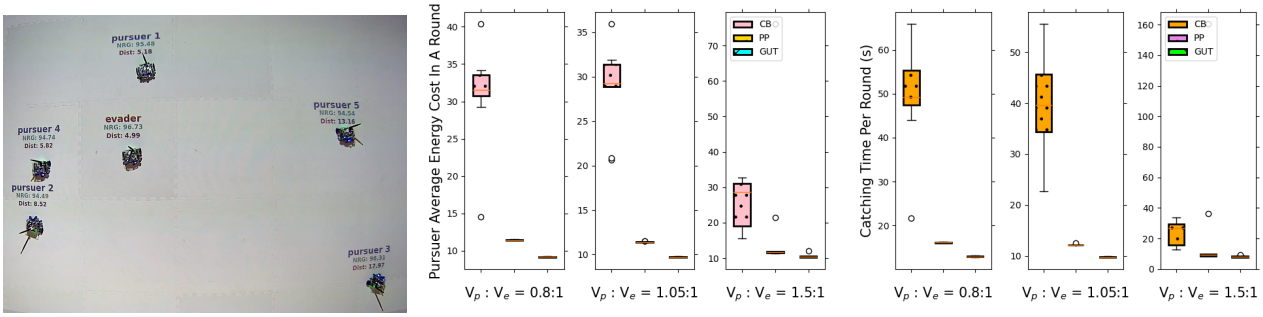
Our experiments assume that the pursuers will capture the evader if all the distances between each pursuer and evader are less than the distance threshold $d_{capture}$. i.e., $\sum_i \|P_i - E_j\| \leq d_{capture}, i \in N_P^j$, where N_P^j are the number of pursuers P_i seeking to capture the evader E_j. We apply three approaches (CB, PP, and GUT) in the Robotarium platform [13] both in simulations and real-robot experiments and analyze the results of ten trials. We consider two performance metrics: the pursuer's energy costs and the efficiency (time to capture).

5.2 Evaluation

We consider three scenarios with different proportions between pursuer and evader: *1 pursuer vs 1 evader* Fig. 4(a), *3 pursuers vs 1 evader* Fig. 5(a), and *5 pursuers vs 1 evader* Fig. 6(a). In each scenario, we conducted ten simulation trials with different speed proportions ($V_p : V_e = 0.8:1$, 1.05:1, and 1.5:1) for the GUT against *constant bearing (CB)* and *pure pursuit (PP)* methods by comparing their average energy cost per round and capture time.

5.3 1 pursuer vs 1 evader

In this scenario, we consider one pursuer pursuit one evader Fig. 4(a). Fig. 4(b) and 4(c) show that the performance of PP and GUT has a great advantage for CB in every speed proportion. However, comparing the GUT with PP, the pursuer's average energy cost per round and



(a) 5 pursuers vs 1 evader (b) Pursuer Average Energy Cost in A Round (c) Time To Catch Evader Per Round

Figure 6 The Performance (5P vs 1E) of Robotarium Experiments with Different Speed Proportion in Pursuit-Evasion Domain.

capture time do not show significant difference.

According to our experiments, when we implemented the GUT, the pursuer only selected the PP tactic chasing the evader in this scenario instead of changing its pursuit strategy. It means that GUT will converge to a unique and constant solution in a simple situation without considering the many factors involved in the current utility or reward mechanism. It also implies that the PP strategy might be the optimal solution for this scenario.

5.4 3 pursuers vs 1 evader

When we add another two pursuers Fig. 5(a) in our scenario to capture the evader cooperatively, the results represent distinguished differences among CB, PP, and GUT with different speed proportions.

Fig. 5(b) and 5(c) describe pursuer average energy cost per round and capture time for three different speed proportions, respectively. As we can see, if the pursuer and evader have a similar velocity ($V_p : V_e$ = 1.05:1), the GUT has a more distinct performance than the larger one ($V_p : V_e$ = 1.5:1) compared with CB and PP. Especially when the evader has a more significant advantage than the pursuer ($V_p : V_e$ = 0.8:1), the GUT shows dramatic performance compared with CB and PP.

Furthermore, if there is a considerable advantage in capabilities (like the speed) between individuals and opponents, the performance of cooperation and independence will be similar. Moreover, it is worth the point that suitable strategy selection and change can help the group with fewer advantages achieve higher performance, such as the GUT. It also verifies that if every group member has outstanding abilities, the cooperation becomes less significant.

5.5 5 pursuers vs 1 evader

Considering more pursuers chasing the evader, Fig. 6(a) presents five pursuers collaboratively capturing one evader in the Robotarium. Fig. 6(b) and 6(c) show that the CB has the worst performance in different speed proportions ($V_p : V_e$ = 0.8:1, 1.05:1, 1.5:1) than others. Furthermore, although the GUT presents some advantages in $V_p : V_e$ = 0.8:1 and 1.05:1 compared with the PP, the performance shows fewer differences in $V_p : V_e$ = 1.5:1. Due to the number of pursuers increasing, the advantages of some strategies become less effective. It means that these strategies which the pursuer can choose have been affected by other factors (like space) in the current dimension.

Through the above *Pursuit* experiments, we demonstrate the effectiveness and generality of the GUT in organizing individual behaviors to present more complex game-theoretic strategies, adapting to various situations to improve system performance. Furthermore, we prove that effective cooperation strategies can help agents with fewer advantages achieve higher performance in the tasks, which effectively organize the system resource and fully take advantage of the capabilities of each group member. We can further improve the performance of GUT's hierarchically structured relationships by adding more strategies to any levels (dimensions) or combining new strategies.

6 Conclusions

Our work extends the Game-theoretic Utility Tree (GUT) in the pursuit domain to achieve multiagent cooperative decision-making in catching an evader. We demonstrate the GUT's performance in the real robot implementing the Robotarium platform compared to the conventional constant bearing (CB) and pure pursuit (PP) strategies. Through simulations and real-robot experiments, the results show that the GUT could effectively organize cooperation strategies, helping the group with fewer advantages achieve higher performance.

In our future work, we plan to improve GUT from different perspectives, such as optimizing GUT structure through learning from different scenarios, designing appropriate utility functions, building suitable predictive models, and estimating reasonable parameters fitting the specific scenario. Besides, optimizing GUT structure through learning from different scenarios with reinforcement learning techniques is also an avenue for future work. Especially, integrating deep reinforcement learning (DRL) into GUT will primarily increase its application areas and effectiveness.

7 Literature

[1] S. Russell and P. Norvig, *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[2] Y. Rizk, M. Awad, and E. W. Tunstel, "Decision making in multiagent systems: A survey," *IEEE Transactions on*

Cognitive and Developmental Systems, vol. 10, no. 3, pp. 514–529, 2018.

[3] B. Browning, J. Bruce, M. Bowling, and M. Veloso, "Stp: Skills, tactics, and plays for multi-robot control in adversarial environments," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 219, no. 1, pp. 33–52, 2005.

[4] J.-l. Zhang, D.-l. Qi, and M. Yu, "A game theoretic approach for the distributed control of multi-agent systems under directed and time-varying topology," *International Journal of Control, Automation and Systems*, vol. 12, no. 4, pp. 749–758, 2014.

[5] E. Bakolas and Y. Lee, "Decentralized game-theoretic control for dynamic task allocation problems for multi-agent systems," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3228–3233.

[6] N. Agmon, G. A. Kaminka, and S. Kraus, "Multi-robot adversarial patrolling: facing a full-knowledge opponent," *Journal of Artificial Intelligence Research*, vol. 42, pp. 887–916, 2011.

[7] Y. Shapira and N. Agmon, "Path planning for optimizing survivability of multi-robot formation in adversarial environments," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4544–4549.

[8] T. H. Chung, G. A. Hollinger, and V. Isler, "Search and pursuit-evasion in mobile robotics," *Autonomous robots*, vol. 31, no. 4, p. 299, 2011.

[9] M. Kothari, J. G. Manathara, and I. Postlethwaite, "A cooperative pursuit-evasion game for non-holonomic systems," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 1977–1984, 2014.

[10] V. R. Makkapati and P. Tsiotras, "Optimal evading strategies and task allocation in multi-player pursuit–evasion problems," *Dynamic Games and Applications*, pp. 1–20, 2019.

[11] S. Nadarajah and K. Sundaraj, "A survey on team strategies in robot soccer: team strategies and role description," *Artificial Intelligence Review*, vol. 40, no. 3, pp. 271–304, 2013.

[12] P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[13] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.

[14] N. Agmon, S. Kraus, G. A. Kaminka, and V. Sadov, "Adversarial uncertainty in multi-robot patrol," in *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[15] R. Yehoshua and N. Agmon, "Adversarial modeling in the robotic coverage problem," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 891–899.

[16] O. Keidar and N. Agmon, "Safe navigation in adversarial environments," *Annals of Mathematics and Artificial Intelligence*, vol. 83, no. 2, pp. 121–164, 2018.

[17] Q. Yang, Z. Luo, W. Song, and R. Parasuraman, "Self-reactive planning of multi-robots with dynamic task assignments," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 89–91.

[18] Q. Yang and R. Parasuraman, "Needs-driven heterogeneous multi-robot cooperation in rescue missions," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 252–259.

[19] T. Liu, J. Wang, X. Zhang, and D. Cheng, "Game theoretic control of multiagent systems," *SIAM Journal on Control and Optimization*, vol. 57, no. 3, pp. 1691–1709, 2019.

[20] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. SIAM, 1998.

[21] 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.

[22] S. Xu and H. Chen, "Nash game based efficient global optimization for large-scale design problems," *Journal of Global Optimization*, vol. 71, no. 2, pp. 361–381, 2018.

[23] R. Gopalakrishnan, J. R. Marden, and A. Wierman, "An architectural view of game theoretic control," *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, no. 3, pp. 31–36, 2011.

[24] M. I. Abouheaf, F. L. Lewis, K. G. Vamvoudakis, S. Haesaert, and R. Babuska, "Multi-agent discrete-time graphical games and reinforcement learning solutions," *Automatica*, vol. 50, no. 12, pp. 3038–3053, 2014.

[25] J.-H. Cho, "Tradeoffs between trust and survivability for mission effectiveness in tactical networks," *IEEE transactions on cybernetics*, vol. 45, no. 4, pp. 754–766, 2014.

[26] A. Kolling and S. Carpin, "Multi-robot pursuit-evasion without maps," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 3045–3051.

[27] S. Shivam, A. Kanellopoulos, K. G. Vamvoudakis, and Y. Wardi, "A predictive deep learning approach to output regulation: The case of collaborative pursuit evasion," *arXiv preprint arXiv:1909.00893*, 2019.

[28] Q. Yang and R. Parasuraman, "A game-theoretic utility network for cooperative multi-agent decisions in adversarial environments," *arXiv preprint arXiv:2004.10950*, 2020.

[29] Y. Shoham and K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[30] Q. Yang and R. Parasuraman, "Hierarchical needs based self-adaptive framework for cooperative multi-robot system," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 2991–2998.

[31] P. C. Fishburn, "Utility theory for decision making," Research analysis corp McLean VA, Tech. Rep., 1970.

[32] R. B. Myerson, *Game theory*. Harvard university press, 2013.

[33] M. K. Sohrabi and H. Azgomi, "A survey on the combined use of optimization methods and game theory," *Archives of Computational Methods in Engineering*, vol. 27, no. 1, pp. 59–80, 2020.

[34] A. X. Jiang and K. Leyton-Brown, "A tutorial on the proof of the existence of nash equilibria," *University of British Columbia Technical Report TR-2007-25. pdf*, vol. 14, 2009.

[35] V. Conitzer and T. Sandholm, "New complexity results about nash equilibria," *Games and Economic Behavior*, vol. 63, no. 2, pp. 621–641, 2008.

[36] Q. Yang and R. Parasuraman, "How can robots trust each other for better cooperation? a relative needs entropy based robot-robot trust assessment model," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, pp. 2656–2663.