IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, October 2022.

Predictive Angular Potential Field-based Obstacle Avoidance for Dynamic UAV Flights

Daniel Schleich and Sven Behnke

***Abstract*— In recent years, unmanned aerial vehicles (UAVs) are used for numerous inspection and video capture tasks. Manually controlling UAVs in the vicinity of obstacles is challenging, however, and poses a high risk of collisions. Even for autonomous flight, global navigation planning might be too slow to react to newly perceived obstacles. Disturbances such as wind might lead to deviations from the planned trajectories.**

In this work, we present a fast predictive obstacle avoidance method that does not depend on higher-level localization or mapping and maintains the dynamic flight capabilities of UAVs. It directly operates on LiDAR range images in real time and adjusts the current flight direction by computing angular potential fields within the range image. The velocity magnitude is subsequently determined based on a trajectory prediction and time-to-contact estimation.

Our method is evaluated using Hardware-in-the-Loop simulations. It keeps the UAV at a safe distance to obstacles, while allowing higher flight velocities than previous reactive obstacle avoidance methods that directly operate on sensor data.

I. Introduction

Unmanned aerial vehicles (UAVs) are increasingly applied to many different tasks requiring observation of objects or environments that are difficult to access. This includes fields like industrial inspection, agriculture, and search and rescue. Their ability for agile flight and high velocities makes UAVs especially suitable for time-critical applications like target tracking and disaster response. Such scenarios often require flights close to obstacles or even indoors, thus posing significant strain on human pilots during manual operation. The application of autonomous flight systems is also challenging, however, since one has to deal with large, initially unknown environments. In our previous work [1], we presented a hierarchical navigation and control pipeline for autonomous flights in GNSS-denied environments. We now propose an additional reactive obstacle avoidance layer (Fig. 2) that can be added to our system to increase flight safety in the vicinity of obstacles.

Obstacle avoidance methods must be able to react quickly to unknown obstacles without depending on other modules like localization or mapping, thus ensuring safe flights even if external disturbances or errors in higher-level modules occur. Our method can be directly applied to LiDAR range images without requiring a local map. However, we still aggregate LiDAR scans over a short time interval into a history range image. This is necessary to be able to avoid small structures like cables, which might not be measured in every scan. Since we only aggregate over a short time interval, the transformation between different LiDAR range images can be approximated by integrating velocities. Thus, we do not depend on global position estimates but only assume that an estimate of the current UAV velocity is available, which is also needed for the low-level controller to track the velocity commands generated by our method.

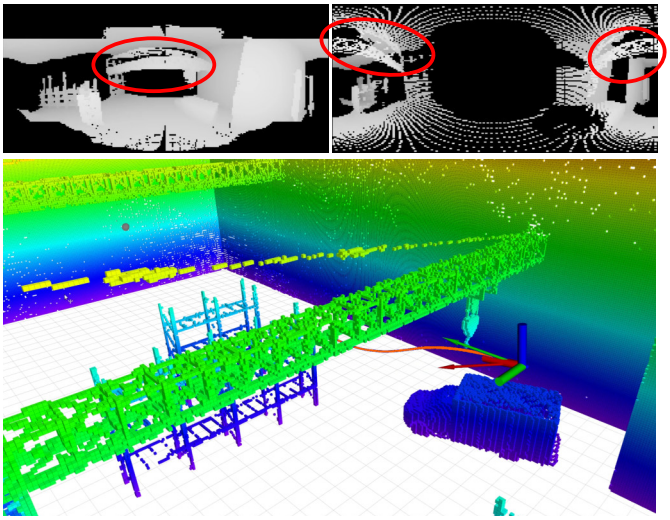

Fig. 1: Our predictive obstacle avoidance method successfully avoids a collision. The UAV (coordinate axes) receives a velocity command (green arrow) that would result in a collision with an obstacle. Our method adjusts the flight direction using angular potential fields, which are computed on aggregated LiDAR range images (top left). Note that distant obstacles are pruned since they do not pose a risk of collision. The velocity magnitude of the final command (red arrow) is chosen based on a time-to-contact estimate along the future trajectory (orange), which is generated by iteratively applying potential fields to range images transformed into the future sensor frames. The predicted range image for the end of the unrolled trajectory is shown on the top right. Note that the obstacle (circled red) is now behind the UAV.

A key advantage for the application of UAVs is their ability for high flight velocities and thus short mission execution times. Many obstacle avoidance methods unnecessarily limit the flight velocity, though. For example, the repulsive forces of classic potential field methods do not only adjust the flight direction but also decelerate the UAV, although obstacles might be passed at a safe distance. To address this issue, we propose a novel method that adjusts the flight direction based on an angular potential field directly defined on LiDAR range images. To choose an appropriate velocity magnitude, the future trajectory is subsequently predicted to estimate the time-to-contact and the velocity command is scaled accordingly.

All authors are with the Autonomous Intelligent Systems group, University of Bonn, Germany; schleich@ais.uni-bonn.de

This work has been funded by the German Federal Ministry of Education and Research (BMBF) in the project “Kompetenzzentrum: Aufbau des Deutschen Rettungsrobotik-Zentrums (A-DRZ)”.

In summary, our proposed method includes:

- the aggregation of LiDAR range images over a short time horizon into a history range image, which does not depend on global localization or mapping modules,
- an angular potential field method with dynamic consideration that is applied to range images to determine the flight direction, and
- trajectory prediction with time-to-contact estimation to scale the velocity magnitude.

II. Related Work

Due to their long planning times, global trajectory planning methods cannot be executed at the high frequencies needed to avoid dynamic or previously unknown obstacles during fast UAV flight. Thus, many approaches to low-level obstacle avoidance locally adjust a global trajectory to newly perceived obstacles. For example, Oleynikova et al. [2] fit continuous-time polynomials to a global geometric path, which are locally optimized with respect to obstacles and control costs. A similar method was proposed by Usenko et al. [3] using B-Splines instead.

Zhang et al. [4] use a hierarchy of multiple precomputed offline trajectories to quickly react to previously unknown obstacles in cluttered environments. If a collision for the currently tracked path is detected, they efficiently switch to an alternative trajectory that locally avoids obstacles. This method significantly reduces onboard computations but depends on a prior map of the environment. Thus, it is extended in [5] to model parts of the environment probabilistically if they are not currently covered by onboard sensors. Instead of searching for the shortest path, the trajectory with highest probability of reaching the goal is executed. For the case where no prior map is known, a heuristic is introduced to estimate this probability.

All of the above methods aggregate an environment map and thus depend on global position estimates. To ensure collision-free flights even if errors in higher-level localization modules occur, obstacle avoidance methods should directly operate on sensor data. For example, Beul and Behnke [6] generate time-optimal trajectories which are checked for collisions against LiDAR point clouds. If necessary, additional trajectories to alternative waypoints are computed until a collision-free trajectory is found. However, the target points do not depend on the environment structure but are sampled around the initial trajectory and the current UAV position. Thus, the performance depends on the parametrization of the sampling process and it is prone to local minima in constrained environments.

A common method for low-level obstacle avoidance is the Dynamic Window Approach (DWA) introduced by Fox et al. [7]. A set of control inputs is sampled and each is evaluated by predicting the corresponding future trajectory. The best control input is chosen based on obstacle clearance, progress towards the goal, and velocity. Multiple extensions to DWA have been introduced. For example, Missura and Bennewitz [8] propose a dynamic collision model that predicts the motion of obstacles and thus can handle non-static environments. Dobrevski et al. [9] adjust DWA parameters dynamically based on the current environment perception using a neural network and reinforcement learning. Due to sampling the control inputs and a limited planning horizon, DWA-based methods are prone to local minima. To increase the look-ahead, Missura et al. [10] propose short-term aborting A* searches, but they again depend on global localization and environment mapping.

Instead of searching for a collision-free trajectory, artificial potential field-based methods [11] are commonly used to instantaneously adjust movement commands to the most recent sensor measurements. Nieuwenhuisen et al. [12] combine potential field-based obstacle avoidance with a learned motion model to predict the future trajectory and reduce the current velocity if necessary. In [13], the requirement to learn a motion model is removed by introducing two different influence spheres around the robot: A larger, passive avoidance sphere, where the UAV is decelerated in the obstacle direction, and an inner sphere, where the UAV is actively pushed away from the obstacle. Thus, smoother trajectories through narrow corridors are achieved. However, classic potential field-based methods do not scale well to high velocities and are prone to local minima.

The idea to choose the movement direction based on range images has been proposed before, e.g., by Sezer et al. [14], who steer a vehicle towards the obstacle gap with largest angle in the current scan. Houshyari and Sezer [15] extend this approach to use the Euclidean distance instead of angles to measure the gap width. Cho et al. [16] define a Gaussian potential field on range images. In contrast to our approach, these methods have only been applied to 2D planning for ground vehicles and they do not consider the vehicle's dynamics.

We propose a reactive obstacle-avoidance method that directly operates on 3D LiDAR range images and does not depend on global localization or mapping. We use velocity-dependent angular potential fields to determine the flight direction and scale the velocity magnitude by predicting the future trajectory and a time-to-contact estimation.

III. Method

In the following sections, we give a detailed description of the different components of our obstacle avoidance approach depicted in Fig. 2. First, the current 3D LiDAR scan is preprocessed (Sec. III-A) by reducing the resolution and removing distant points that do not pose a risk of a collision in the near future. Additionally, we aggregate the current range image with the previous ones over a short time horizon to be able to avoid small obstacles that are not measured in every scan.

In order to keep safe distances to obstacles while still allowing fast flights on collision-free trajectories in narrow passages, we represent velocity commands in spherical coordinates and split their generation into two parts: In Sec. III-B, we determine the flight direction, i.e., the angular components, by applying a potential field method to the sensor range image. Subsequently, we predict the future trajectory

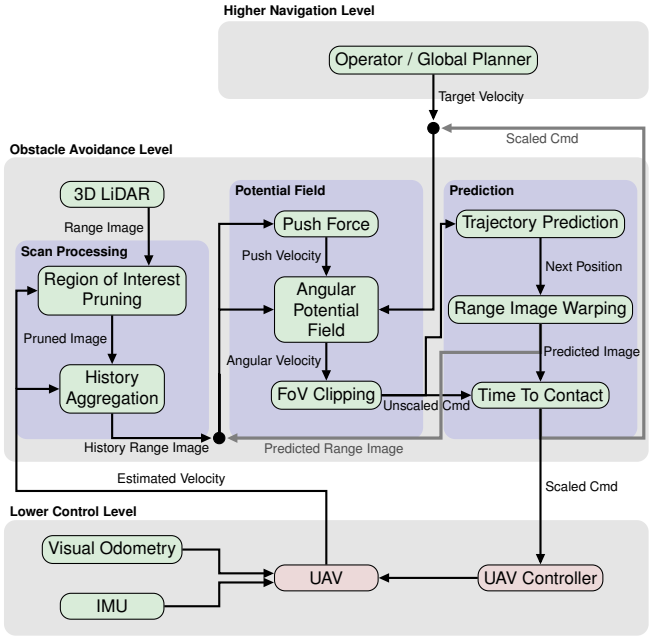

Fig. 2: System overview. The computation blocks *Potential Field* and *Prediction* are executed iteratively to unroll the future trajectory. Black dots mark the entry points to these iterations. Starting from the second iteration, the results of the previous iteration (gray arrows) are forwarded instead of the initial input.

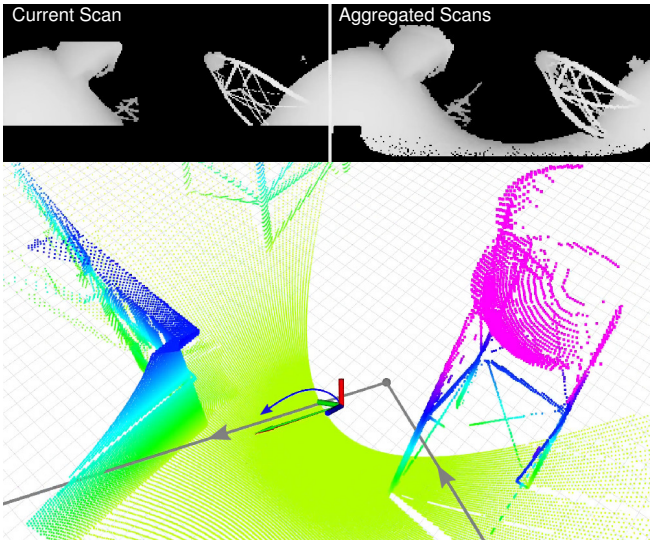

Fig. 3: Comparison of a single range image (top left) against the aggregated range image (top right) during an aggressive change in flight direction. The corresponding 3D view of the environment is shown at the bottom including the UAV (coordinate axes), the target trajectory (gray), the target and adjusted velocity commands (green and red arrows) as well as the predicted trajectory (blue). Due to the UAV orientation, the ground in front of the UAV is not visible in the current scan. Aggregating range images over a short time interval efficiently extends the vertical field of view. As a result, the ground is still considered during obstacle avoidance.

(Sec. III-C) by iteratively applying our potential field method and unrolling the resulting velocity commands (scaled with the commanded target velocity). Thus, we can detect future collisions and scale the actually executed velocity command such that the UAV will not enter the safety region around obstacles within the minimal time-to-contact t_{contact}.

A. LiDAR Scan History Aggregation

Our method is able to directly operate on LiDAR range images. Only considering the most recent scan is dangerous, however, since thin structures might not be represented in every measurement. Furthermore, during dynamic flight, obstacles might be moved out of the sensor's limited vertical field of view. This is illustrated in Fig. 3. During frequent aggressive changes of flight direction, a velocity command might be chosen that steers the UAV closer to currently not visible obstacles, even when restring commands to the sensor's field of view. In combination with control latencies, this might result in the UAV entering the safety region around obstacles or even in a collision. To address this issue, we aggregate measurements over a short time interval $t_{\text{history}} = 1\,\text{s}$ into a history range image $\mathcal{H}$. For each pixel (ϕ, θ), we additionally keep track of the age $t(\phi, \theta)$ of the corresponding measurement.

For lower computation times, we discard all pixels of the current image with range values that are too high to pose a risk of collision in the near future. To this end, we predict the area A_{pos} of possible future UAV positions by applying a maximum acceleration command $\pm a_{\max}$ to the current UAV velocity v_0 for a time interval $t = t_{\text{history}} + t_{\text{contact}}$. Here, t_{contact} denotes the minimum allowed time-to-contact, which equals the prediction horizon used in Sec. III-C. All pixels corresponding to 3D points whose distance to A_{pos} exceed a safety margin d_{safe} can be skipped during the processing described below.

Every time a new 3D scan is obtained, we transform the history range image $\mathcal{H}_{i-1}$ of the previous iteration into the new sensor frame. First, we project every pixel $p_{\text{Image}} = (\phi, \theta)$ into 3D space. For simplicity of notation, we state the transformations for continuous angles and discard the discretization into image coordinates:

$$\mathcal{T}_{\text{Image}\mapsto\text{3D}}(p_{\text{Image}}) = \frac{r(\phi,\theta)}{\|\overline{p}_{\text{3D}}\|}\overline{p}_{\text{3D}}, \quad \text{with } \overline{p}_{\text{3D}} = \begin{pmatrix} \cos(\theta)\cos(\phi) \\ \cos(\theta)\sin(\phi) \\ \sin(\theta) \end{pmatrix}. \tag{1}$$

Here, $r(\phi, \theta)$ denotes the range value associated with the pixel (ϕ, θ). The offset between the previous and current sensor frame can be estimated by integrating UAV velocity measurements. The projected point $p_{\text{3D}} = (x, y, z)$ is then transformed accordingly and reprojected into the new image frame using

$$\mathcal{T}_{\text{3D}\mapsto\text{Image}}(p_{\text{3D}}) = \begin{pmatrix} \operatorname{atan2}(y, x) \\ \frac{\pi}{2} - \operatorname{acos}(\frac{z}{\|p_{\text{3D}}\|_2}) \end{pmatrix}. \tag{2}$$

Subsequently, we merge the transformed history image $\widehat{\mathcal{H}}_{i-1}$ with the current scan $\mathcal{S}_i$. Range values and pixel ages of the history image should be updated if the corresponding obstacle is present in the current scan, although the new range value might be larger than the history value. Thus, instead of taking the pixel-wise minimum of $\widehat{\mathcal{H}}_{i-1}$ and

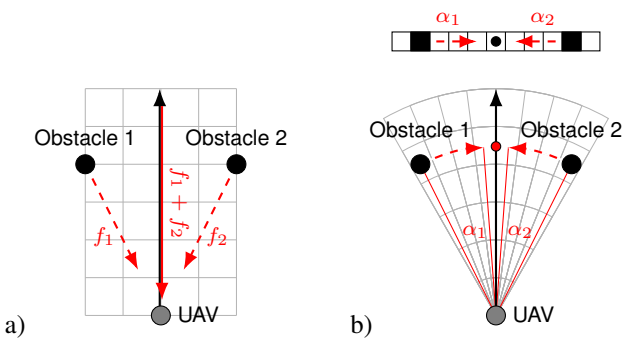

Fig. 4: Comparison of Cartesian and spherical repulsive forces. a) Top-down view in Cartesian coordinates. The sum of the repulsive forces $f_1 + f_2$ cancels out the commanded velocity (black arrow). b) Top: The angular repulsive forces α_1, α_2 are computed in image coordinates on the LiDAR range image. They cancel out each other and guide the robot towards the center of the gap between the obstacles. Bottom: Top-down view of the projection into spherical coordinates.

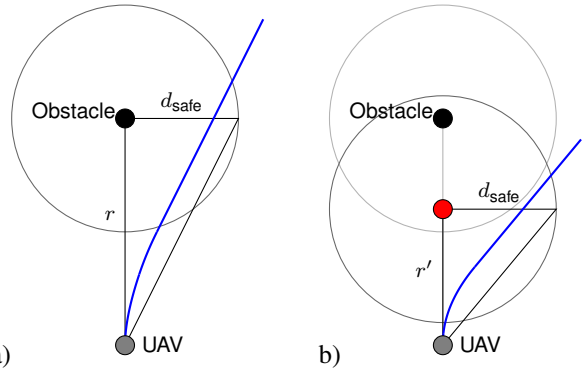

Fig. 5: Definition of the support radius for repulsive force computation. Depicted is the top-down view in Cartesian coordinates of a situation where the UAV has an initial velocity towards the obstacle. a) When defining the support (gray circle) using the Euclidean range value r, the actual UAV trajectory (blue) passes the obstacle at a too close distance since the UAV cannot instantaneously change its flight direction. b) Using the velocity-dependent predicted future range value $r_{\text{vel}} := r - t_{\text{contact}} v$ results in larger acceleration commands. Thus, the UAV does not enter the safety area around the obstacle.

$\mathcal{S}_i$, we keep the most recent range values $r_{\mathcal{S}_i}(\phi, \theta)$ even if they are larger than the history values $r_{\widehat{\mathcal{H}}_{i-1}}(\phi, \theta)$, as long as the difference is below a threshold T. To model the uncertainty of history values $r_{\widehat{\mathcal{H}}_{i-1}(\phi,\theta)}$, this threshold should be different for each pixel and should grow monotonically with increasing pixel age $t(\phi, \theta)$. Thus, we define the range values of the updated history image $\mathcal{H}_i$ as

$$r_{\mathcal{H}_i(\phi,\theta)} = \begin{cases} r_{\widehat{\mathcal{H}}_{i-1}}(\phi,\theta), & \text{if } r_{\widehat{\mathcal{H}}_{i-1}}(\phi,\theta) e^{\frac{t(\phi,\theta)}{\tau}} \leq r_{\mathcal{S}_i}(\phi,\theta) \\ r_{\mathcal{S}_i}(\phi,\theta), & \text{otherwise,} \end{cases} \tag{3}$$

where $\tau > 0$ controls the growth rate of the threshold $T = e^{\frac{t(\phi,\theta)}{\tau}}$.

B. Potential Field Method

The repulsive forces of classical potential field methods are defined in Cartesian coordinates and thus influence both, the UAV flight direction and its velocity. For example, consider the situation depicted in Fig. 4 a, where the UAV is commanded into a gap between two obstacles. The UAV is pushed away from each obstacle by the repulsive forces f_1 and f_2, respectively. Depending on the force magnitudes $\|f_1\|$ and $\|f_2\|$, the commanded velocity is reduced or even completely cancelled out, resulting in a local minimum. To address this issue, we propose to formulate the problem in spherical coordinates and apply the repulsive forces only to the angular components ϕ and θ (Fig. 4 b). Thus, a potential field is computed for the LiDAR range image. The corresponding angular repulsive forces α_1, α_2 steer the UAV towards the center of the gap between the obstacles without influencing the magnitude of the resulting velocity command, which will be determined in a subsequent step based on a time-to-contact estimation. Our experiments show that this method is less prone to local minima and allows fast flights through narrow corridors, while keeping sufficient distance to the walls.

1) Angular Potential Field Method: In a first step, we project the commanded target velocity v_{target} into the range image using (2), obtaining the target pixel $(\phi_{\text{target}}, \theta_{\text{target}})$. The angular components $(\phi_{\text{out}}, \theta_{\text{out}})$ of the adjusted velocity command are then computed by adding repulsive forces $\alpha_i = (\phi_i, \theta_i)$ induced by all other pixels.

For the repulsive force computation, we define the support of a pixel as the area where its repulsive forces are non-zero. Since the UAV should keep a safety distance d_{safe} to each obstacle, the support of a pixel $p_{\text{obs}} = (\phi_{\text{obs}}, \theta_{\text{obs}})$ can be chosen as the projection of a 3D sphere with radius d_{safe} and center $\mathcal{T}_{\text{Image}\mapsto\text{3D}}(p_{\text{Image}})$ into the range image. Depending on the range value r, the resulting support radius is given by $d_{\text{support}} = \text{atan2}(d_{\text{safe}}, r)$. The repulsive force induced by p_{obs} acting on a pixel $p' = (\phi', \theta')$ is then defined as

$$\alpha = \begin{cases} \frac{d_{\text{support}} - \|p' - p_{\text{obs}}\|}{\|p' - p_{\text{obs}}\|}(p' - p_{\text{obs}}), & \text{if } \|p' - p_{\text{obs}}\| \leq d_{\text{support}} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

This definition of repulsive forces adjusts the UAV flight direction such that it has a safety distance d_{safe} to the obstacle once the UAV reaches the obstacle depth (see Fig. 5 a). However, the direct line-of-sight to this position intersects with the safety area around the obstacle. Since the UAV cannot instantaneously change its flight direction—especially at high velocities—the actual future trajectory would pass the obstacle at an even smaller distance. To address this issue, we propose not to use the pixel range value r, i.e., its Euclidean distance to the UAV, for the support radius definition, but a different distance metric that incorporates the UAV velocity. Thus, we define $r_{\text{vel}} := r - d_{\text{contact}}$ as the predicted future range value after moving a distance d_{contact} towards the obstacle. This virtually moves the obstacle closer to the UAV, as depicted in Fig. 5 b. As a result, the change in flight direction will be larger for higher velocities, and a safe distance to the obstacle is maintained. The distance d_{contact} is defined as the predicted movement towards the obstacle p_{obs}. Thus, it depends on the current (scalar) velocity v into the obstacle direction, which is obtained using the vector

projection of the current velocity vector onto the direction towards p_{obs}. To increase the change in flight direction even if the current velocity is zero, we ensure a lower threshold $d_{\text{min\_contact}}$. Thus, we define

$$d_{\text{contact}} := \max\{t_{\text{contact}} v, d_{\text{min\_contact}}\}. \tag{5}$$

For lower computation times, we additionally discard obstacles that are still far away. The support radius of a pixel p_{obs} with range r is thus defined as

$$d_{\text{support}} = \begin{cases} 0, & \text{if } r_{\text{vel}} \geq d_{\text{safe}} \\ \operatorname{atan2}(d_{\text{safe}}, r_{\text{vel}}), & \text{if } r_{\text{vel}} > 0 \\ \frac{\pi}{2}, & \text{otherwise.} \end{cases} \tag{6}$$

When obstacles spread over multiple pixels, the corresponding repulsive forces accumulate and adjust the flight direction stronger than necessary. This can even lead to unstable flight trajectories with frequent large jumps in the direction command. To avoid unbounded accumulation while still allowing repulsive forces with different directions to cancel each other out, we keep track of the minimum and maximum forces $\theta_{\text{min}}, \theta_{\text{max}}, \phi_{\text{min}}, \phi_{\text{max}}$ along each dimension independently. The sum of repulsive forces is then clipped to these values. Thus, using the repulsive forces $\alpha_i = (\phi_i, \theta_i)$, the angular components $(\phi_{\text{out}}, \theta_{\text{out}})$ of the adjusted velocity command are computed as

$$\begin{pmatrix} \phi_{\text{out}} \\ \theta_{\text{out}} \end{pmatrix} = \begin{pmatrix} \phi_{\text{target}} \\ \theta_{\text{target}} \end{pmatrix} + \begin{pmatrix} \min\{\max\{\sum_i \phi_i, \phi_{\text{min}}\}, \phi_{\text{max}}\} \\ \min\{\max\{\sum_i \theta_i, \theta_{\text{min}}\}, \theta_{\text{max}}\} \end{pmatrix}. \tag{7}$$

To ensure that the UAV is only moving into a direction that is covered by the LiDAR, θ_{out} is subsequently clipped to the vertical field-of-view. Finally, the adjusted velocity command is transformed back into Cartesian coordinates, scaled to the target velocity magnitude.

2) Push Force: The angular potential field method changes the commanded flight angle at most by $90°$. If the UAV already is too close to an obstacle, adjustments of up to $180°$ are necessary to move the UAV away from the obstacle. Thus, such situations have to be addressed separately. Note that the angular potential field method was introduced to allow dynamic flights. If the UAV is within the safety area around obstacles, we do not allow high flight velocities anymore. Instead, the UAV should be pushed slowly away from the obstacle. Thus, we use a classical potential field method in Cartesian coordinates for this situation.

Every time a new range images is available, we first check it for obstacles closer to the UAV than d_{safe}. Each such obstacle induces a repulsive force with a magnitude that grows linearly with decreasing distance to the UAV. The total push force F_{push} is obtained as the sum of repulsive forces, scaled to the desired movement velocity.

If the UAV is closer to an obstacle than a threshold $d_{\text{close}} < d_{\text{safe}}$, the target command is completely discarded and the adjusted velocity command equals the push force. However, if the distance to the nearest obstacle exceeds d_{safe}, the adjusted velocity command is generated by the angular potential field method described above. To allow a smooth transition between those two cases, we combine both methods if the closest obstacle distance is between d_{close} and d_{safe}. Thus, we add the push force F_{push} to the target command v_{target}. To avoid speeding up the UAV if the directions of F_{push} and v_{target} align, we additionally subtract the part of the target command that steers the UAV into the direction of the push force, i.e.,

$$v'_{\text{target}} = v_{\text{target}} + F_{\text{push}} - cF_{\text{push}}, \tag{8}$$

where c is the scalar projection of v_{target} onto F_{push}. The resulting command v'_{target} is then further adjusted by the angular potential field method described above. To avoid oscillations where the UAV is repeatedly commanded into the safety area and pushed out again, we additionally limit the acceleration, i.e., the difference between two subsequent velocity commands, if the UAV approaches the distance d_{close}. Afterwards, the velocity command is sent to the trajectory prediction module to determine the velocity magnitude.

C. Trajectory Prediction and Velocity Generation

The angular potential field method only adjusts the angular components of the flight command. In this section, we describe how the velocity magnitude is chosen based on a time-to-contact estimation.

We scale the adjusted velocity command $v_{\text{cmd},0}$ from the potential field method to the initially commanded velocity and predict the future UAV positions. Here, we assume that the low-level controller tries to reach the commanded velocity as fast as possible by applying a maximum acceleration a_{max}. For a single axis with current velocity v_0, the time t_{accel} needed to reach the velocity command v_{cmd} is thus given by

$$t_{\text{accel}} = \frac{v_{\text{cmd}} - v_0}{a_{\text{max}}}. \tag{9}$$

The position after a time step Δt is

$$\begin{aligned} p' = \min\{t_{\text{accel}}, \Delta t\} v_0 + \frac{1}{2} \min\{t_{\text{accel}}, \Delta t\}^2 a_{\text{max}} \\ + \max\{\Delta t - t_{\text{accel}}, 0\} v_{\text{cmd}}. \end{aligned} \tag{10}$$

Note that the initial position $p_0 = 0$, since all calculations are performed in the egocentric sensor frame. The corresponding future velocity is given by

$$v' = v_0 + \min\{t_{\text{accel}}, \Delta t\} a_{\text{max}}. \tag{11}$$

Using a more complex motion model would also be possible. Obtaining an accurate model of the UAV dynamics is challenging, however, and it has to be done for every UAV independently. Instead, we use this simplified generic model, which proved sufficiently accurate in our experiments to allow safe, dynamic flights in the vicinity of obstacles.

We do not only consider the influence of the UAV dynamics on the trajectory, but also future velocity commands generated by our potential field method. Thus, we choose the prediction time step $\Delta t = 0.05\,\text{s}$ corresponding to the frequency with which new LiDAR scans are obtained. We then transform the current history range image into the predicted sensor frame as already described for the history

TABLE I: Results using random velocity commands for the indoor environment (Fig. 6). We report the average flight velocity v_{avg}, as well as the minimal and average obstacle distances d_{min} and d_{avg}. As an indicator for the proneness to local minima, we additionally report the average distances to the commanded waypoints d_{target} at which the UAV stopps.

Method	d_{target} [m]	v_{avg} [m/s]	d_{min} [m]	d_{avg} [m]
PF [13]	-	**2.38**	0.00	1.62
TG [6]	**27.05**	**1.70**	1.21	2.30
Ours (r_{vel})	27.11	1.02	1.30	2.05
Ours ($r_{\text{euclidean}}$)	30.80	1.27	1.20	2.40
Ours (r_{vel}+pred.)	27.73	1.56	**1.41**	**2.59**

aggregation in Sec. III-A and apply the potential field method from Sec. III-A to generate the next velocity command $v_{\text{cmd},1}$. This process is iterated until we reach the prediction horizon t_{contact} or until the closest obstacle distance is smaller than d_{safe}. In the latter case, let t denote the time of the last iteration. Since we want the time the UAV needs to reach the safety area around obstacles to be at least t_{contact}, we scale the velocity command accordingly. Thus, the velocity command sent to the low-level flight controller is

$$v_{\text{out}} = \frac{t}{t_{\text{contact}}} v_{\text{cmd},0}. \tag{12}$$

If the initial UAV position already is within the safety area, the time-to-contact based approach cannot be applied. In this case, we execute the velocity command only if the future obstacle distance increases in every prediction step.

IV. Evaluation

We apply our obstacle avoidance approach to the UAV described in [1]. It was specially designed for search and rescue missions and is based on the DJI Matrice 210 v2 platform equipped with an Ouster OS-0 3D-LiDAR. The experiments were done using the DJI Hardware-in-the-Loop simulation and Gazebo [17]. The velocity commands generated by our method were executed using the onboard DJI flight controller.

We evaluate our approach against the potential field method (PF) from [13], where two influence spheres around obstacles are used. In the larger one, i.e., the passive avoidance sphere, repulsive forces slow down the UAV movement into the obstacle direction, while the smaller active avoidance sphere is used to push the UAV away from the obstacle. Additionally, we compare against the trajectory generation method (TG) from [6], which generates time-optimal trajectories to sampled target points until a collision-free trajectory is found. To evaluate our different design decisions, we additionally apply two variants of our method to all experiments. First, we use the Euclidean distance $r_{\text{euclidean}}$ instead of the velocity-dependent future range value r_{vel} for the support definition of angular repulsive forces. Second, we remove our trajectory prediction and estimate the time-to-contact based on the vector projection of the commanded velocity onto the obstacle directions.

For all experiments, we use the following parameters:

d_{safe}	d_{close}	a_{max}	t_{contact}	$d_{\text{min\_contact}}$
1.5 m	1.0 m	2 m/s	1.5 s	2

TABLE II: Results for the path following experiment (cf. Fig. 7 and Fig. 8). Success denotes whether the path was fully tracked ✓, the method got stuck in a local minimum (✓) or a collision occurred ×. Additionally, we report the length of the flight path l_{path} (until the UAV stops, either due to reaching the end of the target path, getting stuck in a local minima or due to a collision), the average flight velocity v_{avg}, as well as the minimal and average obstacle distances d_{min} and d_{avg}.

v_{max} [m/s]	Method	Suc.	l_{path} [m]	v_{avg} [m/s]	d_{min} [m]	d_{avg} [m]
1	PF [13]	(✓)	81.27	0.91	1.04	1.84
	TG [6]	(✓)	102.00	0.87	1.24	1.76
	Ours (r_{vel})	(✓)	125.17	0.77	1.37	1.79
	Ours ($r_{\text{euclidean}}$)	✓	125.36	0.91	1.37	1.72
	Ours (r_{vel}+pred.)	✓	**124.89**	**0.95**	**1.38**	**1.91**
2	PF [13]	✓	130.19	1.85	0.46	1.75
	TG [6]	(✓)	104.13	1.71	1.31	1.79
	Ours (r_{vel})	(✓)	126.37	0.92	1.37	1.77
	Ours ($r_{\text{euclidean}}$)	✓	128.51	**1.97**	**1.40**	**2.12**
	Ours (r_{vel}+pred.)	✓	**127.88**	1.89	1.39	2.01
3	PF [13]	✓	133.83	**2.79**	0.31	1.72
	TG [6]	(✓)	106.89	2.21	1.20	1.80
	Ours (r_{vel})	(✓)	125.02	0.84	1.37	1.75
	Ours ($r_{\text{euclidean}}$)	✓	**133.82**	2.14	1.35	1.83
	Ours (r_{vel}+pred.)	✓	134.48	2.77	**1.39**	**2.43**
4	PF [13]	×	123.30	**3.70**	0.00	1.63
	TG [6]	(✓)	3.67	1.29	1.29	1.81
	Ours (r_{vel})	(✓)	126.07	0.87	1.37	1.76
	Ours ($r_{\text{euclidean}}$)	✓	**134.04**	2.50	1.34	1.93
	Ours (r_{vel}+pred.)	✓	140.00	**3.48**	**1.39**	**2.55**
5	PF [13]	×	73.82	**4.57**	0.00	1.61
	TG [6]	(✓)	50.44	1.60	1.36	1.74
	Ours (r_{vel})	(✓)	126.12	0.88	1.37	1.77
	Ours ($r_{\text{euclidean}}$)	(✓)	56.91	2.08	**1.45**	1.75
	Ours (r_{vel}+pred.)	✓	**142.33**	**3.84**	1.33	**2.45**
6	PF [13]	×	72.89	**4.90**	0.00	1.58
	TG [6]	(✓)	135.06	2.59	1.38	1.79
	Ours (r_{vel})	(✓)	103.93	0.85	1.37	1.75
	Ours ($r_{\text{euclidean}}$)	(✓)	57.44	1.97	**1.44**	1.88
	Ours (r_{vel}+pred.)	✓	**144.75**	**4.29**	1.39	**2.42**

Our method starts decelerating the UAV if the predicted obstacle distance falls below the safety threshold d_{safe} within the look-ahead t_{contact}. Thus, the radius of the passive avoidance sphere for the potential field method [13] is chosen as $d_{\text{passive}} = d_{\text{safe}} + v_{\text{max}} t_{\text{contact}}$, where v_{max} denotes the maximum allowed flight velocity. The size of the active avoidance sphere corresponds to our safety threshold, i.e., $d_{\text{active}} = d_{\text{safe}}$.

Obstacle avoidance methods must ensure a safe distance to obstacles independent of the—possibly adversarial—input commands. In a first experiment, we thus investigate how the different methods react to randomly chosen velocity commands. For this, we designed the indoor warehouse environment shown in Fig. 6. The UAV starts at the center of the map with an altitude of 2 m and is commanded to move at a velocity of 3 m/s into the direction of a randomly sampled target position, which might even be outside the warehouse. Every 10 s, a new target is sampled.

Tab. I summarizes the results for a flight time of 300 s per method. The flight velocities were too high for the potential field method [13] and it collided with an obstacle. While all other methods successfully avoided collisions, our method maintained the largest distance to obstacles. Using the velocity-dependent range value r_{vel} for the repulsive force definition significantly increases the obstacle distance and the trajectory prediction allows higher velocities.

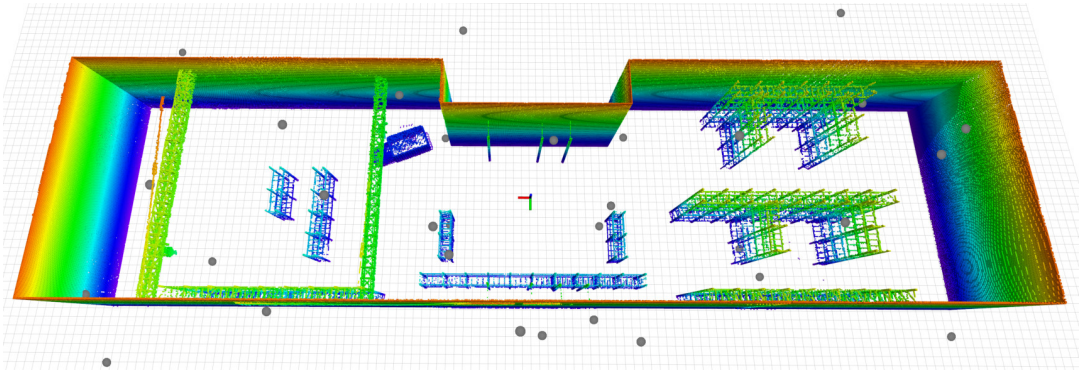

Fig. 6: A simulated indoor environment for the random input experiment. The randomly chosen waypoints are marked by gray spheres. The initial UAV position is at the map center, marked by the coordinate axes.

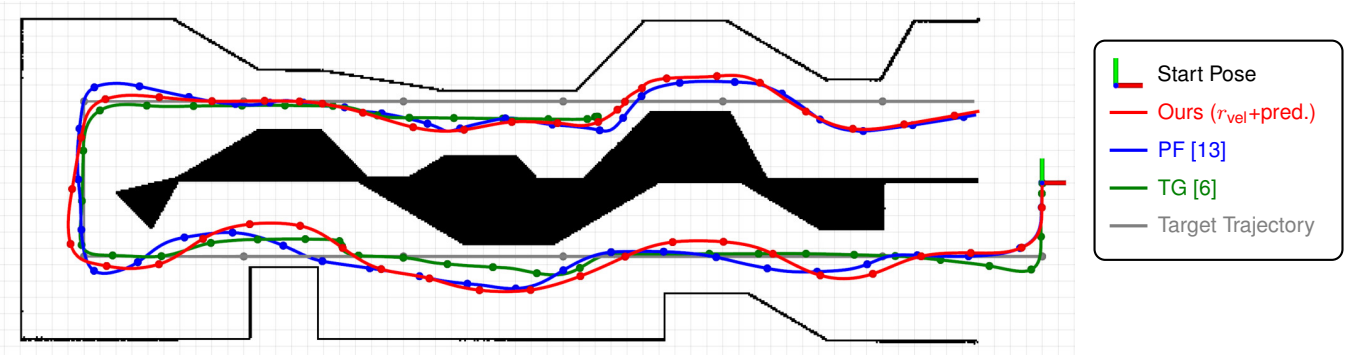

Fig. 7: Top-down view of example trajectories for our method (red), the potential field method from [13] (blue) and the trajectory generation method [6] (green). On each trajectory, spheres are sampled with a time difference of 1 s. The target trajectory (gray) shall be tracked at a velocity of 3 m/s. Our method shows the most far-sighted behavior and thus also maintains the largest distances to obstacles.

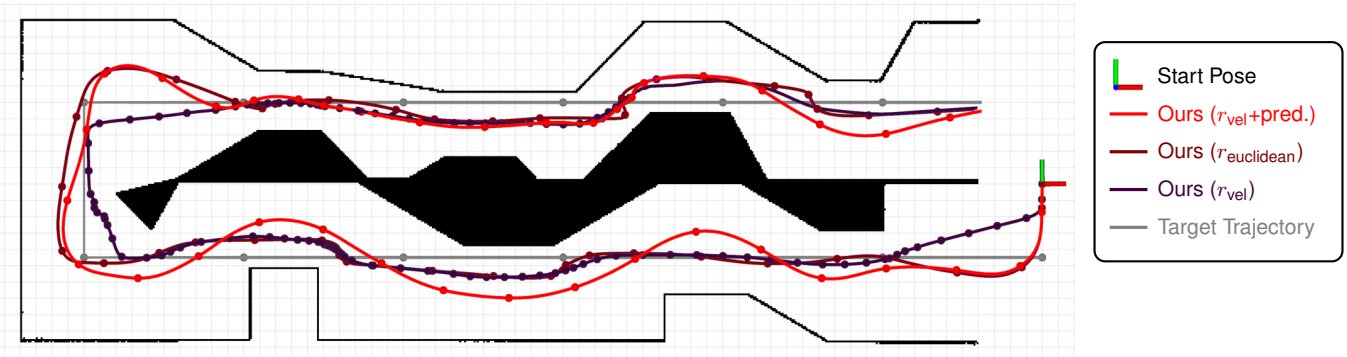

Fig. 8: Top-down view of example trajectories for our method (bright red), our method without velocity-dependent support radius (dark red), and our method without trajectory prediction (purple). On each trajectory, spheres are sampled with a time difference of 1 s. The target trajectory (gray) shall be tracked at a velocity is 4 m/s. Combining the velocity-dependent support radius with trajectory prediction results in a smooth trajectory at high velocities.

In a second experiment, we analyze at which velocities the different obstacle avoidance methods can track a pre-planned path that is too close to (initially unknown) obstacles. The results are summarized in Tab. II and Fig. 7 depicts example trajectories. Our method shows a more far-sighted behavior than the Cartesian potential field method and the sampling-based trajectory generation. Thus, our method can maintain larger distances to the obstacles, even at high flight velocities. Furthermore, it is less prone to local minima. Fig. 8 compares trajectories from the different variants of our method. The velocity-dependent support radius results in smoother trajectories, while the trajectory prediction allows significantly higher flight velocities.

One advantage of our proposed angular potential field-based obstacle avoidance method is the possibility to adapt the flight direction to nearby obstacles without affecting the velocity. To further evaluate this aspect, we designed an experiment where the UAV moves through a corridor with changing height values for both, floor and ceiling. Thus, the vertical flight angle has to be frequently adjusted to keep a safe distance to obstacles. Tab. III shows that the trajectory prediction is essential for our method to obtain

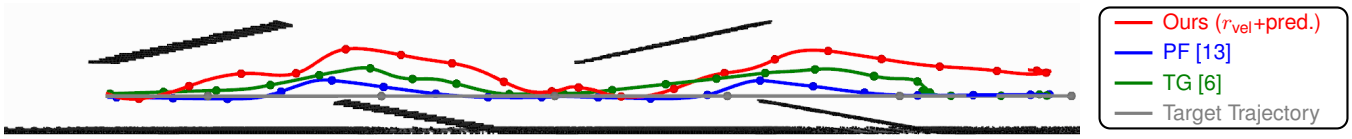

Fig. 9: Side view of an excerpt of the environment with vertical obstacles. Shown are the trajectories of our method (red), the potential field method from [13] (blue) and the trajectory generation method [6] (green). On each trajectory, spheres are sampled with a time difference of 1 s. The target trajectory (from right to left) is depicted in gray. The maximal allowed velocity is $3\,\mathrm{m/s}$.

TABLE III: Results for the path following experiment with vertical obstacles (cf. Fig. 9). We report the length of the flight path l_{path}, the average flight velocity v_{avg}, the flight duration t_{flight}, as well as the minimal and average obstacle distances d_{min} and d_{avg}.

Method	l_{path} [m]	v_{avg} [m/s]	t_{flight} [s]	d_{min} [m]	d_{avg} [m]
PF [13]	128.13	2.85	44.96	1.03	1.68
TG [6]	130.63	2.61	50.05	1.57	1.97
Ours (r_{vel})	**124.98**	1.53	81.67	**1.59**	2.16
Ours ($r_{\text{euclidean}}$)	129.20	**3.00**	**43.07**	1.47	1.94
Ours (r_{vel}+pred.)	130.20	2.92	44.59	1.29	**2.59**

TABLE IV: Comparison of the computations times per control iteration.

Method	t_{min} [ms]	t_{avg} [ms]	t_{max} [ms]
PF [13]	$<$ **1**	**4**	**11**
TG [6]	15	24	227
Ours	1	15	32

high flight velocities which are about twice the velocities of our method without prediction. Interestingly, the classic potential field method achieves similar velocities in this experiment. This is due to the fact that it does not adjust its trajectory to the obstacles much, as can be seen in Fig. 9. Our angular potential field method, however, maximizes the obstacle distance while still achieving fast flights.

Finally, we compare the computation times of each method in Tab. IV. As expected, the classic potential field method is the fastest one. The sampling-based trajectory generation method shows a large runtime variance between different control iterations. Especially in constraint environments, it can be difficult to sample valid target points, resulting in high maximal computation times. Our method reliably achieves runtimes significantly below the sensor measurement frequency of 20 Hz.

V. Conclusion

In this work, we proposed a fast predictive obstacle avoidance method that combines angular potential fields to determine the flight direction with trajectory prediction and time-to-contact estimation to scale the flight velocity magnitude. Our method operates directly on 3D LiDAR range images and does not depend on higher-level localization or mapping. The conducted experiments show that the proposed method reliably keeps a safe distance to obstacles, while showing a more far-sighted behavior than previous obstacle avoidance methods that directly operate on sensor data. Thus, our method is suitable for high-velocity flights in cluttered environments.

References

[1] D. Schleich, M. Beul, J. Quenzel, and S. Behnke, "Autonomous flight in unknown GNSS-denied environments for disaster examination," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021.

[2] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[3] V. Usenko, L. Von Stumberg, A. Pangercic, and D. Cremers, "Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[4] J. Zhang, R. G. Chadha, V. Velivela, and S. Singh, "P-CAP: Pre-computed alternative paths to enable aggressive aerial maneuvers in cluttered environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[5] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "Maximum likelihood path planning for fast aerial maneuvers and collision avoidance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[6] M. Beul and S. Behnke, "Trajectory generation with fast lidar-based 3D collision avoidance for agile MAVs," in *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2020.

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

[8] M. Missura and M. Bennewitz, "Predictive collision avoidance for the dynamic window approach," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[9] M. Dobrevski and D. Skočaj, "Adaptive dynamic window approach for local navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[10] M. Missura, A. Roychoudhury, and M. Bennewitz, "Fast-replanning motion control for non-holonomic vehicles with aborted A*," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[11] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *The International Journal of Robotics Research (IJRR)*, vol. 5, no. 1, 1986, pp. 90–98.

[12] M. Nieuwenhuisen, M. Schadler, and S. Behnke, "Predictive potential field-based collision avoidance for multicopters," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*, vol. 1, p. W2, 2013.

[13] M. Beul, D. Droeschel, M. Nieuwenhuisen, J. Quenzel, S. Houben, and S. Behnke, "Fast autonomous flight in warehouses for inventory applications," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 3121–3128, 2018.

[14] V. Sezer and M. Gokasan, "A novel obstacle avoidance algorithm:"Follow the gap method"," *Robotics and Autonomous Systems*, vol. 60, no. 9, pp. 1123–1134, 2012.

[15] H. Houshyari and V. Sezer, "A new gap-based obstacle avoidance approach: follow the obstacle circle method," *Robotica*, pp. 1–24, 2021.

[16] J.-H. Cho, D.-S. Pae, M.-T. Lim, and T.-K. Kang, "A real-time obstacle avoidance method for autonomous vehicles using an obstacle-dependent gaussian potential field," *Journal of Advanced Transportation*, vol. 2018.

[17] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.