2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Safe Path Planning with Multi-Model Risk Level Sets

Zefan Huang[1], Wilko Schwarting[2], Alyssa Pierson[2], Hongliang Guo[1], Marcelo Ang Jr[3], and Daniela Rus[2]

***Abstract*— This paper investigates the safe path planning problem for an autonomous vehicle operating in unstructured, cluttered environments. While some objects may be accurately with canonical perception algorithms, other objects and clutter may be harder to track. We present an approach that combines two methods of risk assessment: for objects with reliable tracking, we use a Gaussian Process (GP) regulated risk map to describe the risk map information; for unknown objects that we fail to accurately track, we compute a Dynamic Risk Density (DRD) from the overall occupancy and velocity field from LiDAR scan snapshots. Several methods are proposed for combining the GP risk map and DRD, and the resultant hybrid risk map is used for the proposed safe path planning algorithm. Experimental results on an autonomous buggy show that the hybrid risk map is able to yield a safe path planner to navigate the autonomous testbed within the cluttered environments.**

I. Introduction

As autonomous mobility integrates into human-centric environments, these autonomous systems need to perceive the environment and move safely without collisions. For safe navigation, the autonomous systems often identify objects within the environment, then assign a risk of collision. State-of-the-art methods now allow for accurate object tracking and behavior predictions. However, the challenge of perceiving the environment still remains a major hurdle. While recent advances make certain objects easy to track with high precision, these systems are not always robust when facing random changes, dropped detection, or other unknown entities in the environment. In this paper, we explore how to combine a perception pipeline that tracks objects with a "safety net" to the perception system relying on only the occupancy density and velocity field to fill in the gaps in detection.

This paper focuses on navigation in crowded and unstructured or semi-structured environments, such as plazas, pedestrian crosswalks, or other areas where a vehicle may need to interact with humans, bicycles, cars, or other autonomous agents. Our goal is to study cases at the limits of current state-of-the-art perception pipelines. In a prior paper, we introduced a Gaussian Process (GP) regulated risk map for navigation, which is capable of predicting cars and pedestrians [1]. However, it is limited to tracking only a few number of objects within the environment, and limited to the known objects. To complement this risk map, we also incorporate the Dynamic Risk Density (DRD) [2], which creates a "safety net" for the autonomous system, based on the occupancy density and velocity field of the environment. While the DRD does not predict complex behaviors or maneuvers, it does not rely on explicit object tracking and can capture unknown clutter. We combine these two methods into a unified risk map for the vehicle, and then execute the path planning with the combined risk map. Our experiments illustrate how these two modes of combined risk create a better assessment of the environment than either individual method. The main contributions of this paper are:

- Combined pipeline for fusing multiple models of risk assessment;
- Variation of risk based on different levels of conservativeness;
- Risk-aware safe path planner; and
- Experiments on an autonomous buggy in unstructured, and semi-structured environments (illustrated in Figure 1).

(a) Autonomous Buggy Testbed (b) Risk Map Illustration

Fig. 1: Autonomous buggy testbed in the unstructured environment and an illustrative example of the risk map generated and the corresponding re-planned path.

A. Related Work

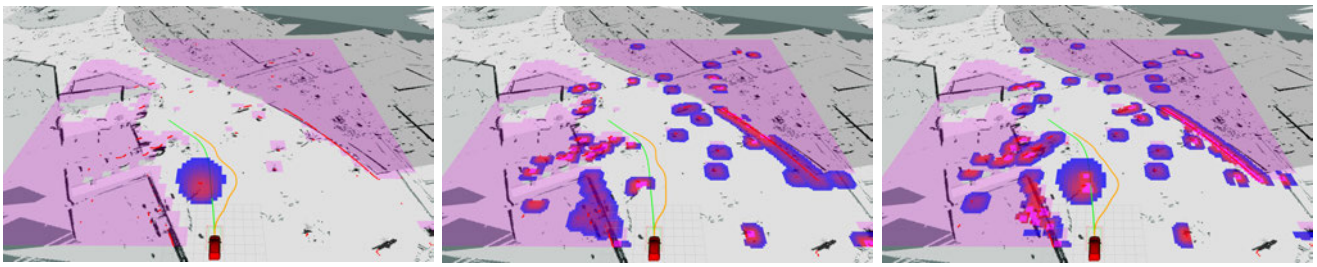
Designing safe and efficient path planning algorithms for autonomous vehicles remains an open problem. These approaches can typically be classified as graph-based, sampling-based, or trajectory optimization methods [3]. With graph-based planners, the environment is discretized such that efficient methods like Dijkstra's algorithm [4], or A* and related algorithms [5], [6] can be applied. Sampling-based planners randomly sample the environment to generate feasible trajectories that conform to vehicle dynamics, including

[1]Singapore-MIT Alliance for Research and Technology, Singapore huang.zefan@smart.mit.edu, hongliang@smart.mit.edu

[2]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA [wilkos, apierson, rus] @mit.edu

[3]National University of Singapore, Singapore mpeangh@nus.edu.sg

This work supported in part by NSF Grant 1723943, the Office of Naval Research (ONR) Grant N00014-18-1-2830, Singapore-MIT Alliance for Research and Technology (SMART) Future Urban Mobility (FM) IRG under the CREATE programme, National Research Foundation, Prime Minister's Office, Singapore, and the Zhejiang Lab's International Talent Fund for Young Professionals. Their support is gratefully acknowledged.



(a) Gaussian Process Risk Map (b) Dynamic Risk Density (c) Hybrid Risk Map

Fig. 2: We combine multiple models of risk into a single risk assessment. Here, we show different risk map representations and the resulting planned safe path for the same scenario snapshots: the green path is the reference path planned beforehand, the yellow path is the re-planned path based on different risk map representations.

RRT, RRT*, and PRM [7], [8], [9], [10], [11]. Trajectory optimization methods formulate a constrained optimization problem and employ techniques from model predictive control or receding horizon control to solve for the optimal path with finite time horizon [12], [13], [14]. Collision avoidance for autonomous systems is a well-studied field within robotics [15]. Common approaches assume that the agent positions are known [16], [17] or focus on static obstacles [18], [19], [20], [21]. For dynamic obstacles, reciprocal velocity obstacles [22], [23] provide collision avoidance, but assume all the agents make the same symmetric decision making. Other approaches may employ probabilistically-safe motion generation [24], or control barrier functions [25], [26], [27].

We are interested in designing safe path planning methods that are risk-averse by design. In particular, we focus on methods that prescribe risk to objects based on their proximity and velocity. This work builds upon the authors' prior work in risk level sets [28], [2] and Gaussian Process (GP) regulated risk [1]. Gaussian processes are a common method to model the uncertainty propagation in the system [29], [30], [31], [32], [33]. Here, we combine these models of risk into a single hybrid metric for the vehicle.

The remainder of the paper is organized as follows: In Section II, we introduce our problem formulation and describe both the GP regulated risk and Dynamic Risk Density. Section III presents our method for creating our multi-model risk map, and our safe path planner. Experiments using an autonomous buggy are presented in Section IV, and we state our conclusions in Section V.

II. Models of Risk

In generating our combined risk map, we consider how to both quantify a high-fidelity estimate of behaviors for objects that we can track, as well as a risk assessment for all the objects that we cannot track. For objects we can track, we choose the GP regulated risk [1], and use the Dynamic Risk Density [2] for everything else. In this section, we provide a background summary of each of these methods and how each method generates a risk map of the environment.

Figure 2 illustrates the two types of risk maps we will combine in this paper, as well as the resulting combined map. The GP regulated risk creates a risk map based on object-tracking and behavior prediction, while Dynamic Risk Density combines the occupancy density of the environment with a velocity field estimate. From Figure 2(a), we can see that the GP risk map performs quite well for the objects that are detected and tracked very well, however, when there are detection dropout and/or undetectable objects, the GP risk map tends to ignore them, and the resulting planned path is unsafe in that case. On the other hand, the DRD method uses the Lidar snapshot difference as the input, and it captures the overall movement of all the objects (without object dropout), therefore, the risk map captures all the moving objects, as is shown in Figure 2(b). However, since the DRD risk map focuses on the overall movement of all the objects, the objects near the ego vehicle are treated equally important as the far-away objects. In this case, the risk map is not accurate enough to describe the nearby areas, and the resulting path is unsafe as well. Therefore, a combination for the two risk map models is needed, displayed in Figure 2(c).

A. GP Regulated Risk Map

For detected and tracked objects in the environment, we generate a risk map that computes the probability that a point in the environment is occupied or in the direct path of an object. For an environment $Q \subset \mathbb{R}^2$, we define points in Q as $q = (x, y)^\top$, where x and y are the coordinates of the 2D environment. We model our object detection module by a Gaussian process, which returns a probability $p(x, y)$ that a point in the environment is occupied by the tracked object or its path.

In our object detection module, we denote $\hat{\boldsymbol{\mu}}_{t:t+k}$ as the streamline output from time t to time $t+k$, and the posterior distribution after k outputs as $p(\mu|\hat{\boldsymbol{\mu}}_{t:t+k})$. We update our risk map by deriving $p(\mu|\hat{\boldsymbol{\mu}}_{t:t+k})$ given $p(\mu|\hat{\boldsymbol{\mu}}_{t:t+k-1})$ and $\hat{\boldsymbol{\mu}}_k$. The complete derivation procedure is presented in [1], and outlined here:

$$\begin{aligned} p(\boldsymbol{\mu}|\hat{\boldsymbol{\mu}}_{1:k}) &\propto p(\hat{\boldsymbol{\mu}}_{1:k}|\boldsymbol{\mu})p(\boldsymbol{\mu}) \\ &= p(\boldsymbol{\mu}) \prod_{i=1}^{k} p(\hat{\boldsymbol{\mu}}_i|\boldsymbol{\mu}) \\ &\propto \exp\Big(-\frac{1}{2}\big((\boldsymbol{\mu}-\boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}-\boldsymbol{\mu}_k)\big)\Big), \end{aligned} \tag{1}$$

where $\mathbf{\Sigma}_k$ denotes the covariance and $\boldsymbol{\mu}_k$ denotes the mean. The mean and covariance are defined

$$\begin{aligned}\mathbf{\Sigma}_k &= (\mathbf{\Sigma}_0^{-1} + k\mathbf{\Sigma}_\mu^{-1})^{-1} \\ \boldsymbol{\mu}_k &= (\mathbf{\Sigma}_0^{-1} + k\mathbf{\Sigma}_\mu^{-1})^{-1}(\mathbf{\Sigma}_0^{-1}\boldsymbol{\mu}_0 + \sum_{i=1}^{k}\mathbf{\Sigma}_\mu^{-1}\hat{\boldsymbol{\mu}}_i).\end{aligned} \tag{2}$$

Additionally, we can update the risk map online by recursively computing $\boldsymbol{\mu}_k$ and $\mathbf{\Sigma}_k$ given the real-time inputs $\hat{\boldsymbol{\mu}}_k$, and the previous posterior parameterization $\boldsymbol{\mu}_{k-1}$, and $\mathbf{\Sigma}_{k-1}$ at time $k-1$. The recursive online update of the posterior is

$$\mathbf{\Sigma}_k = (\mathbf{\Sigma}_{k-1}^{-1} + \mathbf{\Sigma}_\mu^{-1})^{-1}, \tag{3}$$

$$\boldsymbol{\mu}_k = (\mathbf{\Sigma}_{k-1}^{-1} + \mathbf{\Sigma}_\mu^{-1})^{-1}(\mathbf{\Sigma}_{k-1}^{-1}\boldsymbol{\mu}_{k-1} + \mathbf{\Sigma}_\mu^{-1}\hat{\boldsymbol{\mu}}_k) \tag{4}$$

From the one-static-obstacle GP risk derivation procedure, we can see that the posterior distribution of the obstacle given an output stream from the object detection module follows a Gaussian process with $\boldsymbol{\mu}_k$ and covariance function $\mathbf{\Sigma}_k$. For $i = \{1, ..., m\}$ static obstacles, we define the risk map that pertains to obstacle i by

$$p_i(x,y) = \frac{\exp\big(-\frac{1}{2}((x,y)^\top - \boldsymbol{\mu_k})^\top \mathbf{\Sigma}_k^{-1}((x,y)^\top - \boldsymbol{\mu_k})\big)}{\sqrt{|2\pi\mathbf{\Sigma}_k|}}. \tag{5}$$

We define the overall risk map for all the m obstacles as:

$$p(x,y) = 1 - \prod_{i=1}^{m}(1 - p_i(x,y)) \tag{6}$$

We refer the reader to [1] for the full derivation process for dynamic obstacles.

B. Dynamic Risk Density

For the objects we cannot identify and track, we still wish to assign a risk to those objects and regions for our path planner, creating a "safety net" for the autonomous vehicle. Here, we implement the Dynamic Risk Density introduced in [2], summarized below. For points $q \in Q$, let $\rho(q,t)$ define the occupation density at point q in the environment at time t. We also define $\mathcal{V}(q,t)$ as the velocity field of the environment at time t. The Dynamic Risk Density is [2]

$$\mathcal{H}_\rho(q,t,\rho,\mathcal{V}) = \frac{\rho(q,t)}{1 + \exp\left(\alpha\nabla\rho(q,t)\cdot\mathcal{V}(q,t)\right)}, \tag{7}$$

where $\nabla\rho(q,t)$ is the gradient of the density, and α is a user-designed scaling factor. For brevity, we use the superscript t to denote time, letting ρ^t and $\mathcal{V}^t$ denote the density and velocity field, respectively.

While the occupation density is easily obtained from laser scanners, we need to estimate the velocity field of the environment. Here, we estimate the velocity field $\mathcal{V}^t$ from changes in density between ρ^t and ρ^{t-1}, inspired by dense optical flow estimation from the computer vision community. Consider a voxel at location $q = (x,y,t)$, with density $\rho(x,y,t)$. Between measurements, the voxel moves by Δx, Δy, and Δt, which allows us to write the density constancy constraint:

$$\rho^t(x,y) = \rho(x,y,t) = \rho(x+\Delta x, y+\Delta y, t+\Delta t) \tag{8}$$

Assuming this movement between time steps is small, we approximate the density with a first-order Taylor expansion,

$$\begin{aligned}&\rho(x+\Delta x, y+\Delta x, t+\Delta t) = \\ &\qquad \rho(x,y,t) + \frac{\partial\rho}{\partial x}\Delta x + \frac{\partial\rho}{\partial y}\Delta y + \frac{\partial\rho}{\partial t}\Delta t + \text{H.O.T.}\end{aligned} \tag{9}$$

From the density constancy condition (8) and dividing the partial differential equation by Δt

$$\frac{\partial\rho}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial\rho}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial\rho}{\partial t} = 0, \tag{10}$$

we relate 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 then solve (10) similar to [34] to estimate the density flow $\mathcal{V}^t$ given sequential densities $\rho^t, \rho^{t-\Delta t}$ with a local polynomial expansion.

III. Navigation with Multi-Model Risk

In this section, we first present our method for combining the risk maps into a single unified metric, and then describe our safe path planner given the combined risk map. Algorithm 1 summarizes our planning process, which is implemented in the experiments in Section IV.

Algorithm 1 Navigation with Multi-Model Risk

1: Find Dynamic Risk Density $\mathcal{H}(x,y)$ (7)
2: Find GP regulated risk $p(x,y)$ (6)
3: Calculate combined risk $r(x,y)$ with one of the four combination models [(11), (12), (13), or (14)]
4: Choose ϵ-safe threshold (15)
5: Find shortest safe path from (16)
6: **if** no feasible path **then**
7: Find the next safe action from (17)
8: **end if**

A. Multi-Model Risk Map Construction

Within our perception pipeline, we can calculate both models of risk presented in Section II at each point in time. The GP regulated risk provides risk values for the tracked objects in the environment, while the Dynamic Risk Density assigns risk to any objects in the environment that cannot be explicitly tracked, or for which we do not have a reliable prediction model. Given both models, the task is to combine them in a way that is not overly conservative or aggressive. Here, we present four possible alternatives, which are later analyzed in the experiments section.

We denote the combined risk map as $r(x,y)$, which assigns a value of risk to points on a grid (x,y). The combined map $r(x,y)$ is constructed as a combination of the GP regulated risk $p(x,y)$ and the Dynamic Risk Density $\mathcal{H}(x,y)$. We normalize both risk densities, $p(x,y)$ and $\mathcal{H}(x,y)$, to take values between $[0,1]$ to allow for comparisons and combination of the two methods. Furthermore, this also allows the combination result $r(x,y)$ to map to the same range. Here, we present four possible methods for combining

these risk maps, which we refer to as the "Maximum-Safety" model (r_m), the "Joint-Minimum" model (r_j), the "Aggressive" model (r_a), and the "Convex Combination" model (r_c).

1) Maximum-Safety Model (r_m): This model constructs the most conservative estimate of the combined risk by multiplying both risk maps, and r_m is calculated as:

$$r_m(x,y) = 1 - (1 - p(x,y))\,(1 - \mathcal{H}(x,y))\,. \tag{11}$$

We consider this model the most conservative, as any amount of risk in either map will be enlarged in this map.

2) Joint-Minimum Model (r_j): For this model, we choose the minimum between the two models as the risk value. For a node (x, y) to be considered "safe" under this construction, the node must be considered "safe" by both models. We calculate r_j as:

$$r_j(x,y) = 1 - \min\{1 - p(x,y), 1 - \mathcal{H}(x,y)\}. \tag{12}$$

The Joint-Minimum model is appropriate in situations where we are worried about missed detections, and the probability of false positives is low for either map.

3) The Aggressive Model (r_a): In contrast to the Joint-Minimum model, the Aggressive model takes the maximum value between the two risk maps. Thus, a location in $r(x, y)$ is deemed safe when considered safe by either one of the two models.

$$r_a(x,y) = 1 - \max\{1 - p(x,y), 1 - \mathcal{H}(x,y)\}, \tag{13}$$

The Aggressive model is useful when both maps may (independently) produce a large number of false-positive risk detections, and we tend to ignore the falsely detected risk values.

4) The Convex Combination Model (r_c): Our final model finds the convex combination of the two individual risk maps, calculates r_c as:

$$r_c(x,y) = 1 - \big(\theta(1-p(x,y)) + (1-\theta)(1-\mathcal{H}(x,y)\big), \tag{14}$$

where $\theta \in (0, 1)$ is a scaling factor. Here, we can tune θ to bias the weight of one map over another. For instance, a larger value of θ biases the belief of the GP regulated risk model more than the Dynamic Risk Density one.

B. Safe Path Planner

Once the combined risk map $r(x, y)$ has been calculated, we input the map into our safe path planner, summarized in Algorithm 1. The planner calculates the shortest safe path connecting the ego vehicle from its origin location, denoted as O, to a desired destination point, denoted as D in the 2D environment. We define the path on a discretized grid g_t of the environment, where s_O and s_d are the origin and destination in the grid, respectively. Safety is specified by some threshold ϵ, which can be tuned to the given task and application. The following defines the ϵ-safe node and ϵ-safe path.

Definition 1. *An ϵ-safe node: A node in a 2D grid map is said to be 'ϵ-safe' if the risk value of it is less than ϵ,*

$$(x,y) \in \{x, y : r(x,y) < \epsilon\}. \tag{15}$$

Definition 2. *An ϵ-safe path: A path is defined as 'ϵ-safe' if all the nodes contained in the path are ϵ-safe nodes.*

From these definitions, we can then formulate the safe path planning problem as follows:

Problem 1. ***(ϵ-Safe Path Planning)*** *An ϵ-safe path traversing the environment can be found by solving the following optimization:*

$$\begin{aligned}
\underset{g_1,\cdots,g_T}{\textit{minimize}} \quad & T \\
\textit{subject to} \quad & g_t \in \mathcal{A}(g_{t-1}) \qquad \forall 1 \le t \le T, \\
& r(g_t) \le \epsilon \qquad \forall 1 \le t \le T, \\
& g_0 = s_0, \\
& g_T = s_d,
\end{aligned} \tag{16}$$

where $\mathcal{A}(g_{t-1})$ refers to the set of nodes that are reachable from g_{t-1}, i.e., the neighbour set of g_{t-1}, and $r(g_t)$ is the multi-model risk value of the node g_t at time t.

To solve the path planning problem defined in (16), we use Dijkstra's algorithm by removing all the nodes that are not ϵ-safe. The optimization then returns the minimum-time ϵ-safe path from the specified origin to the destination.

In highly cluttered environments, there may be no feasible path between the origin and destination, especially if the chosen risk model is too conservative and ϵ threshold is too small. To address the potentially infeasible path problem, we relax the constraint that all the nodes along the path must be ϵ-safe, and instead only constrain that the immediate next node is ϵ-safe. The relaxed safe path planning problem becomes:

Problem 2. ***(ϵ-safe Next Action)*** *If there is no feasible ϵ-safe path, we can find the immediate safe action from the relaxed problem:*

$$\begin{aligned}
\underset{g_1,\cdots,g_T}{\textit{maximize}} \quad & \sum_{i=1}^{T} \log\big(1 - r(g_i)\big) \\
\textit{subject to} \quad & g_t \in \mathcal{A}(g_{t-1}) \qquad \forall 1 \le t \le T \\
& r(g_1) \le \epsilon, \qquad (17) \\
& g_0 = s_0, \qquad (18) \\
& g_T = s_d.
\end{aligned}$$

Note the constraint in (17) only requires that $r(g_1) < \epsilon$, instead of $r(g_t) < \epsilon$ in (16), and the objective is to maximize the path's overall safety score which is defined as $\sum_{i=1}^{T} \log\big(1 - r(g_i)\big)$. In Algorithm 1, if we cannot find a feasible solution to (16), we use the relaxed planner as defined in (17) to find the next safe action for the ego vehicle.

IV. Experiments and Analysis

We implemented our safe planner from Algorithm 1 together with the hybrid risk map calculation method on an autonomous buggy testbed. More details on our autonomous

buggy platform can be found in [35]. A Sick LMS 151 2D Lidar is the primary sensor for object localization and multi-model risk map assessment. The buggy navigates itself within a $2D$ map of the environment constructed through laser-based SLAM (Simultaneous Localization and Mapping), and the ego autonomous testbed's localization is through an adaptive Monte-Carlo localization (AMCL) method [36].

For our experiments, we compare the various methods of combined risk, as detailed in Section III, in two environments. First, we consider a semi-structured tunnel environment, where the flow of traffic follows the direction of the tunnel. Next, we analyze performance in an unstructured busy pedestrian mall, which has traffic in all directions. We first compare how the chosen risk model impacts the safe planning region, then compare variations in path deviations when re-planning around an obstacle.

A. Object Detection Pipeline

To construct the GP regulated risk map $p(x, y)$, we need to detect, localize and track the objects within the environment. For object detection, we used the spatial-temporal approach for LiDAR object detection as outlined in [37]. The object detection module has been further tuned to enable higher detection rates for pedestrians in order to suit the golf cart test platform in the plaza test environment. Furthermore, the permanent features in the environment, *e.g.* walls, mirrors and pillars, have been masked away to minimize false positives. For object tracking, we employ the Kalman Filter and Hungarian Algorithm [38] as the tracking pipeline, as well as a novel data association maneuver with velocity priors in certain regions to incorporate symbolic road context information. Normally, a constant velocity motion transition model is supplied to the Kalman Filter, however, when the environment contains specific structures such as road curvatures or crosswalks, we change the velocity priors so as to embed this information. For example, we update the belief to include that pedestrians are most likely to cross at crosswalks, or that vehicles will probably follow the road curvatures. This is most used in areas where the constant-velocity model is less precise, *e.g.* entering or leaving road sections, roundabouts, and road crossings.

In the implementation, the obstacle detection and tracking module operates at around 10Hz, and the risk map update and safe path planner of Algorithm 1 is operating at 5Hz. The micro-controller for low-level planning of the vehicle operates at 40Hz. During our experiments, the ego vehicle reacts within 200ms to decisions made by the safe path planner, hence, we call our safe path planner real time. This planning loop also includes any path re-planning whenever the previous path becomes unsafe or infeasible.

B. Safe Path Planning in Semi-Structured Environment

We tested the safe path planner together with the multi-model risk map in a semi-structured environment, *i.e.* a tunnel environment, where the flow of pedestrians and bicycles is, more or less, in a fixed direction, by design of the environment. In this testing scenario, the buggy encountered both the pedestrians and bicyclists going through the tunnel.

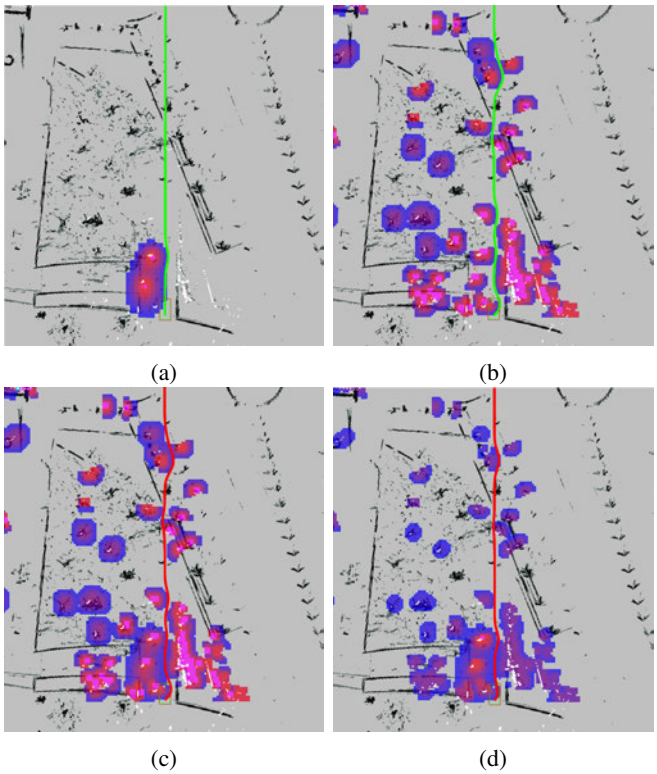
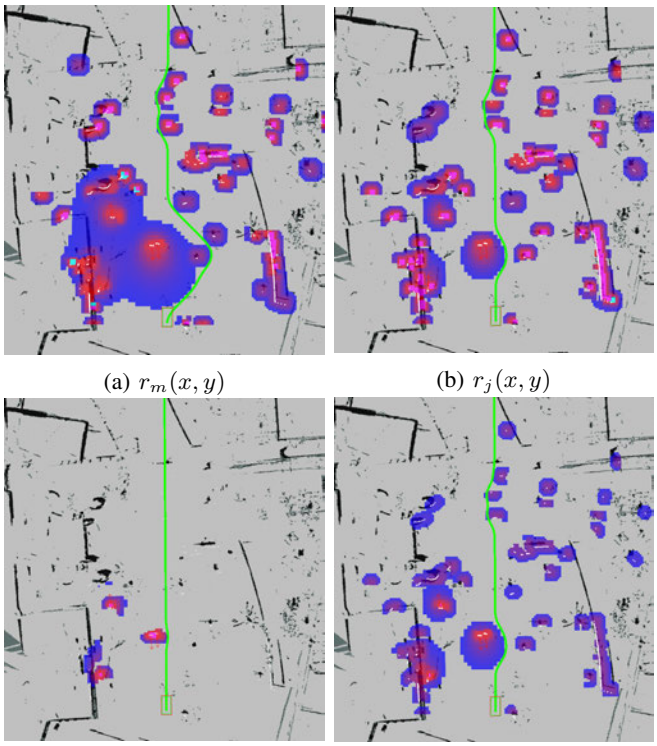


Fig. 3: Risk map and planned path in a tunnel environment. (a) GP regulated risk $p(x, y)$ and (b) Dynamic Risk Density $\mathcal{H}(x, y)$, (c) maximum-safety risk $r_m(x, y)$ and (d) Convex Combination $r_c(x, y)$ (with $\theta = 0.5$) respectively. A feasible path (green) exists when only using a single model of risk in (a) or (b). When using the combined risk in (c) and (d), we can see that the safe path plan is infeasible (red), thus the vehicle should use the relaxed safe path planner, slow its speed, and re-plan at a higher frequency.

Figure 3 illustrates how the original (a) GP regulated risk and (b) Dynamic Risk Density compare to both the (c) Maximum-Safety model $r_m(x, y)$ and (d) Convex Combination model $r_c(x, y)$. Note that we have not plotted the Joint-Minimum model $r_j(x, y)$ nor the Aggressive model $r_a(x, y)$. While the GP regulated risk is good at detecting easily-identifiable pedestrian's motion near the vehicle, the Dynamic Risk Density adds more information about other pedestrians, walls and obstacles in the environment, and the movement motion which is missed by the object detection and tracking pipeline. We also plot the ego vehicle's planned path in Figure 3. When only a single model of risk is considered, the vehicle computes a safe path (green) through the environment. However, when we take both models in combination, the red path indicates that the region is no longer safe, and the vehicle has switched to the relaxed safe path planner, and it needs to slow down and plan at a higher re-planning frequency. A comparison of all four risk models is presented in the following section on unstructured



(a) $r_m(x, y)$ (b) $r_j(x, y)$

(c) $r_a(x, y)$ (d) $r_c(x, y)$

Fig. 4: Comparison of planned paths for various calculation models of risk in an unstructured environment. (a) Maximum-Safety, (b) Joint-Minimum, (c) Aggressive, and (d) Convex Combination Models for combining the GP regulated risk and Dynamic Risk Density. Note (a) provides the most conservative estimate, while (c) is the most aggressive.

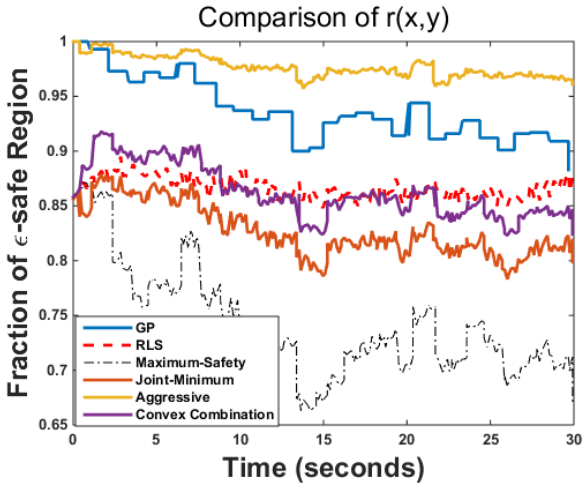


Fig. 5: The evolution of ϵ-safe region percentage over time, where ϵ is set to be 0.01. Visualizations of the environment are shown in Figure 4. The Aggressive model fails to capture most of the obstacles, thus resulting in a high fraction of safe area in the environment. We found the Joint-Minimum model to be the best-performing combination of risk models in our environments.

environments.

C. Navigating Unstructured Environments

As detailed in Section II, our combined risk map is generated as the combination of a GP-regulated risk map as proposed in [1] and the Dynamic Risk Density from [2]. We fuse these maps into a single risk map, computed in four different ways described in Section III and illustrated in Figure 4. In Figure 4, we illustrate how the different models of risk map combinations result in different risk map representations for the unstructured environment. We consider this environment "unstructured" since the pedestrians and bicycles move freely in any direction. For this specific scenario, we considered all four models presented. In our preliminary tests, we came up with several observations about the different models being used.

For the maximum-safety model ($r_m(x, y)$), it can be seen that the risk areas around any obstacles are greatly inflated. Large areas are perceived as too risky to move through, causing the path planner to often fail in returning a viable path. Also, we find that the Aggressive risk model ($r_a(x, y)$) is not conservative enough in defining safe regions, and failed to capture most of the obstacles. While this method may be appropriate in systems with high rates of false detection, for our system, we find that the dropped detection (false negative) is the more common perception issue.

Overall, the Joint-Minimum model ($r_j(x, y)$) showed a good balance between allowing the path planner to generate a feasible yet safe path and demarcating a risky zone around both static and dynamic obstacles. This model avoid the issues in both the maximum-safety model and the Aggressive risk model, while avoiding additional steps to tune the θ value as that in the Convex Combination model ($r_c(x, y)$).

1) Comparison of ϵ-safe regions: We compare the evolution of the total ϵ-safe regions in the environment in Figure 5. Here, we park the buggy in the crowded environment, and record the fluctuations in total ϵ-safe area over 30 seconds. Figure 5 plots the fraction of the ϵ-safe region over time. As expected, the Maximum-Safety model is the most conservative, and the Aggressive model is the least conservative. The Joint-Minimum model is the less conservative than the Convex Combination model. We use the Joint-Minimum as our hybrid risk model for the performance comparisons in the following section.

2) Variation in Path Deviations: To quantify the performance of our path planner, we compare the total path length and the corresponding safety score in a simple benchmark scenario. Here, a single pedestrian walks in front of the buggy, forcing the buggy to plan around the obstacle. We compare the individual models of risk, the GP risk and Dynamic Risk Density, against our hybrid model. For these tests, the Joint-Minimum model $r_j(x, y)$ defined in (12) is our chosen hybrid risk model. Snapshots from the video of these experiments is shown in Figure 2. We compare $n = 20$ trials for each risk model, and report the average path length, standard deviation values and the safety score for both static

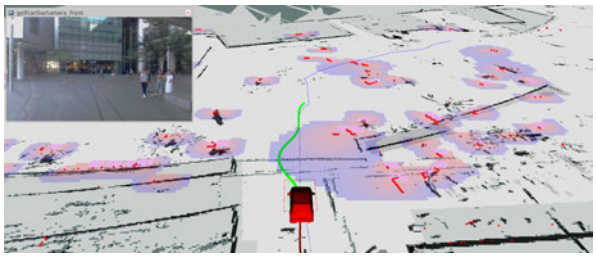
Risk Map	Hybrid Risk	GP Risk	Dynamic Risk Density
Avg (m)	19.7429	19.2716	19.3399
Std (m)	0.3	0.4357	0.2623
Safety score (static)	2.6711	2.1319	2.6619
Safety score (dynamic)	2.9249	2.8608	2.5215

TABLE I: Comparison of the ϵ-safe path to avoid a dynamic obstacle (pedestrian) and a static obstacle (a tree) with different risk maps. The hybrid risk map uses the "Joint-Minimum" $r_j(\cdot)$ from (12). Here, we compare the average path length from $n = 20$ trials. Safety score (static) refers to the path's minimal distance to the static obstacle in the environment, and Safety score (dynamic) refers to the path's minimal distance to the pedestrian.

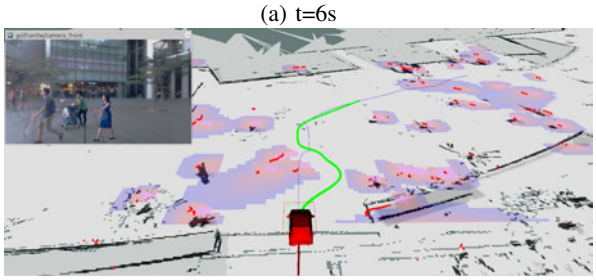
and dynamic obstacles in Table I. Here, a safety score for static obstacle is calculated as the re-planned path's minimal distance to the static obstacle, which is a tree, in the environment. The safety score for the dynamic obstacle is quantified as the re-planned path's minimal distance to the dynamic obstacle, which is a pedestrian. A statistical significance test shows a p-value less than 0.01 when compare the hybrid risk resultant path's length against the re-planned path's length of the other two risk models.

In Table I, we can see that the hybrid risk will make the re-planned path, on average, a little longer than the other two methods, which indicates that it is more conservative and hence safer. We also compare the minimum distance from a static obstacle (tree) and a dynamic obstacle (pedestrian) from the trials. The GP risk keeps a higher distance from the dynamic obstacle versus the static obstacle, which placed the buggy within 2.13 meters. Since the buggy's width is 1.4 meters. Compared to the GP model, the Dynamic Risk Density is less conservative around the dynamic obstacle, but more conservative around static obstacles. These results are expected: the GP risk model excels at modeling well-known dynamic obstacles, like pedestrians, while the Dynamic Risk Density provides a safety net agnostic to obstacle type. By taking the combination of these two models, we increase the safety scores for both static and dynamic obstacles, yielding a safer path.

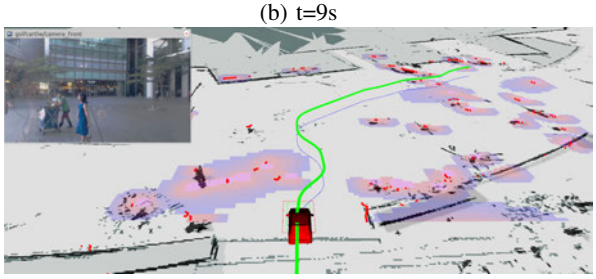
3) Safe Path Planning through Unstructured Environment with the Joint-Minimum Risk Model: For the comparison of ϵ-safe regions in Figure 5 and path deviations in Table I, we kept the autonomous buggy stationary to evaluate the evolution of the risk map and corresponding safe path planner. Here, we evaluate the buggy's performance using the multi-model risk map for its safe path planner when the buggy moves the dynamic environment. We consider the environment unstructured as pedestrians and bicycles move freely in any direction. We choose the Joint-Minimum model from (12) as the combined multi-model risk. Figure 6 shows



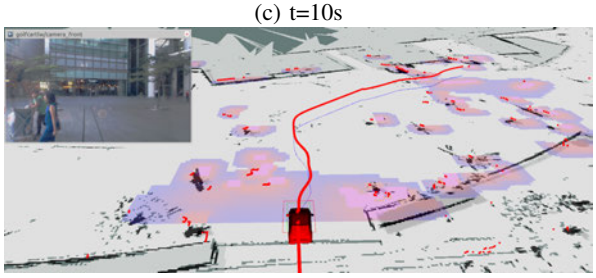
(a) t=6s



(b) t=9s



(c) t=10s



(d) t=12s

Fig. 6: Safe path planning with multi-modal risk map snapshots showcase. The 'Joint-Minimum' model is used to combine the GP and DRD risk maps, and the ego autonomous buggy will re-plan the safe path based on the combined risk map. The re-planned path is visualized in green, and the predefined reference path is plotted in yellow color for comparison.

snapshots from the video as the buggy moves through the environment.

V. Conclusions

This paper presents a method to combine multiple models of risk assessment into a single planning pipeline. We choose to combine a GP regulated risk map, which generates high fidelity predictions of individually-tracked objects, with Dynamic Risk Density, a method of assigning risk based on the occupancy density and velocity field in the environment. We present four methods on how to combine risk metrics, based

on the desired level of conservativeness when combining the estimates. From the combined multi-model risk map, we find the safe path through the environment following a grid-based planning method. Experiments performed on our autonomous buggy demonstrate how fusing the two risk maps provides a more robust assessment of the environment than either individual model.

In the future, we would like to expand the application scenarios to more urban environments with the autonomous vehicle testbed, to see how an efficient combination of multi-model risk maps can help with a better safe path planner. We would also like to embed more contextual information, such as road curvatures, pedestrian intentions with crosswalk information, into the safe path planner.

REFERENCES

[1] H. Guo, Z. Meng, Z. Huang, W. Leong, Z. Chen, M. Meghjani, M. H. A. Jr, and D. Rus, "Safe path planning with Gaussian process regulated risk map," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1–8.

[2] A. Pierson, C. Vasile, A. Gandhi, W. Schwarting, S. Karaman, and D. Rus, "Dynamic risk density for autonomous navigation in cluttered environments without object detection," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 5807–5814.

[3] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[4] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[5] D. Sharma and S. K. Dubey, "Anytime A * Algorithm-An Extension to A* Algorithm," *International Journal of Scientific & Engineering Research*, vol. 4, no. 1, 2013.

[6] S. Koenig and M. Likhachev, "D*lite," in *Eighteenth National Conference on Artificial Intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 476–483.

[7] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep., 1998.

[8] 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.

[9] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug 1996.

[10] M. Naderan-Tahan and M. T. Manzuri-Shalmani, "Efficient and safe path planning for a mobile robot using genetic algorithm," in *IEEE Congress on Evolutionary Computation*, May 2009, pp. 2091–2097.

[11] S. Bouraine, T. Fraichard, and O. Azouaoui, "Real-time Safe Path Planning for Robot Navigation in Unknown Dynamic Environments," in *CSA 2nd Conference on Computing Systems and Applications*, 2016.

[12] 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.

[13] H. Andersen, W. Schwarting, F. Naser, Y. Eng, M. H. Ang, D. Rus, and J. Alonso-Mora, "Trajectory optimization for autonomous overtaking with visibility maximization," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 10 2017, pp. 1–8.

[14] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 174–179, 2017.

[15] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, p. 463–497, 2015.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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

[25] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 6271–6278.

[26] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 68 – 73, 2015, analysis and Design of Hybrid Systems ADHS. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S240589631502412X

[27] D. Panagou, D. M. Stipanovi?, and P. G. Voulgaris, "Distributed coordination control for multi-robot networks using lyapunov-like barrier functions," *IEEE Transactions on Automatic Control*, vol. 61, no. 3, pp. 617–632, March 2016.

[28] 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, pp. 1–8.

[29] M. Turchetta, F. Berkenkamp, and A. Krause, "Safe exploration in finite markov decision processes with Gaussian processes," *CoRR*, vol. abs/1606.04753, 2016.

[30] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Proc. of Neural Information Processing Systems (NIPS)*, 2017.

[31] M. Turchetta, F. Berkenkamp, and A. Krause, "Safe exploration in finite Markov Decision Processes with Gaussian processes," in *Proc. of Neural Information Processing Systems (NIPS)*, 2016.

[32] S. Manjanna, N. Kakodkar, M. Meghjani, and G. Dudek, "Efficient terrain driven coral coverage using Gaussian processes for mosaic synthesis," in *2016 13th Conference on Computer and Robot Vision (CRV)*. IEEE, 2016, pp. 448–455.

[33] A. Wachi, Y. Sui, Y. Yue, and M. Ono, "Safe exploration and optimization of constrained MDPs using Gaussian processes," in *AAAI 2018*, 2018.

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

[35] S. D. Pendleton, W. Liu, H. Andersen, Y. H. Eng, E. Frazzoli, D. Rus, and M. H. Ang, "Numerical approach to reachability-guided sampling-based motion planning under differential constraints," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1232–1239, July 2017.

[36] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, no. 1, pp. 99 – 141, 2001.

[37] B. Qin, Z. J. Chong, S. H. Soh, T. Bandyopadhyay, M. H. Ang, E. Frazzoli, and D. Rus, "A spatial-temporal approach for moving object recognition with 2D LiDAR," in *Experimental Robotics*, M. Hsieh, O. Khatib, and V. Kumar, Eds. Springer, Cham: Springer Tracts in Advanced Robotics, 2016, pp. 807–820.

[38] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.