Safe prediction-based local path planning using obstacle probability sections

Tanja Hebecker[1] and Frank Ortmeier[2]

Abstract— **Autonomous mobile robots gain more and more importance. In the nearest future they will be a part of everyday life. Therefore, it is critical to make them as reliable and safe as possible. We present a local path planner that shall ensure safety in an environment cluttered with unexpectedly moving obstacles. In this paper, the motion of obstacles is predicted by generating probability sections, and collision risks of path configurations are checked by determining whether these configurations lead inevitably to a collision or not. The presented approach worked efficiently in scenarios with static and dynamic obstacles.**

I. INTRODUCTION

Due to the fast development in the field of autonomous mobile robotics the importance of safe motion planning increases. Even in situations with many irrationally moving obstacles local path planners have to guarantee a safe avoidance of obstacles. Therefore, it is necessary to consider the robot's kinematics, the motion of dynamic obstacles and also risks that can arise in the future even if a robot is located at a point of a path (waypoint) that is itself collision-free.

Within the last years many new path planning approaches for autonomous robots were presented but to our knowledge there is no approach that considers both future collision probabilities of a waypoint in the presence of moving obstacles, and also atypical obstacle motion. For guaranteeing freedom from collision, even promising path planning approaches still have some need for improvement.

Schmidt and Berns [1] presented an approach in which an extended A* path planning algorithm is applied to a 2D gridmap with growing regions for obstacles as a minimum safe distance. This approach does only consider static obstacles.

A method named Minimal Risk Motion Planning is presented in [2]. A route cost function minimizes possible failures to reach a goal state. By applying the wavefront algorithm [9], a cost-to-go function within the sensor field of view is calculated. Future collision risks of path states are not considered.

There are also approaches considering the possibilities of a future collision with static and dynamic obstacles. In [5] Petti and Fraichard proposed the Rapidly-Exploring Random Tree (RRT [13])-based motion planning scheme Partial Motion Planning (PMP). The Inevitable Collision State (ICS) approach is applied to check if it is possible in a state to avoid a collision with an obstacle (otherwise the state is an ICS), and to guarantee safety by checking a state whether at least one control command exists which is not leading to a collision state. This approach promises well but atypical obstacle motions are not considered.

Bouraine et al. [6] presented a passive motion safety approach that guarantees that, if a collision takes place, the robot will be at rest. The future obstacle motion is modeled by reachable sets [9] occupying with growing time the whole workspace. Therefore, a Braking ICS-Checking algorithm is integrated in a navigation scheme called PASSAVOID to check whether for all future braking trajectories of a state a collision occurs before the robot is at rest. It is not the task of PASSAVOID to drive the robot to a given goal and in some cases collisions occur that could have been avoided.

Likewise, methods to predict the motion of moving obstacles already exist. Vasquez and Fraichard [3] presented a technique to estimate the motion of a moving object in a structured environment in the long term by determining typical motion patterns of obstacles to predict their future motion. The presented approach is not able to predict atypical trajectories and for a short observation time frame the predicted motion tremendously differs from an obstacle's real motion.

Kushleyev and Likhachev proposed in [4] a representation of dynamic obstacles that models their predicted trajectories and considers the prediction uncertainty with error ellipses. It is assumed that the obstacle maintains constant controls at all future times. Atypical motion of obstacles is not regarded and if dynamic obstacles remain stationary it is possible that the robot gets stuck.

In this paper, we present a local path planning approach for mobile robots in unknown dynamic environments that considers (i) the robot's dynamic properties, (ii) motion probability of obstacles and (iii) future risks. The consideration of the robot's kinematics is important for generating only traceable paths and for a better evaluation of critical situations. This is because planning safe paths that a real robot cannot track due to a high velocity or constrained rotating movement can lead to collisions. We involve motion probabilities of obstacles by motion probability sections to consider realistic and, to our knowledge in contrast to other related works, also atypical obstacle motions with variable velocities in order to plan collision-free paths. Future risks of a chosen waypoint of a path will be estimated to not lead a robot into a situation where a collision is inevitable. Therefore, we apply ICS but we simplify the ICS checks similar to the PMP method for requiring less computation time. For our ICS checks we allow very low collision risks

[1]Chair of Software Engineering, Faculty of Computer Science, Otto-von-Guericke-University, Germany, Tanja.Hebecker@ovgu.de

[2]Chair of Software Engineering, Faculty of Computer Science, Otto-von-Guericke-University, Germany, Frank.Ortmeier@ovgu.de

of the future states to not make path planning impossible because of too strict constraints.

Currently, this path planner is designed for holonomic and non-holonomic ground robots and, therefore, first, it is only required to involve the two-dimensional case. An environmental model representing a mobile robot within a room with static and moving obstacles is implemented in OpenRAVE.

We assume that the parameters and kinematics of detected obstacles are known to our algorithm. Currently, we start from the premise that obstacles have the same kinematics as our robot. Another assumptions are that the obstacles' unseen part is not bigger than the part that is detected by the sensor and that their motions are independent from other obstacles aside from collision avoidance.

II. METHOD DESCRIPTION

In this section, we present the concept of our path planning approach which is summarized in pseudocode 1. First, cells in a 2D grid map are generated following the strategy in Section II-A. Then, motion probability sections are determined in Section II-B to predict the future position of detected moving obstacles. The modified wavefront algorithm [12] assigns cost values to the grid map cells after a collision-free local goal is determined within the sensor field of view (see Section II-A). Then, a path is determined by determining a cell sequence. The neighboring cells of the lastly to the cell sequence added cell are checked if they have a lower cost value. This process continues as long as the path has not reached the local goal. If the condition is true the cell centers of these cells are checked if they do not lead to an inevitable collision (see Section II-C.3), keep a safety distance to obstacles and the sensor range (see Section II-C.1), and are reachable (see Section II-C.2). In this context, we define reachability as the existence of a control command such that the robot can reach this waypoint safely and without unneeded detours. If these criteria are fulfilled the considered cell is added to the path cell sequence. The path is generated by the cell centers from the cell sequence.

Algorithm 1: Generating a collision-free path

Input: sensor range r_{sensor}, initial state of robot s_{init}
Output: Path P

while *global path not safe* **do**
  $Cells \leftarrow CellGeneration(r_{sensor})$;
  $RiskArea \leftarrow MotionProbabilitySections(r_{sensor})$;
  $LocalGoal \leftarrow AssignGoal(RiskArea, Cells)$;
  $Costs \leftarrow ModifiedWavefront(Cells, LocalGoal)$;
  $P \leftarrow FindingPath(Costs, Cells, RiskArea, s_{init})$;
end
return *P*;

A. Generating Potential Waypoints

The first step is to apply an algorithm that provides potential waypoints within the field of view. Many path planning approaches can be applied, e.g., Virtual Force Field by [10] or the enhanced Vector Field Histogram method VFH* by [11]. We propose to use the modified wavefront algorithm presented in [12]. It is an extension of the wavefront algorithm that we previously applied for a UAV online planner in 3D [8]. The modified wavefront algorithm is a real-time capable approach that is applicable in 2D and 3D space and capable of finding shortest paths. It guarantees the achieving of the goal if it is possible to reach, i.e., it does not lead to a local minimum. This method requires a grid map representation of the workspace. Because the applied sensor has a circular field of view we construct the grid map with polar coordinates.

An example of a grid map representation in a polar coordinate system is depicted in Fig. 1. The radii get larger by a constant value and the straight lines along the polar angles have a constant distance to each other. This way cells are generated that are smaller at short distance to the robot and larger with growing distance. A local goal is a cell center which is determined after checking all cell centers for the following conditions: (i) the distance to the global goal is as short as possible, (ii) the distance to static obstacles and to the sensor range limit is not too short and (iii) there is no risk that an obstacle occupies this cell center within the required time frame (see Section II-C.1).

The modified wavefront algorithm labels the cells of the work space with cost values based on their distance to the local goal. First, all these cells in the work space get the cost value zero, and then, the cell containing the local goal receives the cost value one. Orthogonal neighbor cells of the goal get the cost value of the goal plus 3 and diagonal neighbor cells get the cost value of the goal plus 4. This process continues for their neighbors with the cost value of the currently considered cell (see Fig. 1).

Cells that are occupied by obstacles or due to obstacles beyond the sensor's view get an invalid cost value. When assigning cost values we consider only the sensor detected position of dynamic obstacles and not their motion. We include their possible future positions in the determination of motion probability sections that are considered when choosing a waypoint later. The algorithm stops when no cell has the cost value zero anymore.

Beginning from a starting cell, centers from neighboring cells with the lowest (and positive) value are potential waypoints. Before selecting one of these potential waypoints we have to ensure that it does not lead into a collision.

B. Probability Sections

We consider only the last detected position of moving obstacles for the application of the modified wavefront algorithm, hence, we have to include their motion when choosing waypoints for a path. We represent the obstacles' motion with motion probability sections. The possible future position of obstacles is described by an approximated reachable set around the detected part of the moving obstacles. This approximated reachable set is partitioned into areas containing the probabilities of keeping or changing a direction.

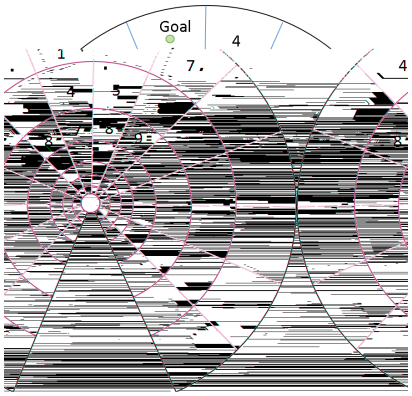


Fig. 1: Generating cells and applying the modified wavefront algorithm within the robot's sensor range

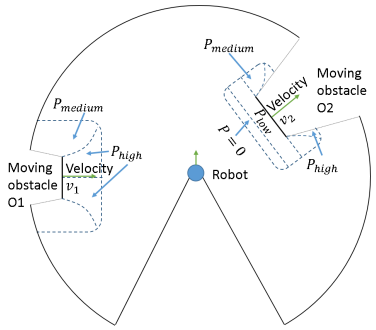


Fig. 2: Probability sections for a dynamic obstacle within the laser range

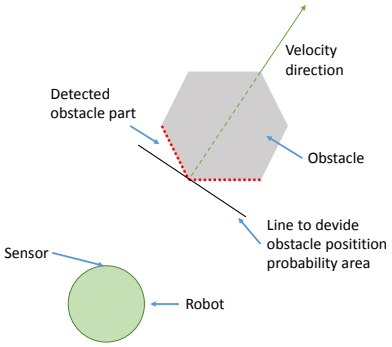


Fig. 3: Determination of a straight line that divides the lowest probability section from the whole future position area

These probability sections are approximations to enable fast calculations.

If an obstacle with a certain velocity direction moves freely in a room the probability for this obstacle to continue with the same velocity and in the same direction is higher than a turning or a braking maneuver (except, e.g., it has an obstacle in front of it). The probability that an obstacle brakes and moves with a sharp curve to the left or to the right is in this case lower, and the probability that an obstacle moves suddenly in the direction where it came from is the lowest one because it is completely irrational (but not impossible) except it has to fulfill a task that demands such a motion behavior.

We refer a probability value to a section giving the probability that in the next time steps the obstacle will be within this area. These probabilities are computed with the standard normal distribution for the motion angle. The peak is in the direction of the current motion.

Examples of probability sections are depicted in Fig. 2. The obstacle O1 of the robot is moving towards the robot and the obstacle O2 is about to move out of the field of view (the velocity arrows show the velocity direction). Both obstacles are unknown and, therefore, the robot has only the information about the obstacles that it gets from its sensor. Around the seen part of the obstacles an area is determined that this obstacle part is able to reach. Considering their current speed a distance is calculated that these obstacles can cover within a certain time frame which is explained in Section II-C.1. This distance determines the limit of the probabilistic future position area. This area for obstacle O1 is divided by maximum possible turning maneuvers to the left and to the right with a constant velocity and by the current motion direction. The reason for the section limitation by these turning maneuvers is that a change of the obstacle motion direction without braking is in unconstrained space almost as possible as continuing in the initial direction. If an obstacle has to avoid another object the probability for continuing in the same direction decreases. But if this object is not in the front but at a side of the obstacle it is still highly probable that the obstacle continues without reducing the velocity. Therefore, the front section is separated by the current direction line into two sections.

In Fig. 2 the robot can see the back view of obstacle O2 which is about to leave the field of view and, therefore, the part of the area around the obstacle that is behind the obstacle has a low occupancy probability. Hence, we divide the area around the detected obstacle part by a straight line going through the obstacle's point with the closest distance to the robot and being at an angle of 90° to the velocity direction. For a better understanding, the determination of the straight line is depicted in Fig. 3. To avoid unnecessary huge areas a part that definitely is not reachable within the given time frame gets the probability 0. This area part is calculated by assuming the obstacle to suddenly move directly in the opposite direction. Within the given time frame it is not possible that the obstacle reaches the area boundary due to its initial velocity. This negligible area part starts at the maximum reachable distance in the reversed direction of the initialized obstacle motion. As in the case of obstacle O1 the section with the highest probability is determined by calculating maximum possible turning maneuvers with a constant velocity.

The probability values are influenced by the presence of other obstacles. If obstacle O2 or its approximated reachable

set intersects the probability section of obstacle O1 the probability value for the concerned section of O1 is calculated with the formula for conditional probabilities.

The motion probability sections are involved for selecting waypoints (see Section II-C.1). Parts of the environment that cannot be perceived by the sensors because they are hidden by obstacles are considered as potential obstacles.

C. Selection of Waypoints

Beside the criterion that a waypoint must have a lower cost value than its predecessor other conditions have to be fulfilled to be chosen as a point of a path. The waypoint has to be collision-free, reachable and with low future collision risks.

1) Safe Distance to Obstacle Regions: A waypoint must not be occupied by an obstacle when the robot reaches it. In our specification, this means that a waypoint is not allowed to be located within an obstacle position probability section for a time frame the robot would need to arrive there. The only exception is the unreachable section with probability equal to zero because there is no collision risk.

The extents of the probability sections depend (beside the obstacle's current velocity and kinematics) on the distance of a waypoint which is checked for freedom from collision to the robot's initial position. That is because we choose the time frame t_{frame} for the calculation of the motion probability area by this distance and a certain velocity. We determine this velocity as the lower bound of the robot's velocity for reachability (see Section II-C.2) v_{bound} because t_{frame} is an assumed time frame the robot needs to get to the considered waypoint in the exclusion of unexpected detours.

Additionally, a safety distance to the obstacle motion area has to be included because if an obstacle really reaches the limit of this area it must not be too close to the robot. This safety distance is added to the outer limits of the obstacle sections and it depends on whether the obstacle is in motion direction of the robot or not for not rendering planning in narrow areas unnecessarily impossible. For obstacles in a planned robot motion direction we determine the safety distance as

$$d_{safe} = \begin{cases} term1 + \frac{l_{robot}}{2}, & \text{if } t_{break} > t_{frame} \\ term2 + \frac{l_{robot}}{2}, & \text{else} \end{cases}, \qquad (1)$$

where $term1 = \frac{a_{max}}{2} t_{frame}^2 + v_0 t_{frame}$ and $term2 = \frac{a_{max}}{2} t_{break}^2 + v_0 t_{break}$. The parameter t_{break} is the time the robot needs to come to a standstill, v_0 is the initial velocity and l_{robot} is the length of the robot. Equation 1 is based on uniform acceleration. The reason for adding $\frac{l_{robot}}{2}$ is that we assume the center of the robot to be located on the waypoint, then half of its extent exceeds this waypoint. We consider the maximum extent of the robot due to safety reasons and, therefore, we include half of the robot's length in the safety distance.

For obstacles that are not located in the planned motion direction of the robot the probability sections are additionally only extended with $d_{safe} = \frac{l_{robot}}{2}$. The occupancy probability of a waypoint is determined for the time when the robot is able to reach the waypoint.

The safety distance d_{safe} holds also for static obstacles. Due to possible obstacles beyond the field of view also a safety distance d_{safe} to the outer sensor range has to be kept. A waypoint must not be located within an obstacle section with a probability higher than zero and has to keep a safety distance d_{safe} to static obstacles and the full sensor range. If these conditions are complied the waypoint is checked for fulfilling the following criteria, if not another waypoint is analyzed.

Even if a waypoint is collision-free, two other criteria have to be fulfilled before it becomes a part of the path: (i) the waypoint has to be reachable by the robot without unnecessary detours due to motion safety and without braking to a too slow velocity, and (ii) the robot's state at the waypoint must not be an ICS.

2) Reachability: To determine if a waypoint is reachable without the need that the robot comes below a certain speed, we determine an approximated reachable set that was already presented in our previous work [8]. The limits of this reachable set are calculated by computing trajectories representing maximum possible turning maneuvers to the left and to the right and the maximum distance d_{reach} that the robot with its initial velocity can cover.

We consider the motion direction that the robot would have if it moves from the waypoint that was lastly added to the path to the potential new waypoint. Turning maneuvers are determined by setting the initial speed into this motion direction starting from the potential waypoint. We do not want to plan a path that is only traceable when the robot stops in front of obstacles or to render path planning impossible due to too high velocities. Therefore, the trajectories concern the case that the robot brakes with $a = k_{reach} a_{max}$ not below $v_{bound} = k_{reach} v_{max}$. The constant value k_{reach} is defined in the interval [0,1] and in our case $k_{reach} = 0.5$. We chose this value because we assume it to represent a lower bound to ensure a smooth motion. Only cell centers between these two trajectories and within the distance d_{reach} to the previous waypoint are reachable.

3) Inevitable Collision States: According to [7] an ICS is a state in which it is not possible to avoid a collision with an obstacle, no matter what the control input of a robot is. ICS checks are important because a waypoint must not be added to a path if it is impossible to avoid a collision after reaching it. Precise ICS checks require a lot of computational time, hence, we apply a simple way of determining ICS. Similar to [5], we check if a predicted state at a waypoint is an ICS by computing three trajectories with the help of the robot's dynamic model for a certain time frame. One trajectory is for the straight forward motion and two trajectories are the ones already calculated for the reachability check with the difference that the trajectories are now determined for the case of full braking with $a = a_{max}$. These trajectories are depicted in Fig. 4. Sampled points of these trajectories are checked for collision risks as the waypoint itself with one difference. If the collision risk for these trajectories is higher than a probability border, which we for the moment set to 5% (because the area behind an obstacle that is not

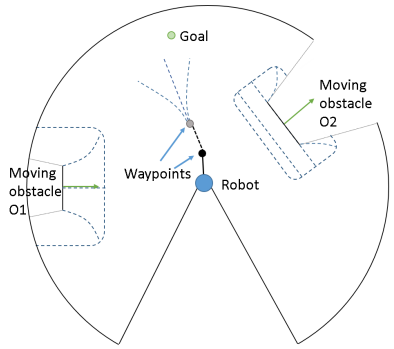


Fig. 4: ICS check of a potential waypoint

influenced by intersections of other obstacles never exceeds this probability value), than the considered waypoint is not safe enough and another waypoint has to be found.

III. EVALUATION

In this section, the evaluation results of our path planning approach are presented and compared to the PMP algorithm. Both algorithms were implemented in OpenRAVE[1].

A. Kinematics and Parameters

For testing the two approaches in simulations, we assume as an example that the robot as well as the obstacles move according to the following simple model:

$$\dot{\mathbf{x}} = \begin{pmatrix} v & cos(\gamma) \\ v & sin(\gamma) \\ & a \\ & \dot{\gamma}_{max} \end{pmatrix} \tag{2}$$

with

$$\mathbf{x} = (x_1, x_2, x_3, x_4)^T ,$$

x_1 and x_2 as positions in x- and y-direction, x_3 as speed and x_4 as the yaw angle. The state space model contains the velocity v, the acceleration a and the yaw angle γ as well as the maximum angular velocity $\dot{\gamma}_{max}$.

In our simulations, we represent obstacles and the robot with our local path planner as KUKA youBots[2]. For the evaluated scenarios, the motion of these robots is constrained according to the mathematical model 2. The KUKA youBot omni-directional mobile platform has the following parameters: a length of 0.580m, a width of 0.380m and a maximum speed of $v_{max} = 0.8$m/s. Additionally, we set for our simulations the maximum acceleration to a constant value of 0.2m/s^2.

The applied sensor is a 2D laser scanner with the following chosen specifications: a scan time of 0.1s, a sensor range of 5m, an angle interval [-135°,135°] and a resolution of 0.35°.

[1]OpenRAVE: see http://www.openrave.org, accessed on August 4, 2015

[2]KUKA youBot: see http://www.youbot-store.com, accessed on August 4, 2015

B. Simulation Results

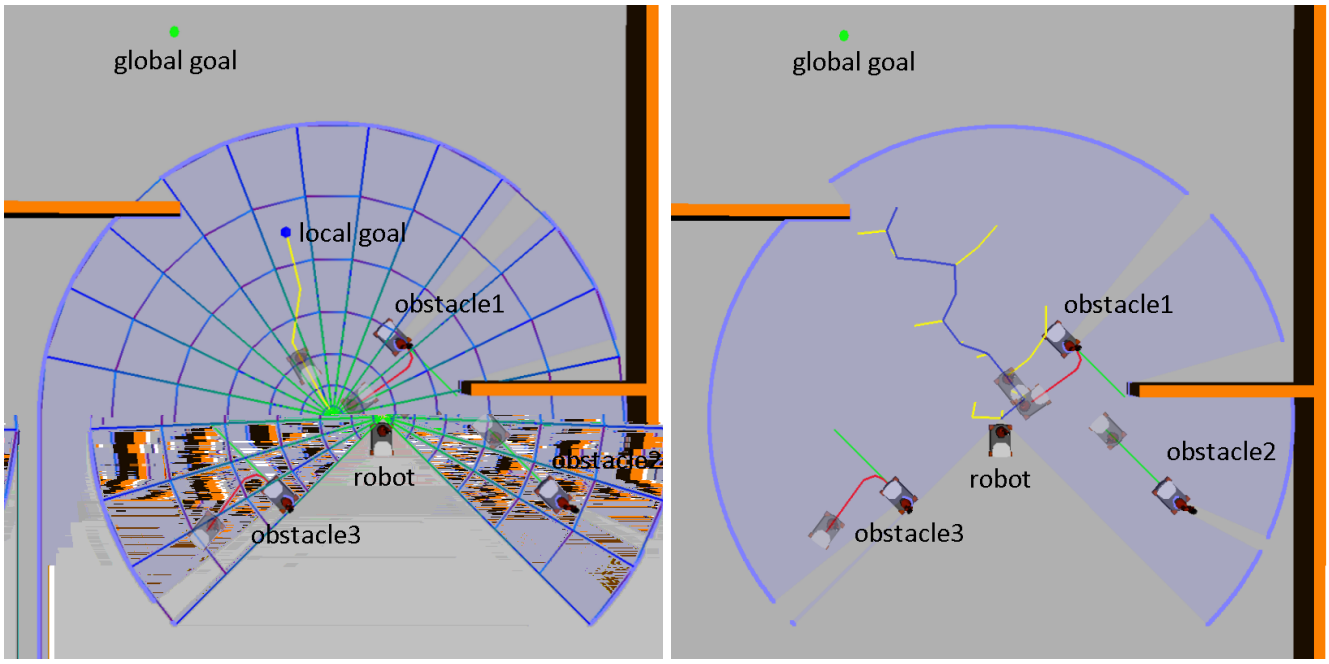
In this section we present evaluation results for our probability section algorithm. We compare the proposed algorithm with the PMP method on scenarios where obstacles move atypically. Path smoothing was for the moment neglected in the simulations. Our environment consists of an 8 8 meters plane limited with walls from every side. The obstacles are other youbots. The environment is depicted in Fig. 5.

In the scenario of Fig. 5 three obstacles are moving with a current speed $v_{obstacle} = 0.75 \ v_{max}$ and the robot has a starting velocity $v_{robot} = 0.5 \ v_{max}$. The green lines starting at each obstacle indicate the assumed future motion direction due to the observed previous movement. The red curves show the real future motion of two obstacles (the third one $obstacle2$ moves in the assumed direction). Fig. 5(a) depicts a path planning result of our algorithm. A grid map overlays the geometry of the maximum possible sensor field of view. Due to the inclusion of unexpected obstacle motion, the planned path, which is represented by a yellow line, avoids $obstacle1$ though with its current position and velocity direction there is no collision risk. The local goal is not within one of the furthest cells in consequence of the consideration of the safety distance to the field of view's limit.

Fig. 5(b) shows the path planned by the PMP algorithm with a blue line. Negligible branches, which are generated in a RRT search towards the global goal, are represented with yellow arms at the path. The chosen path is collision free considering only the current obstacle states but with the actual future motion of $obstacle1$ the robot collides with the obstacle when reaching the critical waypoint. The reason of this is the different inclusion of the obstacle motion because PMP used the predicted trajectories approach presented in [3] that does not consider atypical obstacle motion. In contrast to our algorithm, the missing consideration of the robot's dynamics effects an avoidable need for velocity reduction. The unnecessary detours are caused by applying the RRT with 50 nodes.

In other test cases with more than one obstacle located on the upper part of the grid map, and with obstacles moving directly between the robot and the global goal our algorithm generates a local goal that is closer to the robot than to the global goal. The reason behind this is if the local goal would be further afar from the robot and closer to the global goal the time frame for the probability sections would increase and thus also the extents of these sections. Hence, obstacles in the upper field of view part which are moving in path direction can occupy a local goal with a longer distance to the robot before our robot is able to reach it. Therefore, only short collision-free paths are calculated until the robot escapes the critical situation.

In the tested scenarios, our algorithm always found a safe path to a local goal as long as it was possible to reach the local goal. The adaptation of the local goal to environmental conditions contributes to the planning of a complete path. The only test cases when the path planner fails to reach the



(a) (b)

Fig. 5: We compared our approach with the PMP algorithm in the same scenarios. (a) shows the in this paper presented algorithm and (b) the PMP method.

local goal were when the robot was already at initialization time in an unsafe situation with obstacles moving all around and too close to the robot and blocking all possible ways to the local goal.

IV. CONCLUSIONS

In this paper, we presented a local path planning method for mobile robots that chooses a local goal within the field of view according to the environmental situation, considers future risks of a potential waypoint by applying ICS checks and includes possible irrational motion of obstacles by motion probability sections. We tested the algorithm in the OpenRAVE simulation environment with moving and static obstacles and compared it to the PMP method. In simulations, our algorithm always found a safe path to a local goal. In future works, we have to include the following issues in our algorithm:

- In the case that no safe waypoint can be added to a path a state will be determined where the collision risk is as minimal as possible and where the robot will get the fewest harm in case of a collision (e.g., the robot turns in a position that a collision has the fewest consequences).
- Another future work direction is the extension to 3D cases, e.g., to be capable to plan flight paths for unmanned aerial vehicles (UAVs).

REFERENCES

[1] D. Schmidt, K. Berns, "Construction site navigation for the autonomous excavator Thor", in 6th International Conference on Automation, Robotics and Applications (ICARA), IEEE, 2015, pp. 90–97.

[2] C. Goerzen, M. Whalley, "Minimal risk motion planning: a new planner for autonomous UAVs", in AHS International Specialists Meeting on Unmanned Rotorcraft, 2011.

[3] D. Vasquez, T. Fraichard, "Motion Prediction for Moving Objects: a Statistical Approach", in IEEE International Conference on Robotics and Automation, vol. 4, IEEE, 2004, pp. 3931 – 3936.

[4] A. Kushleyev, M. Likhachev, "Time-bounded Lattice for Efficient Planning in Dynamic Environments", in IEEE International Conference on Robotics and Automation, IEEE, 2009, pp. 1662 – 1668.

[5] S. Petti, T. Fraichard, "Safe Motion Planning in Dynamic Environments", in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 2210 – 2215.

[6] S. Bouraine, T. Fraichard, H. Salhi, "Provably Safe Navigation for Mobile Robots with Limited Field-of-Views in Dynamic Environments", in International Conference on Robotics and Automation, IEEE, 2012, pp. 174 – 179.

[7] T. Fraichard, H. Asama, "Inevitable collision states. A step towards safer robots?", in IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, IEEE, 2003, pp. 388 – 393.

[8] T. Hebecker, R. Buchholz, F. Ortmeier, "Model-Based Local Path Planning for UAVs", in Journal of Intelligent and Robotic Systems, Springer, 2014

[9] S. M. Lavalle, "Planning Algorithms", Cambridge University Press, 2006

[10] J. Ni, W. Wu, J. Shen, X. Fan, "An Improved VFF Approach for Robot Path Planning in Unknown and Dynamic Environments", in Mathematical Problems in Engineering, 2014

[11] I. Ulrich, J. Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification", in IEEE International Conference on Robotics and Automation, vol. 3, IEEE, 2000, pp. 2505 – 2511

[12] J. S. Oh, Y. H. Choi, J. B. Park, Y.F. Zheng, "Complete Coverage Navigation of Cleaning Robots using Triangular Cell Based Map", in IEEE Transactions on Industrial Electronics, vol. 51, IEEE, 2004, pp. 718 - 726

[13] S. LaValle, J. Kuffner, "Randomized kinodynamic planning", in International Conference on Robotics and Automation, MAy 1999, pp. 473-479