Dynamic Lambda-Field: A Counterpart of the Bayesian Occupancy Grid for Risk Assessment in Dynamic Environments

Johann Laconte, Elie Randriamiarintsoa, Abderrahim Kasmi, Francois Pomerleau, Roland Chapuis, Christophe Debain, Romuald Aufrere

Dynamic Lambda-Field: A Counterpart of the Bayesian Occupancy Grid for Risk Assessment in Dynamic Environments

Johann Laconte[1], Elie Randriamiarintsoa[1], Abderrahim Kasmi[1,2], François Pomerleau[3],
Roland Chapuis[1], Christophe Debain[4], Romuald Aufrère[1]

***Abstract*— In the context of autonomous vehicles, one of the most crucial tasks is to estimate the risk of the undertaken action. While navigating in complex urban environments, the Bayesian occupancy grid is one of the most popular types of maps, where the information of occupancy is stored as the probability of collision. Although widely used, this kind of representation is not well suited for risk assessment: because of its discrete nature, the probability of collision becomes dependent on the tessellation size. Therefore, risk assessments on Bayesian occupancy grids cannot yield risks with meaningful physical units. In this article, we propose an alternative framework called Dynamic Lambda-Field that is able to assess generic physical risks in dynamic environments without being dependent on the tessellation size. Using our framework, we are able to plan safe trajectories where the risk function can be adjusted depending on the scenario. We validate our approach with quantitative experiments, showing the convergence speed of the grid and that the framework is suitable for real-world scenarios.**

I. Introduction

In the context of robotics in urban environments, autonomous vehicles are beginning to coexist alongside human drivers. One of the main factors to take into account for each robot's decision is undoubtedly the risk of the undertaken action. The risk is tied to the probability of collision, conventionally defined as the probability of encounter between the robot and static or dynamic obstacles.

To assess this risk, the robot creates a map of the environment and identifies any potential threats or obstacles it must avoid. To create these maps, the robot can either use a semantic approach, namely identifying each obstacle as a unique entity, or use a metric approach by storing the occupancy information of each position in the environment. Although the first approach seems by far the most fitted for risk assessment scenarios, creating semantic maps is not an easy task. Indeed, identifying obstacles from raw measurements is complicated and can leave obstacles undetected if, for example, the obstacle has specific features that were not in the training set (e.g., people wearing a skirt or with strollers [1]). As it is critical to detect any possible threat, the metric map is then a reliable option since it does not require in-depth processing of measurements, thus providing a more conservative approach to risk assessment.

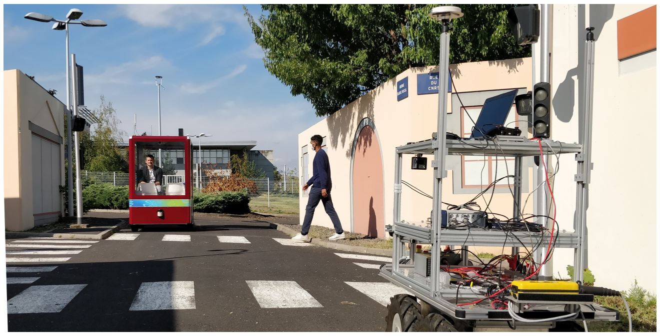

Fig. 1. Example of an urban scenario that the robot can encounter. A pedestrian is crossing the road, while a vehicle is approaching the robot in the other lane. Using the Dynamic Lambda-Fields, the robot is able to estimate the risk of its actions, taking into account a generic risk that can be adapted depending on the scenario.

The most commonly used metric map is the Bayesian occupancy grid: it tessellates the environment into cells where each cell stores the probability of occupancy (i.e., the probability that the cell will cause a collision if crossed). However, as highlighted by Heiden *et al.* [2], this type of map is not suitable for risk assessment. Indeed, the probability of collision depends on the size of the cells: for instance, for a map having cells twice as small as another map of the same environment, the robot would have to cross twice as many cells to reach its goal, increasing the probability of collision. Following this problem, any risk assessment strategy on Bayesian occupancy grids has to take into account the cell size in the equations. As such, the Bayesian occupancy grid cannot naturally yield risks but only probabilities of collision that are themselves dependent on the tessellation size. For instance, Figure 1 shows a scenario where a robot is crossing an urban-like environment: a car is coming in the other lane while a pedestrian suddenly crosses the road despite the vehicles approaching. In the case where the dynamics of the robot does not allow it to stop, a collision is inevitable, either with the pedestrian by simply braking, or with the vehicle in the other lane by swerving. Whereas Bayesian occupancy grid frameworks are not able to differentiate by themselves the risks associated with each decision, a human driver would have a more nuanced approach. If the risk only takes into account the damages to the robot itself, it is indeed safer to collide with the pedestrian since its mass is much lower than that of a car. However, if the risk takes into account the probability that both the robot and the collided obstacle

[1] Université Clermont Auvergne, CNRS, Clermont Auvergne INP, Institut Pascal, F-63000 Clermont-Ferrand, France; johann.laconte@uca.fr
[2] Sherpa Engineering, R&D Department, 333 Avenue Georges Clemenceau, 92000 Nanterre, France; a.kasmi@sherpa-eng.com
[3] Northern Robotics Laboratory, Université Laval, Canada; francois.pomerleau@ift.ulaval.ca
[4] Université Clermont Auvergne, INRAE, UR TSCF, F-63178 Aubière, France; christophe.debain@inrae.fr

survive, changing direction and colliding with the car might become a better solution.

In this article, we present a novel framework, named Dynamic Lambda-Field, specifically conceived to allow risk assessment in metric maps using a 2D lidar sensor. This concept answers the limitations of the Bayesian occupancy grid by proposing a method where the probability of collision does not depend on the tessellation size, thus allowing for continuous risk assessments over the path of the robot. We extend our theory presented in [3] to take into account dynamic environments. Using the framework presented in this article, the robot is able to plan safe trajectories using generic, user-defined risks that do not depend on the tessellation size while keeping their physical units.

II. Related Work

The most widely adopted representation of static environments is the Bayesian occupancy grid, introduced by Elfes [4]. Nègre *et al.* [5] proposed to extend the framework, estimating static and dynamic obstacles in occupancy grids, using particles to represent the dynamic world. Rummelhard *et al.* [6] enhanced the aforementioned framework by conducting a pre-processing step of estimating the state of the space (i.e., static, dynamic, empty or unknown). Our framework draws from this idea, inferring at each step the state of each cell, while providing a means of inferring physical risks from the map. In parallel, Nuss *et al.* [7] proposed to model the dynamic environment using random finite sets and Dempster-Shafer theory. Rexin *et al.* [8] built upon the previous method, using a pure Bayesian approach, leading to analogous results [6]. Guizilini *et al.* [9] proposed a method to learn and store continuous dynamic occupancy maps. Although providing an approach to learn complex pattern and dependencies in the environment, [9] relies on clustering and learning, two aspects where providing safety guaranties cannot be ensured. Furthermore, the problem of the tessellation size remains in continuous maps: the question only becomes how to integrate (instead of sum) probabilities of occupancy. As the information stored into such maps is not suitable for risk assessment, the Bayesian occupancy grid cannot yield meaningful risks.

The notion of risk and how to handle it is not uniquely defined in the literature. Majumdar *et al.* [10] addressed this question, exploring what criteria are needed to define risk metrics. In the context of occupancy grids, Rummelhard *et al.* [11] proposed to define the risk as the probability to collide with a specific area. The authors argued that a simpler approach such as the maximum value of collision over the cells tends to work better. Nevertheless, this approach discards all collisions with the exception of the one with the maximum likelihood, leading to potential hazardous situations (e.g., a path colliding with two pedestrians with chances of $90\,\%$ and $91\,\%$ may be more dangerous than a path colliding with only one pedestrian with a probability of $95\,\%$). Guardini *et al.* [12] inferred risk maps from a Bayesian occupancy grid and defined it as the severity of injury. However, they do not provide a means of inferring the risk of a path as the aforementioned problem of the Bayesian maps also applies here. As such, our framework can be seen as a counterpart of their work where it is possible to assess the risk of a path. Heiden *et al.* [2] proposed a method to assess collision probabilities in Bayesian occupancy grids, relying on product integrals. The main drawback is that there is no natural reason to use the concept of product integrals, as it is here merely a tool to lift the dependence on the tessellation size. In addition, their framework does not take into account the size of the robot. On the contrary, our method takes into account its size while providing a natural means of assessing generic risks. Finally, Fan *et al.* [13] planned trajectories in risk maps (i.e., a grid where each cell stores a risk) and defined a risk as a compound of one-step risk metrics. Each risk is defined as a weighted sum of different metrics (e.g., collision, slippage). However, compounding and summing risks therefore lose their physical meaning, whereas our framework maintains the physical meaning and therefore leads to more informed decisions.

In a different manner, Laugier *et al.* [14] clustered the dynamic obstacles of the Bayesian occupancy grid, hence falling back on more standard path planning problems where the obstacles are identified. However, this approach assumes that every obstacle has been successfully detected. Because of the possible wrong clustering (e.g., two pedestrians being detected as only one cluster), obstacles might have an incorrect velocity estimation hence leading the robot to make wrong decisions. On the contrary, our method does not require performing clustering to assess risk with a physical meaning, hence provides a more conservative approach. The risk function, that we define below as the maximum change of kinetic energy arising from a collision, keeps its physical unit and allows more nuanced decision-making than would be possible with only the probability of collision. Therefore, more adequate decisions can be taken by the robot, where each path possesses a meaningful risk with a physical unit.

III. Risk assessment in Dynamic Lambda-Field

We present here how to construct Dynamic Lambda-Fields as well as how to assess risks using this map as done in the static case in [3]. In order to motivate our approach, we show that the theory of the Lambda-Field naturally emerges from Bayesian occupancy grids when the cell size tends to zero. We assume that when the cell size tends to zero, the probability of a collision occurring in this cell is $\lambda\Delta a$, where $\lambda \in \mathbb{R}_{\geq 0}$ is the rate of the event collision and Δa is the area of the cell. The larger λ is, the more likely a collision will occur. Under these assumptions, the probability to safely cross a path (i.e., without collision) of N cells c_i of areas Δa with a constant rate of λ_i is

$$\prod_{i=0}^{N-1} (1 - \lambda_i \Delta a). \tag{1}$$

Taking the limit $\Delta a \to 0, N \to \infty$ leads to the computation of the Volterra type I product integral. For a path crossing a total area A where each cell of area Δa has a rate $\lambda(a)$, a being the total area crossed from the beginning, the probability of collision is

$$\begin{aligned}\mathbb{P}\left(\texttt{coll}\right) &= 1 - \lim_{\Delta a \to 0} \prod_{i=0}^{A/\Delta a} \left(1 - \lambda(i\Delta a)\Delta a\right) \\ &= 1 - \exp\left(-\int_0^A \lambda(a)\,\mathrm{d}a\right).\end{aligned} \tag{2}$$

It can be shown that taking such a limit leads to the Poisson point distribution. Therefore, the theory of Lambda-Field relies on this distribution. Instead of storing the probability of occupancy for each obstacle (i.e., cell and potential dynamic obstacles), as done in the standard Bayesian occupancy grids, the Lambda-Field stores the intensity of the cells, namely a value $\lambda_i \in \mathbb{R}_{\geq 0}$ for each obstacle. The higher the intensity, the higher is the probability that crossing the cell will result in a collision. Conversely, a value of zero indicates that the cell will never lead to a collision.

In this article, we define the risk as the maximum change of kinetic energy of the robot and the obstacle (static or dynamic) due to a collision. The framework allows the risk function to be changed depending on the application. For example, the risk function could also model the probability of survival for both the vehicle occupants and the collided obstacle if it is human. By keeping the risk in its physical form, the robot is able to make more informed decisions without the need of tuned user-defined thresholds. First, we show how to compute the Lambda-Field, namely the intensities λ_i of both static and dynamic obstacles, using a lidar sensor. Then, we present how to manage the particles used to represent the obstacles in the environment. Finally, we show how to use this field to assess physical risks given a generic risk function, leading to the generation of safe paths for the robot.

A. Mapping

As the environment contains both static and dynamic obstacles, the value of the lambda of a cell c_i (i.e., its intensity representing the likelihood of a collision occurring in this cell) is stochastic. Indeed, the cell can be occupied by a static obstacle, a dynamic obstacle or be free of obstacles. Therefore, we define the probability of collision as the probability of colliding with an obstacle in the expectation of the intensity field, leading to

$$\mathbb{P}\left(\texttt{coll}\right) = 1 - \exp\left(-\sum_{c_i \in \mathcal{C}} \Delta a \mathbb{E}\left[\lambda_{c_i}^{t_i}\right]\right), \tag{3}$$

for a path crossing the cells $\mathcal{C} = \{c_0, \ldots, c_{N-1}\}$ at the times $\{t_0, \ldots, t_{N-1}\}$, of expected intensities $\left\{\mathbb{E}\left[\lambda_{c_0}^{t_0}\right], \ldots, \mathbb{E}\left[\lambda_{c_{N-1}}^{t_{N-1}}\right]\right\}$ and of area Δa. One can note that the probability of collision does not depend on the tessellation size as the sum is weighted by the area of the cells, as shown in [3].

In order to estimate the expectation of the intensity $\mathbb{E}\left[\lambda_c^t\right]$ for each cell c, we use particles that represent the possible obstacles in the environment. The static grid is considered as a set of particles that do not move and have the size of a cell, whereas dynamic obstacles are defined as moving particles of different classes. We define in this work three different classes of particles: the ‘cell’ class, for the static environment, the ‘pedestrian’ class and the ‘car’ class for the dynamic obstacles. Each particle has a position, a speed, a velocity profile (i.e., a car can achieve greater speed than a pedestrian, whereas a pedestrian can change direction quicker). Each particle is also defined by their size. For instance, a pedestrian particle is represented by a $40\,\mathrm{cm}$ diameter circle whereas a vehicle is described by a rectangle of $2 \times 1\,\mathrm{m}$. Note that the particles' footprints are not dependent on the tessellation size. Naturally, particles of the ‘cell’ class have null velocity and acceleration. Using these attributes, the framework is able to represent the different obstacles more efficiently, as only one particle can represent an obstacle even if it spans several cells, contrary to [6] where a particle does not have a size. Also, using the particle classes representing the obstacle, the classes of the obstacles are inferred at the same time, allowing the framework to take into account the classes of the obstacles while assessing the risk of a path. For instance, if the risk function takes into account the probability of survival of collided obstacles, it allows the robot to prefer colliding with a car over a pedestrian.

As an example, Figure 2 gives the Lambda-Field of Figure 1 where the robot wants to cross the environment while a pedestrian abruptly appears on the road, therefore having only two choices: either braking and colliding with the pedestrian or swerving and colliding with the car in the other lane, hence saving the pedestrian. The environment is populated by static particles (one per cell) of different intensities whereas the possible dynamic obstacles are represented using dynamic particles (in this example, only two, one of type ‘pedestrian’ and the other of type ‘car’). The expectation of the cells' intensity that the robot crosses is derived in the following section, taking into account the intensity of the particles that are in the cell at the time of traversal (a static particle and possibly several dynamic ones). Once the expected intensity of the cells is computed, the probability of collision of the robot's path is given by Equation 3, integrating the expected intensity of the cells crossed by the robot. Then, the expectation of a risk function over the path can be computed, giving more insight on which path is the safest.

B. Computation of the expectation of a cell

In order to measure particles and infer the risk of a path, the first step is to compute the expectation of the intensities of the cells. When measuring the environment at a given time, we assume that a cell cannot contain more than one obstacle (e.g., a cell can contain a wall or a car but not both). This consideration allows us to only measure the obstacle that is in the cell, as for instance we do not want to measure the static part of a cell as occupied if a moving car is in the same cell at this time. Furthermore, we also model the probability that a particle still exists: it is indeed unlikely that a particle

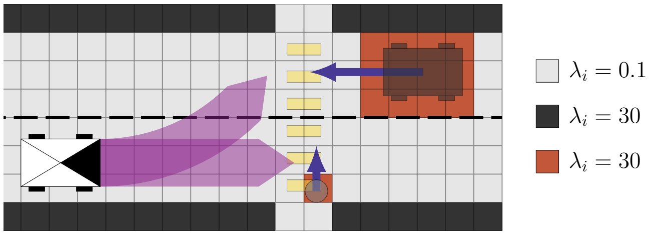

Fig. 2. Example of Dynamic Lambda-Field where a pedestrian emerges unexpectedly on the road in front of the robot (black box with its front represented as a filled triangle) while another vehicle is approaching in the opposite lane. The environment is represented as a set of cells storing the expected intensity, computed using the intensity of the static (resp. dynamic) particles, depicted in gray (resp. red) scale, occupying these cells. Depending on the risk function, the robot can assess which one of the two paths (in purple) is the safest.

still represents an obstacle after having run into a wall. If this case occurs, the existence probability of the particle will drop.

Under these considerations, the expected lambda of a cell c is given by

$$\mathbb{E}\left[\lambda_c^t\right] = \sum_{p_i \in c} \lambda_i \mathbb{P}\left(p_i \in c\right) \mathbb{P}\left(e_i \,|\, p_i \in c\right), \tag{4}$$

where λ_i is the lambda of the particle p_i being in the cell at the time of traversal t, $\mathbb{P}\left(p_i \in c\right)$ depicts the probability that the particle p_i is the one located in the cell and $\mathbb{P}\left(e_i \,|\, p_i \in c\right)$ is the probability that the particle p_i exists. In the case where there is truly no obstacle in the cell, the probability $\mathbb{P}\left(p_i \in c\right)$ will be one for the static particle, which can have a very low lambda, thereby meaning that the cell is safe to cross, or a high lambda, meaning that the cell contains a static obstacle. Assuming that every cell has exactly one obstacle, we can compute the probability of the particle p_i of being in the cell c as the probability that only the particle p_i creates a collision given that there is exactly one obstacle in the cell:

$$\begin{aligned} &\mathbb{P}\left(p_i \in c \,\middle|\, (p_0 \in c) \oplus \cdots \oplus (p_{N_P-1} \in c)\right) = \\ &\quad \frac{\left(1 - \exp\left(-\lambda_i \Delta a\right)\right) \prod_{j \neq i} \exp\left(-\lambda_j \Delta a\right)}{\sum_k \left(1 - \exp\left(-\lambda_k \Delta a\right)\right) \prod_{j \neq k} \exp\left(-\lambda_j \Delta a\right)} \\ &\quad \propto \frac{1 - \exp\left(-\lambda_i \Delta a\right)}{\exp\left(-\lambda_i \Delta a\right)}, \end{aligned} \tag{5}$$

for a cell c of area Δa where N_P particles lie in it, $\oplus$ being the standard XOR operator. The probability of existence $\mathbb{P}\left(e_i \,|\, p_i \in c\right)$ is defined as the joint probability that the particle did not collide with the static environment since its creation and that it still follows an obstacle, computed as

$$\mathbb{P}\left(e_i \,|\, p_i \in c\right) = \exp\left(-\sum_{c \in \mathcal{P}_i} \lambda_{c,s}\right) \cdot \exp\left(-\tau \cdot t_i\right), \tag{6}$$

where $\mathcal{P}_i$ is the path (set of crossed cells) of the particle p_i, $\lambda_{c,s}$ is the lambda of the static particle at the cell c (we allow dynamic particles to cross without collision), t_i is the time since the last measurement 'hit' (i.e., the lidar hits the obstacle) of p_i and τ is the rate of the distribution. Indeed, if the particle has not been measured for a long time, it is very likely that either the particle lost the obstacle or that the obstacle left the field of view of the robot. This probability can also take into account other sensors such as a camera: for instance, the probability of existence of particles of type 'pedestrian' should drop if the camera informs the robot that there is a car at this position.

C. Measuring particles

Using the expressions of the probability of collision given by Equation 3 and the expectation of the intensities of the cells given by Equation 4, we provide a means of estimating the lambdas of the static field as well as the dynamic particles using a lidar sensor. As in [3], we determine the combination of the intensities $\lambda = \{\lambda_i\}$ of the particles that maximizes the expectation of the K beams the lidar has shot since the beginning. The lidar is modelled as a range sensor with an error region of area e: for a lidar measurement, the true position of the obstacle is contained within the error region, itself centered on the measurement. In the case of a perfect sensor, this region is reduced to a point at the measurement position. Using the derivation in [3] and the expectation of the intensity given by Equation 4, the intensities of the particles are given by

$$\lambda_i = \frac{1}{e} \ln\left(1 + \frac{h_i}{m_i}\right), \tag{7}$$

where h_i is the sum of probabilities $\mathbb{P}\left(p_i \in c_h\right) \mathbb{P}\left(e_i \,|\, p_i \in c_h\right)$ each time the particle p_i has been counted as 'hit' in the cell c_h (i.e., was in the region of error of the sensor) and m_i the sum of probabilities $\mathbb{P}\left(p_i \in c_m\right) \mathbb{P}\left(e_i \,|\, p_i \in c_m\right)$ each time the particle p_i has been counted as 'miss' in the cell c_m (i.e., the lidar beam crossed the cell without collision). For this equation to be true, one needs the assumption that for a given measurement, all of the particles measured in the error region of the sensor have the same intensity. As the lidar error region tends to be small, this assumption holds in every situation except for low-intensity static particles. Indeed, high-intensity dynamic particles can easily come to the cell containing the low-intensity particle, thereby the assumption that every obstacle has the same intensity no longer holds. This case is tackled by assuming that the low-intensity static particle has the same intensity as the high-intensity particles, therefore overestimating its intensity and keeping a conservative approach.

D. Particle Management

Using the particles update equation previously derived, the Lambda-Field is updated as described below. At each iteration, the particles evolve, are updated with lidar measurements and are resampled. First, the particles evolve using a simple update equation:

$$\begin{aligned} \boldsymbol{x}_i^t &= \boldsymbol{x}_i^{t-1} + \boldsymbol{v}_i^t \Delta t, \\ \boldsymbol{v}_i^t &= \boldsymbol{v}_i^{t-1} + \boldsymbol{a}_i \Delta t \quad \text{with } \boldsymbol{a} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i), \end{aligned} \tag{8}$$

for the particle p_i of position and speed $\boldsymbol{x}_i^t, \boldsymbol{v}_i^t \in \mathbb{R}^2$ at the time t, where $\boldsymbol{a}_i \in \mathbb{R}^2$ is a centered Gaussian random variable of covariance $\boldsymbol{\Sigma_i}$ depicting the acceleration of the particle. Evidently, static particles have zero speed and acceleration. Then, for each measured cell c, we compute the probability that the particle p_i is indeed the particle in the cell. Each particle in the 'hit' zone (resp. 'miss' zone) of the lidar measurements increments its 'hit' counter h_i (resp. 'miss' counter m_i) of the quantity $\mathbb{P}\left(p_i \in c\right) \mathbb{P}\left(e_i \,|\, p_i \in c\right)$ as shown in Equation 7. Moreover, particles have a low probability to switch classes (i.e., 'pedestrian' switching to 'car' and vice versa). This consideration avoids an incorrect convergence of the particles (e.g., a low velocity car can be represented as a group of pedestrians, however both hypotheses have to be maintained).

Once the particles have been updated, they are resampled according to the joint probability of their existence and selecting one of the cells in which they are located:

$$w_i \propto \mathbb{P}\left(p_i \in c\right) \mathbb{P}\left(e_i \,|\, p_i \in c\right) \sum_{p_i \in c} 1 - \exp\left(-\Delta a \mathbb{E}\left[\lambda_c^t\right]\right) \quad (9)$$

Moreover, particles can be born during the resampling step. Indeed, obstacles can appear on the map and the particles might not converge to an obstacle, leaving a dynamic obstacle unidentified. Even if this situation is less likely the more particles we have, dealing with such a critical case is necessary. Hence, we allow the birth of particles inside any cell c measured 'hit' by the lidar with

$$w_c^{\texttt{birth}} = \gamma \cdot \exp\left(-\Delta a \sum_{c \in \mathcal{E}_k} \mathbb{E}\left[\lambda_c^t\right]\right), \quad (10)$$

for every 'hit' measurement of error region $\mathcal{E}_k$, where γ is a coefficient controlling the proportion of births at each resampling. If the measured cell is already populated by particles of high lambdas, meaning that the obstacle is already represented, the birth probability drops to zero. If the particle which is born is picked during the resampling, random class and speed are drawn from a uniform distribution. Algorithm 1 summarizes the procedure.

Algorithm 1: Particles management

```
repeat
    Evolve particles using Equation 8
    Update particles with measurements using Equation 7
    forall particles $p_i$ do
        Draw a particle $p_k$ with weight $w_k$ and birth weight $w_c^{\texttt{birth}}$
        Replace particle $p_i$ with particle $p_k$
    end
until True;
```

Once every particle has been updated and resampled, we infer the underlying distribution for each cell. Indeed, planning while taking into account every particle would not reach real-time constraints. For each cell, the speed is modeled as a normal distribution $\mathcal{N}(\mu_v, \sigma_v)$ whereas the direction is modeled using the Von-Mises distribution (as the direction lies on a circle) of parameters $\mu_\theta, \kappa_\theta$, where these parameters are estimated as in [15]. In the case where κ_θ is large enough, the Von-Mises distribution can be approximated by a normal distribution of the same mean with a standard deviation of $\sigma_\theta = \sqrt{1/\kappa_\theta}$.

At the end of the mapping process, a grid is created where each cell contains an intensity for the static part, as well as an intensity for each type of dynamic particle (i.e., summing the intensities of the particles of the same class, using Equation 4). We do not sum the intensities of the different classes since the risk can depend on the type of obstacle. We instead provide each class intensity to the planner. As we defined three classes in this article (static, pedestrian and car), each cell contains three intensities, one for each obstacle.

E. Risk assessment and Path Planning

Using the data provided by the dynamic map, we generate safe trajectories for the robot. As in Gerkey *et al.* [16], we sample commands of translational and rotational velocities (v, w) and choose the one leading the closest possible to the objective while being below the allowed risk. In each cell, the robot has to go through all of the obstacles simultaneously (i.e., static obstacle and all of the dynamic ones arriving in the cell at the same time as the vehicle). The probability of colliding with the obstacle o_k in a cell c containing N_O obstacles $o_0, \ldots, o_{N_O-1}$ of intensities $\lambda_0, \ldots, \lambda_{N_O-1}$ at the time t is the joint probability of not colliding the other obstacles $\{o_i\}_{i \neq k}$ and colliding with the obstacle o_k:

$$\begin{aligned} \mathbb{P}\left(\texttt{coll}_{c,k}\right) &= \int_0^{\Delta a} \exp\left(-a \sum_{i \neq k} \lambda_i\right) \cdot \lambda_k \exp\left(-a\lambda_k\right) \mathrm{d}a \\ &= \frac{\lambda_k}{\mathbb{E}\left[\lambda_c^t\right]} \left[1 - \exp\left(-\Delta a \mathbb{E}\left[\lambda_c^t\right]\right)\right], \quad (11) \end{aligned}$$

where $\mathbb{E}\left[\lambda_c^t\right] = \sum_{i=0}^{N_O-1} \lambda_i$ is the expected lambda of the cell c at the time of traversal t, where the intensities λ_i of the incoming obstacles o_i have been pondered by the probability of reaching the cell beforehand. One can note that the expectation is the same as Equation 4 where we lifted the assumption that only one obstacle can be in the cell. Indeed, even if only one obstacle can truly be in a cell, several obstacles can reach the cell from different directions and create a collision with the robot which attempts to cross it. Since several obstacles can hit the robot in the same cell, the assumption of each cell having only one obstacle is lifted, yielding $\mathbb{P}\left(p_i \in c\right) = 1$ for the computation of expectation in risk assessment. Using Equation 11, we compute the risk of a path as the expectation of the risk function $r(a, o_k)$, where a is the traversed area at the time of the collision and o_k is the obstacle the collision occurs with, as

$$\mathbb{E}\left[r(\cdot)\right] = \sum_{i=0}^{N-1} K_i \sum_{o_k \in c_i} \frac{\lambda_k}{\mathbb{E}\left[\lambda_{c_i}^{t_i}\right]} \cdot r(i\Delta a, o_k), \quad (12)$$

with $K_i = \exp\left(-\sum_{j=0}^{i-1} \Delta a \mathbb{E}\left[\lambda_{c_j}^{t_j}\right]\right)\left(1 - \exp\left(-\Delta a \mathbb{E}\left[\lambda_{c_i}^{t_i}\right]\right)\right)$, for a path passing through the cells $\{c_0, \ldots, c_{N-1}\}$ at the times $\{t_0, \ldots, t_{N-1}\}$ of expected lambdas $\mathbb{E}\left[\lambda_{c_i}^{t_i}\right]$. The curvilinear abscissa s is linked to the traversed area by the relation $s = i\Delta a/L$ where L is the width of the robot. In order to determine which obstacles are in the cell at the time of traversal, we convert the velocity distribution of each dynamic cell into a set as depicted in Figure 3. The set corresponds to the shape of the obstacle's (i.e., dynamic cell) path using two sigmas on its velocity and orientation. This shape is then tested to cross the cells of the robot's path, using the Gilbert-Johnson-Keerthi distance algorithm. In the case of a collision between the robot's path and the cell's path (dashed cells in Figure 3), the earliest arrival and latest departure time of the dynamic obstacle and the robot are compared. If the time intervals intersect, the obstacle is said to be in the cell of the robot's path. Using this risk assessment method, the robot is able to effectively assess the risk of a path. In the following section, the risk is defined as the change of kinetic energy arising from the collision, thus taking into account the harmfulness of the collision for both the robot and the obstacle it collides with.

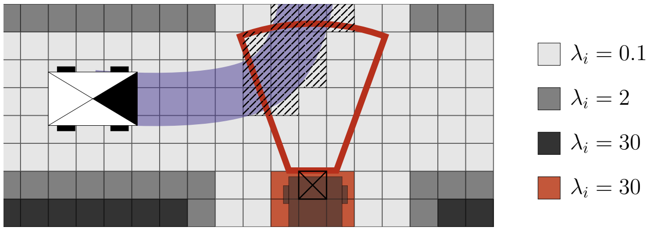

Fig. 3. Example of risk assessment. The robot (left vehicle) wishes to cross the crossroad using the trajectory depicted in blue. A dynamic obstacle arrives from the bottom of the map, with a probabilistic direction and speed, converted into a set (in red) that corresponds to the possible positions of the obstacle. Using these distributions for each dynamic cell (here depicted for the crossed one), the potential collision positions are marked (dashed cells). If the times of traversal intersect, the obstacles are added to the crossed cells for risk assessment.

IV. Validation of the Framework

In order to show the applicability of our theory, we implemented our mapping framework on a Jetson TX2 GPU. We used the robot depicted in Figure 1 (right), equipped with a lidar LMS-151 located at its front. As the aim is to plan short distance trajectories and the map is centered on the robot, we only used the odometry for the relative displacements of the robot. To estimate the states of the dynamic obstacles, we used 2×10^4 particles in the experiments, running at more than $10\,\mathrm{Hz}$ and more than $5\,\mathrm{Hz}$ if the planning segment is carried out on the same GPU. The map size was set to 200×200 cells of size $15 \times 15\,\mathrm{cm}$, resulting in a map of $30 \times 30\,\mathrm{m}$. In order to accelerate the convergence of the particles towards the obstacle, particle velocities were resampled with a Gaussian noise of $\sigma = 0.3\,\mathrm{m\,s^{-1}}$.

A. Evaluation

First, we show that the convergence speed of the framework. Using the same type of validation as Nuss *et al.* [7], we simulated an environment where an obstacle was approaching the robot at a known velocity and orientation. This experiment was repeated 50 times in which the obstacle was either a pedestrian or a car. The velocity of the obstacle was chosen to be $1.5\,\mathrm{m\,s^{-1}}$, as this profile of speed can be matched by either the 'pedestrian' or the 'car' class. Using the fact that only one dynamic obstacle is in the environment, we retrieve the mean and a confidence interval at two sigmas for the velocity and the orientation. Figure 4 shows the results for a pedestrian (in green) and a car (in blue). We can see that both the speed and the orientation converge to a valid value after less than $2\,\mathrm{s}$. Furthermore, as the framework assesses the risk by allowing two sigmas on the velocity and the orientation, the ground truth is always contained in the estimation. At first, the speed of the obstacle is overestimated as both cars of high velocities and pedestrian of low velocities coexist. In the case of the pedestrian, all particles of class 'car' are discarded as their size is too large for the obstacle (i.e., they lie in cells counted as free by the sensor). Once the 'car' particles are removed at $t \approx 1\,\mathrm{s}$, the mean velocity converges to the true value at the next iteration. In the case of the car, the car particles of high velocities are discarded as soon as they leave the measurements zone, which is indicated by the fact that the convergence of the mean is smoother than in the pedestrian experiment.

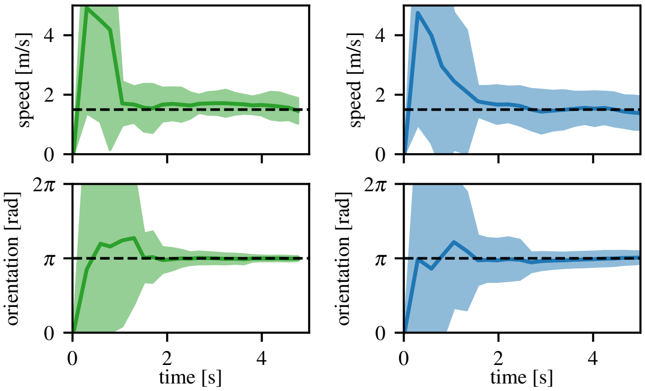

Fig. 4. Convergence of the speed and orientation of a single dynamic obstacle (Left: pedestrian; Right: car) where the ground truth is depicted in dashed black. The mean is displayed in solid line whereas the confidence interval at two sigmas is depicted in light shade.

The convergence of the obstacle class is also studied. We compute the probability of the obstacle to be of a certain class as the probability to collide with dynamic cells of the class. Figure 5 shows the resulting convergence for a pedestrian and a car. In the case of the pedestrian, the framework quickly converges to the real class of the obstacle after $1\,\mathrm{s}$. However, in the case of the car, the framework cannot decide whether the obstacle is a car or a group of pedestrians. Indeed, with only a lidar, the obstacle can match both classes equivalently. Other sensors such as a camera could remove the ambiguity. Also, the class would be determined to be a car if the velocity profile only matched the 'car' class (i.e., the obstacle moves at greater velocity). Therefore, the framework is able to effectively infer the velocity, orientation and class (when possible) of the different obstacles.

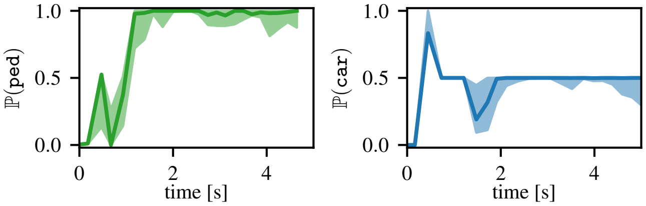

Fig. 5. Convergence of the class probability for a pedestrian (left) and a car (right) with the quantiles at 25% and 75% in light shade. Whereas the pedestrian has its class quickly inferred, the framework cannot decide between a car and a group of pedestrians in the case where the car has a speed matching both classes.

B. Validation of the approach

In the following experiment, the risk is defined as the maximum gain of kinetic energy between the robot and the obstacle it collided with, assuming inelastic collisions, as

$$r(s, o_k) = \max\left(\frac{1}{2}m_R\left(v_R(s) - v_f\right)^2, \frac{1}{2}m_k\left(v_k - v_f\right)^2\right)$$

$$\text{with } v_f = \frac{m_R v_R(s) + m_k v_k}{m_R + m_k} \tag{13}$$

where $m_R, v_R(s)$ is the mass and velocity of the robot at the curvilinear abscissa s, m_k, v_k the mass and velocity of the obstacle o_k, and v_f the final velocity of the obstacles after collision. The mass of the robot was set to $150\,\mathrm{kg}$ while the masses of the 'pedestrian' and 'car' classes were set to $80\,\mathrm{kg}$ and $500\,\mathrm{kg}$ respectively. This risk enables the consideration of both the possible damages suffered by the robot but also by the obstacle it collided with. As such, the decision to collide with a pedestrian takes into account that although the robot will suffer little damage, the pedestrian is at a much greater risk. Note that the risk function can be adapted depending on the context and can take into account other elements if available such as slippage, car deformation and so on. The static environment is assumed to have infinite mass, meaning that collisions with the static environment will always lead the vehicle to stop. As shown in Figure 1, the robot had to move through an urban-like environment consisting of a crossroad, where other agents such as pedestrians and cars were also evolving. A car was approaching in the other lane and a pedestrian entered the field of view of the robot from behind, afterwards crossing the road in front of it. For obvious security reasons, in this experiment, the velocity of the robot was bounded such that it will always be able to instantly stop in the case of hazardous situations where every path is too risky, contrary to the scenario depicted in Figure 2.

Figure 6 shows the resulting Lambda-Field for two different timestamps. The static environment is depicted with a gray scale, whereas the dynamic environment is shown with a red scale. Using our framework, the velocity distribution of each cell is extracted as well as a lambda of the static environment and each type of particle (i.e., in our case a lambda for the 'car' class and a lambda for the 'pedestrian' class). At $t = 19\,\mathrm{s}$, the large dynamic high-lambda zone (in red) of the map corresponds to the car, where the probability of being a car is approximatively $50\,\%$ for the underlying cells. The probability did not converge to $100\,\%$ because without additional sensors, the framework cannot decide whether the obstacle is a car or a group of pedestrians, as the velocity of $1.2\,\mathrm{m\,s^{-1}}$ is possible for both classes. At $t = 28\,\mathrm{s}$, a pedestrian emerges from behind the robot, then crosses the road right in front of it a few seconds later. When the robot detects the new obstacle, an inconsistency between the map and the measurements is found, leading the framework to create many particles on this location. Note the light gray trace which is left behind the pedestrian, as the framework had not yet decided whether the obstacle was static or dynamic. As shown in the right polar distribution (i.e., angle for the orientation, radius for the velocity), the hypothesis of the pedestrian pursuing its northward trajectory (top of the figure) is maintained. This hypothesis models the fact that without other sensors, the obstacle can in fact represent two pedestrians walking together, with the probability that they can change directions. The wrong hypothesis (i.e., pedestrian moving north) is discarded over the next iterations.

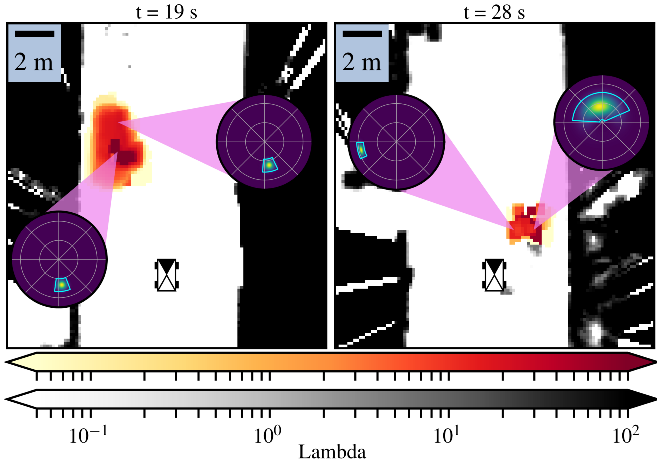

Fig. 6. Example of field resulting from our method, where the robot is located at the bottom center of the map. The static (resp. dynamic) environment is represented using a gray (resp. red) scale. A car is approaching the robot at $t = 19\,\mathrm{s}$, whereas a pedestrian is crossing the road at $t = 28\,\mathrm{s}$ right in front of the robot. Some speed distributions of the cells are displayed in polar plots (angle for the orientation, radius for the velocity), where inner circles correspond to a step of $1\,\mathrm{m\,s^{-1}}$.

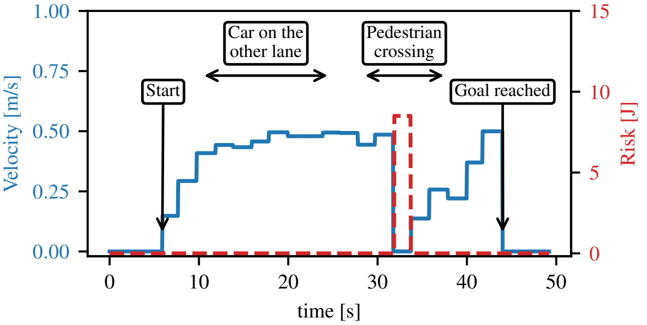

Fig. 7. Risk undergone by the robot during its traversal with the associated speeds. First, a car passed the robot in the other lane, where its speed was precise enough not to cause the robot to brake or change direction. After that, a pedestrian emerges from behind the robot and crosses the road in front of it, leading the robot to stop and wait for the pedestrian to free the way. The robot then rejoins its goal without further obstacles.

Finally, we used this dynamic map to plan safe paths for the robot. The goal of the robot was set at the top of the map, $15\,\mathrm{m}$ away from its position. To do so, we set the maximum risk to be $r_{\text{max}} = 1\,\text{J}$ where any path below this risk is considered safe. Figure 7 shows the risks the robot underwent during the traversal. First, no obstacles were in sight, leading the robot to accelerate to its maximum speed, here at $0.5\,\text{m}\,\text{s}^{-1}$. At $t = 10\,\text{s}$, a car entered the field of view of the robot. As the obstacle was far away enough during the convergence of the speed and orientation, the robot did not stop its course. At $t = 22\,\text{s}$, the car passed on the left side of the robot. As the velocities were precise enough not to encounter the path of the robot, it continued its course at full speed. At $t = 31\,\text{s}$, the pedestrian took a hard left turn, deviating from its expected trajectories thereby leading the framework to birth particles on its position. Consequently, the robot detected a danger on every path it could take, leading it to stop as this decision is the one minimizing the risk. The associated risk is then the risk of the pedestrian running into the robot, thus harming himself. In contrast, the Bayesian occupancy grid would only yield a probability of collision (in this case equal to one) and the robot could not distinguish between the collision at full speed and the collision at rest since both paths lead to a collision with the pedestrian. If the velocity of the robot did not allow it to stop, the robot would then prefer to collide with a parked car of same weight, as the resulting risk is lower. After a few iterations, the velocity of the pedestrian re-converged, and the robot continued on its way as soon as the pedestrian left. The robot accelerated to its maximum velocity and reached its goal safely.

V. Conclusion

In this article, we proposed a novel framework for generic risk assessment in occupancy grids. Using particles, we modeled both static and dynamic environments, deriving at the same time the nature of the obstacles. We first showed how to compute the Dynamic Lambda-Field using lidar measurements. Then, the resulting dynamic map is used to assess the risk for a given path, here defined as the change of kinetic energy due to a collision with an obstacle. On the contrary, the Bayesian occupancy grid would only yield the probability of collision, hence not being able to differ between a collision with a car or a pedestrian. Using this formulation of the risk, the robot was able to plan real-time safe trajectories in a dynamic environment. As this work focused on explaining the theoretical framework and providing a use case for its application, future work will involve extensive experiments with the framework in both real and simulated benchmarks, allowing for more complex and hazardous scenarios. Future work will also address particle convergence guarantees for a wider range of scenarios. Finally, other risks, metrics and particle types will be used to better characterize the risk of a path in more complex situations.

ACKNOWLEDGMENT

We thank Ted Morell for his help with the implementation of the framework. This work was sponsored by a public grant overseen by the French National Research Agency as part of the “Investissements d’Avenir” through the IMobS3 Laboratory of Excellence (ANR-10-LABX-0016), the IDEX-ISITE initiative CAP 20-25 (ANR-16-IDEX-0001) and the RobotEx Equipment of Excellence (ANR-10-EQPX-0044).

References

[1] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, “Pedestrian recognition using on-vehicle LIDAR,” in *IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 405–410.

[2] E. Heiden, K. Hausman, and G. S. Sukhatme, “Planning High-speed Safe Trajectories in Confidence-rich Maps,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2880–2886, 2017.

[3] J. Laconte, C. Debain, R. Chapuis, F. Pomerleau, and R. Aufrère, “Lambda-Field : A Continuous Counterpart of the Bayesian Occupancy Grid for Risk Assessment,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 167–172, 2019.

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

[5] A. Nègre, L. Rummelhard, and C. Laugier, “Hybrid Sampling Bayesian Occupancy Filter,” *Intelligent Vehicles Symposium Proceedings*, pp. 1307–1312, 2014.

[6] L. Rummelhard, A. Negre, and C. Laugier, “Conditional Monte Carlo Dense Occupancy Tracker,” *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 2485–2490, 2015.

[7] D. Nuss, S. Reuter, M. Thom, T. Yuan, G. Krehl, M. Maile, A. Gern, and K. Dietmayer, “A random finite set approach for dynamic occupancy grid maps with real-time application,” *International Journal of Robotics Research*, vol. 37, no. 8, pp. 841–866, 2016.

[8] N. Rexin, D. Nuss, S. Reuter, and K. Dietmayer, “Modeling occluded areas in dynamic grid maps,” *International Conference on Information Fusion*, pp. 1–6, 2017.

[9] V. Guizilini, R. Senanayake, and F. Ramos, “Dynamic hilbert maps: Real-time occupancy predictions in changing environments,” *IEEE International Conference on Robotics and Automation*, pp. 4091–4097, 2019.

[10] A. Majumdar and M. Pavone, “How should a robot assess risk? Towards an axiomatic theory of risk in robotics,” *arXiv*, pp. 75–84, 2017.

[11] L. Rummelhard, A. Nègre, M. Perrollaz, and C. Laugier, “Probabilistic Grid-based Collision Risk Prediction for Driving Application,” *Experimental Robotics*, pp. 821–834, 2014.

[12] L. A. S. Guardini, A. Spalanzani, C. Laugier, P. Martinet, A. L. Do, and T. Hermitte, “Employing Severity of Injury to Contextualize Complex Risk Mitigation Scenarios,” *IEEE Intelligent Vehicles Symposium, Proceedings*, no. Iv, pp. 1839–1845, 2020.

[13] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-A. Agha-Mohammadi, “STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation,” 2021.

[14] C. Laugier, I. Paromtchik, M. Perrollaz, Y. Mao, J.-D. Yoder, C. Tay, K. Mekhnacha, and A. Nègre, “Probabilistic Analysis of Dynamic Scenes and Collision Risk Assessment to Improve Driving Safety,” *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 4, pp. 4–19, 2011.

[15] R. Senanayake and F. Ramos, “Directional grid maps: modeling multimodal angular uncertainty in dynamic environments,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3241–3248, 2018.

[16] B. Gerkey and K. Konolige, “Planning and control in unstructured terrain,” *ICRA Workshop on Path Planning on Costmaps*, 2008.