Dynamic Risk Density for Autonomous Navigation in Cluttered Environments without Object Detection

Alyssa Pierson[1], Cristian-Ioan Vasile[1,2], Anshula Gandhi[1], Wilko Schwarting[1], Sertac Karaman[2], and Daniela Rus[1]

***Abstract*— In this paper, we examine the problem of navigating cluttered environments without explicit object detection and tracking. We introduce the *dynamic risk density* to map the congestion density and spatial flow of the environment to a cost function for the agent to determine risk when navigating that environment. We build upon our prior work, wherein the agent maps the density and motion of objects to an occupancy risk, then navigate the environment over a specified risk level set. Here, the agent does not need to identify objects to compute the occupancy risk, and instead computes this cost function using the occupancy density and velocity fields around them. Simulations show how this dynamic risk density encodes movement information for the ego agent and closely models the object-based congestion cost. We implement our dynamic risk density on an autonomous wheelchair and show how it can be used for navigating unstructured, crowded and cluttered environments.**

I. Introduction

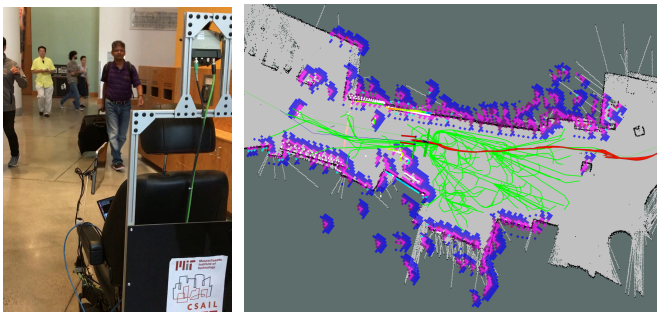
Autonomous mobility comes in many forms and packages: cars, trucks, golf carts, wheelchairs, and delivery buggies, to name a few. As we integrate these vehicles into human-centric environments, autonomous systems must detect and adapt to the presence of moving agents and obstacles. The correct response depends on the level of congestion and the nature of the motion. This paper presents a solution to navigation in clutter with dynamic obstacles agnostic to the classification of the obstacle. We formalize a method of estimating the density and motion of the clutter with a congestion cost function. Using the density and velocity field of the environment, we generate the **dynamic risk density** function, which encodes the location and movement of dynamic obstacles into a single cost function. Using the dynamic risk density, we demonstrate navigating crowded hallways with an autonomous wheelchair that is able to avoid collisions with both static obstacles and pedestrians. Figure 1 shows our autonomous wheelchair platform and an illustration of our path planning algorithm.

This paper focuses on autonomous mobility platforms for individuals, such as autonomous wheelchairs, whose features are needed to navigate crowds, buildings, plazas, airports, and other cluttered and unstructured environments.

[1]Computer Science & Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA [apierson, cvasile, anshula, wilkos, rus]@mit.edu

[2]Laboratory of Information and Decision Systems (LIDS), MIT, Cambridge, MA, USA sertac@mit.edu

This work supported in part by NSF Grant 1723943, the Office of Naval Research (ONR) Grant N00014-18-1-2830, and by Toyota Research Institute (TRI). This article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. Their support is gratefully acknowledged.



(a) (b)

Fig. 1: (a) Autonomous wheelchair platform and (b) dynamic risk density cost function with RRT* path generation.

In recent years, sensing and perception technology has made significant advances in our ability to parse and understand an environment. However, these perception systems remain prohibitively expensive at the consumer scale. Furthermore, many of these approaches require advanced algorithms with significant computational overhead, or are limited in the environmental complexity they can handle. Here, we are interested in studying mobility solutions achievable with minimal sensor data and light computational loads. Our goal is to rapidly assess and translate risks for reliable, collision-free motion in diverse, cluttered, and unstructured environments. Our previous work proposed using "risk level sets" to navigate a cluttered environment. In [1], we introduced a cost function that quantified the level of congestion based on the positions and movement of other agents and obstacles in the system. This function mapped the environment clutter to a metric of risk for an ego agent planning a path through the environment. We also introduced varying risk thresholds for the agents, such that from the congestion cost function, the agents choose their allowable risk tolerance, which results in a range of conservative to aggressive behavior while still avoiding collisions. Here, we extend the results of our previous work by removing the ability to detect and track obstacles. We present the "object-less" version of our congestion cost function, which approximates the previous congestion cost using only the occupancy density of the environment and the velocity flow field around the agent. We show this closely matches the true congestion cost function in the calculation of the safe planning space.

We implement our dynamic risk density cost function on an autonomous wheelchair navigating busy pedestrian corridors. The wheelchair has a prior map of the environment, and given some initial starting location, must navigate to a goal location along a reference path. The map only includes static

features and environment boundaries, but does not include information about dynamic obstacles, such as other pedestrians, or non-permanent obstructions, such as tables and chairs. Due to the random and cluttered nature of this environment, as well as large crowds moving through the corridors, we do not want to rely on object-detection for the wheelchair to navigate. Instead, we will use only 2D laser scanners to construct occupancy grids about the environment. From the changes in the occupancy grid over time, we also construct an estimate of the velocity flow field, and generate our dynamic risk density. Using this cost function, the wheelchair uses an RRT*algorithm to navigate along its desired path through the environment while avoiding obstacles. We illustrate how this dynamic risk density improves the performance of the wheelchair's navigation over approaches that use occupancy grids computed only from measurements at the current time step, which we refer to as *snapshot costmaps*. We present two distinct methods for computing the velocity field: a clustering-based approach, and density flow.

Related Work

This paper focuses on quickly assessing the density and motion of obstacles in the environment for risk-aware motion planning to prevent collisions. Collision avoidance in multi-agent systems is often studied when agent positions are known [2], [3] or with static obstacles [4], [5], [6], [7]. One common technique for collision avoidance is to use reciprocal velocity obstacles [8], [9], however, this strategy relies on all agents using the same policy. In dynamic and uncertain environments, receding horizon control policies may be used to predict the intent of other agents [10]. Other approaches use predictions about dynamic obstacles for probabilistically-safe motion [11].

In cluttered environments, an autonomous agent's control policy must be aware of all other agents. The "opera problem" [12] is defined as many agents attempting to navigate through a narrow aperture, such as a crowd of people leaving through narrow exits. One proposed solution uses potential field and gradient descent for coordinated flow [13]. Another approach is to determine a Voronoi tessellation from obstacle locations, then navigate along the Voronoi boundaries [14]. Alternative Voronoi tessellations can also provide safe planning regions for agents [7]. These approaches are effective for moving through static obstacles, but does not readily extend to dynamic obstacles.

Navigating in crowds is an open problem for socially-competent navigation. Many approaches focus on learning and identifying key features of pedestrian motion in order to predict their intent [15], [16], [17], [18], [19], [20], [21]. Deep reinforcement learning can learn typical human trajectories for the autonomous agent [22]. Other methods rely on multi-policy decision making [23], utilize probabilistic predictive models of cooperative collision avoidance [24], or predict underlying pathways to follow [25]. By understanding features gleaned from data, it is possible to decrease the uncertainty in navigation with pedestrians [26], [27]. While these approaches provide varied and robust ways to interact with pedestrians, they often require either advanced pedestrian tracking, or large training data sets.

Our approach uses the density and velocity information to formulate our cost function. We build upon work in occupancy grid-based navigation [28], [29], [30], [31], which combines all available sensor data into an occupancy probability for navigation. By tracking features, it is possible to create spatio-temporal models of the flow patterns throughout the environment [32], [33]. Here, we use the changes in the densities to predict the velocity flow field of the environment, and then compute a weighted costmap for navigation. In [34], [35], the authors use information about the pedestrians in the environment to weight their presence in a costmap differently than other obstacles. Our approach does not directly detect or track obstacles, treating any occupied space as a region to avoid.

The remainder of the paper is organized as follows: Section II defines the congestion cost and problem formulation. Section III presents our dynamic risk density calculation when object detection is not available. We present simulations in Section IV comparing the performance of the object-based congestion cost and its more general form of the dynamic risk density. Section V details our experiments navigating cluttered pedestrian walkways with an autonomous wheelchair, and we state our conclusions in Section VI.

II. Problem Formulation

In this section, we present our problem formulation and summarize the congestion cost from prior work [1]. From this congestion cost, we then present our approximation when only the local density and velocity flow field information is known. Consider an environment $Q \subset \mathbb{R}^N$, with points in Q denoted q. Here, we consider all agents and obstacles as dynamic obstacles. For n dynamic obstacles in the environment, we write the position of each obstacle as x_i, for $i \in \{1, ..., n\}$. For brevity, we write the position of all the agents as x, with dynamics $\dot{x}$. We define the ego agent with subscript e and denote its position x_e.

The occupancy cost due to all dynamic obstacles in the environment is calculated

$$H(q, x, \dot{x}) = \sum_{i=1}^{n} \frac{\exp\left(-(q - x_i)^T \Omega (q - x_i)\right)}{1 + \exp\left(-\alpha \dot{x}_i^T (q - x_i)\right)}, \quad (1)$$

where α is a scaling factor and Ω is the diagonal matrix of the inverse square of the standard deviation. In $\mathbb{R}^2$, $\Omega = \text{diag}\{\frac{1}{\sigma_x^2}, \frac{1}{\sigma_y^2}\}$ for x, y axes. By construction, the congestion cost is the summation of a Gaussian peak centered at an obstacle's location multiplied by a logistic function in the direction of its motion. By skewing the position by the velocity, the occupancy cost H is greater in the direct path of the obstacle's motion, and lower where obstacles are not actively moving. This particular formulation allows us to make robust claims about the collision avoidance of the ego agent under minimal assumptions of the other agents.

In our previous work, we assume that agents are "self-preserving," which means all agents will avoid collisions when possible with other agents. This assumption implies that there are no adversarial agents in the environment, and that if the ego agent is capable of avoiding collisions with dynamic obstacles, those obstacles will also attempt to avoid collisions. Unlike other work, we do not assume the collision

avoidance happens by any particular control policy, nor do we assume the control policies be reciprocal among agents.

Under this assumption, we can find the thresholds in H that define safe and unsafe values, which then allow us to define the risk thresholds. By computing level sets of the cost function, we restrict the ego agent's planning space to guarantee safe motion through the environment, and further, change the behavior of the agent based on these choices. Let H_c be the value of the cost function when a collision occurs. We define $H_T < H_c$ to be the maximum risk threshold of the system that guarantees an agent can avoid collisions. We also define $H_P \leq H_T$ as the chosen planning threshold of the ego agent. For more aggressive maneuvers, an agent would choose a value of $H_P \approx H_T$, while a conservative agent would chose $H_P << H_T$.

For an ego agent planning a path through the environment Q, we define the risk level set, $L_{\bar{p}}$ as all points in the environment that are below the chosen planning threshold H_P. The risk level set is written

$$L_{\bar{p}} = \{q \mid H(q - x_e, \dot{x}_i) \leq H_P\}. \tag{2}$$

From the level set in (2), it can be shown that an ego agent that only chooses actions within that set will avoid collisions with all other agents [1]. We demonstrate this by proving that the cost function in (1) never exceeds the thresholds H_c, and that for any value of $H < H_c$, a collision cannot occur.

When it is feasible to detect and track obstacles, the congestion cost in (1) is effective for preventing collisions between an ego agent and dynamic obstacles within the environment. However, if the autonomous system is not capable of reliably tracking obstacles, using (1) may fail to capture all hazards. In the next section, we detail our dynamic risk density function, which provides an approximation of the congestion cost function without explicit obstacle detection.

III. Dynamic Risk Density

When obstacle detection is unavailable, we can still compute an approximation of the congestion cost from sensor information. We introduce the approximation of this cost using the density and the velocity field in the environment. Simulations in Section IV show the fidelity of this approximation to the congestion cost presented in (1). For the experiments on the autonomous wheelchair presented in Section V, the local environment density is calculated from laser scanner information, and we present several methods for estimating the velocity field, detailed later in this section.

A. Dynamic Risk Density from Occupancy and Velocity Fields

Consider the congestion cost function presented in (1). To construct our approximation, we look at substitutions to replace the object locations and velocities with environment density and velocity fields.

Recall the numerator of (1) is constructed from Gaussian peaks about the object locations. In the approximation, we instead use the occupation density of the environment in its place. Let $\rho(q,t)$ define the density at point q in the environment at time t. To compute the logistic function without the objects, we use both the density and the velocity field. We define $\mathcal{V}(q,t)$ as the velocity field of the environment at time t. Overall, the dynamic risk density is computed

$$\mathcal{H}_\rho(q,t,\rho,\mathcal{V}) = \frac{\rho(q,t)}{1+\exp\left(\alpha \nabla\rho(q,t)\cdot\mathcal{V}(q,t)\right)}, \tag{3}$$

where $\nabla\rho(q,t)$ is the gradient of the density, and α is a user-designed scaling factor. For brevity in future notation, we use the superscript t to denote functions at a particular time, letting ρ^t and $\mathcal{V}^t$ denote the density and velocity field, respectively. In Section V, we use (3) in conjunction with an RRT*-based controller to navigate a cluttered environment autonomously. We estimate the environment density directly from laser range scanners and occupancy grids. The velocity field may be computed in several ways; here, we present two distinct methods for determining the velocity field from the density. One approach is to use k-means clustering on the density, then map to a velocity field using a Voronoi tessellation. We also present a density flow-based approach, with methods similar to optical flow in image processing.

B. Estimating the Velocity Field from Clustering

To estimate the velocity field from the density information, we applied a k-means clustering algorithm [36] to the occupancy grid, then track the movement of the centroids over time. Using the movement of the centroids, we approximate their velocity. To map velocity for all points in the environment, we calculate the Voronoi tessellation of the environment from the centroids and apply the velocity of the centroid to all points within its associated Voronoi cell. A summary of our approach is given in Algorithm 1.

Let c_i^t denote the position of a centroid at time t, with $C^t = \left[(c_1^t)^T \mid \ldots \mid (c_k^t)^T\right]^T$ representing the positions of all k centroids from the clustering algorithm. At each time t, the k-means clustering algorithm is seeded using the centroids $C^{t-\Delta t}$. We then use the Hungarian algorithm [37] to match the clusters over time and assign the indices from $C^{t-\Delta t}$ to points in C^t. From the matched pairs, we compute the velocity of a centroid as

$$v_i^t = \frac{\left(c_i^t - c_i^{t-\Delta t}\right)}{\Delta t}, \tag{4}$$

where Δt is the time elapsed between computations. To compute the velocity field $\mathcal{V}^t$, we apply the centroid velocity v_i^t to all points within the associated Voronoi cell V_i^t,

$$\mathcal{V}^t = \sum_{i=1}^{k} v_i^t \, \mathbf{1}_{V_i^t}, \tag{5}$$

where V_i^t is the Voronoi cell about centroid c_i^t, computed

$$V_i^t = \{q \mid \|q - c_i^t\| \leq \|q - c_j^t\|, \quad \forall i,j \in k, \quad ,\forall q \subset Q\}.$$

Using this approach, the clusters track areas of high occupancy, yielding a notion of features for the ego agent. In the experiments presented in Section V, we demonstrate this approach on our autonomous wheelchair, as well as a variation that incorporates a Kalman filter on the centroid velocity estimates to reduce the effect of noise.

Algorithm 1 Velocity Field Estimation from Clustering

1: Input: ρ^t, $C^{t-\Delta t}$
2: Compute clusters C^t from ρ^t
3: Match clusters C^t to $C^{t-\Delta t}$
4: Compute cluster velocities v_i^t (4)
5: Estimate velocity field $\mathcal{V}^t$ from Voronoi tessellation (5)

C. Velocity Field Estimation from Density Flow

In contrast to the clustering approach with discrete estimates, here we present a method for estimating velocity field $\mathcal{V}^t$ directly from occupancy density changes ρ^t and $\rho^{t-\Delta t}$. Our approach is inspired by dense optical flow estimation with applications in computer vision. For a 2D+t dimensional case a voxel at location $(q = (x, y), t)$ with density $\rho(x, y, t)$, $\rho(x, y, t)$ will have moved by Δx, Δy, and Δt between the two measurements, and the following density constancy constraint can be formulated:

$$\rho^t(x, y) = \rho(x, y, t) = \rho(x + \Delta x, y + \Delta x, t + \Delta t) \quad (6)$$

Under the assumption of small movement the density can be approximated by first order Taylor expansion to

$$\rho(x + \Delta x, y + \Delta x, t + \Delta t) = \rho(x, y, t) + \frac{\partial \rho}{\partial x}\Delta x + \frac{\partial \rho}{\partial y}\Delta y + \frac{\partial \rho}{\partial t}\Delta t + \text{H.O.T.},$$

such that by employing the density constancy condition (6) and dividing by Δt the partial differential equation

$$\frac{\partial \rho}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial \rho}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial \rho}{\partial t} = 0, \quad (7)$$

relates the density gradient $\nabla\rho = (\frac{\partial \rho}{\partial x}, \frac{\partial \rho}{\partial y})$ to the density flow $\mathcal{V}^t = (\frac{\Delta x}{\Delta t}, \frac{\Delta y}{\Delta t})$. We approximately solve (7) as described in [38] to estimate the density flow $\mathcal{V}^t$ from two sequential densities $\rho^t, \rho^{t-\Delta t}$ based on local polynomial expansion similar to estimating dense optical flow in images.

D. Control Policy

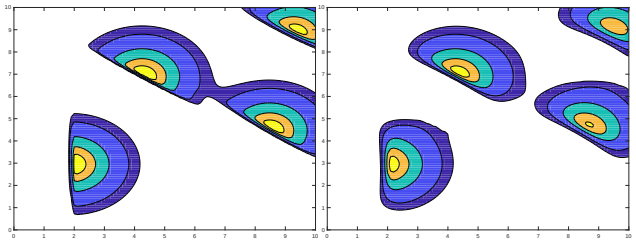
While the risk level set defines the planning space, the user may implement their chosen control policy for navigating the environment. In Section V, we implement an RRT*-based controller on the wheelchair in which the congestion cost is used to update the local path to avoid obstacles. Algorithm 2 provides an overview of our chosen RRT*-based control policy used in our experiments.

Algorithm 2 Dynamic Risk Density Navigation

1: Update ρ^t from sensors
2: Estimate velocity field $\mathcal{V}^t$
3: Compute dynamic risk density (3)
4: Compute risk-bounded path using RRT*along reference path

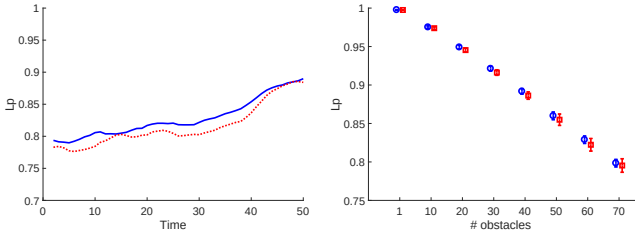
IV. Simulations

To benchmark the dynamic risk density, we ran simulations in Matlab and Python to compare the performance



(a) Object-Based Cost (b) Dynamic Risk Density

Fig. 2: Contours of the (a) object-based cost function and (b) dynamic risk density.



(a) (b)

Fig. 3: (a) From a simulation of $n = 10$ obstacles moving through the environment, we can compare the value of the congestion cost (blue) in (1) to the dynamic risk density (red dashed) in (3). The dynamic risk density yields a similar planning area to (1). (b) For varying numbers of obstacles, we ran 50 trials of randomized configurations and computed the planning area $L_{\bar{p}}$ for the congestion cost (blue) and dynamic risk density (red).

against the congestion cost function in (1) and formalized in [1]. The object-based cost function is computed directly from the object positions and velocities, while the dynamic risk density is computed using the density and velocity flow field. For this comparison, we generate the density field from the position and extent of the objects with added Gaussian noise. The known velocity field is generated for the environment, and objects select their velocity based on their position in the field. From Figure 2, we see that the contours of both cost functions are quite similar. To further compare the variations between the two cost functions, Figure 3a tracks the values of the level set $L_{\bar{p}}$ over time. We simulate 10 obstacles moving through an environment over 50 seconds, and compute the cost at each time step. Over time, the changes in the true congestion cost function are mirrored by the dynamic risk density function.

To compare the average planning set areas, we simulated 50 trials of random object configurations with random velocities for varying numbers of obstacles. Figure 3b shows the comparison of the planning area $L_{\bar{p}}$ between the congestion cost in (1) and the dynamic risk density in (3). Note our dynamic risk density level sets are smaller than the object-based congestion cost, making it more conservative in practice.

For the velocity field computations using our clustering approach presented in Algorithm 1, Figure 4 illustrates the

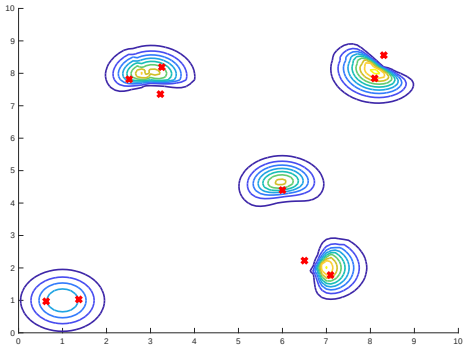


Fig. 4: Contours of the dynamic risk density with the velocity field determined by applying the k-means cluster velocity to a Voronoi tessellation of the environment. Over time, we use the evolution of the clusters to determine the velocity field.

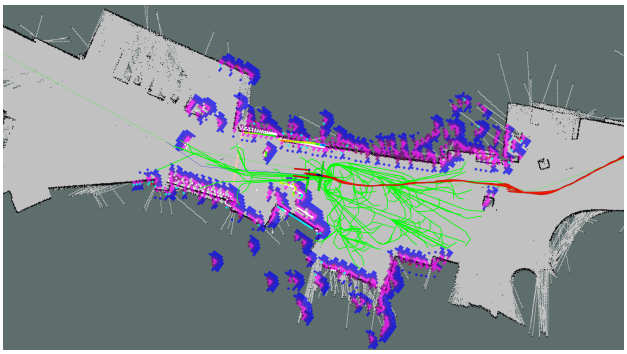


Fig. 5: Visualization of wheelchair's path planning. The red indicates trajectory history, while the green lines are the current exploration of the RRT* algorithm. The local costmap from sensor data is shown in blue and pink.

contours of the dynamic risk density function $\mathcal{H}_\rho$ using the Voronoi-based velocity field computed in (5). The centroids of the k-means clusters are shown on the figure with red X's. The velocity field applies the centroids' movement to the Voronoi tessellation of the environment.

V. Experiments

To validate our dynamic risk density cost function, we performed experiments using an autonomous wheelchair developed at MIT. The hardware platform is inspired by the Singapore-MIT Alliance for Research and Technology (SMART) scooter [39] and golf-cart [40] platforms. The wheelchair uses the same autonomy stack developed for the MIT autonomous Prius platform [41] detailed below. For the experiments, we use laser scanners to construct occupancy grids in the environment that are used for motion planning. Figure 6 shows stills from the video included with the submission of our paper synced to snippets of the costmap.

A. Platform Overview

The experiments are performed on a wheelchair modified for drive-by-wire, which runs an autonomy stack composed of control, navigation, and sensor modules implemented in ROS [42]. The sensor modules interface with the laser scanners, encoders, and an IMU that equip the wheelchair. The control modules provide linear and angular velocity controllers, where the reference signals are given by the navigation modules, as well as emergency stop and human-input control signals. The navigation stack comprises localization, path planner, path following, and costmap modules. Localization is performed using the AMCL package [43], [44], which integrates odometry poses computed from the encoders and IMU data, and laser scan data registered into an *a priori* constructed map. The location of the wheelchair together with a reference path and costmaps are used by an RRT* planner [45] to generate obstacle-free paths in the environment. We use a pure-pursuit controller [46] for path following, which provides the steering control signal. For simplicity, we command constant linear velocity, unless an obstacle is too close to the wheelchair and all velocities are set to zero. Collision checking is performed using a discrete costmap-based approach, where grids are populated with laser scan measurements. The ROS costmap_2d package [35] computes snapshot costmaps that integrate static map information with obstacles generated from the laser scan data, and inflated radially for safety.

Our proposed *dynamic risk density* implementation interfaces with the vanilla costmap and path planning modules. It takes in the costmap stream, and generates discrete (grid) risk cost functions that take into account the underlying velocity fields of cluttered (occupied) space. The cost function is passed to the path planner module that computes obstacle-free paths at the current time, and potentially within a future time horizon. The dynamic risk density computation ran at 60Hz on a Dell Precision 5520 laptop with Core i7-7th gen processor and 32GB RAM, allowing the wheelchair to react quickly to the environment and operate smoothly. A visualization of the path planning is shown in Figure 5.

B. Performance Comparison

We conducted our experiments in the Stata Center at MIT. We ran trials along the a busy pedestrian corridor during weekday afternoons, and compared the performance of our navigation algorithm in Algorithm 2 using several methods of computing the velocity field. These variations were:

- **Snapshot** costmap setting $V^t = 0$ (baseline using only occupancy grid information),
- **Density flow** using (7) to solve for $\mathcal{V}^t$, akin to optical flow methods in image processing,
- **k-means** clustering of the occupancy grid, then estimating $\mathcal{V}^t$ using the Voronoi approach in Algorithm 1,
- **Filtered k-means** clustering with a Kalman filter on the centroid velocity estimates v_i^t before mapping to $\mathcal{V}^t$.

First, we computed the velocity field using the k-means clustering approach with the Voronoi-based velocity field, as described in Algorithm 1. Next, we ran a modified version of the clustering algorithm, this time with a Kalman filter on the generated cluster velocities v_i^t. We also ran several trials with the velocity field generated from the density flow approach outlined in Section III. As a baseline, we compare our results to the snapshot costmap.

Overall, we found the snapshot costmap to be either too conservative in its obstacle padding, or too aggressive and collision-prone. For the two variations on clustering, filtering the velocity helped reduce noise in the estimation

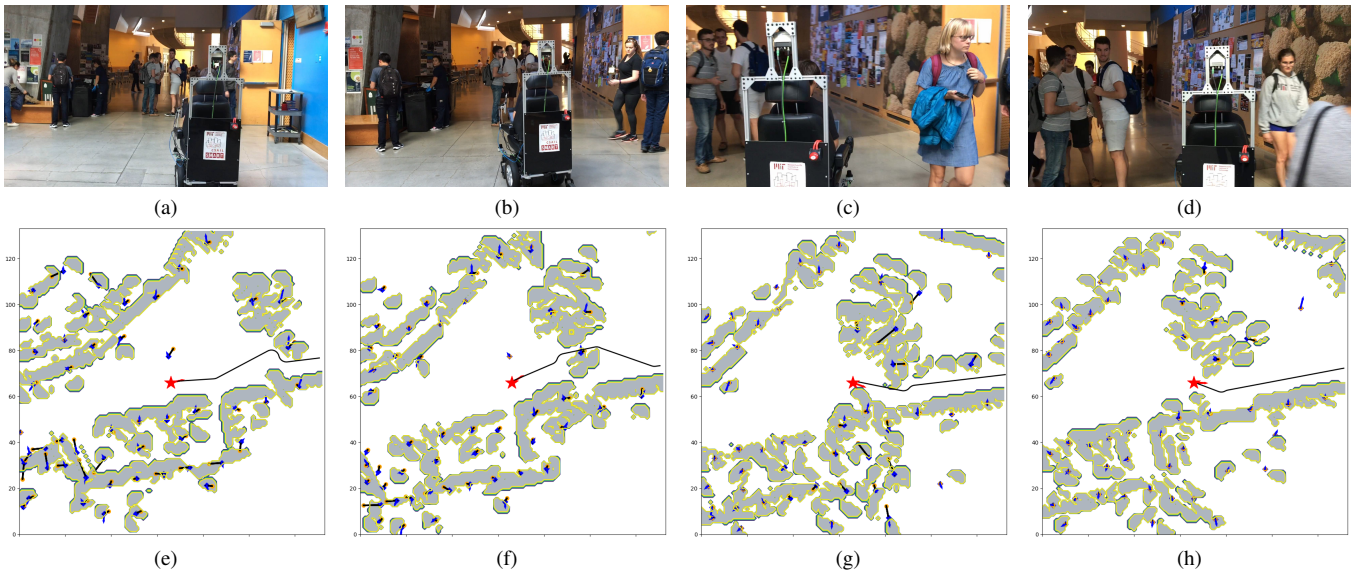


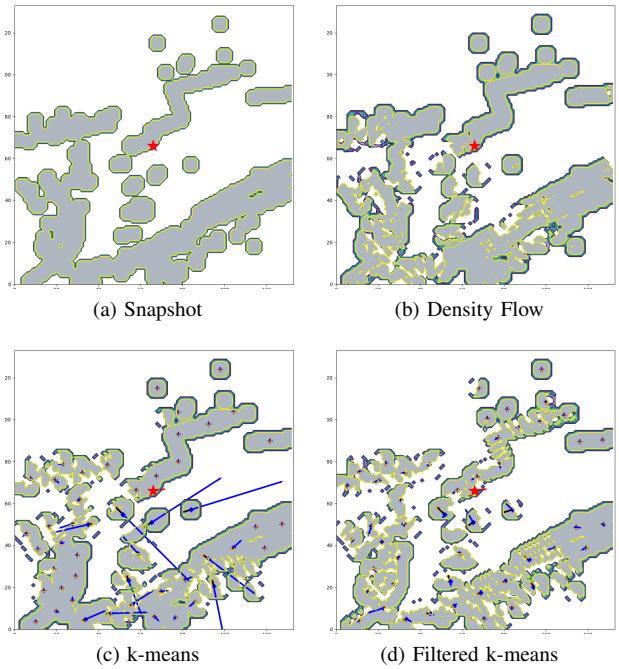
Fig. 6: (a-d) Video frames of the wheelchair navigating a busy section of Stata. (e-h) costmap showing the obstacles (grey) and planned path (black line). Here, the velocity field is generated from the filtered k-means approach. As the wheelchair (red star) approaches the doorway, several people enter the corridor, and the wheelchair adjusts its path around. It then stops and waits for a pedestrian to pass on the right before continuing down the hall.

of static obstacles, but introduced a delay in responding to new dynamic obstacles. Both versions of our clustering algorithm were performed with $n = 50$ clusters over our costmap. As shown in Figure 6 and seen in our video supplement, the clustering approach is able to handle a congested hallway where people are entering and exiting a classroom, introducing new obstacles.

While the clustering approach provides a discrete approximation of points of interest in the environment, the density flow-based velocity field generates a high-resolution, fine-grained estimation of the velocity field. It yielded a better approximation of the environment boundaries, but was more susceptible to artifacts in the density map, and clipped objects within the costmap due to more discontinuities in velocity over the field. Figure 7 shows the four versions of our costmap from a single reference point.

VI. Conclusions

In this paper, we propose the use of a dynamic risk density function for navigating a cluttered environment without explicit object detection and tracking. The dynamic risk density is computed by taking the congestion density and spatial flow of an environment, and mapping it to a cost function that approximates risk for a navigating ego agent. We build upon prior work, which proposed the use of risk level sets to safely navigate a cluttered environment without collision. Simulations illustrate how the dynamic risk density provides a high-fidelity approximation of the congestion cost function. We also propose two methods of calculating the velocity field: first, using a k-means clustering-based approach, which provides discrete approximations of features; and second, a density flow approach that provides a fine-grained approximation of the velocity field. Experiments on an autonomous wheelchair platform demonstrate how the dynamic risk density can be used in cluttered environments, and we perform multiple trials comparing the different instantiations of our velocity field estimation as compared to a "snapshot" costmap.



(a) Snapshot (b) Density Flow

(c) k-means (d) Filtered k-means

Fig. 7: Visualization of the snapshot costmap (a) versus the dynamic risk density costmap with our three variations of velocity field generation using (b) density flow, (c) k-means clustering, and (d) k-means clustering with filtered velocities.

REFERENCES

[1] A. Pierson, W. Schwarting, S. Karaman, and D. Rus, "Navigating congested environments with risk level sets," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018.

[2] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 65–100, 2010.

[3] C. Schlegel, "Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 1. IEEE, 1998, pp. 594–599.

[4] P. Ogren and N. E. Leonard, "A convergent dynamic window approach to obstacle avoidance," *Robotics, IEEE Transactions on*, vol. 21, no. 2, pp. 188–195, 2005.

[5] C. Belta, V. Isler, and G. J. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 864–874, Oct 2005.

[6] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, 1999, pp. 341–346.

[7] A. Pierson and D. Rus, "Distributed target tracking in cluttered environments with guaranteed collision avoidance," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Dec 2017, pp. 83–89.

[8] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 1928–1935.

[9] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 3–19.

[10] N. E. D. Toit and J. W. Burdick, "Robot motion planning in dynamic, uncertain environments," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 101–115, Feb 2012.

[11] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," *Autonomous Robots*, vol. 35, no. 1, pp. 51–76, Jul 2013. [Online]. Available: https://doi.org/10.1007/s10514-013-9334-3

[12] V. Crespi, G. Cybenko, and D. Rus, *Decentralized Control and Agent-Based Systems in the Framework of the IRVS*, 2001.

[13] V. Crespi, G. Cybenko, D. Rus, and M. Santini, "Decentralized control for coordinated flow of multi-agent systems," in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, vol. 3, May 2002, pp. 2604–2609 vol.3.

[14] O. Arslan and D. E. Koditschek, "Sensor-based reactive navigation in unknown convex sphere worlds," in *The 12th International Workshop on the Algorithmic Foundations of Robotics*, 2016.

[15] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 31–48, 2005. [Online]. Available: https://doi.org/10.1177/0278364904048962

[16] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-based prediction of trajectories for socially compliant navigation." in *Robotics: science and systems*, 2012.

[17] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] M. Pfeiffer, U. Schwesinger, H. Sommer, E. Galceran, and R. Siegwart, "Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 2096–2101.

[19] A. Vemula, K. Muelling, and J. Oh, "Modeling cooperative navigation in dense human crowds," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1685–1692.

[20] L. Tai, J. Zhang, M. Liu, and W. Burgard, "Socially compliant navigation through raw depth inputs with generative adversarial imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1111–1117.

[21] M. Pfeiffer, G. Paolo, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–8.

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

[23] D. Mehta, G. Ferrer, and E. Olson, "Autonomous navigation in dynamic social environments using multi-policy decision making," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1190–1197.

[24] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of humanrobot cooperation," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 335–356, 2015. [Online]. Available: https://doi.org/10.1177/0278364914557874

[25] J. Doellinger, M. Spies, and W. Burgard, "Predicting occupancy distributions of walking humans with convolutional neural networks," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1522–1528, July 2018.

[26] C. Mavrogiannis and R. A. Knepper, "Designing algorithms for socially competent robotic navigation," in *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '17. New York, NY, USA: ACM, 2017, pp. 357–358. [Online]. Available: http://doi.acm.org/10.1145/3029798.3034810

[27] C. I. Mavrogiannis, V. Blukis, and R. A. Knepper, "Socially competent navigation planning by deep learning of multi-agent path topologies," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 6817–6824.

[28] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, June 1989.

[29] N. C. Mitsou and C. S. Tzafestas, "Temporal occupancy grid for mobile robot dynamic environment mapping," in *2007 Mediterranean Conference on Control Automation*, June 2007, pp. 1–8.

[30] A. Souza and L. M. G. Gonalves, "Occupancy-elevation grid: an alternative approach for robotic mapping and navigation," *Robotica*, vol. 34, no. 11, p. 25922609, 2016.

[31] S. Hoermann, M. Bach, and K. Dietmayer, "Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2056–2063.

[32] T. Kucner, J. Saarinen, M. Magnusson, and A. J. Lilienthal, "Conditional transition maps: Learning motion patterns in dynamic environments," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1196–1201.

[33] S. Molina, G. Cielniak, T. Krajník, and T. Duckett, "Modelling and predicting rhythmic flow patterns in dynamic environments," in *Towards Autonomous Robotic Systems*, M. Giuliani, T. Assaf, and M. E. Giannaccini, Eds. Cham: Springer International Publishing, 2018, pp. 135–146.

[34] D. V. Lu, D. B. Allan, and W. D. Smart, "Tuning cost functions for social navigation," in *Social Robotics*, G. Herrmann, M. J. Pearson, A. Lenz, P. Bremner, A. Spiers, and U. Leonards, Eds. Cham: Springer International Publishing, 2013, pp. 442–451.

[35] D. V. Lu, D. Hershberger, and W. D. Smart, "Layered costmaps for context-sensitive navigation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 709–715.

[36] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979. [Online]. Available: http://www.jstor.org/stable/2346830

[37] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109

[38] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Scandinavian conference on Image analysis*. Springer, 2003, pp. 363–370.

[39] H. Andersen, Y. H. Eng, W. K. Leong, C. Zhang, H. X. Kong, S. Pendleton, M. H. Ang, and D. Rus, "Autonomous personal mobility scooter for multi-class mobility-on-demand service," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 1753–1760.

[40] S. Pendleton, T. Uthaicharoenpong, Z. J. Chong, G. M. J. Fu, B. Qin, W. Liu, X. Shen, Z. Weng, C. Kamin, M. A. Ang, L. T. Kuwae, K. A. Marczuk, H. Andersen, M. Feng, G. Butron, Z. Z. Chong, M. H. Ang, E. Frazzoli, and D. Rus, "Autonomous golf cars for public trial of mobility-on-demand service," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 1164–1171.

[41] F. Naser, D. Dorhout, S. Proulx, S. D. Pendleton, H. Andersen, W. Schwarting, L. Paull, J. Alonso-Mora, M. H. Ang, S. Karaman, R. Tedrake, J. Leonard, and D. Rus, "A parallel autonomy research platform," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 933–940.

[42] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[43] D. Fox, "Adapting the sample size in particle filters through kld-sampling," *The international Journal of robotics research*, vol. 22, no. 12, pp. 985–1003, 2003.

[44] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[45] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[46] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, September 2009.