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

Perception-aware Path Planning for UAVs using Semantic Segmentation

Luca Bartolomei, Lucas Teixeira and Margarita Chli
Vision For Robotics Lab, ETH Zürich, Switzerland

Abstract— **In this work, we present a perception-aware path-planning pipeline for Unmanned Aerial Vehicles (UAVs) for navigation in challenging environments. The objective is to reach a given destination safely and accurately by relying on monocular camera-based state estimators, such as Keyframe-based Visual-Inertial Odometry (VIO) systems. Motivated by the recent advances in semantic segmentation using deep learning, our path-planning architecture takes into consideration the semantic classes of parts of the scene that are perceptually more informative than others. This work proposes a planning strategy capable of avoiding both texture-less regions and problematic areas, such as lakes and oceans, that may cause large drift or failures in the robot's pose estimation, by using the semantic information to compute the next best action with respect to perception quality. We design a hierarchical planner, composed of an A^* path-search step followed by B-Spline trajectory optimization. While the A^* steers the UAV towards informative areas, the optimizer keeps the most promising landmarks in the camera's field of view. We extensively evaluate our approach in a set of photo-realistic simulations, showing a remarkable improvement with respect to the state-of-the-art in active perception.**

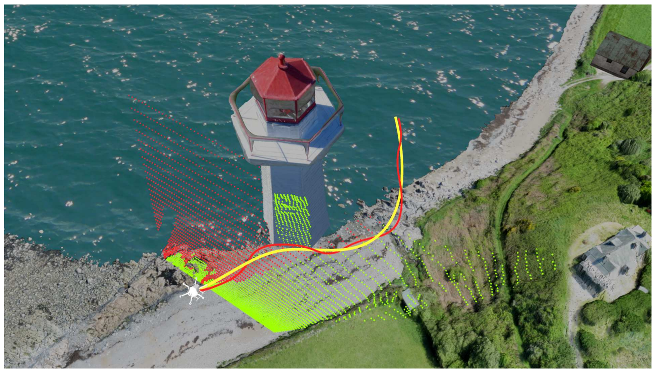


Fig. 1: 3D-view of a planning iteration of the proposed planner in the *Bay* experiment. The informative areas are highlighted with green dots, while the perceptually degraded parts are in red. The planner is able to avoid the collision between the UAV and the obstacle, while steering the robot towards the informative areas. The red line is the trajectory computed by the path-searching algorithm, while the yellow line is the trajectory after optimization.

I. Introduction

Safe navigation in unknown and dynamic environments is a fundamental skill for truly autonomous mobile robots. For successful autonomous navigation, a robot must map the environment with sufficient accuracy to avoid collisions with obstacles and at the same time localize itself in this estimated map. Today, both real-time localization and mapping are still open problems. While GPS is widely used, there are numerous situations that it can fail, such as, around high buildings and mountains, due to bad weather, and jamming. In these cases, ground robots can just stop, but aerial robots need a safe method to reach an emergency landing spot or even better continue to fly towards their destination. Visual-based navigation is an alternative to GPS, usually using Visual-Inertial Simultaneous Localization and Mapping (VI-SLAM) algorithms. In addition, VI-SLAM is also capable of computing the local map to avoid collision with unknown or dynamic objects [1]. However, it also has its limitation; for example, these algorithms do not perform well in texture-less areas. Recently, the emergence of perception-aware path-planning algorithms showed it is possible to avoid paths through texture-less regions, such as [2], [3].

This work proposes an active perception path-planning method that avoids not only texture-less regions, but also is capable of avoiding areas with sufficient texture, albeit unsuitable for robust localization (e.g. due to dynamic objects and specularities). Our approach aims to leverage recent advances in real-time semantic segmentation using deep learning to identify the problematic areas. Today, there is a plethora of algorithms that are capable of detecting well-known problematic areas to fly. Lakes and oceans can be easily detected by algorithms such as [4], as well as roads and people can be recognized by [5]. In brief, the contributions of this work are the following:

- the design of a perception-aware navigation architecture for UAVs in challenging environments,
- the design of a hierarchical path-planning pipeline able to incorporate information from semantic segmentation with real-time re-planning capabilities, and
- an extensive quantitative evaluation of the performances of the proposed system with respect to the state-of-the-art in active perception in photo-realistic simulations.

II. Related Work

The link between active perception and the problem of SLAM, providing estimates about the robot's pose and its workspace, is strong [6], [7]. Both the motion and the path followed by a robot have a great impact on the performance of state estimation algorithms, so the aim in active perception is to fill the gap between path-planning and state estimation by considering the robot's state uncertainty during planning. This constitutes a fundamental step towards the creation of a

This work was supported by Swiss National Science Foundation (SNSF, Agreement no. PP00P2183720), NCCR Robotics, Amazon Research Awards 2019 and the HILTI group.
The video is available at `https://youtu.be/vxAXimQ9CGU`

fully autonomous system, able to cope with the uncertainties present in a real mission.

Active perception has roots in the seminal work of [8], advocating that sensory performance can be improved by proper selection of control actions. Other early works describe the importance of using sensors actively (or *active vision*) for problems, such as structure from motion [9]. Since then, efforts have been made to integrate perception, path-planning and control in a unified framework. One of the first successful approaches is the work in [10], attempting to control two movable cameras on a ground robot to reduce the uncertainty during localization estimation. Based on their earlier results for incremental building and maintaining of maps for a navigating robot, [10] relies on the cross-coupling between the pose of the robot and the mapped features. In the same spirit, [11] and [12] provide more recent, successful examples that integrate path-planning with control and state estimation. In contrast to [10], these works do not use sensors actively and instead focus on motion generation for enhancing the state estimation process.

In general, the problem of active perception can be formulated as Partially Observable Markov Decision Processes (POMDPs) [13], providing a framework for planning under uncertainty. However, the complexity of solving high-dimensional POMDP models motivates more efficient solutions. Belief-space planning emerged to allow the integration of the expected robot belief into its motion planning efficiently [14] and led to powerful tools, such as belief roadmaps [15] and belief trees [6], which were used to approach the problem of exploration and mapping [16]. In recent years, various approaches based on receding-horizon planning strategies emerged, combining efficient exploration of unknown environments with belief-space based planning in order to enhance the on-the-go mapping behaviour of the robot [3], [17]. One of the main objectives is to create a computationally tractable planning framework that can generate optimized navigation paths online, running alongside all other essential computation onboard a small robot [18].

Currently, the most relevant competitor to this work is Zhang and Scaramuzza [3], proposing a perception-aware receding-horizon approach that generates a collection of possible trajectories and evaluates them in terms of landmark concentration, collision probability, and distance to the goal. In this work, we show that landmark concentration alone is not enough for selecting the best areas to fly trough, because unstable landmarks can introduce significant errors in visual-based localization systems. Instead, we employ semantics to evaluate the quality of both the candidate areas for navigation and the landmarks. We propose a path-planning pipeline building on top of [19] able to incorporate this additional information in order to encourage the navigation through informative regions of the space and to favor the triangulation of high-quality landmarks. Our approach shows noticeable improvements in state estimation performance when compared to a purely reactive planning strategy and the perception-aware planner proposed by [3]. Our pipeline is described in the next sections.

III. Problem Description

The overall problem considered in this work is to reach a predefined goal pose while minimizing the drift in a visual-based pose estimation algorithm. The core assumption of this work is that there exist areas which are more suitable for localization than others, and our objective is to fly through them as much as possible. We formulate this as a path-planning problem, where the perception quality extracted from semantic labels plays a primary role. We aim to generate smooth and collision-free trajectories in a receding horizon fashion while considering the constraints of Keyframe-based Visual-Inertial Odometry (VIO) systems. In this work, semantic classes (e.g. water and ground) are manually mapped to a certain level of perceptual informativeness. To incorporate the perception quality, we employ a path-searching algorithm to encourage navigation in well-textured areas, followed by a trajectory optimization step where the objective is to keep the best 3D landmarks estimated by the VIO pipeline in the field of view of the camera.

IV. System Overview

As shown in Fig. 2, the proposed pipeline consists of two main components; a monocular camera-based state estimation module and a path-planning architecture.
The platform is equipped with a front-looking stereo camera with hardware-synchronized IMU [20]. A dense 3D reconstruction of the surroundings of the robot is obtained from the pair of stereo images, while only one camera is used to estimate the robot's state. The dense point cloud is stored as an occupancy map by means of a 3D circular buffer [21]. We employ the keyframe-based VIO system VINS-Mono [22] to estimate the 6-Degrees-of-Freedom (DoF) pose of the camera using the stream of grayscale images and IMU measurements. This estimate is further fused with the readings of an additional IMU mounted on the center of the UAV using the Multi-Sensor Fusion (MSF) framework [23] in order to get better velocity estimates for control purposes. In addition to the poses, the VIO module gives the estimated positions of the 3D landmarks used for localization.

Before being sent to the trajectory generation module, both the dense 3D occupancy map and the sparse landmarks go through a classification step, which provides binary labels for all the points. We consider a point to be of high quality if it belongs to parts of the scene useful for camera-based state estimation. This is motivated by the fact that some parts of the environment, such as water or moving objects, can produce erroneous state estimates because VIO systems assume non-specular surfaces and static scenes. Relying on such features may have disastrous consequences, e.g. increasing drift in the estimation or failures. Given the detection problem of moving objects, such as cars and people, as well as ground classification, is not the core of our contribution here, we used ground-truth semantics in our experiments, but there are several off-the-shelf algorithms available [4], [5] that we could use to this end. The labelled occupancy map and landmarks are used by the path-planning architecture to reason about the next best action. The path-planning module

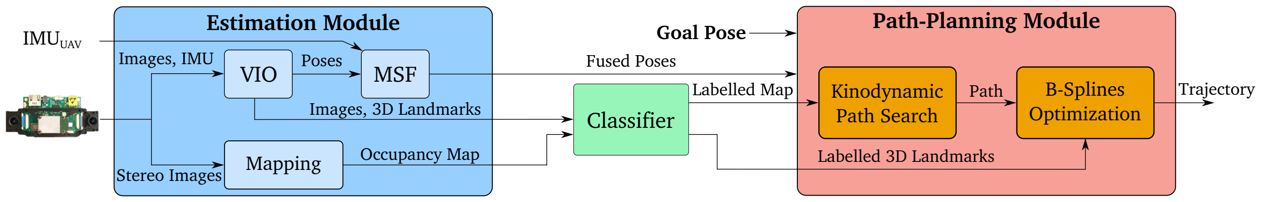


Fig. 2: Schematic representation of the pipeline. In the Estimation Module, we process the sensor inputs (images and IMU readings) to estimate the pose of the UAV and the positions of the 3D landmarks. These landmarks, together with the images and the occupancy map obtained from the stereo camera, go through a classification step, where each point is evaluated on its utility for camera-based state estimation. The labelled data and the poses are used the Path-Planning Module to generate the next best trajectory while considering both the dynamics of the platform and the perception quality.

is based on a hierarchical structure, composed of a path-search step followed by trajectory optimization. The path-search planner uses the labelled occupancy map to steer the robot towards potentially informative areas using the A^* algorithm. The output path might be sub-optimal, so it is later refined by a B-Spline-based planner. We formulate an optimization problem in an attempt to follow the trail of high-quality landmarks by keeping them in the field of view of the camera during navigation. In the next section, the proposed path-planning architecture is explained in details.

V. Path Planning

The first step of the planning pipeline consists of a kinodynamic A^* path search [24]. Its output is used as an initial guess by a second planner, which parametrizes the trajectory as continuous B-Splines and performs trajectory optimization. In the following, we indicate with $\mathcal{M}$ the occupancy map of the environment.

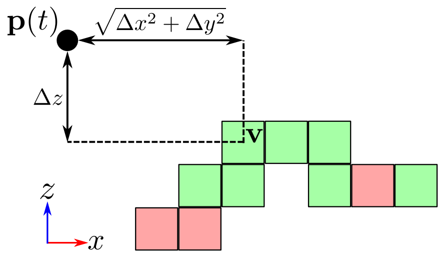
A. Kinodynamic Path Search

The proposed path-searching module builds on top of the kinodynamic A^* algorithm presented in [19]. Similarly to the original implementation, we use motion primitives instead of straight lines as graph edges in order to respect the multi-copter's dynamics. As the first step is the iterative expansion of primitives in $\mathcal{M}$, the trajectories ending in the same voxels are pruned and only the one with the smallest cost is maintained. After expansion and pruning, the remaining primitives are checked for safety and dynamic feasibility. Finally, in order to improve the chances of reaching the goal, we adopt the analytic expansion strategy detailed in [19].
The path search is limited to the position of the robot in $\mathbb{R}^3$, and thanks to the differential flatness of multirotor systems [25], the trajectory is represented as three, independent, time-parametrized polynomial functions $\mathbf{p}(t)$:

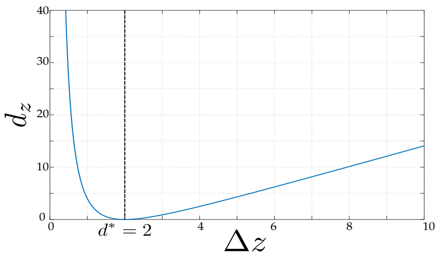
$$\mathbf{p}(t) := [p_x(t), p_y(t), p_z(t)]^T,\ p_d(t) = \sum_{k=0}^{K} a_k t^k \tag{1}$$

with $d \in \{x, y, z\}$. Notice that the orientation is not considered, since at this point the objective is to steer the robot towards informative areas rather than enforcing rotations towards them. Thus, the orientation $\theta(t)$ along the path is computed as:

$$\theta(t) := \arctan \frac{\dot{p}_y(t)}{\dot{p}_x(t)}. \tag{2}$$



(a) Side view of the occupancy map. The green voxels belong to $\mathcal{M}_G$, while the red ones correspond to low-quality areas. The distances used in the computation of d_M for a voxel $\mathbf{v}$ are highlighted.



(b) Plot of the cost d_z as a function of the height difference Δz and the threshold d^*.

Fig. 3: Representation of the terms necessary for the computation of d_M in the path-search problem.

We consider the multirotor system as linear and time-invariant, with state $\mathbf{s}(t) := [\mathbf{p}(t)^T, \dot{\mathbf{p}}(t)^T, \ldots, \mathbf{p}^{(n-1)}(t)^T]^T \in \chi \subset \mathbb{R}^{3n}$ and control input $\mathbf{u}(t) := \mathbf{p}^{(n)}(t) \in \mathcal{U} := [-u_{max}, u_{max}]^3 \subset \mathbb{R}^3$. We set $n = 2$, corresponding to a double integrator.
As the aim is to encourage the navigation in well-textured areas, the cost of a trajectory is defined as

$$\mathcal{J}(T) = \int_0^T \Big(w_u \|\mathbf{u}(t)\|^2 + w_M d_M(\mathbf{p}(t), \mathcal{M}) \Big) dt + w_T T, \tag{3}$$

where $\|\mathbf{u}(t)\|^2$ is the control cost; $d_M(\mathbf{p}(t), \mathcal{M})$ represents a penalty for navigating far away from informative areas; and T is the total time of the trajectory. The terms w_u, w_M and w_T are the constant weights associated to each cost. While the meaning of the first and last terms is clear, we now describe the term $d_M(\mathbf{p}(t), \mathcal{M})$ in more detail.

Given the latest labelled occupancy map $\mathcal{M}$, we treat the informative voxels as attractors. Notice that the remaining voxels do not act as repellers, because $d_M(\mathbf{p}(t), \mathcal{M})$ is a soft

constraint on the position of the robot. This is motivated by extreme situations, where most or all of the area is textureless. In this case, the robot should not interrupt its navigation towards the goal, even if it has to navigate these poorly textured regions. In such situations, the path-search algorithm tries to minimize the control cost and total time required to reach the goal.
If we define $\mathcal{M}_G \subseteq \mathcal{M}$ as the set of voxels corresponding to high-quality areas, $d_M(\mathbf{p}(t), \mathcal{M})$ is computed as

$$d_M(\mathbf{p}(t), \mathcal{M}) := \sum_{\mathbf{v} \in \mathcal{M}_G \subseteq \mathcal{M}} d_v(\mathbf{p}(t), \mathbf{v}) = \sum_{\mathbf{v} \in \mathcal{M}_G \subseteq \mathcal{M}} d_{xy}(\mathbf{p}(t), \mathbf{v}) + d_z(\mathbf{p}(t), \mathbf{v}), \tag{4}$$

where each voxel $\mathbf{v} \in \mathcal{M}_G$ has coordinates $\mathbf{v} = [v_x, v_y, v_z]^T$. The cost $d_M(\mathbf{p}(t), \mathcal{M})$ is composed of two potential functions (Fig. 3). While $d_{xy}(\mathbf{p}(t), \mathbf{v})$ encourages navigation on top of texture-full areas, $d_z(\mathbf{p}(t), \mathbf{v})$ is a soft constraint on the minimum distance from the ground, as shown in Fig. 3a. The two costs are calculated as

$$d_{xy}(\mathbf{p}(t), \mathbf{v}) := (p_x(t) - v_x)^2 + (p_y(t) - v_y)^2, \tag{5}$$

and by defining $\Delta z := |p_z(t) - v_z|$,

$$d_z(\mathbf{p}(t), \mathbf{v}) := d^* \Delta z + \frac{1}{2} \frac{{d^*}^4}{\Delta z^2} - \frac{3}{2} {d^*}^2, \tag{6}$$

where d^* controls the minimum height of the robot with respect to the voxels in $\mathcal{M}_G$. The cost d_z is plotted as a function of Δz in Fig. 3b.
From these definitions, the cost of a motion primitive with discrete inputs $\mathbf{u}(t) = \mathbf{u}_k$ and time duration τ is $e_c = (w_u \|\mathbf{u}_k\|^2 + w_M d_{M,k} + w_T)\tau$. The final cost of the path to reach a state $\mathbf{s}_c$ from the start state $\mathbf{s}_s$ consisting of N primitives is

$$g_c = \sum_{i=0}^{N} (w_u \|\mathbf{u}_{k_i}\|^2 + w_M d_{M,k_i} + w_T)\tau. \tag{7}$$

In order to speed up the search in the A^* algorithm, the choice of an admissible and consistent heuristic is fundamental. We propose to use the heuristic function $h_c := h_{u,T} + h_M$, where $h_{u,T}$ is obtained by minimizing the relaxed cost function from $\mathbf{s}_c$ to the goal state $\mathbf{s}_G$

$$\mathcal{J}(T) = \int_0^T w_u \|\mathbf{u}(t)\|^2 dt + w_T T \tag{8}$$

and $h_M := d_M(\mathbf{p}_G, \mathcal{M})$, where $\mathbf{p}_G$ corresponds to the position in the goal state $\mathbf{s}_G$. Finally, the total cost to reach the goal state is defined as $f_c = g_c + h_c$. Regarding the computation of $h_{u,T}$, the interested reader can refer to [19].

B. Trajectory Optimization

While the trajectory computed in the path-searching step encourages navigation towards the informative areas, the information produced by the VIO module, i.e. the 3D landmarks, is not utilized. Therefore, a trajectory optimization step leveraging this additional level of information is necessary. We propose to parametrize the trajectory $\boldsymbol{\pi}(t)$ as an uniform B-Spline and formulate an optimization problem where the aim is to generate smooth and feasible trajectories encouraging the tracking and triangulation of high-quality 3D landmarks.

1) Uniform B-Splines: A B-Spline $\boldsymbol{\pi}(t)$ of degree K can be evaluated as follows:

$$\boldsymbol{\pi}(t) = \sum_{i=0}^{N} \mathbf{q}_i B_{i,K-1}(t), \tag{9}$$

where $\mathbf{q}_i$ are the control points at time t_i with $i \in \{0, \ldots, N\}$, and $B_{i,K-1}(t)$ are the basis functions.

In our case, each control point in $\{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_N\}$ encodes both the position and orientation of the robot, i.e. $\mathbf{q}_i := [x_i, y_i, z_i, \theta_i]^T \in \mathbb{R}^4$ with $\theta_i \in [-\pi, \pi)$. We adopt a uniform representation, meaning the elements of the knot vector $[t_0, t_1, \ldots, t_{N+1+K}]$ and $t_m \in \mathbb{R}$ have a fixed knot span $\Delta t = t_{m+1} - t_m$. For sake of readability, we drop the explicit dependency on time t in the following.

2) Problem Formulation: The planning problem is formulated as an optimization with the following cost function:

$$\mathcal{F}_{TOT} = \lambda_s \mathcal{F}_s + \lambda_f \mathcal{F}_f + \lambda_c \mathcal{F}_c + \lambda_l \mathcal{F}_l + \lambda_v \mathcal{F}_v, \tag{10}$$

where $\mathcal{F}_s$ is the smoothness cost; $\mathcal{F}_c$ is the collision cost; $\mathcal{F}_f$ is a soft limit on the derivatives (velocity and acceleration) over the trajectory; $\mathcal{F}_l$ is the penalty associated to loosing track of the high-quality landmarks currently in the field of view; and $\mathcal{F}_v$ is a soft constraint on the co-visibility between control points of the spline. The coefficients $\lambda_s, \lambda_c, \lambda_f, \lambda_l$ and λ_v are the constant weights associated to each cost.

For a B-Spline of degree K defined by $N + 1$ control points $\{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_N\}$, our optimization acts on $\{\mathbf{q}_K, \mathbf{q}_{K+1}, \ldots, \mathbf{q}_{N-K}\}$ while keeping the first and last K control points fixed due to boundary constraints. Thus, we do not have to add a cost associated to the end point of the trajectory. Furthermore, in contrast to other recent approaches, such as [21], in the formulation of the smoothness and feasibility costs, we explicitly remove the dependency on time and we perform time re-allocation after optimization; we notice that this approach does not degrade the quality of the trajectory.

We formulate the cost functions as elastic bands [26] by writing fictitious forces $\mathbf{F}_{i,j} := \mathbf{q}_i - \mathbf{q}_j$ acting from the control points $\mathbf{q}_j$ to $\mathbf{q}_i$. From this, we formulate the smoothness cost as

$$\mathcal{F}_s = \sum_{i=K-1}^{N-K+1} \|\mathbf{F}_{i+1,i} + \mathbf{F}_{i-1,i}\|^2 = \sum_{i=K-1}^{N-K+1} \|(\mathbf{q}_{i+1} - \mathbf{q}_i) + (\mathbf{q}_{i-1} - \mathbf{q}_i)\|^2. \tag{11}$$

The purpose of $\mathcal{F}_s$ is to avoid discontinuities in the position and in the velocity and acceleration profiles. Similarly, given the maximum velocity and acceleration v_{max} and a_{max}, the feasibility cost $\mathcal{F}_f$ penalizes the segments of the trajectory exceeding these limits. The penalty for a single dimension velocity v_d with $d \in \{x, y, z\}$ is:

$$F_v(v_d) = \begin{cases} (v_d^2 - v_{max}^2)^2 & \text{if } v_d^2 \geq v_{max}^2 \\ 0 & \text{otherwise} \end{cases}. \tag{12}$$

The cost for the acceleration $F_a(a_d)$ is identical. The final feasibility cost is then given by the contribution of both the velocity and acceleration limits:

$$\mathcal{F}_f = \sum_{d \in \{x,y,z\}} \sum_{i=K-1}^{N-K+1} F_v(v_{i,d}) + F_a(a_{i,d}). \tag{13}$$

Differently from $\mathcal{F}_s$ and $\mathcal{F}_f$, the collision cost acts as a repulsive force, penalizing the trajectory points that are within a threshold distance d_o to the closest obstacles:

$$\mathcal{F}_c = \sum_{i=K}^{N-K} c(\mathbf{q}_i) \tag{14}$$

with

$$c(\mathbf{q}_i) = \begin{cases} (d(\mathbf{q}_i) - d_o)^2 & \text{if } d(\mathbf{q}_i) \leq d_o \\ 0 & \text{otherwise} \end{cases}, \tag{15}$$

where $d(\mathbf{q}_i)$ is the distance between $\mathbf{q}_i$ and the closest obstacle.

Instead, the cost $\mathcal{F}_l$ encourages the triangulation of high-quality landmarks and assigns a high penalty to the control points with few or none such landmarks in the field of view. The choice of using the number of tracked landmarks instead of other information gain formulations, is dictated by the fact that VIO systems benefit more from good spatial distributions of landmarks than sparse informative measurements. The camera frustum is characterized by its horizontal and vertical angular apertures, indicated as θ_{hor} and θ_{ver}, respectively. We consider a landmark to be visible if it lies within the field of view, which is identified by five half-spaces. Notice that, in order to be able to use standard tools for numerical optimization, we need to formulate this visibility condition as a continuous, differentiable indicator function [27]. The half spaces are represented by the set of vectors connecting the optical center of the camera to the vertices of the sensor, as shown in Fig. 4. In the camera reference frame C, these vectors are

$$\begin{aligned} \mathbf{v}_{TR} &= f \cdot \begin{bmatrix} \sin\theta_{hor} & -\sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{LR} &= f \cdot \begin{bmatrix} \sin\theta_{hor} & \sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{LL} &= f \cdot \begin{bmatrix} -\sin\theta_{hor} & \sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{TL} &= f \cdot \begin{bmatrix} -\sin\theta_{hor} & -\sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{Z} &= f \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \end{aligned}, \tag{16}$$

where f is the focal length in meters.
In order to check whether a landmark lies within the camera frustum, it has to be expressed in the camera frame C; however, the set of available 3D landmarks $\mathcal{L}^W$ from the VIO module are expressed in the odometry frame W. If we indicate the subset of high-quality landmarks as $\mathcal{L}_G^W \subseteq \mathcal{L}^W$,

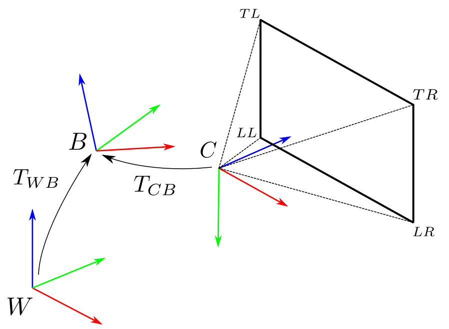


Fig. 4: Representation of the field of view of the camera and of the reference frames. W corresponds to the odometry reference frame, while B is the body (IMU) and C the reference frame of the camera.

we can express its elements $\mathbf{l}_W$ in C as follows:

$$\mathbf{l}_C = \mathbf{R}_{CW}\mathbf{l}_W + \mathbf{t}_{CW}, \tag{17}$$

where $\mathbf{R}_{CW} \in SO(3)$ and $\mathbf{t}_{CW} \in \mathbb{R}^3$ are the rotation matrix and the translation vector from W to C, respectively. For ease of representation, we adopt homogeneous matrices $\mathbf{T}_{AB} \in SE(3)$ to indicate transformations from B to A:

$$\mathbf{T}_{AB} = \begin{bmatrix} \mathbf{R}_{AB} & \mathbf{t}_{AB} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}. \tag{18}$$

To obtain $\mathbf{T}_{CW}$ for a given control point $\mathbf{q} = [x, y, z, \theta]^T$, we proceed as follows. As the VIO estimation corresponds to the pose of the body B (generally the IMU in VIO systems) in W (i.e. $\mathbf{T}_{WB}$), we need to concatenate the fixed transformation $\mathbf{T}_{CB}$, which is known beforehand from calibration:

$$\mathbf{T}_{CW}(\mathbf{q}) = \mathbf{T}_{CB}\mathbf{T}_{WB}^{-1}(\mathbf{q}). \tag{19}$$

Since the control points of the B-Spline represent the body in W, we can obtain the homogeneous transformation $\mathbf{T}_{WB}$ for a control point $\mathbf{q}$ as

$$\mathbf{T}_{WB}(\mathbf{q}) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & x \\ \sin\theta & \cos\theta & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{20}$$

We can now define the following indicator function over the set of all high-quality landmarks expressed in the camera frame (i.e. $\mathcal{L}_G^C$) and current control point $\mathbf{q}$:

$$F_l(\mathbf{q}) = \sum_{\mathbf{l}_C \in \mathcal{L}_G^C} \prod_{j=0}^{5} o_j(\mathbf{q}, \mathbf{l}_C). \tag{21}$$

where o_j are the elements of

$$\mathbf{o}(\mathbf{q}, \mathbf{l}_C) = \begin{bmatrix} \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{TR} \times \mathbf{v}_{LR}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{TL} \times \mathbf{v}_{TR}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{LL} \times \mathbf{v}_{TL}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{LR} \times \mathbf{v}_{LL}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh(\mathbf{v}_{Z} \cdot \mathbf{l}_C)\Big) \end{bmatrix}. \tag{22}$$

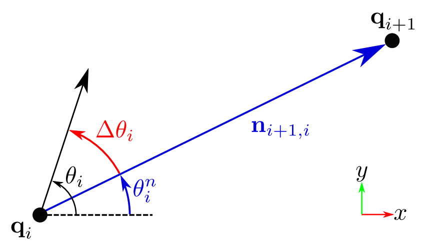


Fig. 5: Terms for the computation of the co-visibility cost. For two consecutive control points $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$, we compute the angular difference $\Delta\theta_i$ between the orientation θ_i in $\mathbf{q}_i$ and the direction vector $\mathbf{n}_{i+1,i}$.

The final cost for the spline optimization corresponds to

$$\mathcal{F}_l = -\sum_{i=K}^{N-K} F_l(\mathbf{q}_i), \tag{23}$$

maximizing the number of the tracked high-quality landmarks.

The last cost function $\mathcal{F}_v$ is associated to the co-visibility between consecutive control points in the B-Spline. The purpose is to add a soft constraint on the orientations of the robot, in order to keep them aligned with the direction of travel. For a two consecutive control points $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$, we compute the orientation vector in the xy plane as

$$\mathbf{n}_{i+1,i} = \begin{bmatrix} x_{i+1} - x_i & y_{i+1} - y_i \end{bmatrix}^T \in \mathbb{R}^2. \tag{24}$$

We do not use the z coordinates, as the orientation is expressed as a rotation around the Z-axis; therefore, the z component does not have any effect on the cost. As shown in Fig. 5, the orientation of $\mathbf{n}_{i+1,i}$ is $\theta_i^n = \theta_i^n(\mathbf{q}_i, \mathbf{q}_{i+1}) = atan2(n_{i+1,i}^y, n_{i+1,i}^x)$, where the superscript indicates the corresponding component in the vector. For the control point $\mathbf{q}_i$, we can compute the absolute difference in orientation with respect to the direction vector $\mathbf{n}_{i+1,i}$ as $\Delta\theta_i := |\theta_i - \theta_i^n|$. The cost for a couple of two consecutive control points is

$$F_v(\mathbf{q}_i, \mathbf{q}_{i+1}) = \begin{cases} (\Delta\theta_i - \Delta\theta^*)^2 & \text{if } \Delta\theta_i \geq \Delta\theta^* \\ 0 & \text{otherwise} \end{cases}, \tag{25}$$

where $\Delta\theta^*$ is the maximum allowed angular deviation. Finally, the cost associated to the co-visibility is

$$\mathcal{F}_v = \sum_{i=K}^{N-K-1} F_v(\mathbf{q}_i, \mathbf{q}_{i+1}). \tag{26}$$

This cost tries to translate both the control points, but for a given $\mathbf{q}_i$, the successive $\mathbf{q}_{i+1}$ is not subject to changes in orientation, as the gradient of $\mathcal{F}_v$ with respect to θ_{i+1} is zero.

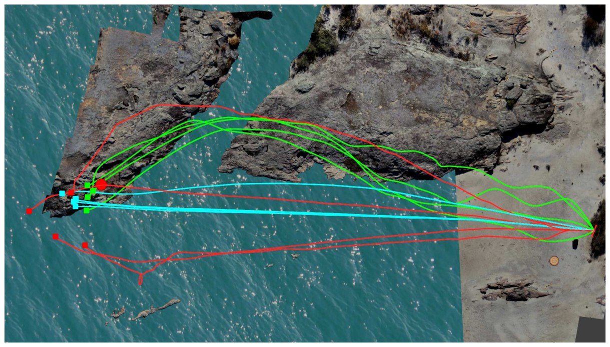
Moreover, notice that the costs $\mathcal{F}_l$ and $\mathcal{F}_v$ are in competition with each other since the former encourages to look in the direction of the well-localized landmarks, while the latter aligns the robot orientation with the direction of travel.

VI. Experiments

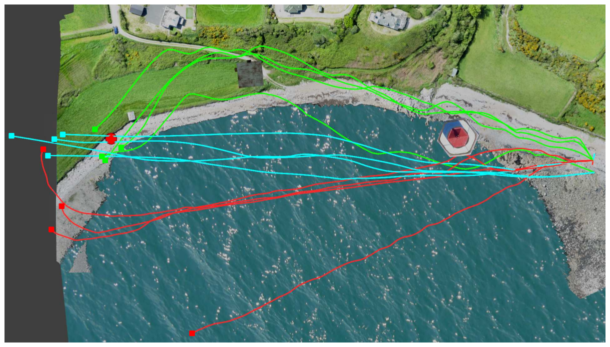
A series of challenging experiments in photo-realistic simulations using the Gazebo and RotorS frameworks [28]

Experiment	Dimensions	Ours	Reactive	[3]
Rocks	$30m \times 50m$	**1.46 m**	2.98 m	4.96 m
Bay	$40m \times 55m$	**1.62 m**	6.58 m	7.28 m
Long Beach	$100m \times 190m$	**6.05 m**	16.67 m	9.24 m

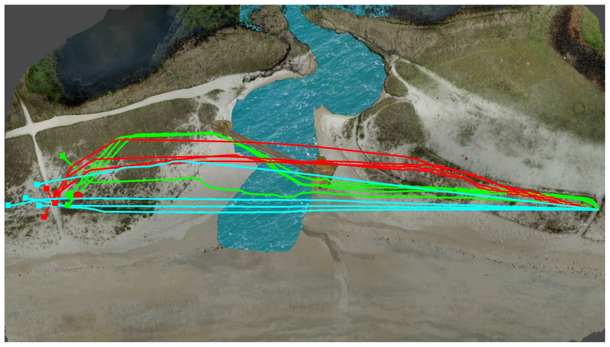
TABLE I: Results of the experiments in the three simulations. We report the average missed distance, i.e. the distance between the final position of the UAV and the real goal position.



(a) Top-view of the experiment *Rocks*. Only our planner (green) is able to consistently fly the UAV on top of the coast and reach the goal.



(b) Top-view of the experiment *Bay*. Despite the presence of the obstacle, our planner (green) is able to steer the robot towards more informative areas. In this experiment, [3] (red) performs worse than the reactive planner (cyan), causing a complete failure case, since it follows the low-quality landmarks extracted on the water surface. Moreover, both the reactive planner and [3] fly the robot over the boundaries of the model.



(c) Top-view of the experiment *Long Beach*. Our planner (green) is able to steer away from dangerous areas (e.g. lake), by flying close to texture-full structures, such as the bridge in the middle of the model.

Fig. 6: Top-views of the experiments in photo-realistic simulations. The outputs of the proposed planner are depicted in green, while the paths in cyan and in red correspond to a purely reactive planning and the perception-aware planner proposed in [3], respectively. The goal position is depicted as a red blob and the end positions of the trajectories are indicated with a square.

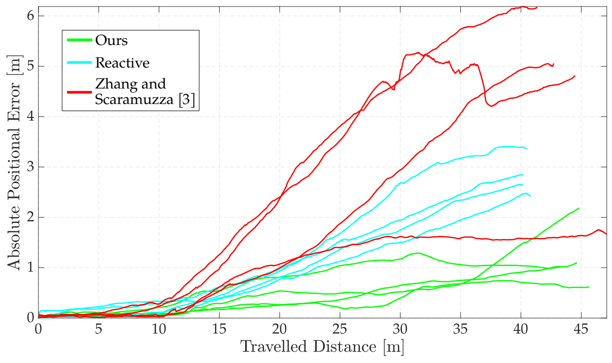
were conducted to show the effectiveness of the proposed method. The simulated UAV is an AscTec Firefly with a front-looking camera mounted with a pitch angle of 30°. Differently from the experiments reported in [3], the simulation environments do not present texture-less areas; instead, some parts of the models were specifically built to create hard challenges for camera-based state estimators, such as water with moving waves. We show that our planner is able to steer the robot away from these problematic areas, enhancing the performance of the VIO system.

We tested three different models[1], shown in Fig. 6. In each scenario, we commanded the UAV to fly to a predefined destination 4 times. The input goal is initially expressed in the reference frame of the simulator and then is transformed into the local reference frame of the robot. Notice that the initial positions of the robot can slightly vary between different runs of the same experiment, due to the initialization procedure of the VIO and MSF modules.

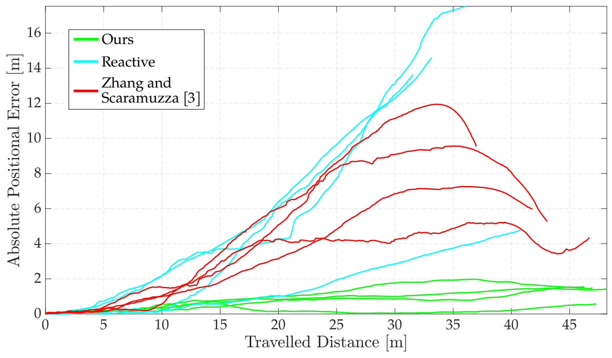
In order to evaluate the approach, we use the state estimation error as a metric, especially because the UAV did not crash in any obstacles in any experiment, despite the high error in localization. This is an advantage of using local information (e.g. local occupancy maps) for obstacle avoidance. For each experiment, we report the missed distance to the goal, i.e. the distance between the end position of the robot and the real position of the goal. The results are reported in Table I. In Fig. 7 we plot the absolute error in position with respect to travelled distance, where the error is computed as the absolute difference from the estimated position of the UAV with respect to the ground truth.
We compare the performance of our method with a purely reactive navigation approach and the perception-aware planner proposed by Zhang and Scaramuzza [3]. The reactive planner is based on our proposed planner without considering the perception costs, i.e. setting w_M in Eq. (3) and the weights λ_l and λ_v in Eq. (10) to zero.
The experiments are visualized in the accompanying video.

The first scene *Rocks* (Fig. 6a) consists of a rocky coast cut in between by a medium-size river. The left half of the model contains just water, which is moving and reflecting light. As there are no obstacles blocking the path, we observe that the purely reactive planner computes mostly straight line trajectories to the goal to the other end of the scene. However, these fly the robot in areas of low-quality perception, resulting in large state estimation errors. The perception-aware planner of [3] is also not able to steer the robot towards more informative areas (e.g. the coast), since it uses all the available landmarks extracted from VINS-Mono and does not filter the high-quality from the low-quality ones. It even performs worse than the reactive planner, since it tries to keep track of the landmarks extracted on the water surface, causing failures in the estimation. On the contrary, our planner is able to use information on more informative areas, encouraging the navigation along the coast. This leads to a noticeable improvement in accuracy with respect to

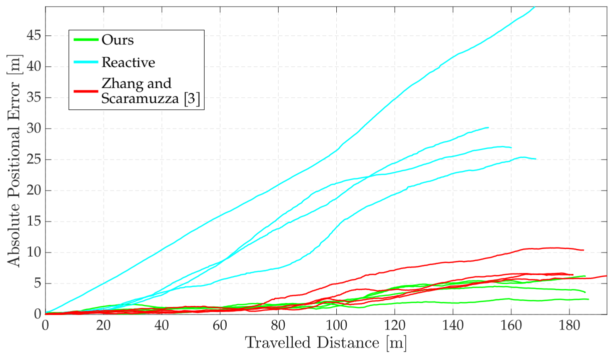
[1]https://sketchfab.com/3d-models



(a) Plot of the absolute positional error in the *Rocks* experiment.



(b) Plot of the absolute positional error in the *Bay* experiment.



(c) Plot of the absolute positional error in the *Long Beach* experiment.

Fig. 7: Absolute error in position of the state estimate with respect to increasing travelling distance.

the other two planners, at the expense of slightly higher trajectory lengths, as shown in Fig. 7a.

Similarly, the second scene *Bay* contains moving water in the middle, which lies between the starting point and the destination. A top view is shown in Fig. 6b. Moreover, close to the initial position of the robot, we place a large obstacle (lighthouse), which we label as useful for localization purposes (Fig. 1). The aim of this experiment is to show both the obstacle avoidance capability of the planners and the impact obstacles have on the perception cost. The lighthouse acts as an attractor for the perception cost, as it is rich in high-quality features. On the other hand, it also works as a repeller from the obstacle avoidance perspective. It is noticeable how our planner is able to satisfy the perception constraint while avoiding the lighthouse, by flying to its right along the coast. The other methods are also able to avoid collisions, but the obstacle pushes the trajectories towards low-quality parts of the scene, causing large drift and, consequently, higher missed distances.

The last experiment *Long Beach* (Fig. 6c) presents a set of dangerous areas, such as water and almost texture-less desert-like parts. In the middle of the model, we place a bridge of top of water, which can be utilized for localization during navigation. Despite that our planner does not navigate the robot on top of it, the drift in the estimation remains limited because our planning strategy encourages the robot to face towards the landmarks extracted on the structure of the bridge. While the path-search step initializes the trajectory next to this structure, the trajectory optimizer enhances the triangulation and tracking of the high-quality landmarks. In this experiment, [3] performs better than the reactive planner, because it steers the robot towards textured areas, while the purely reactive strategy flies the robot on top of the desert-like part of the scene. Thus, [3] improves the localization accuracy by avoiding texture-less areas, but it shows worse performance in comparison to our approach.

Overall, our method performs better than purely reactive planning and the perception-aware planner [3], as in all the experiments, the missed distance to the goal is significantly reduced, at the expense of slightly bigger trajectory length. Moreover, as shown in Fig. 7, the estimation error in position remains bounded, showing a noticeable increase in performance with respect to our competitors.

VII. Conclusion

In this paper, we propose a perception-aware path-planning architecture for goal reaching tasks. Our approach pushes the boundaries of the state of the art by incorporating the semantic labels of the 3D landmarks tracked by a VIO system into the path-planning instead of using only the existence of landmarks in certain areas as information.

This design allows us to navigate through areas with harder perceptual conditions. We demonstrate that we can reach the assigned goals with limited error and drift, showing noticeable improvements in performance with respect to purely reactive planning and the latest available solution in state-of-the-art of active perception.

Future directions include investigating the incorporation of the uncertainty of labels in semantic segmentation and feature classifiers when deployed in aerial vehicles. Another direction is to extend this pipeline to work with emergency landing spots detection, where the robot should also favourite to stay at a close distance from them.

References

[1] L. Teixeira, I. Alzugaray, and M. Chli, "Autonomous Aerial Inspection using Visual-Inertial Robust Localization and Mapping," in *Field and Service Robotics*. Springer, 2018.

[2] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza, "Exploiting Photometric Information for Planning Under Uncertainty," in *Robotics Research: Volume 1*. Springer, 2018.

[3] Z. Zhang and D. Scaramuzza, "Perception-aware Receding Horizon Navigation for MAVs," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[4] L. Steccanella, D. D. Bloisi, A. Castellini, and A. Farinelli, "Waterline and obstacle detection in images from low-cost autonomous boats for environmental monitoring," *Robotics and Autonomous Systems (RAS)*, 2020.

[5] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, 2018.

[6] A. Bry and N. Roy, "Rapidly-exploring Random Belief Trees for Motion Planning under Uncertainty," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[7] B. Mu, L. Paull, A.-a. Agha-mohammadi, J. Leonard, and J. How, "Information-based Active SLAM via Topological Feature Graphs," in *IEEE 55th Conference on Decision and Control (CDC)*, 2016.

[8] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, "Revisiting Active Perception," in *Autonomous Robots*. Springer, 2018.

[9] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active Vision," *International Journal of Computer Vision (IJCV)*, 1988.

[10] A. J. Davison and D. W. Murray, "Simultaneous Localization and Map-building using Active Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2002.

[11] N. Atanasov, "Active Information Acquisition With Mobile Robots," Ph.D. dissertation, University of Pennsylvania, 2015.

[12] A.-m. Ali-Akbar, C. Suman, and M. A. Nancy, "FIRM: Sampling-based Feedback Motion-planning under Motion Uncertainty and Imperfect Measurements," *The International Journal of Robotics Research (IJRR)*, 2014.

[13] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and Acting in Partially Observable Stochastic Domains," *Journal of Artificial Intelligence (JAI)*, 1998.

[14] V. Indelman, L. Carlone, and F. Dellaert, "Towards Planning in Generalized Belief Space," in *Robotics Research: The 16th International Symposium ISRR*. Springer, 2016.

[15] S. Prentice and N. Roy, "The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance," *The International Journal of Robotics Research (IJRR)*, 2009.

[16] R. Sim and N. Roy, "Global A-Optimal Robot Exploration in SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[17] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[18] M. Achtelik, S. Lynen, S. Weiss, M. Chli, and R. Siegwart, "Motion- and Uncertainty-aware Path Planning for Micro Aerial Vehicles," *Journal of Field Robotics (JFR)*, 2014.

[19] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight," *IEEE Robotics and Automation Letters (R-AL)*, 2019.

[20] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart, "A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[21] 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," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[22] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics (TRO)*, 2018.

[23] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A Robust and Modular Multi-sensor Fusion Approach applied to MAV Navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[24] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments," *The International Journal of Robotics Research (IJRR)*, 2010.

[25] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[26] Zhijie Zhu, E. Schmerling, and M. Pavone, "A convex optimization approach to smooth trajectories for motion planning with car-like robots," in *54th IEEE Conference on Decision and Control (CDC)*, 2015.

[27] V. Murali, I. Spasojevic, W. Guerra, and S. Karaman, "Perception-aware trajectory generation for aggressive quadrotor flight using differential flatness," in *American Control Conference (ACC)*, 2019.

[28] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "RotorS – A Modular Gazebo MAV Simulator Framework," in *Studies in Computational Intelligence*, 2016.