Multi-Robot Social Navigation with Cooperative Occupancy Prediction

Jiaqi Li, Yiping Li, Zehao Wang, Albert Dang, Jiachen Li*

***Abstract*— In this paper we introduced a novel Multi-Robot Social Navigation system that incorporates cooperative perception to tackle the challenges posed by visual occlusions and incomplete observations in environments with both static and dynamic obstacles. By employing a multi-agent reinforcement learning (MARL) strategy, our approach enhances predictive capabilities and situational awareness, enabling robots to anticipate and avoid potential collisions in the complex dynamic environments more effectively. Cooperative perception facilitates the exchange of sensory data among robots, significantly refining the accuracy of perception. This improvement in data fidelity enhances the performance of downstream tasks such as occupancy grid map(OGM) prediction and navigation.**

I. INTRODUCTION

Social navigation involves robots operating in environments with both static obstacles, such as walls, and dynamic obstacles, like pedestrians or other moving entities. Although recent advances have enabled single-robot navigation to achieve promising results, significant challenges remain. Single-robot systems are often hindered by occlusions and limited local observations, making it difficult to consistently identify globally optimal navigation paths. For example, a robot may become stuck in a confined area due to surrounding static obstacles, or it may fail to detect a pedestrian hidden behind a wall when turning a corner. These limitations can result in ineffective planning or unsafe navigation choices.

In contrast, in a multi-robot connected setting, robots can exchange information to enhance their situational awareness. By sharing observations, such as a nearby robot detecting a pedestrian beyond an occluded area, the robots can augment their perception and make more informed decisions. While multi-agent cooperative perception techniques have been extensively studied to address these perception challenges, their application to downstream decision-making in social navigation remains underexplored.

One of the most widely used representations for robot navigation is the occupancy grid map (OGM), which provides a structured model of the environment by assigning occupancy probabilities to each grid cell. While OGMs are effective, most research has focused either on single-robot OGM prediction, where the goal is to predict the future states of dynamic environments, or on cooperative perception, which aims to detect distant or occluded objects by sharing raw sensor data between autonomous agents. However, the concept of cooperative OGM prediction remains underexplored. Cooperative OGM prediction obstacles. By predicting the future state of the environment, robots can anticipate potential collisions with moving obstacles, such as humans, and take preemptive actions to avoid them. In order to enable effective multi-robot cooperation, we model the environment in social navigation as a decentralized partially observable semi-Markov decision process (Dec-POSMDP), and adopt a multi-agent reinforcement learning (MARL) algorithm to learn joint policies and optimize cooperative navigation behaviors. By integrating MARL with cooperative OGMs prediction, robots can not only leverage shared perception information but also refine their long-term strategies based on anticipated future states of the environment. The combination of real-time data sharing and predictive modeling allows for more intelligent and adaptive behavior, particularly in environments with partial observability and dynamic obstacles.

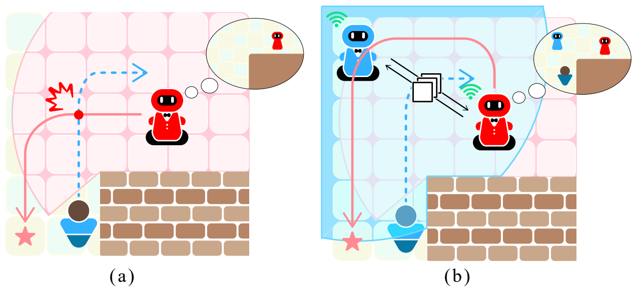

Fig. 1. A comparison between the multi-robot navigation with and without cooperative prediction. (a) Traditional navigation conducts perception based on a single robot's raw sensor data and fails to detect dynamic obstacles. (b) The proposed navigation involves multiple cooperative robots, which share information to enhance cooperative prediction and avoid future collisions.

To the best of our knowledge, this is the first work in multi-robot navigation that integrates cooperative occupancy prediction to overcome the limitations of single-robot systems. Our approach enhances perception through different cooperative settings for multiple robots and enables cooperative prediction using uncertainty-aware OGMs predictors. By augmenting the partially observable environment with future OGMs generated from cooperative predictions, robots can share observations and anticipate future environmental states more effectively. This leads to improved navigation performance in complex, dynamic settings.

The main contributions of this work are as follows:

- We introduce a novel framework for multi-robot social navigation that integrates cooperative occupancy prediction into downstream decision-making, addressing the challenges in multi-robot navigation.
- We develop cooperative OGM predictors under different cooperation settings, enhancing each robot's ability to

* Corresponding author
J. Li is with the University of California, Riverside, CA, USA. {jiachen.li}@ucr.edu.

anticipate future environmental states and improving navigation performance.
- Our approach achieves state-of-the-art (SOTA) performance in both occupancy prediction metrics (SSIM, WMSE) and multi-agent navigation, as demonstrated by higher success rates and lower collision rates compared to several baseline methods.

II. Related Work

A. Multi-Robot Social Navigation

As the development of mobile robotics and autonomous driving advances, physical artificial intelligence agents are required to share limited spaces with humans and adhere to social norms [1] [2]. Typically, social navigation is defined as a Partially Observable Markov Decision Problem (POMDP), where agents cannot access the ground truth of the environmental state but instead make decisions based on local observations. Therefore, the key lies in the modeling of the incomplete observations prevalent in real-world environments [3]. To address highly complex and dynamic crowd environments, using reinforcement learning for real-time decision-making has become a popular approach [4]. A series of previous works primarily focused on the trajectory information of dynamic obstacles. They ideally models obstacles as tuples of position and radius, optimistically assuming that the robot can acquire information about all pedestrians within a certain range around it. This series of studies achieves well performance in simulated environments by modeling human intentions and predicting human behaviors [5] [6] [7] [8]. However, trajectory based navigation faces challenges when applied to the real world, primarily due to the diversity and variability of obstacle shapes in reality.

B. Occupancy Grid Map

Another series of work use occupancy grid maps (OGMs) as observations to model dynamic crowd environments [9] [10], which are similar to the methods used in the autonomous driving. OGMs model obstacles as occupations on each map grid. Although OGMs consume more resources compared to trajectory-based information, OGMs offer superior representational capabilities and can be easily extracted from raw Lidar data. Some variants of OGMs can carry additional information. For example, Dynamic Occupancy Grid Maps (DOGMs) annotate each grid with kinematic information, and Semantic Occupancy Grid Maps (SOGMs) incorporate semantic segmentation information, adding semantic labels to each pixel. This enhancement facilitates the execution of downstream tasks.

C. Cooperative Perception and Prediction

Cooperative perception has become crucial for overcoming the limitations of single-agent systems, particularly in environments with occlusions or restricted fields of view. Multiple agents share sensor data to construct a more complete view of the environment, improving navigation performance in crowded or occluded settings. Early fusion approaches share raw sensor data (e.g., LiDAR or RGB camera data), providing detailed information but requiring high bandwidth, which limits real-time use. On the other hand, late fusion techniques transmit only the processed results, reducing bandwidth requirements but often losing valuable contextual information. Intermediate fusion [11] strikes a balance between these two approaches. In this method, robots process raw sensor data locally, extracting intermediate features, which are then shared with other agents. By combining these features, each robot improves its perception and prediction capabilities while minimizing the need for excessive data transmission. This approach enables agents to collaboratively build a more accurate and comprehensive understanding of their environment, enhancing tasks such as navigation, future state prediction, and collision avoidance.

III. Preliminaries

A. Perception

Considering the limited computational resources of mobile robots and the need for real-time operations, the combination of RGB cameras and 2D LIDAR is a common configuration for the perception module of mobile robots [2]. Specifically, the 2D LIDAR provides 360-degree obstacle distance information around the robot by scanning, while the camera, empowered by computer vision techniques, can provide semantic or kinematic information about the obstacles within the field of view. In our work, we employ this configuration to ensure that robots can extract sufficient and useful environmental information with a practically feasible hardware setup.

B. Multi-Agent Reinforcement Learning

The multi-agent version of the Proximal Policy Optimization algorithm (MA-PPO) is widely regarded as an effective method for multi-agent reinforcement learning [12]. Unlike centralized algorithms, which require agents to transmit observations to a central server for task assignment, distributed algorithms such as MA-PPO rely on local information for decision-making. This decentralized approach reduces communication latency and improves both the safety and feasibility of navigation in dynamic environments.

IV. Methods

In this section, we briefly overview the approach of our multi-agent reinforcement learning, with a primary focus on the observation space, cooperative occupancy perception, action space, and the configuration of the reward function.

A. Problem Formulation

Formally, the multi-robot social navigation problem could be represented by Dec-POMDP problem defined by the tuple $\langle n, N, S, A, \Omega, O, P, R, \gamma, \rangle$, where n / N denotes the he number of Robots / Agents. S represents the state space of Agents, and $s_t = [s_t^1, s_t^2, \ldots, s_t^N] \in S^N$ represents the joint state of all Agents in environment at timestep t.

A is the shared action space for each agents i in environment. Ω represents the observation space of all robots, and the joint observation $\omega_t = [\omega_t^1, \omega_t^2, \ldots, \omega_t^n] \in \Omega^n$.

$O : S^N \rightarrow \Omega^n$ specifies how robots perceive Observations from environment, influenced by the configuration of sensors and the collaborative perception/prediction methods. It should be noted that we use $\hat{\omega}_t^i$ as the **original observation** obtained by the i-th robot at time step t directly from the environment. In contrast, ω_t^i denotes the **final observation** for the i-th robot at time step t, which is derived after processing and integrating shared observations from other robots within the cooperative network in this timestep .

$P(s_i \mid s_j, a_t)$ denotes the transition probability from s_j to s_i given the joint action $a_t \in A^N$ for all N agents.

$R : S^N \times A^N \rightarrow \mathbb{R}$ denotes the joint reward function, and γ is the discount factor.

B. MARL

1) Training Algorithm: Robots use a policy

$$\pi_\theta(a_t^i \mid \omega_t^i, H_t^i)$$

with parameters θ to produce an action a_t^i from the local observation ω_t^i and hidden state H_t^i. All robots share the same policy and jointly optimize the reward $J(\theta) = \mathbb{E}_{a_t, s_t}\left[\sum_t \gamma^t R(s_t, a_t)\right]$.

Algorithm 1 training algorithm

```
$\pi \leftarrow$ Initialize parameters; $step \leftarrow 0$
while $step \leq total\_step$ do
    $s_0 = [s_0^1, s_0^1, \ldots, s_0^N] \leftarrow$ Reset Environment
    $\omega_0 = O(s_0) \leftarrow$ Initial observations
    $H_0 \leftarrow$ Initial hidden states, Rollout$\leftarrow$ []
    for $t$ = 0 to $T$ do
        for $i$ = 0 to $n$ do
            if Robot i is activated then
                sample $a_t^i, v_t^i$ from $\pi_\theta(\omega_t^i, H_t^i)$, update $H_{t+1}^i$
            else
                $a_t^i, v_t^i \leftarrow nothing$
        get human actions, update Env $s_{t+1} \sim P(s_t, a_t)$
        compute reward $r_t = R(s_t, a_t)$
        $\omega_{t+1} = O(s_{t+1}) \leftarrow$ get fused observation
        append $(\omega_t, a_t, v_t, H_t, r_t)$ into Rollout
        for $i$ = 0 to $n$ do
            if Robot i collides or reaches goal then
                Deactivate Robot i
        if all Robots are deactivated then
            Reset Environment, observations, hidden states
    Use Rollout update $\theta$ parameters on MAPPO loss
    step = step + 1
```

Specifically, in each step, each robots independently get their original observations, then share observation information with nearby robots in their local area, and integrate these to form a more precise and complete local observation. Subsequently, using these observations, each robot locally sample actions based on the policy and evaluates their value, incorporating this information into a global rollout. Finally, the loss is computed globally and the policy is optimized based on this aggregated information.

2) Observation Space: To align with the real-world sensors, we utilize an observation space $\omega_t = [I_t, \mathbf{o_t}, d_t]$ comprising three preprocessed components. The first $I_t = [p_x, p_y, g_x, g_y, v_x, v_y, r, \theta]$ contains the robot's state information.

$\mathbf{o_t}$ represents a semantic occupancy map defined in the local reference frame, stored as a two-channel matrix with the shape $[2, \text{map\_size}, \text{map\_size}]$. The first channel indicates the occupancy rate at the current location, while the second channel provides semantic segmentation information, classifying the space into five categories: unknown, open space, pedestrian, obstacle, and robots.

d_t includes the state information of other robots within the vicinity, based on the assumption that robots can communicate and share information. This information, more direct than that from an ordinary grid map (OGM), is crucial for collision avoidance among robots.

C. Cooperative Occupancy Grid Map Prediction

As in Fig.2, the cooperative OGM prediction in the intermediate fusion setting consists of three parts, where multi-robots cooperatively perceive the environment based on LiDAR data and extra the spatiotemporal features with ConvLSTM. Then, the cooperative predictor fuses the dynamic and static features and provides the final probabilistic future prediction for navigation strategy learning. The prediction problem for current time t given the past τ frame can be formulated as:

$$p_\theta(\mathbf{o}_{t+1} \mid \mathbf{y}_{t-\tau:t}^{1:n}, I_{t-\tau:t}^{1:n}) \tag{1}$$

where θ are the model parameters. The state information of robot i at time t from observation space is denoted by $I_t^{1:n}$. The lidar measurements for robot i at time t, comprising range r_t^i and bearing b_t^i, are represented as $\mathbf{y}_t^i = [r_t^i \; b_t^i]^T$, and the predicted OGM for next step $t+1$ is denoted as $\mathbf{o}_{t+1}^i$. The goal is to find the optimal θ that maximizes (1), allowing the robots to predict the future state of the environment through cooperation accurately.

Static / Dynamic Objects Map Conversion. In dynamic environments containing both static and dynamic obstacles, multiple robots equipped with LiDAR sensors cooperatively sense their surroundings by sharing occupancy grid maps (OGMs) or intermediate features. Each robot estimates its pose and velocity using odometry or localization algorithms, and generates a local OGM based on LiDAR measurements and its own states. Both dynamic object maps and static environment maps are considered for prediction as [13]. The function $c(\cdot)$ represents the conversion process that transforms LiDAR measurements into binary dynamic maps within the robot's local coordinate frame. To maintain consistency in the robot's sequence of OGM states, the locally observed OGMs $\mathbf{o}_{t-\tau}$ are transformed into the robot's local coordinate frame at the current time step t. This ensures that the robot's environmental perception remains aligned with its real-time position and orientation.

Early Fusion. Each robot shares its local dynamic and static OGMs with surrounding robots within a defined communi-

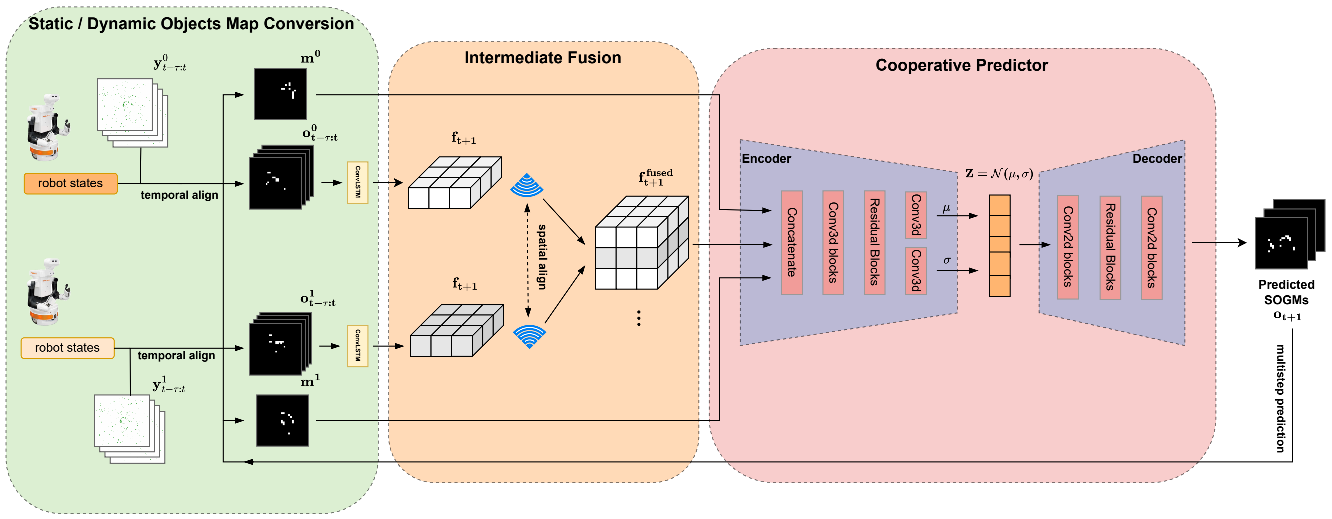

Fig. 2. Cooperative prediction (intermediate fusion) overall architecture

cation range. The ego robot then transforms the received OGMs into its local coordinate frame and fuses them with its own OGMs to create a collective understanding of the environment. The fused dynamic OGM is denoted as $\mathbf{o}^{fused}_{t-\tau:t}$, and the fused static OGM as $\mathbf{m}^{fused}_t$. Next, a ConvLSTM-based model $h(\cdot)$ processes the history of the fused dynamic OGM sequences to obtain spatiotemporal features $\mathbf{f}^{fused}_{t+1}$. These spatiotemporal features, along with the fused static OGM, are used to predict the future occupancy states:

$$\mathbf{f}^{fused}_{t+1} = h(\mathbf{o}^{fused}_{t-\tau:t}), \tag{2}$$

$$p_\theta(\mathbf{o}_{t+1} \mid \mathbf{f}^{fused}_{t+1}, \mathbf{m}^{fused}_t) \tag{3}$$

Intermediate Fusion. In intermediate fusion, each robot i processes its local dynamic OGM $\mathbf{o}^i_{t-\tau:t}$ through $h(\cdot)$, to extract intermediate features $\mathbf{f}^i_{t+1}$ from its historical states. These features are then shared and spatially aligned across robots. The ego robot aggregates both its features and the transformed features from surrounding robots to create a fused feature map $\mathbf{f}^{fused}_{t+1}$, which is used as input for the prediction model along with the fused static map $\mathbf{m}^{fused}_t$ as in previous early fusion setting (3).

Cooperative Predictors. Once the fused dynamic feature, $\mathbf{f}^{fused}_{t+1}$, and fused static map $\mathbf{m}^{fused}_t$ are obtained, we use a Variational Autoencoder (VAE) to model and predict a distribution over the future occupancy grid map (OGM). The VAE consists of an inference network (encoder) and a generative network (decoder). The encoder first concatenates the dynamic and static features and then compresses the features into a latent representation z. The decoder generates the predicted future OGM $\mathbf{o}_{t+1}$ from the latent representation z:

$$z = \mathbf{Enc}_\phi(\mathbf{f}^{fused}_{t+1}), \quad \mathbf{o}_{t+1} = \mathbf{Dec}_\theta(z), \tag{4}$$

where $\mathbf{Enc}_\phi$ represents the encoder parameterized by ϕ, and $\mathbf{Dec}_\theta$ represents the decoder parameterized by θ. The VAE introduces probabilistic components that capture the inherent uncertainty of the future states. This cooperative perception approach, augmented by uncertainty-aware prediction, allows the system to collaboratively predict future states across multiple robots. Moreover, the model estimates future occupancy states in an autoregressive manner, further enhancing prediction accuracy over time by feeding the predicted state back into the model for subsequent time steps.

D. Policy Network

The input to the network consists of three components: the robot state, predicted stochastic occupancy grid maps (SOGMs), and detected robot states. The robot state includes positional and velocity information such as $(P_x, P_y, g_x, g_y, V_x, V_y, \theta)$. The predicted SOGMs are generated from the cooperative perception prediction module introduced in the previous section and can be devide to dynamic obstacle maps and static maps.The dynamic maps capture the movement of surrounding objects, while the static map contains environmental features such as obstacles.

After feature extraction, each input features are concatenated and passed through a ResNet block for feature fusion. The temporal information is then integrated using a Gated Recurrent Unit (GRU), providing a time-dependent representation of the input.

The final fused features are used as input to the Actor-Critic module. The Actor head outputs the action features used to sample actions, while the Critic network's output is used to compute the value of the current state.

E. Reward Function

The total reward function R_{total} for all robots is defined as the sum of one-time rewards and ongoing rewards:

$$R(s^i_t, a^i_t) = \begin{cases} r_g & \text{if reach goal,} \\ r_c & \text{if collision,} \\ r_{\text{danger}} & \text{if too close to obstacles,} \\ r_{pot} & \text{otherwise.} \end{cases}$$

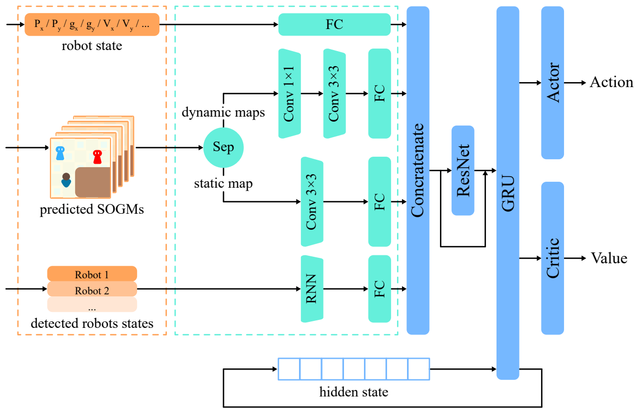

Fig. 3. The architecture of the deep reinforcement learning model. The **robot state** information is processed through two fully connected (FC) layers, each outputting a 64-d feature vector. The **dynamic predicted maps** first passes through a 1×1 convolution layer to integrate temporal features. Then further encoded by CNN into a 256-d feature.The **static map** is also encoded via a CNN network to a 256-d feature. The **detected robot information** is treated as sequential data, where robots will be sorted by their relative distances to the robot, and then passed through a recurrent neural network (RNN), followed by an FC layer, encoded to a 64-d feature. Residual block contains two layers, with the widths of 1024 and 512 dimensions. The **Gated Recurrent Unit (GRU)** has a hidden layer of 512 dimensions. **Actor** and **Critic** networks consist of two 128-d FC layers, and finally output the action and value.

where $r_g = 20$. The collision reward r_c depends on the type of object collided with. We consider a collision with a pedestrian to be a more severe outcome, hence:

$$r_c = \begin{cases} -20 & \text{if the collision object is a pedestrian,} \\ -10 & \text{otherwise.} \end{cases}$$

The danger reward r_{danger} represents a penalty applied when the minimum distance d_{min} between the robot and any other object is less than 3 meters. It is given by:

$$r_{\text{danger}} = \frac{0.3}{d_{\text{min}} + 0.03}$$

Finally, a potential function r_{pot} has been added to guide the robot towards its target. Specifically, r_{pot} is calculated as the change in the distance to the target multiplied by a scaling factor f:

$$r_{\text{pot}} = \Delta d \cdot f$$

where Δd is the change in distance to the target. The scaling factor f typically equals 3. However, when there are obstacles obstructing the path between the robot and the target, f is reduced to 1.5. This reduction in f is intended to encourage the robot to navigate around obstacles to reach the target.

V. Experiments

In this section, we introduce our experimental setup and results, including navigation experiments in a simulator environment, data collection, and real-world experiments.

A. Simulation Environment and Dataset

We built a 20m * 20m square on a 2D simulator, which includes multiple robots, pedestrians, and simple static obstacles. Static obstacles are stored in the form of a bitmap, and several simple convex polygons are randomly generated using a randomized BFS algorithm. The movement of pedestrians follows the topology guided ORCA method [14], which can effectively simulate and handle crowd movements in environments with static obstacles. The robot is set to be invisible to pedestrians, assuming that the robot is responsible for all collision avoidance obligations. In terms of perception, we simulated 2D lidar data, and the robot was able to obtain semantic information of obstacles ahead within a FOV of $\frac{\pi}{2}$, in order to simulate the role of cameras and visual models.

In terms of kinematics, we tested two different movements. One assumes that the robot can adjust its speed arbitrarily, with a maximum speed of 1m/s, which is the same as that of a human; In addition, we also tested a more realistic motion model, referring to a two wheeled robot with differential motion control, limiting the maximum speed to 0.5m/s and the linear velocity to decrease with increasing angular velocity, with a maximum angular velocity not exceeding $\frac{\pi}{2}$/s.

We collected a dataset using our simulator to train and evaluate our cooperative OGM predictors. We first trained a policy using ground truth OGM as observation. Then, we employed the pretrained policy to control robots and stored LIDAR data and position sequences of each robot. During data collection, Our simulator includes 3 robots, 3 pedestrians and 10 different static obstacle maps. We used data from 576 episodes to train and 64 to evaluate our cooperative OGM predictors, in which all robots successfully reached goals.

B. Evaluation Metrics and Baselines

Cooperative Prediction Metrics. We evaluate our cooperative OGM predictors on the following two metrics: weighted mean square error (WMSE), and structural similarity index measure (SSIM). Specifically, we define the predicted OGM as $\bar{\mathbf{o}}$ and the ground truth OGM as $\mathbf{o}$. The WMSE defines as:

$$WMSE = \frac{\sum_{i=1}^{m} w_i \left(\bar{\mathbf{o}}_i - \mathbf{o}_i\right)^2}{\sum_{i=1}^{N} w_i}, \tag{5}$$

where m is the number of cells in the OGM, and w_i is the weight for the cell i in the OGM, calculated by the median frequency balancing method [15]. This metric is used to evaluate the weighted absolute errors (balancing the imbalance in the percentage of occupied and free cells) between the predicted OGM and its corresponding ground truth OGM, describing the predicted quality of a single OGM cell. The SSIM is defined as:

$$SSIM = \frac{\left(2\mu_{\bar{\mathbf{o}}}\mu_{\mathbf{o}} + C_1\right)\left(2\delta_{\bar{\mathbf{o}}\mathbf{o}} + C_2\right)}{\left(\mu_{\bar{\mathbf{o}}}^2 + \mu_{\mathbf{o}}^2 + C_1\right)\left(\delta_{\bar{\mathbf{o}}}^2 + \delta_{\mathbf{o}}^2 + C_2\right)}, \tag{6}$$

where $\mu_{(\cdot)}$ and $\delta_{(\cdot)}$ denote the mean and variance/covariance, respectively, and $C_{(\cdot)}$ denotes constant parameters to avoid

TABLE I
THE COMPARISONS OF OCCUPANCY GRID MAP PREDICTION.

Method	Cooperation Setting	Prediction Horizon (steps)	SSIM ↑		WMSE ↓		IoU ↑	Bandwidth (M/s) ↓	FPS
			no delay	100ms delay	no delay	100ms delay			
SOGMP++ [13]	No Cooperation	1	0.48	-	0.29	-			
		4	0.45	-	0.30	-			
Ours	Early Fusion	1	0.80	0.69	0.10	0.18			199.2
		4	0.76	0.67	0.12	0.20	0.34		112.1
	Intermediate Fusion	1	0.75	0.69	0.13	0.19			82.3
		4	0.70	0.66	0.18	0.21	0.25		30.4
	Late Fusion	1	0.69	0.61	0.18	0.23			
		4	0.65	0.58	0.21	0.20			

instability. We use $C_1 = 1\mathrm{e}{-4}$ and $C_2 = 9\mathrm{e}{-4}$. This metric is used to evaluate the structural similarity between the predicted OGM and its corresponding ground truth OGM, describing the predicted quality of the scene geometry.

Navigation Metrics. We follow the standard evaluation metrics, including both navigation and social metrics. The navigation metrics measure the quality of the navigation and include the success rate (SR), collision rate (CR), timeout rate (TR), average navigation time (NT) in seconds, and path length (PL) in meters for successful episodes.

Additionally, we aim to quantify the smoothness of the paths. To achieve this, we introduce two new metrics: the average acceleration (AA), which is defined as the average velocity changes between each step, and the sharp turn rate (STR), which is defined as the proportion of time during which the angular velocity exceeds 80% of the maximum allowable angular velocity.

Cooperative Prediction Baselines. To demonstrate the effectiveness of incorporating cooperative perception, we conduct ablation studies on various model components and compare our method with SOGMP++ [13], an effective OGM stochastic predictor that leverages motion prediction of surrounding dynamic objects and the robot itself. We compare perception performance based on the absolute error, structural similarity, and prediction time range.

Navigation Baselines. For navigation, we adopt ORCA [14] and DRL-VO [9] as our baseline algorithms. ORCA is a widely used non-learning collision avoidance strategy, while DRL-VO employs an occupancy grid map augmented with velocity information for deep reinforcement learning-based single-robot navigation. We utilize the network and reward function proposed by DRL-VO, training it within our environment using the Proximal Policy Optimization (PPO) algorithm. During testing, the single-robot policy obtained from DRL-VO training is applied individually to each robot in a multi-robot environment. We evaluate the performance data in scenarios both with and without cooperative perception.

C. Quantitative and Qualitative Results

Cooperative Prediction.

Social Navigation. The experimental results are presented in Table II. The ORCA algorithm shows significantly higher collision rate (CR) and timeout rate (TR) in environments with both static obstacles and dynamic targets. This is primarily because the ORCA algorithm assumes that both parties involved in a potential collision share equal responsibility in avoiding each other. However, this assumption does not hold for static obstacles, leading to numerous episodes where the robot either collides with a building or becomes trapped in corners by human agents, ultimately resulting in a timeout.

Additionally, in the ablation study of the cooperative perception module, the success rate (SR) of policies using cooperative perception as observations outperforms those using only local independent perception, both in our method and the baseline methods. This indicates that cooperative perception improves the overall navigation success rate.

It can also be observed that policies using cooperative perception tend to have a higher path length (PL), but a lower average acceleration and sharp turn rate. This can be interpreted as the cooperative perception module enabling robots to detect occluded potential obstacles in advance, allowing them to reroute or decelerate earlier, thus navigating more smoothly through crowds. In contrast, robots relying solely on local observations must resort to abrupt stops or sharp turns to avoid obstacles that suddenly appear in their field of view.

Figure 4 presents a representative case comparing the performance of robots with and without cooperative perception (CP). The first row illustrates how cooperative perception allows the robot to gracefully avoid potential collisions with pedestrians, while the second row shows the performance of a robot without CP in the same scenario.

In the first image of the first row, Robot 1, through cooperative perception with Robot 2, detects the approaching pedestrian (inside the green circle) despite the LIDAR signal being blocked by an obstacle between them. This early detection prompts Robot 1 to decelerate, as shown in the second image, thereby avoiding an impending collision. After the pedestrian passes through the narrow passage, Robot 1

TABLE II

Navigation Performance(TODO: mid fusion

Methods	Perception	**SR**↑	**CR**↓	**TR**↓	**NT**↓	**PL**↓	**AA**↓	**STR**↓
ORCA [14]	No cooperation	43.34%	30.59%	26.06%	45.72	**19.46**	0.21	31.35%
DRL-VO		70.44%	35.26%	**0.91%**	39.16	19.73	0.13	12.51%
Ours		69.53%	30.42%	1.04%	**36.29**	22.86	0.09	14.60%
DRL-VO	Early Fusion	75.15%	21.77%	3.08%	40.06	21.84	0.09	9.75%
Ours		**80.22%**	**18.05%**	1.73%	38.30	23.11	**0.05**	**8.60%**
DRL-VO	Intermediate Fusion							
Ours								
DRL-VO	Late Fusion							
Ours								

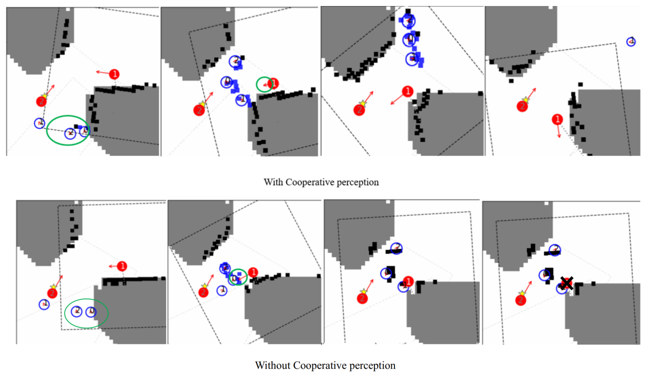

Fig. 4. Illustration of a representative case comparing robots with and without cooperative perception (CP). The solid red circle represents the robot, with the red arrow indicating its velocity. Hollow blue circles represent pedestrians, while large grey areas represent static obstacles. The black and blue grids within the dashed boxes represent the Occupancy Grid Map (OGM) perceived by Robot 1. The first row shows the robot with CP gracefully avoiding collisions by decelerating early, while the second row shows the robot without CP failing to react in time, leading to a collision.

accelerates to continue its path.

In contrast, the second row demonstrates the behavior of a robot without CP. Without early warning, the robot fails to decelerate in advance and only reacts when the pedestrian fully enters its field of view. At this point, the robot makes an abrupt stop, but it is still unable to avoid the collision.

VI. CONCLUSIONS

In conclusion, our research demonstrates the advantages of integrating cooperative perception into multi-robot navigation systems for improved situational awareness and decision-making in complex environments, enabling robots to anticipate and avoid potential collisions more effectively.

However, our current study is limited to tests within a 2D simulator using simplistic sensor configurations and lacks an in-depth analysis of intermediate features like occupancy grid maps (OGMs). Future work will explore the fusion of multiple sensor modalities and aim to develop end-to-end trained models that do not rely on manually specified intermediate features. This approach is expected to further enhance the robustness and applicability of autonomous navigation systems in more dynamically challenging environments.

References

[1] H. Ishiguro, T. Ono, M. Imai, T. Maeda, T. Kanda, and R. Nakatsu, "Robovie: A robot generates episode chains in our daily life," in *Proceedings of the 32nd ISR (International Symposium on Robotics)*, vol. 19, 2001, p. 21.

[2] C. S. Chen, C. J. Lin, and C. C. Lai, "Non-contact service robot development in fast-food restaurants," *IEEE Access*, vol. 10, pp. 31 466–31 479, 2022.

[3] R. Mirsky, X. Xiao, J. Hart, and P. Stone, "Conflict avoidance in social navigation—a survey," *ACM Transactions on Human-Robot Interaction*, vol. 13, no. 1, pp. 1–36, 2024.

[4] W. Zhu and M. Hayashibe, "Learn to navigate in dynamic environments with normalized lidar scans," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 7568–7575.

[5] S. Liu, P. Chang, Z. Huang, N. Chakraborty, K. Hong, W. Liang, D. L. McPherson, J. Geng, and K. Driggs-Campbell, "Intention aware robot crowd navigation with attention-based interaction graph," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 015–12 021.

[6] W. Wang, L. Mao, R. Wang, and B.-C. Min, "Multi-robot cooperative socially-aware navigation using multi-agent reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 353–12 360.

[7] J. Yao, X. Zhang, Y. Xia, Z. Wang, A. K. Roy-Chowdhury, and J. Li, "Sonic: Safe social navigation with adaptive conformal inference and constrained reinforcement learning," *arXiv preprint arXiv:2407.17460*, 2024.

[8] J. Li, C. Hua, H. Ma, J. Park, V. Dax, and M. J. Kochenderfer, "Multi-agent dynamic relational reasoning for social robot navigation," *arXiv preprint arXiv:2401.12275*, 2024.

[9] Z. Xie and P. Dames, "Drl-vo: Learning to navigate through crowded dynamic scenes using velocity obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023.

[10] D. Dugas, J. Nieto, R. Siegwart, and J. J. Chung, "Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7829–7835.

[11] R. Xu, Z. Tu, H. Xiang, W. Shao, B. Zhou, and J. Ma, "Cobevt: Cooperative bird's eye view semantic segmentation with sparse transformers," in *Conference on Robot Learning*, 2022, pp. 989–1000.

[12] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 611–24 624, 2022.

[13] Z. Xie and P. Dames, "Stochastic occupancy grid map prediction in dynamic scenes," in *7th Annual Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=fSmkKmWM5Ry

[14] F. C. Pouria, Z. Huang, A. Yammanuru, S. Liu, and K. Driggs-Campbell, "Topology-guided orca: Smooth multi-agent motion planning in constrained environments," *arXiv preprint arXiv:2407.16771*, 2024.

[15] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,"

in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.