Multi-Agent Pickup and Delivery in Human-Populated Environments

Benedetta Flammini[1], Leo D'Amato[2], and Francesco Amigoni[1]

***Abstract*— In Multi-Agent Pickup and Delivery (MAPD), a team of mobile agents must execute incoming pickup and delivery tasks while avoiding reciprocal collisions. In this work, we consider a setting in which agents operate in an environment shared with humans, whose movements are not controllable. This setting is inspired by real-world applications, from warehouses, where robots move in areas shared with human workers, to shopping malls, where cleaning robots collect trash while moving among customers. In these contexts, agents cannot rely on communication with humans or restrict their behavior, yet must still ensure efficient task execution and safe navigation. To address this challenge, we propose a planning framework that models human mobility using a Diffusion Convolutional Recurrent Neural Network (DCRNN), capturing both spatial dependencies and temporal dynamics to predict future pedestrian flows. These predictions are incorporated into the agents' planning process to enable effective coordination in dynamic environments. Preliminary results on a real-world pedestrian trajectory dataset show that human-aware modeling can be successfully integrated into MAPD in shared environments.**

I. INTRODUCTION

Autonomous robots are increasingly deployed in environments shared with people: hospitals, shopping malls, and warehouses, where mobile platforms operate alongside human workers and customers. In these settings, robots must complete incoming tasks, which can be formulated as Multi-Agent Pickup and Delivery (MAPD) [1] problems. In MAPD, a team of agents must coordinate to service tasks that arrive over time, which are composed of a pickup and a delivery location. MAPD has been widely studied in structured and controlled settings, such as automated warehouses. Classical MAPD solvers [1] assume robots are the only dynamic entities, while human-aware navigation methods either target single-robot navigation [2], [3] or use simplistic occupancy models that ignore the rich spatiotemporal structure of crowd behavior [4].

Human movement in public spaces is structured: people follow habitual routes, concentrate in predictable areas, and exhibit periodic patterns [5]. Modeling and exploiting these regularities during path planning, rather than reacting to humans online, can reduce both task completion time and disturbance to people. To this end, we propose a unified framework for *MAPD in Human-Populated Environments* (*MAPD-HPE*) that integrates learned predictions of human occupancy into multi-agent path planning. We predict future crowd presence using a Graph Neural Network (GNN), specifically a Diffusion Convolutional Recurrent Neural Network (DCRNN) [6], and incorporate these predictions into the path planner as time-dependent edge weights of the graph that represent the environment in which robots plan their movements. At the coordination level, Token Passing (TP) [1], a common MAPD solver, is extended with a rolling horizon strategy [7] so that updated predictions are continuously exploited. A reactive safety layer enforces hard proximity constraints at execution time, independent of prediction quality. We preliminarily evaluate the proposed framework on real-world pedestrian trajectory data [8], comparing against a purely reactive baseline.

Our main contributions are: (i) the formulation of the MAPD-HPE problem; (ii) the introduction of a GNN-based model for predicting human occupancy; and (iii) a planning algorithm that integrates these predictions into a rolling horizon multi-agent coordination strategy.

II. RELATED WORK

Navigation is a fundamental capability of autonomous mobile robots, tightly linked to their internal world representation. Effective operation requires capturing environmental dynamics and incorporating them into path planning. When it comes to autonomous robots interacting with humans, some approaches focus on modeling and prediction, while others aim to integrate predictions into the navigation process.

A large body of work addresses the prediction of human motion in indoor and public spaces [9] [10]. Statistical approaches model long-term periodicity: FreMEn [11] uses harmonic analysis to predict the future visibility of environment features, while STeF-map [12] adapts this idea to occupancy grids. Vintr et al. [13] use a multidimensional hypertime vector to cluster events by periodicity. CLiFF-map [14] encodes directional flow as a mixture of Gaussians and has been extended to long-term trajectory prediction [15]. Deep learning approaches have produced strong short-term predictors: Social LSTM [16] and Social GAN [17] model pedestrian interactions through pooling mechanisms over individual trajectory sequences. Trajectron++ [18] frames prediction as structured probabilistic inference conditioned on environment context. Vintr et al. [19] provide a benchmark for pedestrian flow models that evaluates their suitability for safe robot navigation. In contrast to methods that aim to predict the trajectory of single individuals, we predict *grid-level occupancy* directly, which is the representation most compatible with grid-based path planning and does not require tracking individual identities at runtime.

Some works aim to bridge the gap between modeling human trajectories and planning robot paths. In [4], the

[1]Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milan, Italy name.surname@polimi.it

[2]Institute of Cognitive Sciences and Technologies, National Research Council, Rome, Italy leodamato@cnr.it

authors assume that some external agents are operating in an environment where a multi-robot system is performing MAPD, and model them using Markov Chains or a simple occupancy map; however, the time aspect is not taken into account, making this approach not suitable for human behavior. In [2], [3], authors incorporate spectral environment models into Markov Decision Process frameworks to maximize the success probability of navigation actions, but consider only a single robot. Lo et al. [20] model robot path planning in a shared human workspace as a stochastic game, but do not address task assignment. Reactive collision avoidance methods such as ORCA [21] avoid dynamic obstacles based on instantaneous observations and require no prediction, but provide no optimality guarantees for task-oriented navigation and can produce overly conservative behaviors in dense crowds. Heuer et al. [22] propose a benchmark for multi-robot coordination in human-shared environments and evaluate two MAPF (Multi-Agent Path Finding, a one-shot offline version of MAPD) algorithms, finding that human presence substantially degrades performance. A successive work [23] proposes Flow-Aware MAPF, which integrates learned motion patterns of uncontrollable agents into MAPF algorithms to reduce conflicts.

III. BACKGROUND AND PROBLEM FORMULATION

A. Classical MAPD

The Multi-Agent Pickup and Delivery (MAPD) problem [1] considers a team of n agents, $\mathcal{A} = \{a_1, \ldots, a_n\}$, operating in an environment modeled as a connected undirected graph $G = (V, E)$, which we assume to be a 4-connected grid. Vertices V represent locations and edges E represent connections between locations. Time is discrete, and at each step an agent can either wait at its current vertex or move to an adjacent one, with each action taking one time step.

A set $\mathcal{T}$ contains all unassigned tasks, with new tasks potentially arriving over time. Each task $\tau_j = (s_j, d_j) \in \mathcal{T}$ consists of a pickup location s_j and a delivery location d_j. Agents are *occupied* if assigned to a task and *free* otherwise (they become free upon completing their task). A task requires one agent and an agent can perform one task at a time. To complete a task $\tau_j = (s_j, d_j)$, an agent must first reach s_j and then d_j, following a path π_i. Paths must be conflict-free: agents cannot occupy the same vertex at the same time (*vertex conflicts*) or traverse the same edge in opposite directions simultaneously (*swapping conflicts*). The objective is to compute collision-free paths that complete all tasks, typically minimizing either the *service time* (average completion time per task) or the *makespan* (total completion time for all tasks).

Not all MAPD instances are solvable. A sufficient condition is *well-formedness* [1], which requires: (1) a finite number of tasks, (2) at least as many non-task endpoints (parking locations) as agents, and (3) connectivity between every pair of endpoints (parking, pickup, and delivery locations) via paths that do not pass through other endpoints.

B. MAPD-HPE

Differently from classical MAPD, where agents are the only dynamic entities in the environment, we define a variant called *MAPD in Human-Populated Environments* (*MAPD-HPE*), which explicitly accounts for human presence.

In addition to the classical MAPD elements described in Section III-A, we now consider also a set of humans $\mathcal{H}$. Agents and humans coexist within the same environment G. While agents are constrained to move on G with wait and move actions along the edges (see above), humans can move everywhere with a varying speed (i.e., they can jump over multiple cells per time step when G is a grid). Moreover, while agents are restricted to move within the environment, humans can enter and exit freely, and new individuals may arrive. Thus, $\mathcal{H}$ has no fixed cardinality, and the number of humans in the environment is not constant.

The aim of MAPD-HPE is for agents to complete all tasks while minimizing task completion time, by finding collision-free paths not only with respect to other agents (as in classical MAPD) but also with respect to humans, thus reducing any disturbance caused to them. We assume that agents have dedicated parking locations and that pickup and delivery locations cannot be accessed by humans and respect conditions (2) and (3) of well-formedness (Section III-A). Well-formedness is not ensured since humans may block access to these locations. However, it is reasonable to assume that blocks are temporary, and agents will be able to access their endpoints eventually. Note that the introduction of humans in the environment requires the addition of safety constraints, described in Section IV-B.

IV. PROPOSED ALGORITHM

Given the grid structure of the environment, each cell is associated with a binary human occupancy value; thus, a GNN is a natural choice to model occupancy, as graph convolutions capture the spatial correlations induced by the space topology: human flow at a given cell is inherently influenced by human flows in neighboring cells.

A. GNN-Based Human Occupancy Prediction

The dataset we use for training and test is the ATC dataset [8], described in Section V, which tracks the motion of people in a business and shopping center in Japan.

Each vertex (cell) $v \in V$ is associated with a feature vector $\mathbf{x}_v^t = (o_v^t, \mathrm{tod}_t, \mathrm{dow}_t) \in \mathbb{R}^3$, where $o_v^t \in \{0, 1\}$ is a binary occupancy indicator (whether at least one human was detected in v at time t), $\mathrm{tod}_t \in [0, 1]$ is a normalized time-of-day encoding, and $\mathrm{dow}_t \in [0, 1]$ is a day-of-week encoding. The latter two features are equal for all the vertices and capture the periodic structure of human flow.

The encoding tod_t is computed by mapping the timestamp to the interval $[0, 1]$. Since the dataset recordings are concentrated between 09:00 and 21:00, we normalize time t such that 09:00 corresponds to 0.0 and 21:00 to 1.0; this allows the model to differentiate between early-morning lulls and mid-day peak flows. The encoding dow_t is included by scaling the weekday index to the range $[0, 1]$.

Given a sequence of T_{in} past observations $\{\mathbf{X}^{t-T_{\text{in}}+1}, \ldots, \mathbf{X}^{t}\}$, where $\mathbf{X}^{t} \in \mathbb{R}^{|V|\times 3}$ is the matrix collecting the feature vectors of all vertices at time t, the objective is to predict the occupancy probability at each vertex over the next T_{out} steps:

$$\hat{\mathbf{P}}^{t+1:t+T_{\text{out}}} = f_\theta\big(\mathbf{X}^{t-T_{\text{in}}+1:t},\, G\big)\,. \tag{1}$$

Architecture. We employ a Diffusion Convolutional Recurrent Neural Network (DCRNN) [6], in which every linear transformation of the standard GRU is replaced by a diffusion graph convolution approximated via Chebyshev polynomial filters of order $K{=}2$ ($\mathrm{ChebConv}$) [24]. Concretely, the DCRNN cell computes reset and update gates as:

$$[\mathbf{r}^t,\, \mathbf{u}^t] = \sigma\big(\mathrm{ChebConv}_x\big(\mathbf{X}^t\big) + \mathrm{ChebConv}_h\big(\mathbf{H}^{t-1}\big)\big)\,, \tag{2}$$

where $\mathrm{ChebConv}_x$ and $\mathrm{ChebConv}_h$ denote two Chebyshev graph convolution operators with independent learnable parameters, applied to the input $\mathbf{X}^t$ and the previous hidden state $\mathbf{H}^{t-1}$, respectively. It also computes the candidate hidden state as:

$$\tilde{\mathbf{H}}^t = \tanh\big(\mathrm{ChebConv}_{\text{cand}}\big(\big[\mathbf{X}^t \,\|\, \mathbf{r}^t \odot \mathbf{H}^{t-1}\big]\big)\big)\,, \tag{3}$$

with the standard GRU gating update $\mathbf{H}^t = \mathbf{u}^t \odot \mathbf{H}^{t-1} + (1-\mathbf{u}^t) \odot \tilde{\mathbf{H}}^t$. Here $\mathbf{H}^t \in \mathbb{R}^{N\times d_h}$ is the hidden state matrix ($d_h = 128$, $N = |V|$), σ is the sigmoid activation, and $\odot$ denotes element-wise multiplication. The operator $\mathrm{ChebConv}$ aggregates information from K-hop neighborhoods on G, allowing the model to capture spatial correlations induced by the topology of the space.

Decoding. The model encodes the T_{in}-step input sequence through the recurrent cell to obtain a final hidden state, then decodes autoregressively. At each future step $\tau \in \{1, \ldots, T_{\text{out}}\}$, the network produces a raw scalar output $\hat{y}_v^\tau$ via a shared linear projection; the occupancy probability is then $\hat{p}_v^\tau = \sigma(\hat{y}_v^\tau) \in [0,1]$. The input to the next decoding step is formed by concatenating the predicted occupancy with the tod and dow encodings for that future instant, which are deterministic and thus available at test time.

Training. The model is trained offline by minimizing binary cross-entropy over all cells and prediction steps. The used GPU is a NVIDIA RTX A6000 48GB. At inference time, the trained model is loaded once and kept frozen. A forward pass is executed at most once per simulation step t, producing the full prediction tensor $\hat{\mathbf{P}} \in [0,1]^{T_{\text{out}}\times N}$, which is cached for the remainder of that step and queried by index during path planning.

B. Path Planning Algorithm for MAPD-HPE

Planning Framework. Our planner has two layers: a token-based coordinator that manages task assignment and high-level coordination, and a low-level planner that computes collision-free paths for individual agents.

Token Passing (TP) [1] is a MAPD algorithm in which agents coordinate through a shared data structure, the token, that stores the current planned paths and task assignments. At each time step, free agents are considered for task assignment. Tasks are assigned sequentially by minimizing the Manhattan distance between the agent's position and the pickup location of the task. When an agent needs to plan or replan, it acquires the token, computes a conflict-free path via time-extended A* by treating all paths stored in the token as obstacles, updates its own entry, and releases the token.

Once assigned a task, the robot plans a two-phase path: it first navigates to the pickup location, then to the delivery location. An explicit phase flag (`to_pickup` → `to_delivery`) tracks progress: the transition is triggered when the robot reaches the pickup during execution, ensuring correct phase tracking even under reactive replanning.

Rolling Horizon Strategy. Rather than committing to complete paths, agents plan over a finite horizon H and replan when the remaining path shrinks to ρ or fewer steps. To prevent degenerate loops in which an agent replans indefinitely from the same position, a *replan guard* checks that the last cell of the current plan differs from the assigned goal before triggering a new call; if the path already ends at the goal, no replan is issued.

Reactive Safety Layer. We assume that each agent a has an observation radius obs_a, defined in terms of Manhattan distance, within which it can detect the current positions of nearby humans. Before an agent executes its next planned move, it is passed through a *physical safety filter*: a move m is retained only if it lies at Chebyshev distance greater than one from the current position of every observable human. This constraint defines a $3{\times}3$ exclusion zone around each human, preventing agents from entering any of the eight cells immediately surrounding an occupied cell. An exception is granted for the robot's own current cell, which is always retained as a valid option.

If the next planned move cannot be retained, it means that the agent has to replan. First, all agent's possible moves are passed through the physical safety filter. Second, an *accepted-moves filter* removes any candidate that coincides with the next planned position of another robot. The robot selects the accepted move that maximizes Manhattan distance from the nearest conflicting human, and replans.

C. Integration of DCRNN Predictions into Planning

The occupancy probabilities produced by the DCRNN are incorporated into the low-level search by modifying the cost of each graph edge. When agent a moves from cell u at time t to v at time $t+1$, the edge weight is defined as:

$$w(u, v, t) = (1-\alpha)\frac{d(u,v)}{D_G} + \alpha\, \hat{p}_v^{t+1}, \tag{4}$$

where $d(u,v)$ is the graph edge distance (1 for cardinal moves in our grid), $\hat{p}_v^{t+1}$ is the DCRNN-predicted probability of human presence at v at time $t+1$, D_G is the diameter of the graph G (used to normalize distances to $[0,1]$) and α is a parameter $\in [0,1]$. Note that $\alpha = 0$ recovers pure shortest-path planning and $\alpha = 1$ makes the planner purely occupancy-avoidant; intermediate values trade path length against predicted human exposure.

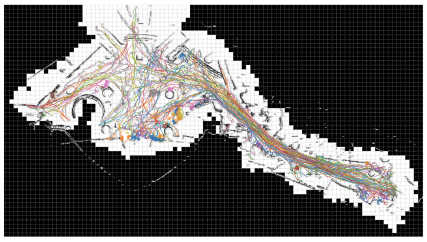

Fig. 1: Discretized environment with humans paths over time.

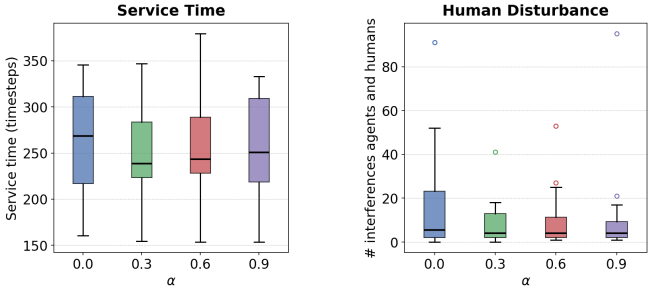

Fig. 2: Service time and number of interferences between agent and human paths varying α.

The DCRNN prediction $\hat{p}_v^{t+1}$ is obtained via an index lookup into the pre-computed $[T_{\text{out}} \times N]$ cache tensor produced once per simulation step (Section IV-A), making per-edge cost evaluation negligible at planning time.

V. EXPERIMENTS

A. Dataset Preprocessing

The considered dataset [8] describes the behavior of people moving in a portion of the ATC business and shopping center in Osaka, Japan. The data were collected between October 24, 2012 and November 29, 2013. Data collection was performed every week on Wednesday and Sunday, from morning until evening (9:40-20:20), consisting of 92 days in total (the last 10 days are used for testing, the rest for training). Each dataset record includes: (i) the time, (ii) the person identifier, (iii) the x and y coordinates of the person, and other properties that are not relevant for this work.

We transform the raw trajectories into a format suitable for our architecture by discretizing both space and time. For space, the environment is divided into a grid with a spatial resolution of 1 m $\times$ 1 m. We define the set of walkable cells, which are those that can be traversed both by robots and humans, as those grid cells that are visited by at least one person across the entire dataset duration, resulting in 1883 grid cells (Fig. 1). Each cell corresponds to a vertex in the environment graph G. For time discretization, we aggregate detections into 1-second intervals. For each discrete time step t, we then compute a 2D histogram of pedestrian positions; each vertex v in G is assigned a binary occupancy value o_v^t which is 1 if at least one person was detected within its corresponding grid cell during that interval, and 0 otherwise.

B. Experimental Setting

The DCRNN receives a window of $T_{\text{in}} = 20$ past occupancy snapshots and predicts the next $T_{\text{out}} = 15$ steps. The path planning horizon is set to $H = 30$ steps; replanning is triggered whenever fewer than $\rho = 15$ steps remain in the path. The α parameter, which balances travel distance and predicted occupancy in the edge weight, is varied over the set $\{0, 0.3, 0.6, 0.9\}$. We test 10 robots with 50 tasks and an observation radius obs_a of 10. Results are averaged over 25 runs, where each run differs in the disposal of tasks and of human paths. Human trajectories are sampled from the test set. Experiments are conducted on a machine equipped with an Intel(R) Xeon(R) Silver 4114 CPU running at 2.20 GHz and with 64 GB of RAM.

The metrics we consider are the service time and the robots' disturbance to humans, quantified by the number of times the paths of the agents interfere with those of humans. Our algorithm is compared with a purely reactive approach, i.e., agents do not use the model when planning their paths, but avoid collisions with humans reactively ($\alpha = 0$).

C. Experimental Results

Fig. 2 reports the distribution of service time and number of interferences between agent and human paths as a function of α across the completed runs. Across all human-aware configurations ($\alpha > 0$), both metrics improve with respect to the purely reactive baseline ($\alpha = 0$). Mean service time decreases from 263.7 timesteps at $\alpha = 0$ to 248.3 at $\alpha = 0.3$ which achieves the best performance; $\alpha = 0.6$ and $\alpha = 0.9$ yield mean values of 254.5 and 255.1 timesteps, respectively. The number of interferences drops from a mean of 14.7 at $\alpha = 0$ to around 8–10 for $\alpha > 0$, confirming that proactive occupancy-aware planning effectively reduces disturbance to humans. Beyond $\alpha = 0.3$, performance does not improve monotonically: higher values of α cause agents to overweight human avoidance at the expense of path efficiency, suggesting that a moderate trade-off between the two objectives is preferable. Overall, these preliminary results support the effectiveness of integrating learned human occupancy predictions into MAPD planning.

VI. CONCLUSION

We address MAPD in human-populated environments, where agents must operate alongside dynamic human behavior. We propose a unified framework that combines GNN-based prediction of human occupancy with a rolling horizon multi-agent planning strategy, enabling agents to adapt to human dynamics. Preliminary results on real-world pedestrian data show the benefits of integrating learning-based models with classical planning in dynamic environments.

This work is still in a preliminary form; thus, future work will aim to address some limitations, such as the generalization aspect, by evaluating the framework on additional environments and comparing it with other behavioral models. Further directions include studying scalability with larger agent fleets and task sets, and integrating online model updates to handle distributional shifts in human behavior.

REFERENCES

[1] H. Ma, J. Li, T. Kumar, and S. Koenig, "Lifelong multi-agent path finding for online pickup and delivery tasks," in *Proc. AAMAS*, 2017, pp. 837–845.

[2] J. P. Fentanes, B. Lacerda, T. Krajník, N. Hawes, and M. Hanheide, "Now or later? Predicting and maximising success of navigation actions from long-term experience," in *Proc. ICRA*, 2015, pp. 1112–1117.

[3] B. Lacerda, F. Faruq, D. Parker, and N. Hawes, "Probabilistic planning with formal performance guarantees for mobile service robots," *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1098–1123, 2019.

[4] L. Bonalumi, B. Flammini, D. Azzalini, and F. Amigoni, "Multi-agent pickup and delivery with external agents," *Robotics and Autonomous Systems*, vol. 191, p. 105000, 2025.

[5] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[6] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. ICLR*, 2018, pp. 1–16.

[7] J. Li, A. Tinka, S. Kiesel, J. W. Durham, T. S. Kumar, and S. Koenig, "Lifelong multi-agent path finding in large-scale warehouses," in *Proc. AAAI*, 2021, pp. 11 272–11 281.

[8] D. Brščić, T. Kanda, T. Ikeda, and T. Miyashita, "Person tracking in large public spaces using 3-D range sensors," *Transactions on Human-Machine Systems*, vol. 43, no. 6, pp. 522–534, 2013. [Online]. Available: https://dil.atr.jp/crest2010_HRI/ATC_dataset/

[9] T. P. Kucner, M. Magnusson, S. Mghames, L. Palmieri, F. Verdoja, C. S. Swaminathan, T. Krajník, E. Schaffernicht, N. Bellotto, M. Hanheide *et al.*, "Survey of maps of dynamics for mobile robots," *The International Journal of Robotics Research*, vol. 42, no. 11, pp. 977–1006, 2023.

[10] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[11] T. Krajník, J. P. Fentanes, J. M. Santos, and T. Duckett, "FreMEn: Frequency map enhancement for long-term mobile robot autonomy in changing environments," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 964–977, 2017.

[12] S. Molina, G. Cielniak, T. Krajník, and T. Duckett, "Modelling and predicting rhythmic flow patterns in dynamic environments," in *Proc. TAROS*, 2018, pp. 135–146.

[13] T. Vintr, S. Molina, R. Senanayake, G. Broughton, Z. Yan, J. Ulrich, T. P. Kucner, C. S. Swaminathan, F. Majer, M. Stachová, A. J. Lilienthal, and T. Krajnik, "Time-varying pedestrian flow models for service robots," in *Proc. ECMR*, 2019, pp. 1–7.

[14] T. P. Kucner, M. Magnusson, E. Schaffernicht, V. H. Bennetts, and A. J. Lilienthal, "Enabling flow awareness for mobile robots in partially observable environments," *Robotics and Automation Letters*, vol. 2, no. 2, pp. 1093–1100, 2017.

[15] Y. Zhu, A. Rudenko, T. P. Kucner, L. Palmieri, K. O. Arras, A. J. Lilienthal, and M. Magnusson, "CLiFF-LHMP: Using spatial dynamics patterns for long-term human motion prediction," in *Proc. IROS*, 2023, pp. 3795–3802.

[16] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proc. CVPR*, 2016, pp. 961–971.

[17] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proc. CVPR*, 2018, pp. 2255–2264.

[18] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. ECCV*, 2020, pp. 683–700.

[19] T. Vintr, Z. Yan, K. Eyisoy, F. Kubiš, J. Blaha, J. Ulrich, C. S. Swaminathan, S. Molina, T. P. Kucner, M. Magnusson, G. Cielniak, J. Faigl, T. Duckett, A. J. Lilienthal, and T. Krajník, "Natural criteria for comparison of pedestrian flow forecasting models," in *Proc. IROS*, 2020, pp. 11 197–11 204.

[20] S.-Y. Lo, B. Fernandez, P. Stone, and A. L. Thomaz, "Towards safe motion planning in human workspaces: A robust multi-agent approach," in *Proc. ICRA*, 2021, pp. 7929–7935.

[21] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. ISRR*, 2011, pp. 3–19.

[22] L. Heuer, L. Palmieri, A. Mannucci, S. Koenig, and M. Magnusson, "Benchmarking multi-robot coordination in realistic, unstructured human-shared environments," in *Proc. ICRA*, 2024, pp. 14 541–14 547.

[23] L. Heuer, Y. Zhu, L. Palmieri, A. Rudenko, A. Mannucci, S. Koenig, and M. Magnusson, "Conflict mitigation in shared environments using flow-aware multi-agent path finding," *arXiv preprint arXiv:2603.12736*, 2026.

[24] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NeurIPS*, 2016, pp. 3844–3852.