

# Localization-aware Trajectory Planning on $SE(3)$ for Intravehicular Robots

Caroline Specht<sup>1,2</sup>, Tommaso Faraci<sup>1,3,5</sup>, Victor Kowalski Martins<sup>4,5</sup>, and Roberto Lampariello<sup>2</sup>

**Abstract**—Visual-based localization paradigms for autonomous robots depend on a reliable presence and detectability of landmarks in the robot’s surroundings. Without this information, localization will fail, which can result in unpredictable robot behavior as it attempts to re-establish its self-awareness. Localization-aware motion planning attempts to minimize the possibility of unsuccessful feature detection by maximizing the number of features in view. In this paper, a differentiable perception-based objective function is developed to enable an efficient gradient-based optimal control approach for localization-aware optimal trajectory planning in  $SE(3)$ . The differentiable metric specifically describes the density of features in view of the robot camera, based on a given feature map of the robot’s working environment. The optimal control algorithm is embedded into the RRT\*-GBO algorithm to provide a global search capability, thus efficiently handling local minima. Results are presented in simulation for an Astrobee robot traversing the Japanese Experiment Module on the International Space Station, using real sparse maps of the same module provided by NASA.

## I. INTRODUCTION

Robotic maneuvers require careful and exhaustive planning - even more so under complex and safety-critical conditions. Trajectory planning for a given robot should take into account not only its kinematic and dynamic capabilities, but also how available sensors can be effectively utilized.

Visual-based localization paradigms for autonomous robots utilize cameras which rely on the presence of reliable and easily detectable landmarks in the robot’s surroundings [1]. Several popular methodologies exist – including Astroloc [2], which is implemented on the Astrobees on the International Space Station (ISS) and was specifically designed to mitigate the gravity-dependence of visual-inertial localization methods [3]. It was noted by several Astrobee Guest Science campaigns, e.g. [4], [5], and [6], that successful localization was impacted by visibility of features - despite the use of highly populated *a priori* landmark maps of the environment. Fig. 1 demonstrates the feature extraction process in the Astrobee simulation environment, where successful feature matching along the trajectory leads to a highly-connected factor graph of poses. The presence of recognizable landmarks in view allows optimization of the entire trajectory, hence reducing the estimation error.

1. Caroline Specht and Tommaso Faraci have contributed equally to this paper. 2. Caroline Specht and Roberto Lampariello are with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Münchener Str. 20, 82234 Wessling, Germany, email: firstname.lastname@dlr.de. 3. Tommaso Faraci is with the Department of Informatics and Computer Science of the University of Trento: tommaso.faraci@unitn.it 4. Victor Kowalski Martins is at the Autonomous Systems Lab, Technische Universität Wien, email: victor.martins@tuwien.ac.at. 5. Tommaso Faraci and Victor Kowalski Martins participated in this work at DLR in partial fulfillment of their Master studies at the Technical University Munich.

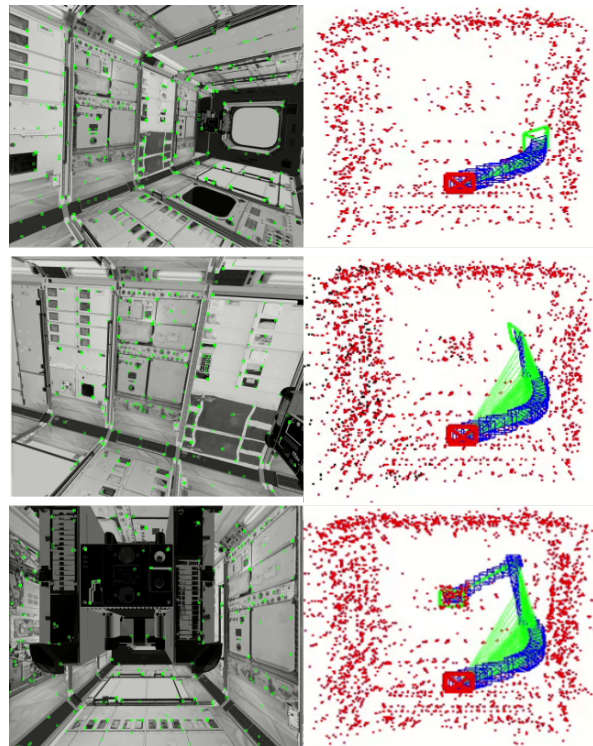


Fig. 1: The camera view of an Astrobee robot in a simulated Japanese Experiment Module of the ISS (left) as the camera frame (blue) moves along a trajectory in  $SE(3)$  (right). Successful feature matching and a dense factor graph are shown by green markers (left) and the green lines (right).

### A. Related work

It is well known that features must be visible, detectable, and correctly matched [7]–[11] for successful visual-based localization. This has important ramifications in real-world applications, including critical deterioration of performance - with two possible options for resolution: choose a different localization method (which may not be possible [3]) or generate motion plans which consider the distribution of features within the environment - with the logic that detection success increases with feature density [8] [10], and matching success increases with detection success [9] [11].

A multitude of trajectory planning methodologies exist in literature, including gradient-based [12]–[15] and sample-based [16] [17] methods. Gradient-based nonlinear optimization (GBO) is a common motion planning approach, but produces locally optimal results which are highly dependent on the initial guess. Sample-based methods evade the local optimality issue through a structured exploration of state space. An effort has been made recently to incorporate the advantages of optimization-based and sampling-based

methods [8] [10] [11] [18]. For example, RRT\*-GBO [18] leverages the gradient information of the cost and constraint functions, while making use of the global search capability.

In localization-aware motion planning (LA), a feature-influenced element is included into the objective function. For example, the feature density can be incorporated into the objective function to maximize the count of visible features along the planned trajectory. However, in the case of GBO methods, the objective function must be differentiable (of class  $C^2$ ). This eliminates the option of simple feature count and factor-graph based objectives or constraints, as they are discontinuous in nature.

The bulk of perception-based trajectory planning literature considers UAV applications. To incorporate the maximization of visible features into a differentiable objective function, [1] introduced the relaxed visibility function, with the goal of keeping features visible in the camera view from one keyframe to the next. The presented algorithm neglects the planning of the position trajectory and plans only the sequence of orientations along a pre-planned position trajectory. In [8], the same objective function was wrapped within an A\* motion planner to obtain an initial guess of the trajectory. In particular, [8] aimed to not only maintain a large number of landmarks in view, but also to only select areas where the features extracted are reliable. This initial guess trajectory was optimized by using a B-spline discretization with elastic-band cost functions. The cost accounts for smoothness and collision avoidance, as well as visibility of landmarks using the same relaxed function from [1]. Additionally, [8] introduces a co-visibility term, which penalizes large changes of direction of the camera over adjacent poses. The method in [10] likewise optimizes the yaw trajectory of the vehicle using a soft co-visibility constraint based on yaw primitive guidance along a position trajectory determined by an unrelated sample-based method. The algorithm presented in [11] introduces a safety corridor which is applied as a cost penalty to enforce that the generated position trajectory does not enter into collision with known stationary obstacles in the environment while keeping them within peripheral view of the on-board cameras.

### B. Contributions

While these methods tackle the difficult task of motion planning for agile, fast-moving, flying robots, they are computationally intensive online and only consider local optimality of the trajectory. This work shifts focus to on-orbit free-flying intravehicular robotic applications. Like the state-of-the-art methods, the foundation of the trajectory generation is based on a local optimal control problem. In contrast, the proposed LA objective function is a differentiable density function, developed for a given *a priori* map (the current operational standard on-board the ISS [3]), which allows for more computationally efficient online planning. As the resulting optimal control problem is non-convex, the local GBO problem is embedded in an RRT\*-based algorithm to provide global optimality. This is achieved through an extension of the RRT\*-GBO method developed in [18]. Unlike the state-of-the-art application to UAVs, the given

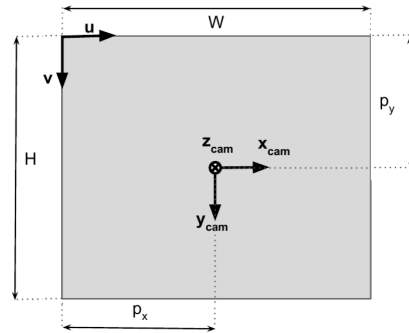


Fig. 2: Diagram of the camera frame and image plane.

application requires a further extension of the RRT\*-GBO framework to problems in the space of the special euclidean group  $SE(3)$ . This method is demonstrated in simulation using the free-flying Astrobees, see Fig. 1.

The remainder of this paper is structured as follows: In Sec. II, the preliminary aspects of the relaxed feature visibility metric relevant to later discussion are outlined. The motion planning problem is expressed in III. The proposed extensions to the sample-based RRT\*-GBO method are discussed in Sec. IV. The method is then demonstrated and discussed in Sec. VI. Conclusions are drawn in Sec. VII.

## II. RELAXED FEATURE VISIBILITY

The proposed LA motion planner requires a smooth description of where features exist in the environment to use in the objective function of the GBO, to be discussed in Sec. III. A simple feature count or factor-graph covariance analysis are out of the question, as these are only differentiable on a per landmark basis and only while the landmark is in view. The works [1] and [8] suggested an alternative and differentiable relaxed visibility function, which characterizes the visibility of a feature with a real number between 0 and 1 (where 1 is complete visibility). The remainder of this section describes this function, utilized in the following sections.

The basic pin-hole projection model is used to represent the camera mapping [19]. Fig. 2 illustrates the image plane relative to the camera frame.  $W$  and  $H$  are the width and height of the camera frame. The pixel in the top left corner defines the origin of the 2D camera frame, or principal point of the camera frame, with axes  $u$  and  $v$ . The camera frame is defined by  $\{x_{cam}, y_{cam}, z_{cam}\}$ , the origin of which is located at  $[p_x, p_y]$  relative to the principal point.

The transformation of feature coordinates from the world frame to the camera frame is expressed through the extrinsic matrix  $H_{cam}^W = [R_{cam}^W \ t_{cam}^W]$ , comprised respectively of the rotation from the world frame to the camera frame and the translation vector from the origin of the camera frame to the origin of the world frame.  $K$  is the intrinsic matrix of the camera,

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

where  $f$  is the camera focal length. The map of a point in the image plane can then be described by

$$u = f x_{cam} / z_{cam} + p_x$$

$$v = fy_{\text{cam}}/z_{\text{cam}} + p_y.$$

The limits of the image plane are defined by the width  $W$  and the height  $H$ .

The characterization of the visibility of a feature is then achieved by modeling the camera view frustum as vectors from the camera's optical center to the vertices of the image plane:

$$\begin{aligned} \mathbf{v}_{\text{tr}} &= [W - p_x, -p_y, f]^\top, & \mathbf{v}_{\text{tl}} &= [-c_x, -c_y, f]^\top \\ \mathbf{v}_{\text{lr}} &= [W - p_x, H - p_y, f]^\top, & \mathbf{v}_{\text{ll}} &= [-p_x, H - p_y, f]^\top \end{aligned}$$

and computing the visibility operator as [8]

$$\mathbf{o}(\mathbf{f}_{i,j}) = \begin{bmatrix} \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{tr}} \times \mathbf{v}_{\text{lr}}) \cdot \mathbf{f}_{i,j}}{b})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{tl}} \times \mathbf{v}_{\text{tr}}) \cdot \mathbf{f}_{i,j}}{b})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{ll}} \times \mathbf{v}_{\text{tl}}) \cdot \mathbf{f}_{i,j}}{b})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{lr}} \times \mathbf{v}_{\text{ll}}) \cdot \mathbf{f}_{i,j}}{b})) \\ \frac{1}{2}(1 + \tanh(\frac{\mathbf{e}_z \cdot \mathbf{f}_{i,j}}{b})) \end{bmatrix}, \quad (1)$$

where  $b$  is a tuning parameter which determines the ‘‘sharpness’’ of the function,  $\mathbf{e}_z = [0 \ 0 \ 1]^\top$  and  $\mathbf{f}_{i,j} = \mathbf{H}_{\text{cam},i}^W \mathbf{w}_j$  indicates the position  $\mathbf{w}$  of a feature  $j$  with respect to the camera frame at time  $t_i$ .

Finally, the relaxed visibility function is

$$h_{\text{RV}}(\mathbf{o}(\mathbf{f}_{i,j})) = \prod_{k=1}^5 (\mathbf{o}(\mathbf{f}_{i,j}))_k,$$

the summed feature visibility for a camera pose  $i$  is

$$\gamma_{\text{RV}}(\mathbf{H}_{\text{cam},i}^W) = \sum_{j=1}^{\mathfrak{n}_{\text{features}}} h_{\text{RV}}(\mathbf{o}(\mathbf{f}_{i,j})), \quad (2)$$

where  $\mathfrak{n}_{\text{features}}$  is the number of features, and the relaxed visibility objective function  $\Gamma_{\text{RV}}$  sums over  $\mathfrak{n}_t$  time steps to

$$\Gamma_{\text{RV}} = \sum_{i=1}^{\mathfrak{n}_t} \gamma_{\text{RV}}(\mathbf{H}_{\text{cam},i}^W). \quad (3)$$

### III. LOCALIZATION-AWARE GRADIENT-BASED OPTIMIZATION

Motion planning for a robot in  $SE(3)$  is achieved by exploiting the group's geometrical structure, by first recognizing that  $SE(3) = \mathbb{R}^3 \times SO(3)$ . Through this simple outer product, the translation and rotation are decoupled into a tuple  $(\mathbf{x}(t), \mathbb{R}(t))$ . The translation and orientation of the robot can then be expressed by six trajectories, one for each dimension, which can each be parameterized by an order-4 B-spline.

These trajectories are nonlinear with respect to the orientation parameters and are generally highly constrained for the application of interest. They can be planned from an initial robot pose to a final one in  $SE(3)$  using optimal control-based methods. Let the robot pose be described in  $SE(3)$  by  $\mathbf{s} = [\mathbf{x}^\top \ \boldsymbol{\xi}^\top]^\top = [x \ y \ z \ \xi_x \ \xi_y \ \xi_z]^\top$ , where  $x, y$ , and  $z$  are position and  $\boldsymbol{\xi} = \theta \mathbf{e} \in \mathbb{R}^3$  describe a rotation of angle  $\theta$  about arbitrary unit vector  $\mathbf{e}$ .

To this end, consider an optimal control problem which has been discretized in time  $t$  with  $\mathfrak{n}_t$  via points to produce the following nonlinear program (NLP):

$$\min_{\mathbf{p}_{\text{free}}} \Gamma(\mathbf{p}_{\text{free}}) \quad \text{s.t.} \quad \mathbf{h}(\mathbf{p}_{\text{free}}, t) \leq 0, \quad (4)$$

where  $\mathbf{p}_{\text{free}}$  indicates the composite  $6 \times \mathfrak{n}_{\text{free}}$  free parameters of the B-splines parameterizing the 6D trajectory,  $\Gamma$  is the objective function, and  $\mathbf{h}$  are the relevant motion constraints. By parameterizing the trajectory through order-4 B-splines, the position, velocity, acceleration, and jerk in each dimension are simultaneously considered, and the boundary conditions of the B-splines on  $k$  derivatives

$\mathbf{s}^{(k)}(t_0) = \mathbf{b}_{t_0,i}$ ,  $\mathbf{s}^{(k)}(t_f) = \mathbf{b}_{t_f,i}$ ,  $k = 0, \dots, 3$ ,  $i = 1, \dots, 6$  are implicitly satisfied.

The NLP is solved using the gradient-based second-order SQP method (see Sec. V-B). The optimality conditions of the latter require that the objective function  $\Gamma$  and constraint functions  $\mathbf{h}$  are twice continuously differentiable.

A common objective is to minimize the expended energy on the duration of a maneuver

$$\Gamma = \Gamma_{\text{energy}} = \sum_{i=1}^{\mathfrak{n}_t} \mathbf{u} \cdot \mathbf{v},$$

where  $\mathbf{u} = [\mathbf{F} \ \boldsymbol{\tau}]^\top \in \mathbb{R}^6$  is the robot actuation, comprised of forces  $\mathbf{F} \in \mathbb{R}^3$  and torques  $\boldsymbol{\tau} \in \mathbb{R}^3$ , and  $\mathbf{v} = [\dot{\mathbf{x}} \ \boldsymbol{\omega}] \in \mathbb{R}^6$  is the robot velocity  $\dot{\mathbf{x}} \in \mathbb{R}^3$  and angular velocity  $\boldsymbol{\omega} \in \mathbb{R}^3$ .

However, in order to generate trajectories which provide the robot with the best chances of correctly localizing itself within its environment, feature visibility is also taken into account through a perception-aware cost function  $\Gamma_{\text{perception}}$ . Given a sparse feature map of the environment,  $\Gamma_{\text{perception}}$  quantifies the features the robot's camera sees along a trajectory. One option for  $\Gamma_{\text{perception}}$  which satisfies the GBO differentiability requirements is  $\Gamma_{\text{RV}}$ , as presented in Sec. II. While differentiable, this objective function requires the visibility of every feature to be evaluated at each via point along the trajectory in each optimization iteration, resulting in a computationally expensive execution. An extension of this, the Summed Feature Visibility Interpolation function  $\Gamma_{\text{SFVI}}$ , is proposed here. A multi-dimensional interpolation of  $\Gamma_{\text{RV}}$  over all possible robot camera poses is conducted for a given feature map of the environment, allowing  $\Gamma_{\text{SFVI}}$  to move the bulk of this computation offline.

As the tuning parameter  $b$  tends to 0, (1) tends to a point in the camera frame. As demonstrated in [1], this results in sharp corners in the function, which are not suitable for use in gradient-based optimization. The work of [1] arbitrarily tuned  $b$  to be the average distance of features from the camera. In this work,  $b$  is chosen to normalize the frustum vectors, and (1) becomes

$$\mathbf{o}'(\mathbf{f}_{i,j}) = \begin{bmatrix} \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{tr}} \times \mathbf{v}_{\text{lr}}) \cdot \mathbf{f}_{i,j}}{|\mathbf{v}_{\text{tr}} \times \mathbf{v}_{\text{lr}}|})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{tl}} \times \mathbf{v}_{\text{tr}}) \cdot \mathbf{f}_{i,j}}{|\mathbf{v}_{\text{tl}} \times \mathbf{v}_{\text{tr}}|})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{ll}} \times \mathbf{v}_{\text{tl}}) \cdot \mathbf{f}_{i,j}}{|\mathbf{v}_{\text{ll}} \times \mathbf{v}_{\text{tl}}|})) \\ \frac{1}{2}(1 + \tanh(\frac{(\mathbf{v}_{\text{lr}} \times \mathbf{v}_{\text{ll}}) \cdot \mathbf{f}_{i,j}}{|\mathbf{v}_{\text{lr}} \times \mathbf{v}_{\text{ll}}|})) \\ \frac{1}{2}(1 + \tanh(\mathbf{e}_z \cdot \mathbf{f}_{i,j})) \end{bmatrix}. \quad (5)$$

Similarly to the application in [8], this adjustment emphasizes the co-visibility of features from one viapoint to the next. Unlike in [8], no component of  $\mathbf{w}$  is neglected.

The summed feature visibility function  $h_{RV}(\mathbf{o}'(\mathbf{f}_{i,j}))$  is evaluated for this modified visibility operator at each of  $m$  grid points on a six-dimensional grid of robot pose parameters. At each grid point, the summed feature visibility for the camera pose is

$$\gamma'_{RV}(\mathbf{s}) = \sum_{j=1}^{n_{\text{features}}} h_{RV}(\mathbf{o}'(\mathbf{f}_{i,j})).$$

The function is then interpolated using cubic splines between the grid points to generate a differentiable, approximated summed feature visibility function  $\hat{\gamma}'_{RV}(\mathbf{s})$ , defined for any pose in the robot's workspace.

Finally, the interpolated perception objective function is given as:

$$\Gamma_{\text{SFVI}} = \sum_{i=1}^{n_t} \hat{\gamma}'_{RV}(\mathbf{s}(t_i)),$$

where  $\hat{\gamma}'_{RV}(\mathbf{s}(t_i))$  is sampled from the interpolated function  $\hat{\gamma}'_{RV}$  for pose  $\mathbf{s}$  at time  $t_i$ . In effect, this function informs the optimizer of the density of visible features for a given sampled pose.

A weighted objective function permits a synergy of  $\Gamma_{\text{energy}}$  and  $\Gamma_{\text{perception}} = \Gamma_{\text{SFVI}}$ :

$$\begin{aligned} \Gamma_{\text{weighted}} & \quad (6) \\ & = w_{\text{energy}} \left( \frac{\Gamma_{\text{energy}}}{\Gamma_{\text{energy,max}}} \right) + (1 - w_{\text{energy}}) \left( 1 - \frac{\Gamma_{\text{SFVI}}}{\Gamma_{\text{SFVI,max}}} \right) \end{aligned}$$

where

$$\begin{aligned} 0 & \leq w_{\text{energy}} \leq 1, \\ \Gamma_{\text{energy,max}} & = n_t \mathbf{u}_{\text{max}} \cdot \mathbf{v}_{\text{max}}, \\ \Gamma_{\text{SFVI,max}} & = n_t n_{\text{features}}, \end{aligned}$$

where  $\mathbf{u}_{\text{max}} \in \mathbb{R}^6$  is the maximum actuation effort and  $\mathbf{v}_{\text{max}} \in \mathbb{R}^6$  is the maximum velocity and angular velocity achievable by the robot. The resolution of this GBO problem is prone to falling into local minima. However, when wrapped into the RRT\*-GBO algorithm, as described in the following section, a search of the solution space with globality properties works to mitigate this complication.

#### IV. RRT\*-GBO ON $SE(3)$

RRT\*-GBO on  $SE(3)$  derives from the work of [18]. This section outlines the RRT\*-GBO algorithm and describes how to extend it from three-dimensional translational space to include rotation in three dimensions.

##### A. RRT\*-GBO

There are two overbounding phases involved in trajectory generation using RRT\*-GBO: first, an RRT\*-tree is constructed, through the solutions of edge computations formulated as optimal control problems, to connect the starting condition to the goal condition. The resulting path constructed by these so-called ‘‘GBO edges’’ is then used as an initial guess for a final smoothing step: the edges are merged into one curve, which is then used as an initial guess for the original optimization problem.

The first phase is comprised of three steps, depicted in Fig. 3. *Steps (1)-(3)* occur iteratively until a successful

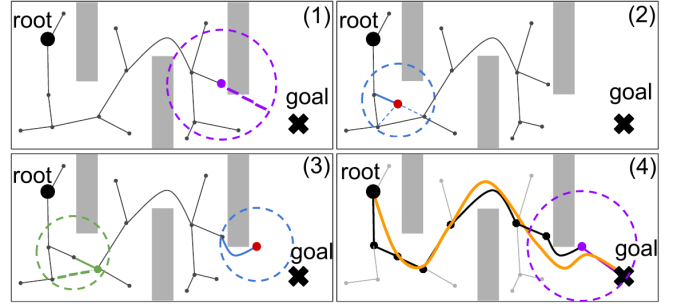


Fig. 3: RRT\*-GBO procedure: (1) Attempt to connect node (purple dot) to goal, provided that the latter is within  $r_{\text{connect}}$  (purple circle). (2) Sample a new node (red) and connect to the nearest node in the blue neighborhood. (3) Re-wire connections to new node (green). Iterate (1) and (2) until goal is reached (red dot and blue circle). (4) Smoothing of the complete trajectory.

connection to the Goal condition is made and certain other stopping criteria are met, such as a maximum number of solutions found or a maximum execution time. In *Step (1)*, a connection between the last added node and the goal node is attempted with a GBO edge. If this connection is successful, then the path is complete and the iteration can be exited to the second phase. Otherwise, the iteration progresses to *Step (2)*. *Step (2)* is the sampling of a new node in the search space. Then, for every existing node in a ‘‘neighborhood’’ of the sampled node (defined below), a GBO edge is constructed from the existing neighboring node to the sampled node. The successfully built edges are compared, and the cheapest one is chosen to attach the sampled node to the existing tree. In *Step (3)*, the newly attached node is used in a rewiring procedure to check if any nodes in the previously defined neighborhood should have their incoming edge replaced by an edge departing from the new node, in case that reduces the cost to get to the neighboring node.

Finally, in the second phase, or *Step (4)* in Fig.3, the GBO edges are connected together to provide an initial guess to the NLP to finally provide a smooth, optimal path to the Goal condition.

The algorithm contains some parameters that must be tuned:  $r_{\text{prune}}$ ,  $r_{\text{ball}}$ , and  $r_{\text{connect}}$ . The parameter  $r_{\text{prune}}$  is the required minimum distance between a newly sampled node and any other node in the existing tree. The sampled node is discarded if it is too close to another node, and a new node is sampled. The parameter  $r_{\text{ball}}$  has a dual function. First, a newly sampled node should be at most  $r_{\text{ball}}$  distance away from the node that is closest to it. If the sampled node is located further than  $r_{\text{ball}}$  away from the closest node, this sample should be discarded and a *steering step* must be conducted: The vector that connects the sampled node from its nearest neighbor is scaled to have a norm equal to  $r_{\text{ball}}$ , and a new sampled node is placed at the tip of this normalized vector. Second,  $r_{\text{ball}}$  dictates the size of the neighborhood within which to search for nodes to be wired to the sampled node. The parameter  $r_{\text{connect}}$  is the maximum distance that a node may be from the goal node such that an attempt to connect them is made. In [18], each of these requirements was evaluated using the Euclidean distance.

## B. Sampling on $SE(3)$

As the nodes of the proposed RRT\*-GBO are members of  $SE(3)$ , some adaptation of the algorithm is necessary. The sampling of nodes and parameter tuning must be handled considering both translational and rotational distances. For the translational part (in  $\mathbb{R}^3$ ), the Euclidean distance metric can be used:  $\phi_{\text{trans.},1 \rightarrow 2} = \|\mathbf{x}_2 - \mathbf{x}_1\|$ . The Geodesic on the unit sphere metric from [20] is chosen to represent the distance between two rotations ( $SO(3)$ ):

$$\phi_{\text{rot.},1 \rightarrow 2}(R_1, R_2) = \|\log(R_1, R_2)\| = \|\xi_{1 \rightarrow 2}\| = \theta_{1 \rightarrow 2},$$

where  $R_{(\cdot)}$  represented rotation matrices and  $\log(\cdot)$  is the matrix logarithmic mapping. The distance metric adopted in this work consists of the angle in the angle-axis representation of rotations.

As a consequence, the three tuning parameters  $r_{\text{ball}}$ ,  $r_{\text{prune}}$ , and  $r_{\text{connect}}$  become six – one for each in translation and rotation.

In the RRT\*-GBO algorithm, the generalized distance between two nodes as a single scalar value is needed. A unified distance metric is therefore necessary:

$$\phi_{\text{unified}}(\phi_{\text{trans.}}, \phi_{\text{rot.}}) = \frac{\phi_{\text{trans.}}}{r_{\text{ball,trans.}}} + \frac{\phi_{\text{rot.}}}{r_{\text{ball,rot.}}}.$$

For each node  $N_i, i = 1, \dots, n_{\text{nodes}}$ , the index of the closest node to the sampled node is computed as:

$$i_{\text{closest}} = \underset{i}{\operatorname{argmin}} \phi_{\text{unified}}(\phi_{\text{trans.},i \rightarrow \text{sampled}}, \phi_{\text{rot.},i \rightarrow \text{sampled}}).$$

The steering step becomes:

$$\mathbf{x}_{\text{sampled}} = \mathbf{x}_{\text{closest}} + r_{\text{ball,trans.}} \frac{\mathbf{x}_{\text{sampled}} - \mathbf{x}_{\text{closest}}}{\phi_{\text{trans.},\text{closest} \rightarrow \text{sampled}}}$$

$$\xi_{\text{sampled}} = \xi_{\text{closest}} + r_{\text{ball,rot.}} \frac{\xi_{\text{closest, sampled}}}{\phi_{\text{rot.},\text{closest} \rightarrow \text{sampled}}}$$

A sampled node is discarded if

$$\phi_{\text{unified}}(\phi_{\text{trans.},\text{closest} \rightarrow \text{sampled}}, \phi_{\text{rot.},\text{closest} \rightarrow \text{sampled}}) < r_{\text{prune,unified}}$$

where  $r_{\text{prune,unified}} = \phi_{\text{unified}}(r_{\text{prune,trans.}}, r_{\text{prune,rot.}})$ , and a node  $N_i$  belongs to the neighborhood of the sampled node if

$$\phi_{\text{trans.},i \rightarrow \text{sampled}} \leq r_{\text{ball,trans.}},$$

$$\phi_{\text{rot.},i \rightarrow \text{sampled}} \leq r_{\text{ball,rot.}},$$

$$\phi_{\text{unified}}(\phi_{\text{trans.},i \rightarrow \text{sampled}}, \phi_{\text{rot.},i \rightarrow \text{sampled}}) \geq r_{\text{prune,unified}}.$$

Finally, an attempt to build an edge between the sampled node  $N_{\text{sampled}}$  and the goal node  $N_{\text{goal}}$  is made if

$$\phi_{\text{trans.},\text{sampled} \rightarrow \text{goal}} \leq r_{\text{connect,trans.}},$$

$$\phi_{\text{rot.},\text{sampled} \rightarrow \text{goal}} \leq r_{\text{connect,rot.}}.$$

The rates  $\dot{\mathbf{x}}$  and  $\dot{\xi}$ , defining the velocity terms of the sampled node, are randomly sampled within the constraint bounds used within the GBO, ensuring that the optimization views the boundary constraint stipulated by this new node as feasible. A dynamic weight is applied, which reduces the sampling window proportionally to the elapsed search time. The accelerations and jerks are set to zero to ease sample mismatch.

## C. Edge creation

Each GBO edge is parameterized as a set of order-4 B-splines  $s_{\text{edge}}$ , describing the  $SE(3)$  pose of the robot in time, with  $n_{\text{free,edge}}$  free B-spline vertices  $\mathbf{p}_{\text{free,edge}}$  and duration  $\Delta t_{\text{edge}} < \Delta t_{\text{maneuver}}$ . The initial and final boundary condition poses of the edge B-splines comprise the translation and rotation components of the parent and child nodes, respectively. The B-splines defining the edge are set using the GBO discussed in Sec. III, where the optimization parameters are  $\mathbf{p}_{\text{free,edge}}$  and the objective function is (6).

## D. Smoothing

The smoothing step compiles the optimal path of GBO edges  $s_{\text{edge}}^j, j = 1, \dots, n_{\text{edges}}$  from the Root to the Goal conditions into a single, smooth, optimal B-spline. First, the duration  $\Delta t_{\text{edge}}$  of each edge B-spline must be scaled by a factor  $\lambda$  such that their summed duration matches the chosen duration  $\Delta t_{\text{maneuver}}$  of the full trajectory B-spline:

$$\lambda = \frac{\Delta t}{n_{\text{edges}} \Delta t_{\text{edge}}}$$

Then, the sequence of edge B-splines is sampled at the sampling times  $t_i, i = 1, \dots, n_t$  of the full trajectory B-spline. This requires identifying which edge B-spline  $s_j(t)$  spans the time  $t_i$  and sampling this B-spline at its "local" time  $t_i^j = t_i - \lambda \Delta t_{\text{edge}}(j-1)$ . The samples are stored in a matrix  $S \in \mathbb{R}^{6 \times n_t}$ . The full trajectory B-spline is then fit to the edge B-splines samples using a least-squares optimization. The resulting free control vertices  $\mathbf{p}_{\text{free,LS}}$  are then used as an initial guess to the GBO problem (4) with the weighted objective function  $\Gamma_{\text{weighted}}$ . The result of this optimization is a set of free control vertices  $\mathbf{p}_{\text{free,smoothed}}$  which define the final, smoothed trajectory.

## V. SPACE ROBOT RENDEZVOUS

The scenario under consideration in the following section for the demonstration of perception-aware RRT\*-GBO on  $SE(3)$  is based on the rendezvous maneuver described in [4] and [15], and is reported below. This section first describes the simulation environment and parameters in which the experiments are conducted to produce the results described in Sec. VI. The NLP and the tuning parameters to be used in an RRT\*-GBO search for suitable trajectories is then outlined.

### A. Simulation Scenario

This scenario involves the rendezvous of a free flying chaser spacecraft with a target spacecraft - in this case the Astrobees free-flying robotic testbed on the ISS. The chaser robot is brought from a known initial pose relative to the target robot to a given final pose - or mating pose (MP) - while satisfying motion and collision constraints and while maximizing feature visibility on the walls of the ISS. The MP is located on the far side of the target robot relative to the initial pose of the chaser robot (see Fig. 6).

Along with open-source models of the Astrobees [21], mapping data of the ISS [22], and base flight-software [21], a custom Gazebo Simulation [23] permits generated trajectories for navigating the Columbus Module and the Japanese

Experiment Module (JEM) to be analyzed in simulation. In order to simulate the unreliability of visual-based feature matching, the map data was downsampled.

The Astrobbee robots have a mass of 9.58 kg and an inertia tensor  $I = \text{diag}(0.153, 0.143, 0.162)$  kg m<sup>2</sup> [24] [21]. It is sufficient to consider Newton-Euler dynamics [25] [15]:

$$\begin{aligned} \mathbf{F} &= m\ddot{\mathbf{x}} \\ \boldsymbol{\tau} &= I\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times I\boldsymbol{\omega}. \end{aligned}$$

The chaser robot motion is constrained here such that

$$[-2.7 \quad -0.75 \quad 0.6]^\top \text{m} \leq \mathbf{x}_c \leq [1.6 \quad 0.75 \quad 2.1]^\top \text{m} \quad (7)$$

$$|\dot{\mathbf{x}}_c| \leq [0.1 \quad 0.1 \quad 0.1]^\top \text{m/s} \quad (8)$$

$$|\boldsymbol{\omega}_c| \leq [0.1 \quad 0.1 \quad 0.1]^\top \text{rad/s} \quad (9)$$

$$|\mathbf{F}_c| \leq [0.849 \quad 0.406 \quad 0.486]^\top \text{N} \quad (10)$$

$$|\boldsymbol{\tau}_c| \leq [0.0849 \quad 0.0406 \quad 0.0486]^\top \text{Nm}. \quad (11)$$

In this simulation, the chaser and target robots each possess a collision geometry, respectively  $\mathbb{C}_c$  and  $\mathbb{C}_t$ . Each consists of a sphere large enough to overbound their  $0.32 \times 0.32 \times 0.32$  m cubic hull.

The simulation makes use of a pin-hole type camera, with parameters - introduced in Sec. II - reflecting those of the Astrobbee robots [21]. The transform from the camera frame to the body frame is given by

$$\mathbf{T}_{\text{COM}}^{\text{cam}} = \begin{bmatrix} 0 & 0 & 1 & 0.1177 \\ -1 & 0 & 0 & -0.0422 \\ 0 & -1 & 0 & -0.08260 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (12)$$

and the extrinsic matrix is given by

$$\begin{bmatrix} & \mathbf{H}_{\text{cam}}^W & & \\ 0 & 0 & 0 & 1 \end{bmatrix} = (\mathbf{T}_{\text{COM}}^{\text{cam}})^{-1} (\mathbf{T}_W^{\text{COM}})^{-1}. \quad (13)$$

### B. Formulation of RRT\*-GBO in SE(3)

RRT\*-GBO is used to generate trajectories with the parameters given in Table I.

The NLP considered in the GBO step of the RRT\*-GBO search is based on (4), with

$$\begin{aligned} \min_{\mathbf{p}_{\text{free}}} & \Gamma_{\text{weighted}}(\mathbf{p}_{\text{free}}) \\ \text{s.t.} & (7), (8), (9), (10), (11), \\ & c_{\text{collision}}(\mathbb{C}_c, \mathbb{C}_t, \mathbf{s}_c, \mathbf{s}_t) \leq 0, \end{aligned} \quad (14)$$

where  $c_{\text{collision}}$  is implemented as a penetration depth constraint [15], and  $\mathbf{s}_c$  and  $\mathbf{s}_t$  respectively represent the pose of the chaser and target robots. This NLP is solved using the NLOpt implementation of the *slsqp* algorithm [26]. The feature visibility function for use in the  $\Gamma_{\text{SFVI}}$  term of  $\Gamma_{\text{weighted}}$  has been interpolated using cubic splines, with the parameters outlined in Table II.

TABLE I: RRT\*-GBO parameters

Parameter	Value	Parameter	Value
$\Gamma_{\text{ball,trans.}}$	2	$\Gamma_{\text{prune,trans.}}$	0.5
$\Gamma_{\text{ball,rot.}}$	$\pi/2$	$\Gamma_{\text{prune,rot.}}$	$\pi/12$
$\Gamma_{\text{connect,trans.}}$	2	$\Delta t_{\text{edge}}$	60 s
$\Gamma_{\text{connect,rot.}}$	$\pi$		

## VI. RESULTS

In this section, the methods defined in the previous sections are demonstrated, highlighting the ability of RRT\*-GBO to perform motion planning on SE(3) and handle complex motion constraints, nonlinear dynamics, and non-convex cost formulations such as LA perception-based cost objectives. Unless otherwise indicated, the scenario of Sec. V is employed.

### A. Perception-Aware Trajectory Optimization in SE(3)

Solutions for the point-to-point motion traversing the JEM module longitudinally are analyzed. Fig. 4 illustrates how solving the NLP in (14) delivers trajectories which steer the camera toward areas of higher feature density, keeping highly textured areas in the FOV of the camera. Fig. 4a visualizes the interpolated perception objective function (3D surface), relative to the features on the  $y-z$  plane of the JEM (i.e., assuming that the camera has a clear view of one wall of the JEM). 2D trajectories in the  $y-z$  plane are projected on to this 3D surface, demonstrating the tendency of the LA objective function to move towards the local maximum of visible feature density. Fig. 4b illustrates these maneuvers in the same  $y-z$  plane along with the landmark features on this wall.

The antagonism between the two terms in the weighted cost function is made evident. In each maneuver illustrated in Fig. 4, the robot begins at Start, facing the Goal position. The Goal pose is appropriately defined in the boundary conditions of the six B-splines. The mechanical energy-based objective ( $w_{\text{energy}} = 1$ ) results in a straight line motion, with no rotation, as expected. In the combined results ( $w_{\text{energy}} = 0.6$  in Figs. 4a and 4b and  $w_{\text{energy}} = 0.4$  in Fig. 4a), the robot ventures into the region of highest feature density, rotating to face the wall and then to the Goal orientation at the end of the maneuver. The motion remains smooth - balancing the objectives according to their weights. The feature-based objective ( $w_{\text{energy}} = 0$ ) considers only the LA term of  $\Gamma_{\text{weighted}}$ . This objective does not smooth sharp motion, caring only to maximize the density of the features in view.

Localization accuracy is next evaluated, comparing the perception-aware trajectory and perception agnostic, energy optimal trajectory against the ground truth in terms of RMSE. Two methods are used to evaluate generated trajectories. First, SLAM is used to estimate the robot's trajectory in the JEM environment. ORB-SLAM3 [27] is selected, for its robustness and SE(3) optimization built-in capability. The RMSE for the energy-optimal and LA trajectories are, respectively, 8.2946 cm and 5.2331 cm. Inclusion of the LA term in (6) therefore provides in an overall improvement of

TABLE II: Visibility function interpolation parameters

	Linearly spaced grid points	Range
$x$ [m]	20	$[-3, 3]$
$y$ [m]	5	$[-0.75, 0.75]$
$z$ [m]	5	$[0.6, 2.1]$
$\xi_x$ [rad]	19	$[0, \pi]$
$\xi_y$ [rad]	19	$[0, \pi]$
$\xi_z$ [rad]	10	$[0, \pi]$

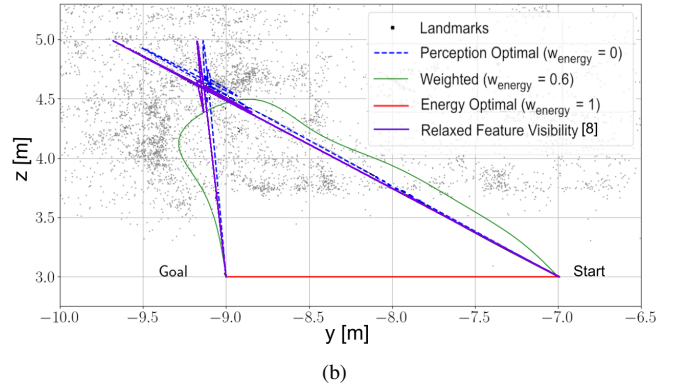
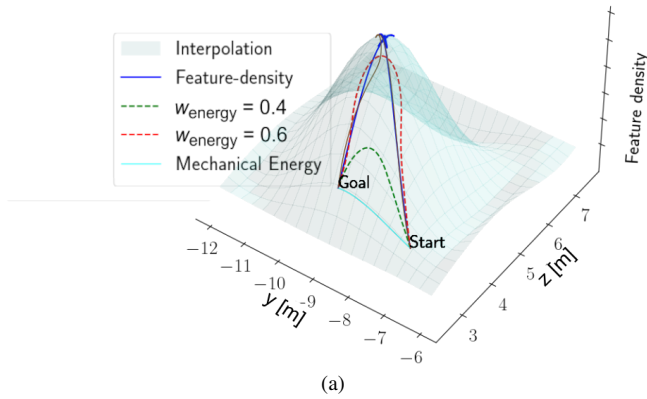


Fig. 4: A visualization of point-to-point trajectories in the  $y - z$  plane of the JEM module (see Fig. 7), using various weights  $w_{\text{energy}}$  in the weighted cost function, with (a) the interpolated feature density function (translucent surface), and (b) comparing 2D projections of the proposed and the state-of-the-art objective functions.

36.9% along this trajectory. A simulated AstroLoc pipeline is next employed. Unlike SLAM, Astrobees native onboard localization pipeline relies solely on pre-computed maps which are not updated online to perform visual-inertial localization. The presented method has been demonstrated to decrease the RMSE by a factor of  $10^{-2}$  when executing LA trajectories.

### B. Comparison to the State-of-the-Art

A performance comparison is made to the existing methods presented in [1] and [8]. However, sample-based perception-aware kinodynamic motion planning on  $SE(3)$  is a novel approach, and the methods of [1] and [8] were extended to permit planning in  $\mathbb{R}^4$ . To enable a fair comparison, these otherwise state-of-the-art methods were implemented on  $SE(3)$  to fit within the framework of (14). Only the GBO step, using the weighted objective function, is considered in this section.

Using the relaxed visibility objective function of [1] and [8] achieves a very similar result to using  $\Gamma_{\text{SFVI}}$ . However, due to the large amount of landmarks in the map, the method of [1] and [8] requires significantly longer to converge – on the order of  $10^3$  s – while the interpolated density function never exceeds 100 s. The comparison to [8] is illustrated in Fig. 4b.

### C. RRT\*-GBO on $SE(3)$

To demonstrate the efficacy of the algorithm presented in Sec. IV, it was challenged to find a feasible trajectory through “the Maze”, a cluttered environment where the robot needs to traverse a narrow passage to reach the goal. The RRT\*-GBO algorithm efficiently explores the environment along 19 edges and finds an optimal solution in 1400 seconds. Once the goal is reached, the trajectory is smoothed and the optimal solution to the point-to-point motion problem depicted in Fig. 5 is delivered. The algorithm showcases its ability to account for nonlinear constraints, as it guarantees obstacle avoidance by concurrently affecting translational and rotational motion of the body.

### D. Perception-Aware RRT\*-GBO

Finally, RRT\*-GBO on  $SE(3)$  is applied to the scenario described in Sec. V using the weighted objective function

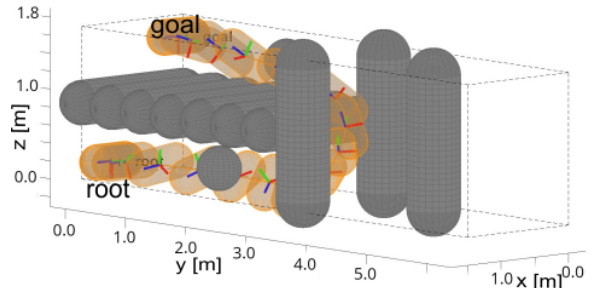


Fig. 5: Visualization of the trajectory planned using RRT\*-GBO in  $SE(3)$  to navigate a cluttered environment. The brown capsule navigates around the gray obstacles to move from root to goal. Note: it is necessary to plan in  $SE(3)$  so that the capsule’s orientation – indicated by the tri-color axes – permits a collision-free trajectory.

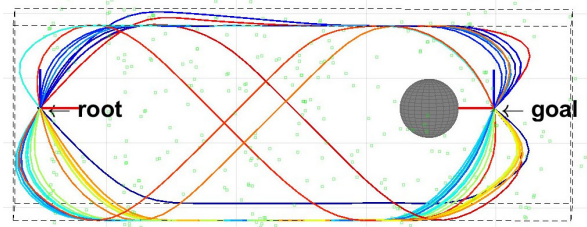


Fig. 6: Tree of solutions for the rendezvous task generated using perception-aware RRT\*-GBO. A given trajectory’s color reflects its cost – with lowest cost in dark blue to highest cost in red. The presence of multiple local minima is shown.

for 3D motion in the JEM. The resulting RRT\*-GBO tree is presented in Fig. 6. The color of the trajectory indicates the relative cost (blue is lowest cost, red is highest cost). There are clear clusters of solutions falling into local minima. The best solution is chosen and smoothed. The resulting motion is simulated, see Fig. 7, and the onboard camera feed and estimated trajectory snapshots are shown in Fig. 1.

## VII. CONCLUSIONS

On-orbit, intravehicular localization is a challenging task. A localization-aware motion planner which uses a perception-based optimization metric was proposed to mitigate the risk of landmark feature loss. To achieve this, a gradient-based optimization approach using a novel twice continuously differentiable perception-aware objective function was presented. The proposed RRT\*-GBO motion plan-

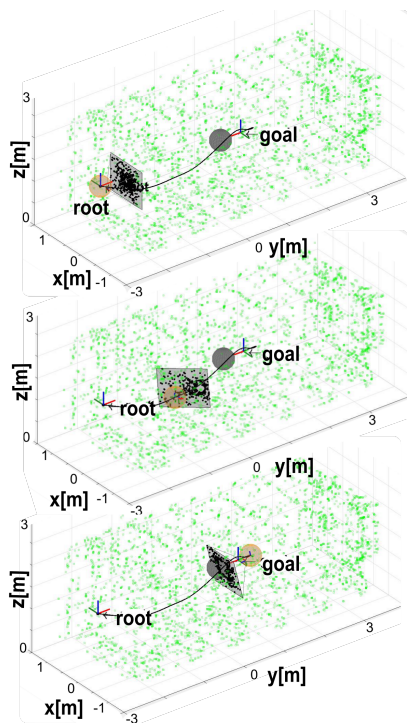


Fig. 7: Progression of the trajectory of an Astrobee during rendezvous. The camera frustum and image plane are visualized, showing the robot's orientation and that the planned solution keeps landmarks visible along the path. For clarity, the landmarks are located only on the walls of the JEM.

ning algorithm was then used to generate trajectories on  $SE(3)$ , which prioritize keeping a high density of features in view of the robot's camera while also minimizing the expended mechanical energy.

The approach was validated with experiments in a high-fidelity ISS simulation with the Astrobees using real sparse maps provided by NASA. The proposed gradient-based optimization framework for perception-aware trajectory optimization on  $SE(3)$  was compared to an adaptation of the state-of-the-art approach from [1] and [8]. The proposed interpolated feature-density cost function demonstrated significant improvement on state-of-the-art methods by reducing the online computational effort through an offline interpolation step, while also reducing localization error in simulation. The efficacy of RRT\*-GBO on  $SE(3)$  was demonstrated in the navigation of a cluttered environment and trajectory planning for the Astrobee robots in an on-orbit rendezvous maneuver in simulation.

Future work includes deployment for motion planning on a robotic test bed, as well as methodological extensions introducing online computational capability for the interpolation of the density function and for receding horizon-based motion planning approaches. Environmental uncertainties - such as air currents [4] - will also be examined.

#### REFERENCES

[1] V. Murali, I. Spasojevic, W. Guerra, and S. Karaman, "Perception-aware trajectory generation for aggressive quadrotor flight using

differential flatness," in *ACC 2019*, 2019, pp. 3936–3943.

[2] R. Soussan, V. Kumar, B. Coltin, and T. Smith, "Astroloc: An efficient and robust localizer for a free-flying robot," in *2022 Int. Conf. on Robot. and Automat. (ICRA)*, Philadelphia, PA, USA, 2022.

[3] L. Mao, R. Soussan, B. Coltin, T. Smith, and J. Biswas, "Semantic masking and visual feature matching for robust localization," in *iSpaRo 2024*, 2024, pp. 1–7.

[4] K. Albee and et al., "Autonomous rendezvous with an uncertain, uncooperative tumbling target: The tumblelock flight experiments," in *ASTRA 2022*, Katwijk, Netherlands, 2022.

[5] B. Doerr, K. Albee, M. Ekal, R. Ventura, and R. Linares, "The reswarm microgravity flight experiments: Planning, control, and model estimation for on-orbit close proximity operations," *J. of Field Robot.*, vol. 41, no. 6, pp. 1645–1679, 2024.

[6] H. Dinkel, J. Di, J. Santos, K. Albee, P. Borges, M. Moreira, and et al., "Astrobeed: Change detection in microgravity with free-flying robots," *Acta Astronautica*, vol. 223, pp. 98–107, 2024.

[7] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, "Revisiting active perception," *Autonomous Robots*, vol. 42, pp. 177–196, 2018.

[8] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware path planning for uavs using semantic segmentation," in *IROS 2020*, 2020, pp. 5808–5815.

[9] M. Flores, D. Valiente, A. Gil, O. Reinoso, and L. Payá, "Efficient probability-oriented feature matching using wide field-of-view imaging," *Eng. Appl. of Artif. Intell.*, vol. 107, 2024.

[10] X. Chen, Y. Zhang, B. Zhou, and S. Shen, "Apace: Agile and perception-aware trajectory generation for quadrotor flights," in *ICRA 2024*, 2024, pp. 17 858–17 864.

[11] G. Sun, X. Zhang, Y. Liu, X. Zhang, and Y. Zhuang, "Safety-driven and localization uncertainty-driven perception-aware trajectory planning for quadrotor unmanned aerial vehicles," *IEEE Trans. on Intell. Transp. Syst.*, vol. 25, no. 8, pp. 8837–8848, 2024.

[12] L. E. Scales, *Introduction to non-linear optimization*. Berlin, Heidelberg: Springer-Verlag, 1985.

[13] R. Lampariello, D. Nguyen-Tuong, C. Castellini, G. Hirzinger, and J. Peters, "Trajectory planning for optimal robot catching in real-time," in *ICRA 2011*, Shanghai, China, May 2011.

[14] R. Lampariello and G. Hirzinger, "Generating feasible trajectories for autonomous on-orbit grasping of spinning debris in a useful time," in *IROS 13*, Tokyo, Japan, November 2013.

[15] C. Specht, A. Bishnoi, and R. Lampariello, "Autonomous spacecraft rendezvous using tube-based model predictive control: Design and application," *J. of Guid., Control, and Dyn.*, vol. 46, no. 7, 2023.

[16] S. M. LaValle and J. J. Kuffner Jr., "Randomized kinodynamic planning," *Int. J. of Robotics Res.*, vol. 20, no. 5, pp. 378–400, 2001.

[17] L. E. Kavraki and S. M. LaValle, *Motion Planning*. Cham: Springer International Publishing, 2016, p. 139–162. [Online]. Available: [https://doi.org/10.1007/978-3-319-32552-1\\_7](https://doi.org/10.1007/978-3-319-32552-1_7)

[18] S. Stoneman and R. Lampariello, "Embedding nonlinear optimization in rrt\* for optimal kinodynamic planning," in *CDC 2014*, 2014, pp. 3737–3744.

[19] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[20] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *J. of Math. Imag. and Vis.*, vol. 35, no. 2, pp. 155–164, 2009.

[21] NASA, "Astrobee robot software," <https://github.com/nasa/astrobee>, 2024.

[22] —, "Astrobee-released-data," <https://nasagov.app.box.com/s/4ign43svk39guhy9ev8t5xkzui6tqjm1>, 2022.

[23] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IROS 2004*, Sendai, Japan, Sep 2004, pp. 2149–2154.

[24] F. Turchetti, M. Ekal, N. Lii, and M. Roa, "Analysis of intra-vehicular robotic free-flyers and their manipulation capabilities," in *IAC 2024*, 2024.

[25] K. Albee and et al., "A robust observation, planning, and control pipeline for autonomous rendezvous with tumbling targets," *Frontiers in Robotics and AI*, vol. 8, 2021.

[26] S. G. Johnson, "The nlopt nonlinear-optimization package," 2011. [Online]. Available: <http://github.com/stevengj/nlopt>

[27] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.