2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Optimizing Field-of-View for Multi-Agent Path Finding via Reinforcement Learning: A Performance and Communication Overhead Study

Hoi Chuen Cheng, Ling Shi, C. Patrick Yue
The Hong Kong University of Science and Technology

***Abstract*—This work investigates the impact of Field-of-View (FOV) on the performance of Reinforcement Learning (RL) models in Multi-Agent Path Finding (MAPF) problems. The study measures the effects of different FOV settings on RL performance, communication overhead, and computation time. Results show that the tested smallest FOV (3×3) reduces communication frequency by 28.9% with only a 1.65% reduction in success rate compared to the baseline (9×9). The study also compares computation time for different FOV for efficiency analysis and provides insights into FOV selection considering computation cost.**

***Index Terms*—Reinforcement Learning, Multi-Agent System, Multi-Agent Path Finding**

I. Introduction

Large-scale deployment of autonomous robots has been transforming manufacturing and warehouses [1]. To support these applications, a solution to the MAPF problem [2], i.e., a set of collision-free paths on a given map, has to be provided. Even though MAPF is NP-hard to solve optimally, numerous solutions for the MAPF problem have been derived. Some approaches exploit search-based techniques such as Conflict Based Search (CBS) [3], [4], while some simplify the problem into Boolean Satisfiability Problem (SAT) [6]. However, these approaches can only scale effectively in a system with a small number of agents.

To overcome the scalability issue, decentralized executions with Imitation Learning (IL) or RL have been used [19]–[21], [24]. Decentralized executions often modeled the MAPF task as a partially observable Markov game. They reduce the overhead by allowing agents to make decisions based on their observation instead of thoroughly observing the environment. IL-focused approaches aim to reduce the deviation between expert guidance and actual agent actions, while RL-focused approaches receive guidance from behavior cloning [19], [20] and heuristics [25].

To further improve multi-agent collaboration, researchers have recently shifted the focus on communication-based solutions [21], [22], [25]. These solutions, however, emphasize broadcast communication, in which messages are sent to the surrounding agents without specific targets. Even though significant advantages have been demonstrated in broadcast communication over previous works, it creates a large amount of communication overhead. In addition, it adds unnecessary burdens and latency to the system, as not all information from the surroundings is helpful for decision-making. The superfluous information might confuse the agent and eventually degrade the learning process. Therefore, RL frameworks that can lower communication overhead have been suggested [15]–[17], [26], [27]. Particularly in [26], agents communicate to any neighbor who can potentially affect its temporary decision.

The FOV setting is critical in multi-agent systems for several reasons. FOV impacts agents' perception, navigation, and awareness of opportunities. Larger FOVs allow more perspective but demand more resources, while smaller FOVs are more focused and efficient and require less computation load to operate. FOV also affects coordination and collaboration, as overlapping views enable communication and shared goals. However, oversized FOVs cause unnecessary exchanges. Additionally, smaller FOVs help agents concentrate on their immediate surroundings, like decreasing the detection distance of a LiDAR sensor to focus only on nearby objects. Narrower FOVs conserve energy and extend battery life, especially where charging is limited. In some contexts, limiting FOV restricts sensitive data access for security.

FOV configuration is critical in real-world deployment, given hardware and cost constraints. The FOV scope must be judiciously chosen when engineering and fielding multi-agent robots. Intuitively, the FOV scale should adequately support navigation and teamwork while remaining energy-efficient. In order to bring more insight into the selection of FOV and study its effect on performance and communication overhead, we conduct experiments based on a recent communication-based MAPF framework [26]. On one hand, the performance of the MAPF framework can easily be improved by adjusting the FOV. Conversely, we suggest a way to view the performance when considering computation cost, and it allows

This work is in part supported by Bright Dream Robotics (BDR) and the HKUST-BDR Joint Research Institute Funding Scheme under Project HBJRI-FTP-005 (OKT22EG06).

researchers and engineers to examine the trade-offs more easily when designing a cost-efficient system.

II. Related Works

A. Multi-Agent Path Finding (MAPF)

Even though there exists approximating optimal solutions [7], MAPF is still an NP-hard issue. We can generally categorize MAPF planners as coupled, decoupled, and dynamically coupled methods.

Coupled methods such as A* suffer tremendously from the curse of dimensionality. Meanwhile, decoupled methods such as [8] can plan and modify large amounts of paths for collision avoidance in low-dimensional search spaces. However, it is possible that the decoupled techniques are not complete, as the low-dimensional search spaces only represent a limited piece of joint configuration space [9]. Dynamically-coupled techniques allow broader agent interactions while preventing planning in the entire configuration space. For example, CBS and its variants [3]–[5] avoid searching in higher dimensional space by defining a set of constraints.

B. MAPF with RL

Single-agent path planning using RL has seen considerable accomplishments [11], [12]. Researchers have recently focused on using RL approaches to solve MAPF problems, and most rely on expert guidance from existing planners. For instance, OD-recursive-M^* (ODrM*) [13] is used in PRIMAL [19], A* is used in MAPPER [24] and a graph-based planner [10] is used in Global-to-Local Autonomy Synthesis (GLAS) [23]. Particularly in PRIMAL, the framework uses an Asynchronous Advantage Actor-Critic (A3C) network as the RL module and ODrM* planner for behavior cloning. However, these expert planners suffer from high computation complexity with the growing number of agents, and the paths designed for single-agent environments might not be optimal for multi-agent environments. One potential way is to utilize communication to encourage collaboration between agents. Targeted Multi-Agent Communication (TarMAC) [15] and Graph Neural Network (GNN) [21] are two recent examples of how communication is used to boost collaboration. DHC [25] utilizes both heuristics and graph convolution for navigation and communication. In Decision Causal Communication (DCC) [26], the authors extend DHC by using a Decision Causal Unit to communicate selectively with neighbors. It removes redundant messages, and the communication overhead is greatly reduced.

III. Problem Setup

The models are trained and tested in a classical MAPF setting or benchmark defined in [2]. The basic rules of the benchmark test are as follows:

1) Each agent in the environment takes one of the five available actions (Up, Down Left, Right and Stop).
2) Agents should stop once their destinations are reached.

A. Formal Definition

In a classical MAPF problem with n agents, an undirected graph $G = (V, E)$ is used as the input, where the start vertices and destination vertices are defined as $\{s_1, \ldots, s_n\} \in V$ and $\{d_1, \ldots, d_n\} \in V$ for the n agents correspondingly. The location change $v \rightarrow v'$ caused by an agent's movement corresponds to an edge in the graph (i.e., $(v, v') \in E$). Time is discretized so that each agent will move or stop at its position for each time step.

We denote a sequence of actions taken by agent i from the beginning to time t as $\pi_i = \{a_1, \ldots, a_t\}$ and the location of the agent i as $l_i(t) = a_t(...a_1(s_1))$. A MAPF problem solution comprises n action progressions $\pi = \{\pi_1, \ldots, \pi_n\}$ for each of the n agents.

B. Types of Conflicts

MAPF has a lot of variants and each allows different conflicts defined in [2]. However, all MAPF surveyed by [2] and this work forbids the below conflicts:

1) Vertex collision: A vertex collision happens when 2 agents i and j try to arrive at the same vertex at time t such that $l_i(t) = l_j(t)$.
2) Edge collision: An edge collision occurs iff agents i and j try to move with the same edge $(v, v') \in E$ such that $l_i(t) = l_j(t)$.

C. MAPF Environment

In the predefined MAPF task, the agents move simultaneously in a discrete grid world. In a map of size $k \times k$ with n agents, there will be n corresponding start positions and goals. In the grid world, agents can move to the adjacent vertices in the four cardinal directions or remain stationary. Therefore, the action space has a size of 5.

We consider the MAPF task as a partially observable world for each agent in order to simulate real-world deployment. As a result, each agent has its own limited FOV for its decision-making. More on FOV settings will be discussed in Section V-A.

Table I shows the reward structure, which is adopted from DHC [25] and DCC [26]. To encourage giving ways to other agents when blocking occurs, staying on non-goal vertices is not punished as heavily as in PRIMAL [19] and MAPPER [24].

Actions	Reward
Move (Up/Down/Left/Right)	-0.075
Stay on goal vertices	0
Stay on non-goal vertices	-0.075
Collision	-5
Completion	3

TABLE I
Reward structure

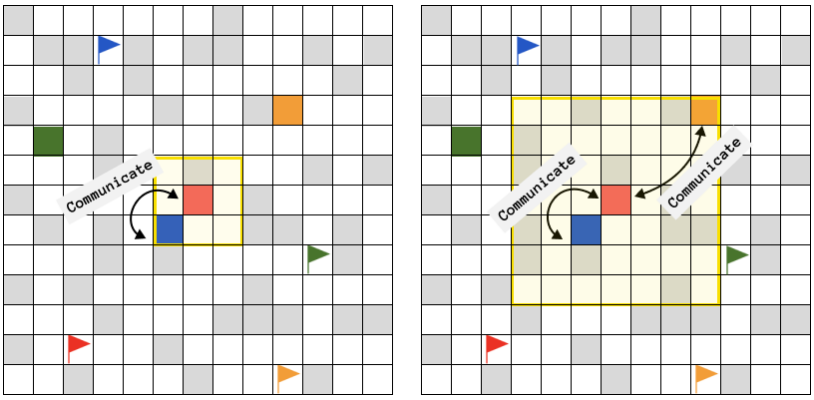

Fig. 1. Examples of different FOV settings (3×3 on the left and 7×7 on the right) of the center agent (red square) in a 13 x 13 grid. Agents are displayed as colored squares, and their corresponding destinations are displayed as colored flags. In addition, the obstacles are displayed as grey squares and white squares are the walkable tiles. Agents that fall into the red agent's FOV (yellow area) can communicate in a request-reply style. In the 3×3 FOV setting, the red agent can only communicate with the blue agent. While in the 7×7 FOV setting, the red agent can communicate with both the blue and orange agents. Note that the computation load will increase indefinitely in an agent-dense environment.

IV. RL Model Architecture

Our model architecture is based on DCC, which utilizes a selective communication mechanism based on causal inference and Individually Inferred Communication (I2C) [27]. The model contains an observation encoder, a decision causal unit, a graph convolution-based communication block, and a Dueling Deep Q Network (DQN).

With a FOV size of $l \times l$, each agent i receives a 6-channel input observation of size of $l \times l \times 6$ and is passed to an observation encoder. The input observation includes two binary matrices marking the positions of agents and obstacles inside the FOV, respectively. The other four matrices are the heuristic channels from DHC. Path information is embedded in four action channels (Up, Down, Left, Right) sized to the FOV. A location is marked one if its action moves the agent closer to the goal; otherwise, it is zero.

The observation encoder is made up of convolutional layers and followed by a Gate Recurrent Unit (GRU). An encoded observation $\hat{o}_i$ is first generated by the convolutional layers using the original observation o_i. The GRU then takes $\hat{o}_i$ and hidden state from the last communication outcome $e_i^{[2]}$ to output an intermediate message e_i.

The decision causal unit determines whether the communication should be triggered between agent i and its neighbors $\mathbb{N}_i$, by judging if the neighbors' existence affects the agent i's decision. To achieve this, the observation encoder first generates modified observation embeddings $\{e_{i,-j}\}_{j\in\mathbb{N}_i}$ from modified observations $\{o_{i,-j}\}_{j\in\mathbb{N}_i}$ (agent i's observation without agent j) for all of agent i's neighboring agents $j \in \mathbb{N}_i$. The embeddings $\{e_{i,-j}\}_{j\in\mathbb{N}_i}$ are then fed to a Dueling network [18] (discussed later in this session) to get temporary actions $\tilde{a}_i$ and $\{\tilde{a}_{i,-j}\}_{j\in\mathbb{N}_i}$. By comparing $\tilde{a}_i$ and $\{\tilde{a}_{i,-j}\}_{j\in\mathbb{N}_i}$, we define the communication scope as

$$\mathbb{C}_i = \{j|\tilde{a}_i \neq \tilde{a}_{i,-j}\}_{j\in\mathbb{N}_i}. \tag{1}$$

The communication block is based on graph convolution, creating a graph by connecting the neighboring agents as adjacent nodes. We communicate with neighboring agents in a request-reply scenario to improve efficiency. Given a communication scope $\mathbb{C}_i$ for agent i, the intermediate message e_i generated from the observation encoder of i, and relative positions of agent i's neighbors' l_i are passed to all the neighboring agents $j \in \mathbb{C}_i$.

The message e_i is projected to query with matrix W_Q^h and the concatenation of e_i and l_i are projected to key and value with matrices W_K^h and W_V^h. Let $\mathbb{O}_{j+}$ be the set $\{\mathbb{O}_j, j\}$ where the receiving scope for agent j is defined as $\mathbb{O}_j = \{i|j \in \mathbb{C}_i\}$. The relation in h-th attention head between agent j and agents $\bar{i} \in \mathbb{O}_{j+}$ is computed as follows

$$\mu_{j\bar{i}}^h = \text{softmax}\left[\frac{W_Q^h e_j \cdot (W_K^h[e_{\bar{i}}, l_{\bar{i}}])^T}{\sqrt{d_K}}\right], \tag{2}$$

where d_K is the dimensions of key and $\sqrt{d_K}$ is used to stabilize the training. The attention head's outputs are concatenated over H heads and passed to a single neural network layer f_o to generate a final output

$$\hat{e}_j = f_o\left[concat\left[\sum_{\bar{i}\in\mathbb{O}_{j+}} \mu_{j\bar{i}}^h W_V^h[e_{\bar{i}}, l_{\bar{i}}], \forall h \in H\right]\right]. \tag{3}$$

Using a GRU, the output $\hat{e}_j$ and the initial message are aggregated, and the output $e_i^{'}$ acts as the input message for the next round by repeating Equation (1) and (2). The final output of the communication module is denoted as $e_i^{''}$.

We use a Dueling DQN model [18], which utilizes the advantage functions to estimate the Q-value using the output from the communication block. In this work, we use the mean of the advantages

$$\frac{1}{|N|}\sum_a A(e_i^{''}))$$

to stabilize the training, where N is the size of the action space. The Equation is defined as

$$Q_{i,s,a} = V_s(e_i^{''}) + \left[A(e_i^{''}) - \frac{1}{|N|} \sum_a A(e_i^{''})) \right]. \quad (4)$$

A multi-step Temporal Difference (TD) error is calculated as

$$L(\theta) = MSE(R_t - Q_{s_t,a_t}(\theta)), \quad (5)$$

where the total reward $R_t = r_t + \gamma r_{t+1} + \cdots + \gamma^n Q_{s_{t+n},a_{t+n}}(\bar{\theta})$ and r_t is the reward received at time t. The discount factor applied to the future rewards is represented as γ, and the periodic target network copy of the model parameters θ is denoted as $\bar{\theta}$.

V. Experiments

A. FOV Settings

The importance of the FOV of an agent in a robotic multi-agent system cannot be overstated. It is a crucial factor in enabling the agent to perceive, navigate, and collaborate effectively with other agents in its environment. A larger FOV can provide an agent with more information and a broader perspective on its surroundings. However, there are also some advantages to using a smaller FOV: reduced computational load, improved energy efficiency, collaboration, and security.

FOV controls agents' information intake and communication scope during MAPF tasks. Our focus is studying FOV's effects on performance and communication overhead. To mimic realistic deployment in large environments like factories and warehouses, assuming a fully observable environment (FOV with the size of the map) is not practical. Thus, a FOV with size $l \times l$ must be smaller than map size $k \times k$, i.e., $l < k$.

Intuitively, minimizing FOV reduces information to save computing bandwidth for path planning and networking. An appropriate FOV should maximize energy efficiency without compromising performance significantly. To find a more energy-efficient FOV, we tested sizes smaller than 9×9 (used in DCC). Note that FOV width and height must be odd to center agents, making 3×3 the minimum. To investigate increased information intake on decision-making and performance, we also evaluated larger FOVs. We use the original DCC model with a 9×9 FOV as the baseline, trained with a batch size of 192 using curriculum learning. Training starts with 2 agents on a 10×10 map, up to 20 agents, and a 40×40 map size. We conduct experiments with the following FOV settings: $\{3 \times 3, 5 \times 5, 7 \times 7, 9 \times 9, 11 \times 11\}$.

B. Test Settings and Hardware Specifications

To assess navigation ability and generalization, our tests used 40×40 and 80×80 maps with 30% obstacle vertices. The 80×80 size evaluates generalization since training used maps smaller or equal to 40×40. We randomly generated 500 test cases varying map size and number of agents {4, 8, 16, 32}, with a maximum of 256 steps. All metrics were averaged across cases.

16 Intel Xeon Gold 6230 CPUs and 2 RTX Quadro RTX 6000 GPUs from HKUST's HPC3 are used during the model training.

VI. Result

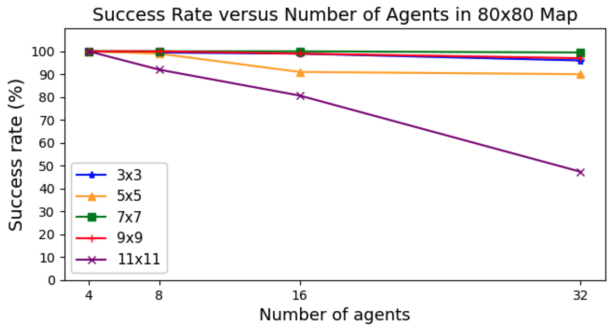

Fig. 2. Success Rate versus Number of Agents in 80×80 Map

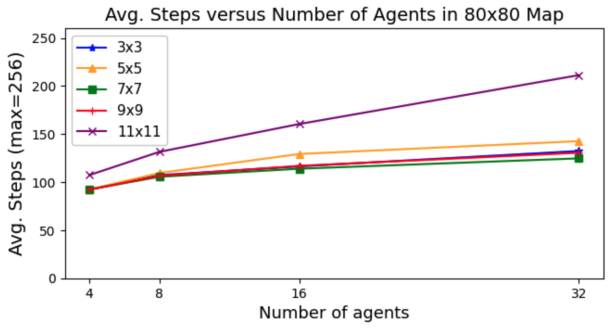

Fig. 3. Average Steps versus Number of Agents in 80×80 Map

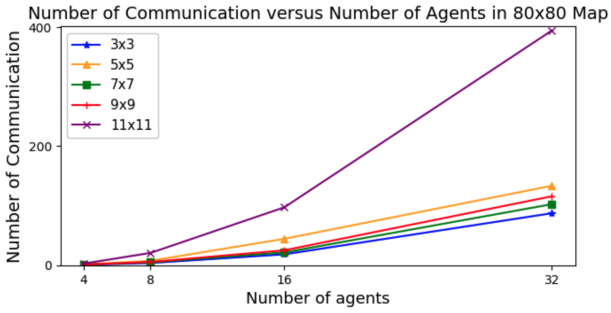

Fig. 4. Number of Communications versus Number of Agents in 80×80 Map

A. Success Rate, Average Steps and Number of Communications

We evaluate the performances of different FOVs by measuring success rate, average steps taken, and number of communications required. The success rate corresponds to the percentage of agents reaching the destination within the maximum number of steps. Average steps measure the average number of steps to finish a MAPF task across all agents. The

maximum number of steps that each agent can take is 256 steps. Finally, the number of communications indicates the number of request-reply pairs generated throughout a MAPF task.

Fig. 2-3 illustrate the success rate and average steps comparison with different FOV in a 80×80 map. On average, the 7×7 FOV outperforms the original baseline (9×9) by 4.2% in terms of success rate and 3.0% in average steps. Although the 3×3 FOV receives the least amount of information, it demonstrated a relatively small sacrifice in performance across the metrics mentioned above. In terms of success rate, 3×3 was 5.85% lower than 7×7 and 1.65% lower than 9×9. Regarding average steps across all the settings, the 3×3 FOV exhibited 4.2% more steps than 7×7 and 1.0% more steps more than 9×9.

Fig. 4 shows the number of communications conducted during different MAPF tasks in a 80×80 map. The 3×3 FOV, which receives the least amount of information regarding neighboring entities, is restricted in its communication capacity, thereby reducing communication overheads. On average, the number of communication drops by 28.9% when compared with the baseline 9×9 FOV and by 24.4% when compared to the most performant 7×7 FOV. The 3×3 FOV's combination of the least communication required and small performance impact makes it optimal when bandwidth is limited, like deploying many robots with few networking devices.

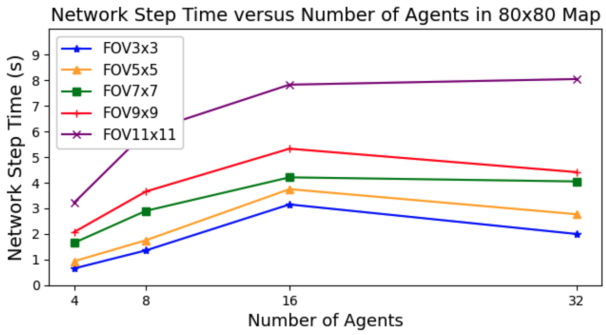

Fig. 5. Network Step Time versus Number of Agents in 80×80 Map

B. Network Step Time

High-performing robotic systems often utilize large FOVs despite high computation costs. However, real-world multi-robot systems face constraints on computing power due to limited resources. Thus, weighing the computational cost of different FOV options is crucial. This study uses network step time to evaluate computation cost. Network step time measures the time an agent takes to select an action given an observation. Longer step times indicate higher decision-making computation costs.

Fig. 5 compares network step times across FOVs. Results confirm step time scales with FOV size. 3×3 has the lowest computation cost among tested FOVs, while 11×11 has the highest. It demonstrates the need to balance performance and computational efficiency when configuring FOV.

FOV	Network Step Time Ratio
3×3	1
5×5	1.34
7×7	1.95
9×9	2.27
11×11	3.69

TABLE II
NETWORK STEP TIME RATIO OF DIFFERENT FOV

However, considering only computation cost is insufficient for selecting FOV, as performance impact is not weighed. It motivates devising a metric incorporating performance and cost, further detailed in Section VI-C. To obtain a universal ratio between step times across FOVs, we normalized the step times by the shortest time for each map and agent count. Table II shows the average ratios of networked step times for the tested FOVs.

C. Normalized Success Rate

In order to consider both computation cost and success rate concurrently, this study introduces a novel metric referred to as the normalized success rate, denoted as r'. The value of r' is determined by dividing the original success rate r by the normalized network step time $\bar{t}$, i.e.,

$$r' = \frac{r}{\bar{t}}. \tag{6}$$

Fig. 6 shows the normalized success rate across all the FOV tested in the two map sizes. Our analysis incorporates both success rate and computation cost metrics, with the understanding that optimal performance demands satisfactory outcomes in both domains. Our findings indicate that the 3×3 FOV demonstrates the most robust performance in terms of normalized success rate, and it is followed by 5×5, 7×7, 9×9, and 11×11. In essence, 3×3 is the most cost-efficient FOV among all the FOVs we tested.

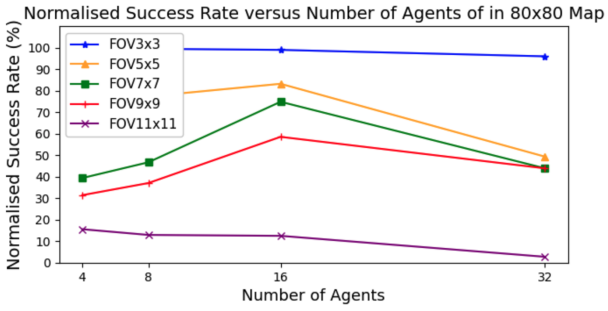

Fig. 6. Normalized Success Rate versus Number of Agents in 80×80 Map

In this study, we have observed that the network step time and normalized success rate of varying FOV dimensions exhibit in the same order, i.e., 3×3, 5×5, 7×7, 9×9, and 11×11. This finding suggests a high sensitivity of

the normalized success rate to the network step time. The effect is further exaggerated when the range of success rate of different FOV is small (typically within 10%), while the maximum network step time can be three times longer than the minimum. Given the reduced computation time associated with smaller FOV dimensions, it is likely that these smaller dimensions will exhibit a comparatively high normalized success rate.

D. Key Findings and Recommendations:

- Increasing the size of the FOV may not necessarily lead to improved performance but could weaken performance. Smaller FOV sizes may be more effective since performance does not decrease proportionally with FOV size.
- Use normalized success rates to compare performance across FOV sizes. A thorough evaluation is critical to determine optimal FOV per application since no universal best size exists.

VII. Conclusion

In this work, we studied agent FOVs' impact on MAPF performance and communication overhead using DCC. Experiments with varied FOV settings showed that adjusting FOV alone can improve prior DCC results for success rate, average steps taken, and number of communications required. We also estimated decision-making time for different FOVs and presented a computation cost ratio. Accounting for this cost, 3×3 FOV is the most efficient overall. These insights on optimizing the performance and communication overhead will assist real-world, large-scale MAPF deployment under resource constraints.

References

[1] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses," The AI Magazine, 2007.

[2] R. Stern, N. R. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, et al., "Multi-agent pathfinding: Definitions, variants, and benchmarks," in Symposium on Combinatorial Search, pp. 151–159, 2019.

[3] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," Artificial Intelligence, vol. 219, pp. 40–66, 2015.

[4] V. Rybář, and P. Surynek, "Highways in Warehouse Multi-Agent Path Finding: A Case Study," International Conference on Agents and Artificial Intelligence, vol 1, pp. 274-281, 2022.

[5] M. Barer, G. Sharon, R. Stern and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," Seventh Annual Symposium On Combinatorial Search, 2014.

[6] P. Surynek, A. Felner, R. Stern, and E. Boyarski, "Efficient sat approach to multi-agent path finding under the sum of costs objective," in Proc. European Conference on Artificial Intelligence (ECAI), pp. 810–818, 2016.

[7] S. M. LaValle, Planning Algorithms. Cambridge: Cambridge University Press, 2006.

[8] J. Berg, S. Guy, M. Lin and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*. pp. 3-19, 2011.

[9] G. Sanchez and J.-C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in Proc. IEEE international conference on robotics and automation, vol. 2, pp. 2112–2119, 2002.

[10] W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," IEEE Transactions on Robotics, vol. 34, no. 4, pp. 856–869, 2018.

[11] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3052–3059, 2018.

[12] F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1343–1350, 2017.

[13] C. Ferner, G. Wagner, and H. Choset, "ODrM* optimal multirobot path planning in low dimensional search spaces," IEEE International Conference on Robotics and Automation (ICRA) , pp. 3854–3859, 2013.

[14] Z. Ding, T. Huang, and Z. Lu, "Learning individually inferred communication for multi-agent cooperation," Advances in Neural Information Processing Systems (NeurIPS), pp. 22069-22079, 2020.

[15] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, J. Pineau, "Tarmac: Targeted multi-agent communication," in International Conference on Machine Learning (ICML), pp. 1538–1546, 2019.

[16] J. Foerster, I. A. Assael, N. De Freitas, and S. Whiteson, "Learning to Communicate with Deep Multi-Agent Reinforcement Learning," Advances in Neural Information Processing Systems (NeurIPS), vol. 29, 2016.

[17] S. Q. Zhang, Q. Zhang, and J. Lin, "Efficient Communication in Multi-Agent Reinforcement Learning via Variance Based Control," Advances in Neural Information Processing Systems (NeurIPS), vol. 32, 2019.

[18] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," International Conference on Machine Learning (ICML), pp. 1995–2003, 2016.

[19] G. Sartoretti, J. Kerr, Y. Shi, G. Wagner, T. S. Kumar, S. Koenig, and H. Choset, "PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning," IEEE Robotics and Automation Letters (RA-L), vol. 4, no. 3, pp. 2378–2385, 2019.

[20] M. Damani, Z. Luo, E. Wenzel, G. Sartoretti1, "PRIMAL2: Pathfinding via Reinforcement and Imitation Multi-Agent Learning - Lifelong," IEEE Robotics and Automation Letters (RA-L), vol. 6, no. 2, pp. 2666-2673, 2021.

[21] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11 785–11 792, 2020.

[22] Q. Li, W. Lin, Z. Liu, and A. Prorok, "Message-aware graph attention networks for large-scale multi-robot path planning," IEEE Robotics and Automation Letters (RA-L), vol. 6, no. 3, pp. 5533–5540, 2021.

[23] B. Riviere, W. Hönig, Y. Yue, and S. Chung, "Glas: Global-to-local safe autonomy synthesis for multi-robot motion planning with end-to-end learning," IEEE Robotics and Automation Letters (RA-L), vol. 5, no. 3, pp. 4249–4256, 2020.

[24] Z. Liu, B. Chen, H. Zhou, G. Koushik, M. Hebert, and D. Zhao, "Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11 748–11 754, 2020.

[25] Z. Ma, Y. Luo, and H. Ma, "Distributed heuristic multi-agent path finding with communication," IEEE International Conference on Robotics and Automation (ICRA), pp. 8699-8705, 2021.

[26] Z. Ma, Y. Luo, and J. Pan, "Learning Selective Communication for Multi-Agent Path Finding," IEEE Robotics and Automation Letters (RA-L), vol. 7, no. 2, pp. 1455-1462, 2022.

[27] Z. Ding, T. Huang and Z. Lu, "Learning individually inferred communication for multi-agent cooperation," Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 22069-22079, 2020.