2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

A congestion-aware path planning method considering crowd spatial-temporal anomalies for long-term autonomy of mobile robots

Zijian Ge, Jingjing Jiang, and Matthew Coombes
Department of Aeronautical and Automotive Engineering
Loughborough University, Leicestershire, LE11 3TU, UK.
Email: {Z.Ge, j.jiang2, M.J.Coombes}@lboro.ac.uk

***Abstract*—A congestion-aware path planning method is presented for mobile robots during long-term deployment in human occupied environments. With known spatial-temporal crowd patterns, the robot will navigate to its destination via less congested areas. Traditional traffic-aware routing methods do not consider spatial-temporal anomalies of macroscopic crowd behaviour that can deviate from the predicted crowd spatial distribution. The proposed method improves long-term path planning adaptivity by integrating a partially updated memory (PUM) model that utilizes observed anomalies to generate a multi-layer crowd density map to improve estimation accuracy. Using this map, we are able to generate a path that has less chance to encounter the crowded areas. Simulation results show that our method outperforms the benchmark congestion-aware routing method in terms of reducing the probability of robot's proximity to dense crowds.**

***Index Terms*—Congestion-Aware Path Planning, Partially Updated Memory, Long-Term Autonomy, Crowd Spatial-Temporal Anomalies**

I. INTRODUCTION

Due to huge advances in recent years in the area of AI, and the general decreasing cost of robotic systems, more intelligent robots have been deployed in human-populated environments, such as museums, train stations and shopping malls. Its crucial for safe interactions between humans and robots in these co-exist spaces [1].

Traditionally, routing in robotics aims to find a path based on the minimal travel cost and the calculated route may cross the area with dense crowds. Many motion planners have been developed to avoid dynamic obstacles in real time, such as the Dynamic Window Approach (DWA) [2]. However, the computational costs for such local planning methods could be incredibly high if the number of the objects (*i.e.*, dynamic and static obstacles) increases, which potentially undermines both efficiency and safety in navigation. Even worse situations (such as robots stop moving) may occur if there are too many obstacles nearby. Therefore, the understanding of human spatial-temporal distributions could let robots navigate with minimal interactions with crowds, hence reduce risks of collisions with humans.

Most crowds' spatial-temporal distributions have specific features and are predictable based on their historical data [3]. Despite the patterns collected from long periods of historical data, there are unusual crowd activities at unexpected moments or places which cannot be easily predicted. Hence, how to deal with these anomalous crowd patterns that are unrecognized from the historical knowledge becomes a challenging problem. In our work, we build a Partially Updated Memory (PUM) by utilizing these observed anomalies to better estimate crowds spatial-temporal distributions. And then, we propose a congestion-aware path planning algorithm which incorporates the PUM model, enable robots to plan an end-to-end path that evaluates the total costs (including congestion costs and travel distance costs), and significantly reduce the computational cost in local planning.

The crowd prediction in our work is very similar to traffic flow prediction since both are aiming to predict the volume and distribution changes of agents (cars or pedestrians) in a network. Furthermore, the ability to forecast traffic congestion in intelligent transportation systems for autonomous vehicles has been widely studied. An Artificial Neural Network (ANN)-based prediction model using historical vehicle GPS data is proposed to predict real-time traffic status at a given location [4]. To make use of the sequential nature of data (in a dataset, such as videos) in time series, long short-term memory algorithm is applied to capture the traffic temporal features in [5] and [6]. To predicting dynamic environment in robotics, Frequency Map Enhancement, a novel approach proposed in [7] based on their early concepts in [8] and [9], formulates the dynamic patterns of the environment as a multiple spectrum map that frequencies can be calculated by Fourier Transform. By utilizing such frequency map, robots can predict the future environmental states and improve their long-term performance in the changing space. Other crowd spatial-temporal formats, such as the dynamic clustering algorithm in [10] and Gaussian process regression model in [11], demonstrate the importance of reasoning human behaviours for robot long-term autonomy. In this paper, we do not work on prediction models of crowds spatial-temporal distributions. Instead, we will make use of these models to help robots to plan congestion-aware paths.

For path planning using the predicted crowd distribution, [1] integrates the crowd's macroscopic properties into a 4D map with spatial-temporal density values, then a CNN-RNN

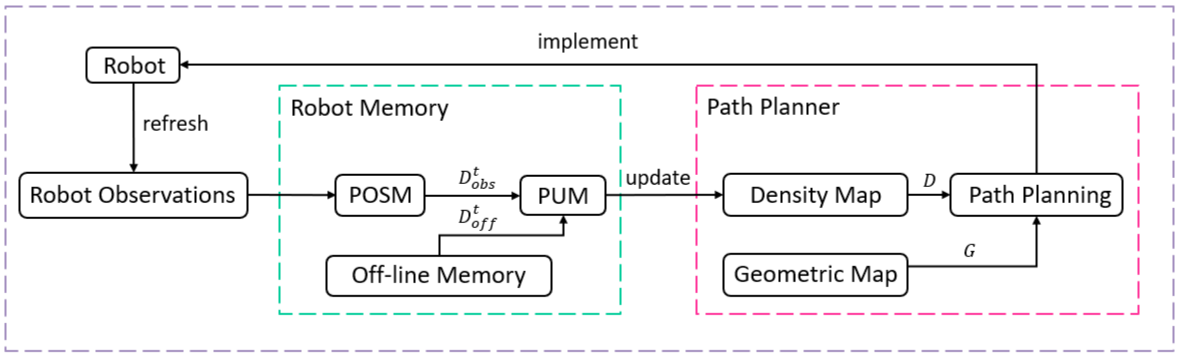

Fig. 1: Architecture of the proposed congestion-aware path planning using PUM.

network is used to predict the pedestrian density and plan a path with less invasion to the dense crowd. There are also learning-based algorithms which apply Markov-chain to guide vehicles to avoid congestion [12] or utilize deep reinforcement learning methods on traffic network edges to improve travel efficiency [13]. However, most aforementioned approaches make predictions based on environments' spatial-temporal regularities while discarding anomalies as they could potentially reduce the prediction accuracy and cause over-fitting in learning-based methods. This results in reduction of travel efficiencies in some scenarios. Therefore, we propose a congestion-aware path planning method, the structure diagram of which is shown in Fig. 1, using partially updated memory (PUM) to reduce the impacts of crowd anomalous activities and improve robots' long-term adaptivity in crowded environments. The details of PUM can be seen in Section II-B.

This paper offers three novel contributions towards addressing the problem of robot long-term adaptivity in crowded environments:

- A time-dependent PUM algorithm, taking anomalies of the crowds' spatial-temporal regularities into consideration, is proposed to reduce estimated error on unpredictable but observable events.
- A multi-layer crowd density map based on kernel density and PUM is generated, which estimates spatial-temporal features of crowd behaviour in a more accurate way than the off-line memory.
- By taking PUM dynamic density map into consideration, a novel adaptive congestion-aware path planner is built aiming to seek a path with not only shortest distance, but also minimal probability of encountering dense crowds.

The remainder of the paper is organized as follows. Section II introduces the construction of the crowd density map by the PUM algorithm. The congestion-aware path planning algorithm using the proposed estimated density map is proposed in Section III, while its effectiveness is demonstrated by simulation results, as well as the comparison with the benchmark method in a shopping center In Section IV. Finally, conclusions and future work are discussed in Section V.

II. Construction of the Crowd Density Map

In this section we introduce the multi-layer crowd density map and explain how our PUM model is used to update the density map over time. This will then be used to predict the crowd density distribution which is important to our congestion-aware path planning algorithm introduced in Section III.

A. Spatial kernel density estimation

Kernel Density Estimation (KDE) is a popular machine learning technique applied to estimate the spatial pattern of a data distribution. It works as a non-parametric geostatistical tool widely used in traffic risk assessments for autonomous vehicles [14] and city crime mapping [15]. In this paper, KDE is used to produce a smooth crowd density surface over Euclidean space by calculating each pedestrian's location. Furthermore, mobile robots (studied in this paper) can make use of the estimated crowd spatial density pattern and avoid interactions with dense crowds. The formulation of the spatial KDE is given as:

$$\hat{f}_s(x,y) = \frac{1}{nh_s^2}\sum_{i=1}^{n} K_s\left(\frac{x-x_i}{h_s}, \frac{y-y_i}{h_s}\right), \tag{1}$$

where (x_i, y_i) represents the location of Pedestrian i, for all $i \in \{1, \ldots n\}$ in X-Y space domain. Note that we use the symbol n to denote the sample size. Moreover, $\hat{f}_s(x,y)$ represents the estimated density at position (x,y). h_s and K_s are the spatial bandwidth and the kernel function respectively. There are many available kernel selections, such as Gaussian, Epanechnikov and Quartic [16]. However, research in [17] and [18] have proved that the choice of kernel function is of little importance, especially when n increases. In this paper, K_s is defined as the most commonly employed two-dimensional Gaussian kernel, which is written as:

$$K_s(x,y) = e^{-\left(\frac{||x-x'||^2}{2h_s^2} + \frac{||y-y'||^2}{2h_s^2}\right)}, \tag{2}$$

where (x', y') is the kernel center with search bandwidth h_s which determines the spread of the data in 2D space. Note that h_s is also considered as the most critical parameter in any kernel density estimation. If h_s is too small, the results are under-smoothed and is hard to interpret, while a larger h_s returns a surface with smoother but more clustered data

[15]. In the theory of machine learning, the approximated density is expected to fit the real data distribution. In our work, since the spatial density map is used for robot routing, a proper data-driven bandwidth value directly determines the robot's path planning performance. Various methods have been studied to seek an optimized bandwidth value, such as cross-validation [19] and multivariate plug-in [20]. In this case, the error function, in terms of bandwidth value, is defined as:

$$MISE(h) = E \iint (\hat{f}(x,y) - f(x,y))^2 dxdy, \tag{3}$$

where $\hat{f}(x,y)$ and $f(x,y)$ are the estimated density and the real density at position (x,y) respectively. From [21], we know that the $MISE(h)$ can be formulated as an asymptotic mean integrated squared error $AMISE(h)$ and the optimal h_s can be found by calculating the pole of the $\partial AMISE(h)/\partial h$. Therefore, the optimal value can be initially obtained by:

$$h_s = (\frac{4\hat{\sigma}^5}{3n})^{\frac{1}{5}} \approx 1.06\hat{\sigma}n^{-\frac{1}{5}} \tag{4}$$

where $\hat{\sigma}$ is the standard deviation of pedestrian's samples.

B. Partially updated memory

To navigate a mobile robot to a target position while avoiding crowded areas, we have to anticipate the distribution of people at any given time and location. We assume that the long-term memory for the crowd density distribution is available and can be used to estimate the crowd's spatial density patterns. Such spatial-temporal regularities can be derived from a large dataset collected via long-term (*i.e.*, months and even years) sensor deployment. [22] gives an example of how a fixed LIDAR mounted to a mobile robot can be used to detect and track human beings. Despite long-term regularities, there are always anomalies in dynamic environments which deviate the real-time density from the off-line memory. Hence, we propose PUM to model such anomalies based on temporal observations of unexpected events.

In this work, we assume that new observations can be accessed from the global shared information in an intelligent network which includes smart agents, as well as deployed fix sensors, operating in the environment. The new observed data at certain area can be used to partially update the global crowd pattern. Unlike the Markov chain-based probability equation which outputs the binary state for a given time [23], we update each map grid by a continuous value evolved over time. For each cell in the grid map, the real-time density estimation at time t, S^t_{est}, is given as follows:

$$S^t_{est} = k_\alpha \cdot S^{t_o}_{obs} + k_\beta \cdot S^t_{off}, \tag{5}$$

where the partially observed short-term memory (POSM), denoted by $S^{t_o}_{obs}$, is the last observation at time t_o and is weighted by k_α, while S^t_{off} denotes the off-line memory at time t which is weighted by k_β. Note that k_α and k_β are two time-dependent parameters that can be calculated as:

$$k_\alpha = \frac{1}{e^{\delta\cdot(t-t_o)}}, \quad k_\beta = \frac{e^{\gamma\cdot(t-t_o)} - 1}{e^{\gamma\cdot(t-t_o)} + 1}, \tag{6}$$

where δ denotes the forgetting speed coefficient which determines the POSM effect duration, γ represents the recovery speed coefficient of the off-line memory. By (6) it is easy to conclude that both k_α, $k_\beta \in [0,1]$. When there is no observed data saved in the POSM (*i.e.*, $t_d = -\infty$), k_α= 0, k_β= 1, PUM only relies on the off-line memory. When observation emerges (*i.e.*, $t = t_o$), then k_α= 1, k_β= 0, POSM starts to work. Eventually, with the increase of t, POSM is decayed as k_α drops to 0, whereas the off-line memory recovers as k_β grows back to 1.

Remark 1: k_α and k_β are two independent variables, which means that (k_α + k_β) may not equal to 1. This is because, in practice, the anomalies in a certain area could overlay with the original spatial pattern. Such phenomenon can be reflected by k_α + k_β > 1.

Algorithm 1: Partially updated memory

Input: Current time t ;
Last observation time t_o, ;
Off-line memory S^t_{off} ;
POSM $S^{t_o}_{obs}$
Output: estimated spatial pattern S^t_{est} at time t
1 **Initialize**: $k_\alpha = 0$, $k_\beta = 1$;
2 Update time interval $t_z = 5s$;
3 **if** *observation occurs* **then**
4   POSM decay time $t_d = 0$
5   $k_\alpha = 1$, $k_\beta = 0$;
6   **while** *POSM is not empty* **do**
7     $t = t + t_z$;
8     $t_d \leftarrow t - t_o$;
9   k_α, k_β *update* according to (6) ;
10     $S^t_{est} \leftarrow k_\alpha \cdot S^{t_o}_{obs} + k_\beta \cdot S^t_{off}$;
11   **if** $S^{t_o}_{obs} > S^{t_o}_{off}$ **then**
12   **if** $S^t_{est} < S^t_{off}$ **then**
13         $S^t_{est} = S^t_{off}$
14   **end**
15   **end**
16   **else if** $S^{t_o}_{obs} < S^{t_o}_{off}$ **then**
17   **if** $S^t_{est} > S^t_{off}$ **then**
18         $S^t_{est} = S^t_{off}$
19   **end**
20   **end**
21   **end**
22 **end**

Physical boundaries are added to the estimated density S^t_{est}, aiming to ensure it is no smaller than the off-line prediction if POSM is higher than the off-line value, while keep its value smaller or equal to the off-line value if the POSM is lower than the off-line value (see Lines 11-19 in Algorithm 1). The reason for setting such physical boundaries for S^t_{est} is to make sure that the robot does not overestimate or underestimate the anomalies. The details of PUM can be seen in Algorithm 1, while Fig. 2 is an example intuitively shows how the density map is updated according to PUM.

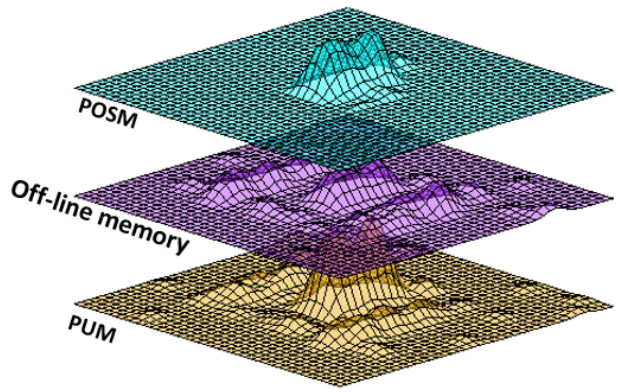

Fig. 2: Crowd density map update: PUM layer is generated by overlaying two layers (POSM, off-line memory).

Remark 2: In reality, observations can occur at any moment within the duration time of an anomalous activity. Thus, the robot is difficult to accurately estimate PUM which is determined by two parameters: forgetting speed δ and recovery speed γ. If δ is large, the estimation curve tends to be more quickly converge to the off-line memory, while the POSM duration is too short to be utilized. On the other hand, if δ is small, the POSM effect lasts for a quite long period which may overestimate the anomaly. Instead of arbitrarily selecting δ and γ, they can be tuned to fit the historical anomalies. In other words, the mean value calculated from those outlier data is selected in our algorithm. Such mean value can be obtained from the robot's long-term deployment.

III. Congestion-aware path planning

With the crowd density map generated by the PUM model in the previous section, we propose a path planning method which finds an optimal path that will avoid crowded areas completely, or at least lower the probability of entering crowded areas. Unlike pure distance-based path planning methods, we consider both travel distance costs and congestion costs by utilizing the geometry map, as well as the estimated density map.

To be more specific, the cost of each cell $n.C$ in the grid map is calculated by:

$$n.C = dist(n, n.s, n.g) + \epsilon \cdot den(n, n.s, n.g, D), \qquad (7)$$

where $dist(n, n.s, n.g)$ describes the overall distance cost of the current node (n) to the goal ($n.g$) and start ($n.s$) nodes, while $den(n, n.s, n.g, D)$ represents the sum of the estimated density costs along the Manhattan distance from the current node to the goal node and along the traveled path from the start node to the current node. It should be noticed that the estimated density map D used in this planning algorithm is derived from Algorithm 1. ϵ is the weighting that determines the trade-off between the distance and the density costs. Greater ϵ value makes the planner be more sensitive to congestion areas.

The pseudo code of the proposed congestion-aware path planning algorithm is given by Algorithm 2.

Algorithm 2: Congestion-aware path planning

Input: Current time t ;
Last observation time t_o, ;
Geographical map G ;
Off-line density map D^t_{off} ;
POSM density map $D^{t_o}_{obs}$;
Start location: $n.s$;
Goal location: $n.g$;
Output: A calculated path from start node to goal node

Initialize waiting list: $WaitList = \emptyset$
Initialize Used list: $UsedList = \emptyset$
$D \leftarrow Algorithm\ 1(D^t_{off}, D^{t_o}_{obs}, t, t_o)$
Insert Node $(n.s, n.g)$ to $WaitList$
while *WaitList is not empty* **do**
  Find a father node with minimal cost:
  $n_f \leftarrow \min(WaitList.n.C)$
  Check if n_f reaches the goal node:
  if $n_f == n.g$ **then**
    GetPath(n.s,$UsedList$)
    return $path$
  end
  $next \leftarrow$ SearhMotion(n_f, G)
  foreach $n \in next$ **do**
    n.dist $\leftarrow$ dist(n, n.g, n.s)
    n.den $\leftarrow$ den(n, n.s, n.g, D)
    n.C $\leftarrow \epsilon.n.den + n.dist$
    if $n \notin$ *WaitList*, $n \in$ *ObstalceFree* **then**
      insert n to WaitList
      set n_f as the father node
    end
    else if $n \in$ *WaitList* **then**
      if *the new path from* n_f *to* n *is shorter* **then**
        update father node n_f to n
      end
    end
  end
  Insert n_f to UsedList
end

IV. Simulation Results

The proposed PUM model was first evaluated at a small-scale virtual shopping mall with crowds to validate its necessity on mitigating the effects of crowd anomalies. Once the PUM model is evaluated, the proposed congestion-aware path planning algorithm was assessed and compared with the benchmarks.

A. Simulation Setup

Pedestrian Dynamics, a software allows users to simulate large volumes of pedestrians coming and going in a customized map, is used to simulate crowds behaviour in a virtual shopping mall, by letting pedestrians follow a certain pre-set macroscopic regularity with random distributions. Such scenario is pretty close to the pedestrians regulated behaviours in real world [3]. Note that the microscopic behaviours of pedestrians in simulation follow the social force model in [24].

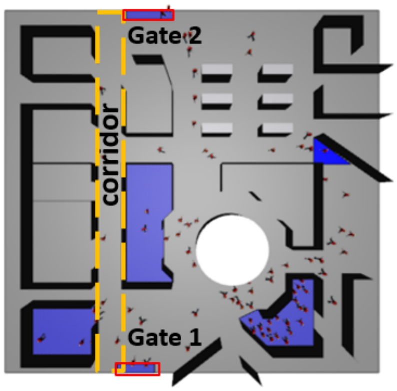

Fig. 3: Geometry map of the virtual shopping mall.

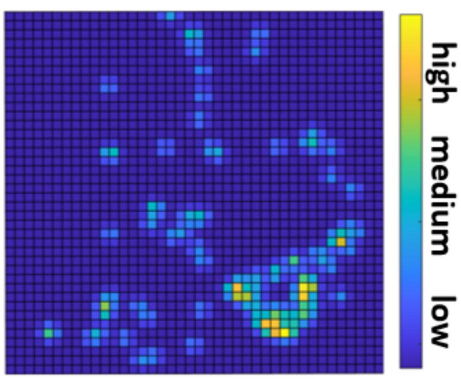

Fig. 4: The density map generated based on the crowd distribution shown in Fig. 3.

In a scenario created in *Pedestrian Dynamics*, we first define all the pedestrians follow the pre-set action preferences with random walking speeds within a fixed range (*i.e.,* 1.2-1.5 m/s). There are two gates, Gate 1 and Gate 2, at the shopping mall highlighted in Fig. 3. The crowds were set to enter the mall via these gates every 5 minutes. After visiting all pre-defined shopping areas, they left through the gates. This represents the crowds' regulated pattern. In addition, to represent unpredictable spatial-temporal anomalies, another twenty groups of people with random sizes were set to enter the mall at random time and then leave after around 5 minutes.

To quantify the crowd distribution in the shopping mall, the first step is to find pedestrians' Cartesian coordinates. This can be obtained from the top camera view of *Pedestrian Dynamics* via the pedestrian tracking techniques described in [25]. Furthermore, with those extracted pedestrians' locations, kernel density estimation in Section II-A can be applied to generate a density map in MATLAB as shown in Fig. 4. Thus, with the generated density map, we can implement the proposed congestion-aware path planner, as well as the benchmark algorithms, to plan paths with randomly defined start and goal positions.

B. Evaluation of PUM

Although the off-line density map needs to be generated based on historical data in reality, we directly use the pre-specified spatial-temporal patterns applied in *Pedestrian Dynamics* in this work, *i.e.,* assuming that we have an excellent off-line density map to represent regular patterns of the crowds. Note that such assumption is reasonable as there are plenty reasonable good works in [10] and [11] on off-line density map generation.

Anomalies in the environment are defined as events or activities happening at unusual places and moments, that are not consistent with the off-line memory. Nevertheless, such anomalies only last for a short period of time, and do not influence the long-term spatial-temporal regularities. If such unexpected activities are observed and transferred to a POSM, it is going to affect the routing results according to Eq. (5). Finally, with the POSM fading, the PUM gradually converges to the original off-line memory.

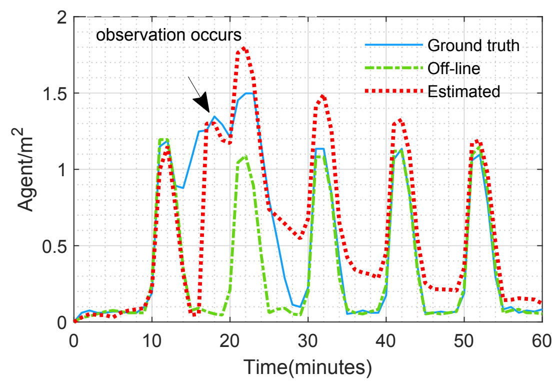

Fig. 5: Comparison of density estimation based on different methods (blue solid: ground truth, green dash-dotted: off-line memory, red dotted: estimated density according to Algorithm 1) in a virtual shopping mall, sample extracted from 60 minutes' crowd flow at the corridor (labeled in Fig. 3).

In the virtual shopping mall as shown in Fig. 3, suppose the robot has received an observation on the unusual crowd behaviour in the corridor. The robot's memory update can be reflected by the sudden increase of the estimated density curve shown in Fig. 5 where the observation emerges at around 16 minutes. By comparing the ground truth time history of the corridor's crowd density, we can see that the proposed PUM is able to follow the trend and performs better than the none anomaly-aware off-line memory.

C. Benchmark methods and evaluation metrics

The benchmark methods selected in this paper are the traditional A* algorithm, and the traffic-aware evacuation routing algorithm in [26] which only considers the off-line memory of the density map.

To analyze the performance of our PUM based congestion-ware path planning method, we define two evaluation indexes as:

$$V_{density} = \frac{density_{ben} - density_c}{density_{ben}}, \quad (8)$$

$$V_{distance} = \frac{length_{ben} - length_c}{length_{ben}}, \quad (9)$$

where $V_{density}$ and $V_{distance}$ are the reduced percentages of the density and the distance cost for our method (compared to the benchmark methods), respectively. Moreover, $density_{cs}$ and $density_{ben}$ represent the total encountered density cost of the proposed method and the benchmark methods, respectively, while $length_{cs}$ and $length_{ben}$ denote the path lengths calculated from the corresponding methods respectively.

D. Performance analysis

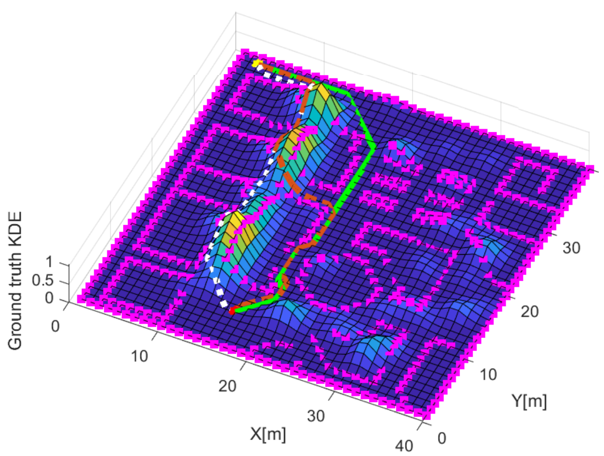

Fig. 6: The results of the proposed method with memory update (green solid path) compared with the benchmark [26] (red dotted path), and the classic A* algorithm (white dotted path). The high dense crowds in the corridor, simulated as anomalies, are not included in the off-line memory but are observable.

To verify the generality of the simulation results, 30 path planning trials have been conducted. Fig. 6 shows how both algorithms perform in a typical scenario when an unexpected dense crowd occurs in the corridor (labeled in Fig. 3). The overall simulation results for the 30 trials is summarized, in terms of density cost and path length, in Table I. Note that 'observation' in the table means the number of anomalies has been used as POSM to calculate the path. Note that the computational costs for different path planning methods are almost the same as PUM is not computationally expensive.

From Table I, it can be seen that the path calculated by our method encounters less density cost than other methods, and the improvement can be significant, especially when more observations on the anomalies are received. However, for the path length, both our method and the benchmark traffic-aware method [26] perform less efficiently compared to the traditional A* algorithm, because both crowd-aware paths need to take detours in certain areas to avoid the dense crowds resulting in longer travel distance. In addition, the differences between the path lengths from two traffic-aware methods can be ignored, since such factor can be influenced by the map geometry features, as well as the locations and types of anomalies. When observation is zero, *i.e.,* without POSM updates, our method performs similar to the benchmark method [26] in terms of reducing density cost.

$mean\ V_{distance}$ $mean\ V_{density}$	Proposed congestion-aware path planning		
	observation:0	observation:10	observation:20
A*	-43.84% 43.77%	-54.29% 48.57%	-32.12% 63.63%
benchmark [26]	-4.01% 3.58%	2.90% 15.07%	-6.12% 58.33%

TABLE I: Comparison results of the proposed congestion-aware path planning method against benchmarks, in terms of density cost and path length.

V. Conclusions and Future Work

A congestion-aware path planning method has been developed that uses the estimated kernel density map derived from the PUM. The method uses not only the crowd's long-term spatial-temporal patterns (*i.e.,* off-line memory), but also observed anomalies as auxiliary information to the robot navigation. A case study was conducted in a virtual scenario in *Pedestrian Dynamics*, to prove the feasibility of the PUM's estimation. Although it cannot perfectly predict the real-time density, it can track the right trend and improve the robot's awareness to those unexpected situations. Simulation results of comparisons with benchmark methods show that our method depends largely on the number of observations on anomalies. In fact, with the increase of the observations, the proposed congestion-aware path planning method can significantly outperform other path planning methods with respect to the reduction in the possibility of navigating to the congestion areas.

The current crowd density map we built is deterministic and the crowd spatial-temporal patterns in experiments are synthetic. However, due to disturbances and uncertainties in environments, as well as sensor noises, we will consider non-deterministic situations and use Bayesian belief model to improve the accuracy of the estimated crowd distribution and verify the method in the real experiments considering travel time as the evaluation factor.

References

[1] S. H. Kiss, K. Katuwandeniya, A. Alempijevic, and T. Vidal-Calleja, "Probabilistic Dynamic Crowd Prediction for Social Navigation," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, no. Icra, pp. 9269–9275, 2021.

[2] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[3] S. Molina, G. Cielniak, and T. Duckett, “Go with the flow: Exploration and mapping of pedestrian flow patterns from partial observations,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 9725–9731, 2019.

[4] D. Zhu, H. Du, Y. Sun, and N. Cao, “Research on path planning model based on short-term traffic flow prediction in intelligent transportation system,” *Sensors (Switzerland)*, vol. 18, no. 12, 2018.

[5] X. Luo, D. Li, Y. Yang, and S. Zhang, “Spatiotemporal traffic flow prediction with knn and lstm,” *Journal of Advanced Transportation*, vol. 2019, 2019.

[6] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, “Long short-term memory neural network for traffic speed prediction using remote microwave sensor data,” *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[7] 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.

[8] T. Krajník, J. P. Fentanes, G. Cielniak, C. Dondrup, and T. Duckett, “Spectral analysis for long-term robotic mapping,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3706–3711, 2014.

[9] T. Krajnik, J. P. Fentanes, O. M. Mozos, T. Duckett, J. Ekekrantz, and M. Hanheide, “Long-term topological localisation for service robots in dynamic environments using spectral maps,” *IEEE International Conference on Intelligent Robots and Systems*, no. Iros, pp. 4537–4542, 2014.

[10] R. Ambrus, J. Ekekrantz, J. Folkesson, and P. Jensfelt, “Unsupervised learning of spatial-temporal models of objects in a long-term autonomy scenario,” *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5678–5685, 2015.

[11] M. P. Zapf, M. Kawanabe, and L. Y. M. Saiki, “Pedestrian Density Prediction for Efficient Mobile Robot Exploration,” *IEEE International Conference on Intelligent Robots and Systems*, no. 2, pp. 4615–4622, 2019.

[12] Y. Zheng, Y. Li, C. M. Own, Z. Meng, and M. Gao, “Real-time predication and navigation on traffic congestion model with equilibrium Markov chain,” *International Journal of Distributed Sensor Networks*, vol. 14, no. 4, 2018.

[13] C. Chen, J. Jiang, N. Lv, and S. Li, “An Intelligent Path Planning Scheme of Autonomous Vehicles Platoon Using Deep Reinforcement Learning on Network Edge,” *IEEE Access*, vol. 8, pp. 99 059–99 069, 2020.

[14] C. Ryan, F. Murphy, and M. Mullins, “Spatial risk modelling of behavioural hotspots : Risk-aware path planning for autonomous vehicles,” *Transportation Research Part A*, vol. 134, no. January, pp. 152–163, 2020. [Online]. Available: https://doi.org/10.1016/j.tra.2020.01.024

[15] Y. Hu, F. Wang, C. Guin, and H. Zhu, “A spatio-temporal kernel density estimation framework for predictive crime hotspot mapping and evaluation,” *Applied geography*, vol. 99, pp. 89–97, 2018.

[16] Z. Xie and J. Yan, “Kernel density estimation of traffic accidents in a network space,” *Computers, environment and urban systems*, vol. 32, no. 5, pp. 396–406, 2008.

[17] O. Schabenberger and C. A. Gotway, *Statistical methods for spatial data analysis: Texts in statistical science*. Chapman and Hall/CRC, 2017.

[18] T. C. Bailey, A. C. Gatrell *et al.*, *Interactive spatial data analysis*. Longman Scientific & Technical Essex, 1995, vol. 413, no. 8.

[19] C. Brunsdon, “Estimating probability surfaces for geographical point data: An adaptive kernel algorithm,” *Computers & Geosciences*, vol. 21, no. 7, pp. 877–894, 1995.

[20] M. P. Wand, M. C. Jones *et al.*, “Multivariate plug-in bandwidth selection,” *Computational Statistics*, vol. 9, no. 2, pp. 97–116, 1994.

[21] B. W. Silverman, *Density estimation for statistics and data analysis*. Routledge, 2018.

[22] T. Vintr, Z. Yan, T. Duckett, and T. Krajník, “Spatio-temporal representation for long-term anticipation of human presence in service robotics,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 2620–2626, 2019.

[23] J. M. Santos, T. Krajník, and T. Duckett, “Spatio-temporal exploration strategies for long-term autonomy of mobile robots,” *Robotics and Autonomous Systems*, vol. 88, pp. 116–126, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2016.11.016

[24] D. Helbing and P. Molnar, “Social force model for pedestrian dynamics,” *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[25] X. Wu, B. Xu, Y. Zheng, H. Ye, J. Yang, and L. He, “Neurocomputing Fast video crowd counting with a Temporal Aware Network,” *Neurocomputing*, vol. 403, pp. 13–20, 2020. [Online]. Available: https://doi.org/10.1016/j.neucom.2020.04.071

[26] Z. Zhang, H. Liu, Z. Jiao, Y. Zhu, and S.-C. Zhu, “Congestion-aware evacuation routing using augmented reality devices,” *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2798–2804, 2020.