CHAPTER 9

Efficient Anytime CLF Reactive Planning System for a Bipedal Robot on Undulating Terrain

9.1 Introduction

Motion planning as a central component for autonomous navigation has been extensively studied over the last few decades. Algorithms such as RRT*, A*, and their variants focus on finding an (asymptotically) optimal path as computationally efficiently as possible [193–195, 219–224]. The application of these algorithms relies on designing a control policy to track the planned path, resulting in waypoint following or pathway tracking. In turn, the tracking of path segments (between waypoints) leads to non-smooth motion of the actual robot, due to abrupt acceleration or heading changes when transitioning between waypoints/pathways.

This chapter seeks to develop a reactive planning system for bipedal robots on unexplored, unmapped, challenging terrains and to provide high-rate (directional) velocity and heading commands to be realized by the robot's low-level feedback-control gait-generation algorithm. For this application, the non-smooth aspects of the planned motions arising from waypoints/pathways transitions are detrimental to stability of the overall system.

Several approaches have been developed to address the non-smooth aspects of paths produced by motion planning, such as reactive motion planning [225–233] and feedback motion planning [234–236]. Fundamentally, these approaches replace paths to be followed with smooth vector fields whose solutions guide the robot's evolution in its configuration space.

We are inspired by the work of [235, 236], which proposes a CLF to realize reactive planning for a non-holonomic differential-drive wheeled robot. A CLF is Lyapunov function for a closed-loop system where at any given time instant, there exists a control input that renders negative definite the derivative of the Lyapunov function along the system dynamics. Hence, a CLF is associated with asymptotically approaching a goal; see Sec. 9.3.1. While their underlying model designed for differential robots is not applicable to a Cassie bipedal robot due to different dynamics and control laws, their basic concept is applicable; see Sec. 9.3.2 for detailed discussion and its deficiency. As part of our work, we design an appropriate CLF for robots capable of walking in any direction with

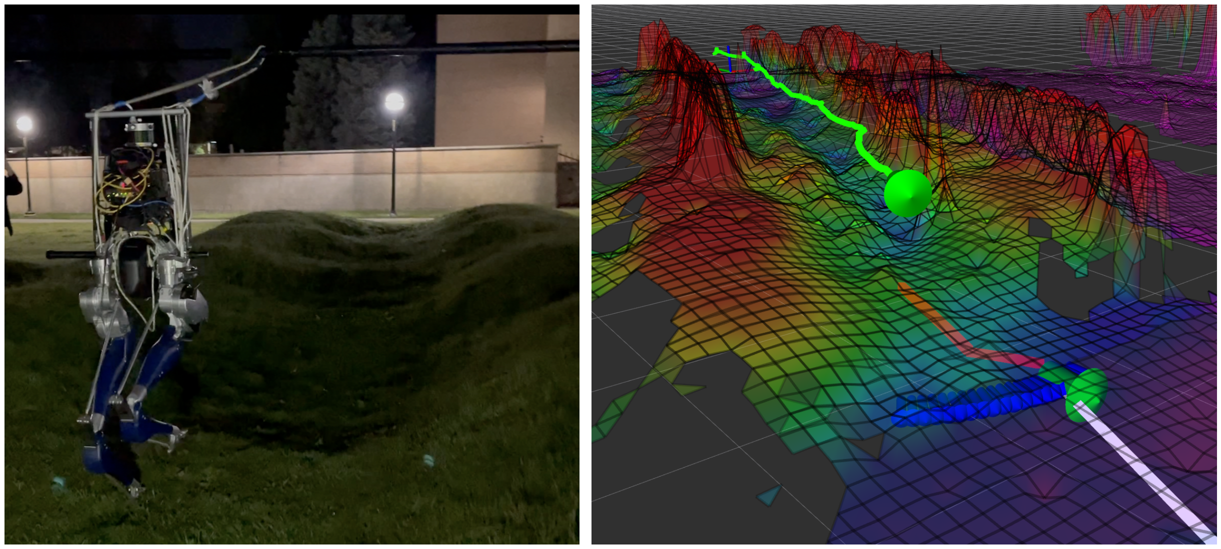


Figure 9.1: In the top figure, Cassie Blue autonomously traverses the Wave Field via the proposed reactive planning system, comprised of a planning thread and a reactive thread. The planning thread involves a multi-layer local map to compute traversability, a sub-goal finder, and an omnidirectional Control Lyapunov Function (CLF)RRT*. Instead of a common waypoint-following or path-tracking strategy, the reactive thread copes with robot deviation while eliminating non-smooth motions via a vector field (defined by a closed-loop feedback policy) that provides real-time control commands to the robot's gait controller as a function of instantaneous robot pose. The bottom figure is the elevation map built online. The red peaks are from the experimenters walking alongside Cassie.

any orientation. Moreover, we take into account features specific to bipeds, such as the limited lateral leg motion that renders lateral walking more laborious than sagittal plane walking.

The feedback motion planning algorithm in [235, 236] has not yet been evaluated on hardware. In general, there is a significant chasm between a planning algorithm and autonomous navigation on real robots. Most planning algorithms assume not only that a fully-explored, noise-free, perfect map is given but also that the robot's destination will always lie within this map. Moreover, the algorithms also assume a perfect robot pose and a perfect robot with ideal actuators that can execute an arbitrary trajectory. These assumptions are not practical. Therefore, utilizing a planning algorithm for autonomous navigation with real robots remains challenging. We propose and demonstrate experimentally an autonomous navigation system for a Cassie bipedal robot that is able to handle a noisy map in real-time, a distant goal that may not be in the initial map when the user decides where to send the robot, and importantly, a means to smoothly handle robot deviation. Additionally, a rudimentary finite-state machine is integrated to handle actions such as where to turn at intersections.

9.2 Related Work and Contributions

Motion planning, an essential component of robot autonomy, has been an active area of research for multiple decades with an accompanying rich literature. In this section, we review several types of planning algorithms and summarize our main contributions.

9.2.1 Sampling-Based Motion Planning

RRT [193] stands out for its low complexity and high efficiency in exploring unknown configuration spaces. Its asymptotically optimal version — RRT* [220] — has also gained much attention and has contributed greatly to the spread of the RRT family. RRT, RRT*, and variations on the basic algorithms, generate a collision-free path comprised of piece-wise linear paths between discrete poses of the robot [193–195, 219–224, 237, 238]. However, abrupt (non-differentiable) transitions between waypoints/pathways are an inherent issue with this family of planning algorithms and in addition, the generated trajectories do not account for control constraints. Therefore, to ensure the produced trajectories are feasible, additional expensive computations such as trajectory smoothing or optimization are often involved. A great deal of attention has been directed to this area, resulting in versions of RRT* [239–244] that utilize different smoothing techniques or steering functions.

Trajectory smoothing (B-spines, Dubins, or other parametric curves) is often designed independently of robot dynamics [245–247], which can lead to unbounded turning rate, acceleration, or jerk. Therefore, additional computations are necessary to validate the resulting smoothed trajectory. Furthermore, these methods are often ambiguous about how they treat robot deviations about the planned path and in the end provide open-loop control laws for tracking.

9.2.2 Optimization-based Planning and DARPA Subterranean Challenge

Point-wise in time optimization- and model-based algorithms, such as Control Lyapunov Functions (CLFs) paired with Quadratic Programs (QPs), (CLF-QP), or, when integrated with Control Barrier Functions (CBFs), (CLF-CBF-QP) [248–251] have been developed to provide low-level control for safety and object avoidance. Data-driven planning and control algorithms algorithms for safety-critical systems are combining machine-learning [252,253], Model Predictive Control (MPC), reinforcement learning [254], or belief-space learning [255].

Uncertainty-aware planning through Networked Belief-aware Perceptual Autonomy (NeBula) [256] from the DARPA Subterranean Challenge (DARPA SubT) [257] probabilistically fuses various sensing modalities to allow the robot to create belief-aware local maps. Reference [258] presents a quadruped with higher levels of autonomy to explore a tunnel environment in the 2019 DARPA SubT. To achieve autonomous exploration, MARBLE [259] proposes graph and frontier-based path planning algorithms on four-wheeled and tracked ground robots complimented with multi-rotor platforms. The GBPlanner proposed in [260] builds and uses a real-time topological map during subterranean exploration. An overview of ground robotics systems for underground environments is provided in [261]. The research challenges faced in such robotics exploration missions span the domains of communications, perception, SLAM, planning, and control.

9.2.3 Reactive Planning

Reactive planning contributes another significant concept to the motion planning literature [225–233], namely potential fields. In other words, the reactive planning replaces the concept of trajectory with that of a vector field arising as the gradient of a potential function. The method of potential fields seems to address all the issues raised in Sec. 9.2.1 for sampling-based methods. However, most of the experimental work has been carried out on flat ground and it is unclear how extensions to undulating terrain can be performed.

The concept of combining sampling-based algorithms with reactive planning was developed in [234–236], which not only provides a feasible path to follow from RRT*, but also a smooth feedback control law that instantaneously replans a path to the next goal as the robot deviates due to imperfections in the robot model in the robot's hardware or terrain. The feedback laws greatly ameliorates the issue of non-smooth paths. The feedback motion planning in [234] is based on a family of CLFs designed via linearization of the robot's model around a sufficiently large set of points in the robot's state space, Linear Quadratic Regulator (LQR), and Sum of Squares (SoS), whereas the feedback motion planning of [235, 236] uses a single CLF and varies the associated equilibrium to set sub-goal poses.

The feedback motion planning of [235,236] is the starting point for the work in this chapter. Park and Kuipers provide a novel form of RRT* for differential-drive wheeled robots, where a CLF is utilized as the steering function in the RRT* algorithm to evaluate the cost between nodes in the trees associated with RRT*, and it replaces the waypoints that are typically used in planning algorithms. Together, these innovations result in a system where robot control and motion planning are tightly coupled. As explained in Sec. 9.3.2, the CLF in [235,236] is designed for differential-drive wheeled robots which must respect nonholonomic constraints associated with wheels. In Sec. 9.3.2 we propose a new goal-centric coordinate system and CLF that are appropriate for bipedal robots that are omnidirectional. While bipedal robots are omnidirectional, they typically have limited agility in the lateral direction. We show how to account for the relative ease of walking forward and backward, sideways, and turning as a function of distance from goal. In Sec 9.4, we take these features of bipedal robots into account when connecting, exploring, and rewiring the trees in RRT*. In addition, we show how to take terrain features, such as friction and elevation changes, into account; see Sec. 9.5.2. This allows our Cassie robot to navigate undulating terrain.

We note that sampling-based approaches exist that are more efficient than RRT*, such as bi-directional RRT*, informed RRT* [262, 263], and RRT*-AB [264, 265]. We choose RRT* because the CLF used in the growing, pruning and rewiring of the tree is asymmetric, meaning the cost from node i to j is not equal to the cost from node j to node i. Therefore, the cost in growing the tree forward and backward are not the same. Search-based planners, such as A* [185], Bi-directional A* [183], or ANA* [184], are not efficient for growing a tree in $\mathrm{SE}(2)$.

9.2.4 Contributions

In particular, the present work has the following contributions:

1. We propose a novel 2D smooth Control Lyapunov Function (CLF) with a closed-form solution to the feedback controller for omnidirectional robots. The 2D CLF is designed such that when a goal is far from the robot position, the CLF controls the robot orientation to align with the goal while moving toward the goal. On the other hand, the robot walks to the goal disregarding its orientation if the goal is close. Additionally, we study the behaviors of the CLF under different initial conditions and parameters.

2. We define a closed-form distance measure from a pose (position and orientation) to a target position for omnidirectional robots under a pose-centric polar coordinate. This distance metric nicely captures inherent features of Cassie-series robots, such as the low-cost of longitudinal movement and high-cost of lateral movement.

3. We utilize the proposed CLF and the distance measure to form a new variation of RRT* (omnidirectional CLF-RRT*) to tackle undulating terrains, in which both distance and traversability are included in the cost to solve the optimal path problem. Moreover, as in [235], the optimal path is realized as a sequence of subgoals that are connected by integral curves of a set of vector fields, thereby providing reactive planning: in response to a disturbance, each vector field associated with the optimal path automatically guides the robot to a subgoal along a new integral curve of the vector field.

4. We integrate all the above components together as a reactive planning system for challenging terrains/cluttered indoor environments. It contains a planning thread to guide Cassie to walk in highly traversable areas toward a distant goal on the basis of a multi-layer map being built in real-time and a reactive thread to handle robot deviation via a closed-loop feedback control instead of a commonly used waypoint-following or path-tracking strategy.

We evaluate the reactive planning system by performing three types of experiments: **1)** A simplified biped pendulum model (inputs are piece-wise constant, similar to Cassie-series robots) navigates various synthetic, noisy, challenging outdoor terrains and cluttered indoor scenes. The system guides the robot to its goals in various scenes, both indoors and outdoors, with or without obstacles. The system also guides the robot to completion of several high-level missions, such as turning left at every intersection. **2)** To verify that the outputs of the control commands are feasible for Cassie-series robots, the system gives commands to a Cassie whole-body dynamic simulator [266], which simulates 20 DoF of Cassie in Matlab Simmechanics on a 3D terrain. **3)** Lastly, the reactive planning system successfully allows Cassie Blue to complete several indoor and

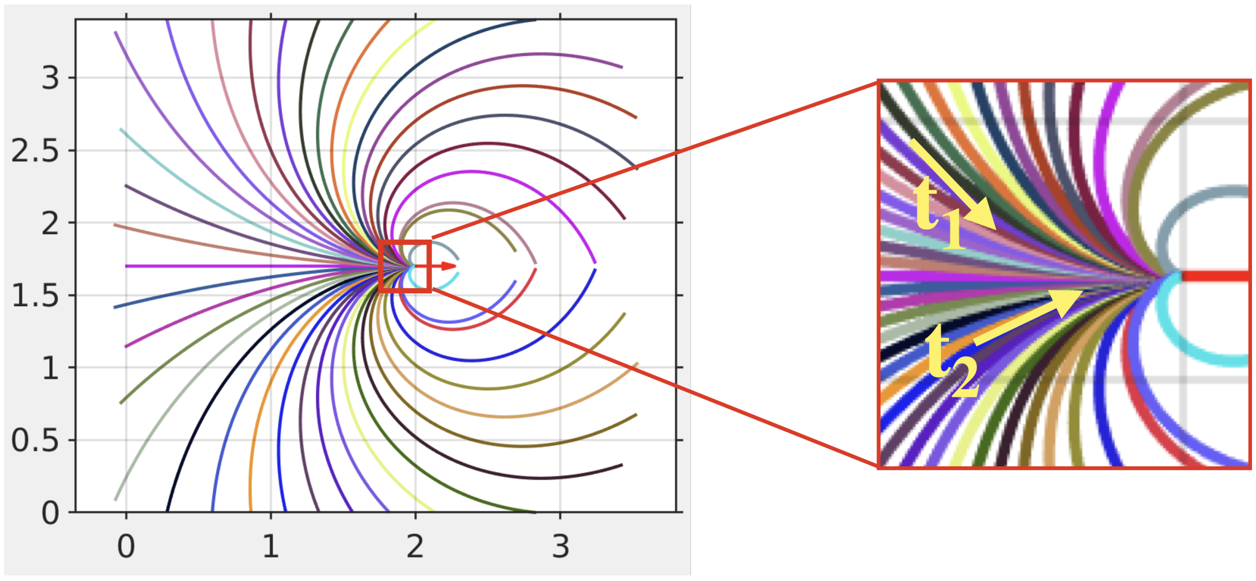


Figure 9.2: The plots show paths in 2D generated by the CLF of [235, 236] for Dubins cars. At each point, the tangent to a path (t_1 and t_2 in the blowup) is the heading angle for the robot. These paths clearly fail to account for a biped's ability to move laterally. Moreover, in practice, an underactuated robot such as Cassie Blue would experience chattering in the heading angle when approaching the goal (red arrow). Moreover, if the robot overshoots the goal, it would have to walk along a circle to return to the goal. For these reasons, a new CLF is needed.

outdoor missions: a) walking in corridors and avoiding furniture in the Ford Robotics Building (FRB) at the University of Michigan; b) turning left when detected intersections of corridors and return to its initial position in FRB; and c) traversing parts of the Wave Field on the North campus of the University of Michigan, as shown in Fig 9.1.

The videos of the autonomy experiments can be found at [49]. All of the simulated environments, the experimental data, and the C++ implementations for the reactive planning system are made available at https://github.com/UMich-BipedLab/CLF_reactive_planning_system [267].

The remainder of this chapter is organized as follows. Section 9.3 constructs the new CLF for bipeds and omnidirectional robots. The omnidirectional CLF-RRT* is introduced in Sec. 9.4. Section 9.5 integrates all the above components as a reactive planning system. Simulated and experimental evaluations of the proposed reactive system is presented in Sec. 9.6. Finally, Sec. 9.7 concludes the chapter and provides suggestions for future work.

9.3 Construction of a Control Lyapunov Function

This section first provides an introduction to CLFs and then describes the reasons for creating a new CLF function, the construction of the CLF, and an analysis of its parameters.

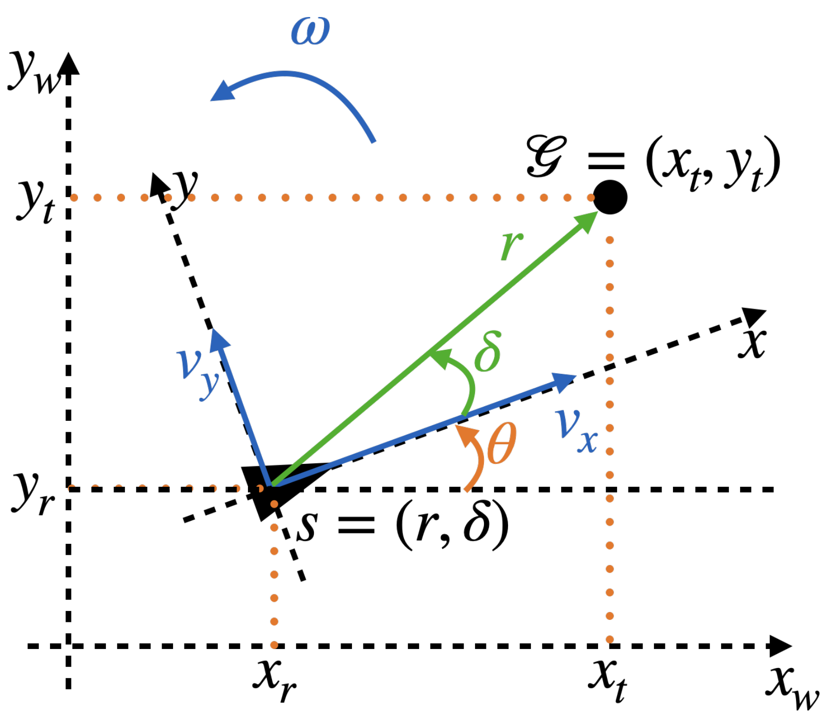


Figure 9.3: Illustration of the robot pose-centric polar representation $(s = (r, \delta))$ for robot pose (x_r, y_r, θ) and target position $\mathcal{G} = (x_t, y_t)$. Here, r is the radial distance to the target and δ is the angle between the heading angle θ of the robot and the line of sight from the robot to the goal. The longitudinal velocity, lateral velocity, and angular velocity are v_x, v_y and ω, respectively.

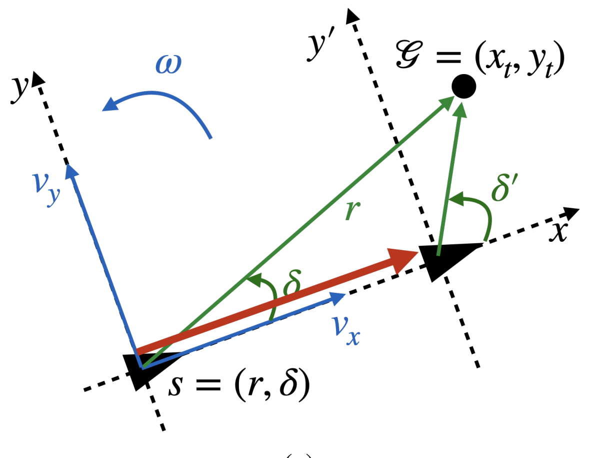
9.3.1 Lyapunov and Control-Lyapunov Functions

In this chapter, our goals and pseudo-goals are chosen to be equilibrium points of the center of mass dynamics of Cassie. A candidate Lyapunov function is a (locally) positive definite function that vanishes at an equilibrium point of a given dynamical system. If at each point of its definition, there exists a control input such that the derivative of the candidate Lyapunov function along the dynamics is negative definite, then it is called a Control Lyapunov Function (CLF), or CLF for short. Hence, CLFs provide a means to specify a goal as well as a family of trajectories that converge to the goal from "arbitrary" points (sufficiently near) the goal. The trajectories are the solutions of the underlying dynamic model that are compatible with the Lyapunov function monotonically decreasing.

We refer the reader to [268] for the formal definition. Consider $\mathcal{O}$ an open set about the origin of $\mathbb{R}^n$, and

$$\dot{x} = f(x, u), \tag{9.1}$$

a control system with $x \in \mathbb{R}^n$ is system state and $u \in \mathbb{R}^m$ control commands. The differentiable function $V : \mathcal{O} \to [0, \infty)$ is a CLF if (i) $V(x) = 0 \implies x = 0$ and for each $0 \neq x \in \mathcal{O}$, there exists $u \in \mathbb{R}^m$ such that $\nabla V(x) f(x, u) < 0$. Any feedback law $u = \alpha(x)$ such that $\nabla V(x) f(x, \alpha(x)) < 0$ then renders the origin asymptotically stable.



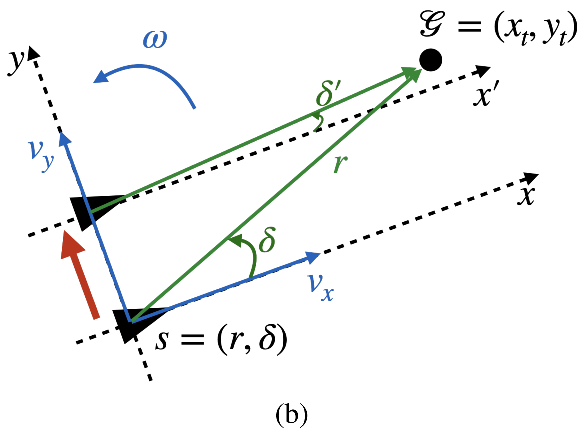


Figure 9.4: This figure explains the signs in (9.3). On the left, δ increases when the robot moves parallel to the x-axis. Therefore, $\frac{v_x}{r}\sin\delta$ is positive. Similarly, $\frac{v_y}{r}\cos\delta$ is negated because δ decreases when the robot moves toward the y-axis.

In [235], the output of their planner is a feedback function $u = \alpha(x)$ rendering a particular CLF negative definite. When the CLF is associated with a goal or pseudo-goal of the planner, this becomes a particularly astute means for the planner to communicate its intentions to a low-level controller: the CLF specifies how the planner wants the robot to approach the goal from an entire open set of current states of the robot. This allows the immediate reaction to disturbances. We adopt this means in this chapter as well.

9.3.2 Redesign of CLF Proposed in the Literature

The 2D CLF planner of [235, 236] has been designed for differentially driven non-holonomically constrained robots, whose dynamics and control laws are inappropriate for bipedal robots. Like most robot models, the work [235] assumes that the robot is able to continuously change its velocity and heading. However, this is not possible for underactuated bipeds such as Cassie Blue. According to the Angular Linear Inverted Pendulum (ALIP) model used for low-level feedback control of Cassie Blue [216, 269–272], the heading angle and the longitudinal and lateral velocity commands can only be updated at the initiation of a step and not within a step. In particular, the low-level controller on a bipedal robot during the swing phase is controlling body posture and **regulating foot placement at the end of the current step** so that the center of mass can achieve velocity and orientation goals over the **next step** or **next few steps**. Only minor instantaneous corrections to body velocity can be achieved during a given step of the robot. In other words, bipedal robots such as Cassie are not able to implement changes in velocity control commands during the current swing phase, but will instead execute the received control commands during the following swing phase. When piece-wise constant commands are applied to the existing 2D CLF of [235, 236], built around a Dubins car model, the closed-loop system will oscillate about the discrete heading directions as the robot approaches the goal pose, as explained in Fig. 9.2. This oscillation is undesirable as it can affect the robot's balance.

With a Dubins car model as used in [235, 236], the linear velocity is always aligned with the heading angle of the vehicle, and hence this is also true as the vehicle approaches an equilibrium pose. Consequently, a CLF for a target position must also include a target heading, therefore, a target pose. The vehicle must steer and align itself as it approaches the target. Cassie Blue, on the other hand, similar to an omnidirectional robot, is able to move laterally with zero forward velocity, which allows the robot to start with an arbitrary pose and arrive at a goal position with an arbitrary heading (i.e., start with a pose and end with a position). Lateral walking, however, requires more effort due to the limited workspace of the lateral hip joints on the robot and this should be taken into account when designing a CLF.

To avoid undesirable oscillating movement and account for lateral walking, a new candidate CLF is designed on the basis of an appropriate kinematics model for underactuated bipeds and other omnidirectional robots.

9.3.3 State Representation

As mentioned in Sec. 9.3, Cassie Blue is able to walk in any direction. Therefore, we model Cassie Blue as an omnidirectional robot and reduce it to a directional point mass. We will account for the increased effort required to walk laterally when we design the CLF.

Denote $\mathcal{P} = (x_r, y_r, \theta)$ the robot pose and $\mathcal{G} = (x_t, y_t)$ the goal position in the world frame. Let s be the state of an omnidirectional robot represented in a *robot pose-centric polar coordinate*:

$$s = \{(r, \delta) | r \in \mathbb{R}, \text{ and } \delta \in (-\pi, \pi]\}, \tag{9.2}$$

where $(-\pi, \pi]$ is open on the left, $r = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}$, and δ is the angle between the heading angle of the robot (θ) and the line of sight from the robot to the goal, as shown in Fig. 9.3.

Remark 31. *References [235, 236] used target pose-centric polar coordinates because the wheelchair robot needed to arrive at a target position with a target heading angle. In our case, we can use robot pose-centric coordinates because we have the freedom to arrive at the target position with any heading angle. For bipeds, turning in place is easy, and thus, if orientation at the goal is critical, it can be handled as a final maneuver.*

9.3.4 Construction of Control Lyapunov Function

The kinematics of an omnidirectional robot is defined as

$$\begin{bmatrix} \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} -\cos(\delta) & -\sin(\delta) \\ \frac{1}{r}\sin(\delta) & -\frac{1}{r}\cos(\delta) \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \end{bmatrix}. \tag{9.3}$$

In the above expression, we view v_x, v_y and ω as control variables. Because the matrix

$$\begin{bmatrix} -\cos(\delta) & -\sin(\delta) \\ \frac{1}{r}\sin(\delta) & -\frac{1}{r}\cos(\delta) \end{bmatrix}$$

is negative definite (and hence invertible) for all $r > 0$, the model (9.3) is over actuated for $r > 0$.

Remark 32. *Observe that $\delta < \delta'$ when the robot moves along the x-axis, as shown in Fig. 9.4(a). Therefore, $\frac{v_x}{r}\sin(\delta)$ is positive. Similarly, $\frac{v_y}{r}\cos(\delta)$ is negated because $\delta < \delta'$ when the robot moves toward the y-axis, as shown in Fig. 9.4(b).*

We next note that the change of control variables

$$\begin{bmatrix} v_r \\ v_\delta \end{bmatrix} := \begin{bmatrix} \cos(\delta) & \sin(\delta) \\ \frac{\sin(\delta)}{r} & -\frac{\cos(\delta)}{r} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \end{bmatrix},$$

allows us to feedback linearize the model to a pair of integrators.

$$\begin{bmatrix} \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} -v_r \\ v_\delta \end{bmatrix}.$$

We note that for this model, any positive definite quadratic function is automatically a CLF. For later use, we note that for all $r > 0$,

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} := \begin{bmatrix} \cos(\delta) & r\sin(\delta) \\ \sin(\delta) & -r\cos(\delta) \end{bmatrix} \begin{bmatrix} v_r \\ v_\delta - \omega \end{bmatrix}. \tag{9.4}$$

As mentioned in Sec. 9.3, lateral walking is more expensive than longitudinal walking because movement in the lateral hip joint is limited. A candidate control Lyapunov function[1] ℓ, in terms of the robot's current pose and target (end) position, is defined as

$$\ell = \frac{r^2 + \gamma^2 \sin^2(\beta\delta)}{2}, \tag{9.5}$$

[1]In polar coordinate, the function ℓ is positive definite in the sense that $\ell = 0 \implies r = 0$, and when $r = 0$, the angle δ is arbitrary or undefined.

where γ is a weight on the orientation and the role of $\beta > 0$ will be described later. We next check that ℓ is a Control-Lyapunov function. The derivative of ℓ is

$$\begin{aligned}\dot{\ell} &= r\dot{r} + \frac{\beta\gamma^2}{2}\sin\left(2\,\beta\,\delta\right)\dot{\delta} \\ &= r(-v_r) + \frac{\beta\gamma^2}{2}\sin(2\beta\delta)v_\delta.\end{aligned} \tag{9.6}$$

The feedback

$$\begin{aligned}v_r &= k_{r1}\frac{r}{k_{r2}+r} \\ v_\delta &= -\frac{2}{\beta}k_{\delta 1}\frac{r}{k_{\delta 2}+r}\sin(2\beta\delta)\end{aligned} \tag{9.7}$$

results in

$$\dot{\ell} = -\frac{k_{r1}}{k_{r2}+r}r^2 - k_{\delta 1}\gamma^2\frac{r}{k_{\delta 2}+r}\sin^2(2\beta\delta), \tag{9.8}$$

which is negative for all $r > 0$, $\beta > 0$, $k_{r1} > 0,\ k_{r2} > 0,\ k_{\delta 1} > 0,$ and $k_{\delta 2} > 0.$

It is emphasized that the proposed CLF under the robot-centric coordinate system is 2D. Later, the cost function for the RRT*-based planner will be 3D; see Sec. 9.4.2. Work in [269, 273, 274] shows how to adapt the local model to the terrain in such a way that the model (7) is always valid and hence the 2D CLF is applicable.

Remark 33. *From* (9.7)*, it follows that* $\dot{\delta} = v_\delta = 0$ *for* $2\beta\delta \in \{0, \pm\pi\}$*. Therefore, the manifolds*

$$M_\delta := \{(r,\delta) \mid r \geqslant 0, \delta \in \{0, \frac{\pi}{\beta}, \pm\frac{\pi}{2\beta}\}\}$$

are invariant for the closed-loop system. From (9.8)*, the manifold* M_δ *is locally attractive for* $\delta \in \{0, \frac{\pi}{\beta}\}$ *and repulsive for* $\delta = \pm\frac{\pi}{2\beta}$*. By selecting* $\beta > 0$*, the repulsive invariant manifold can be placed outside the Field of View (FoV) of Cassie, as shown in Fig. 9.6. In practice, a Finite-State Machine (FSM) is needed so that the robot will initially turn in place so that it starts with the goal located within the FoV of its sensor suite.*

The next step is to set up an optimization such that the control variables (v_x, v_y, ω) satisfy (9.7) and take into account that walking sideways takes more effort than walking forward, for Cassie. Because the camera faces forward, walking backward is only selected if the robot is localized into an already built portion of the map.

Table 9.1: The default values of parameters.

α	β	γ	k_{r1}	k_{r2}	$k_{\delta 1}$	$k_{\delta 2}$
10	1.2	1	1	5	0.1	10

9.3.5 Closed-form Solution

Taking (9.4) as a constraint, we propose to select ω so as to keep v_y small (limit lateral walking) by optimizing

$$J = \min_{v_y,\omega} \;\; (v_y)^2 + \alpha\omega^2. \tag{9.9}$$

The parameter $\alpha > 0$ allows us to penalize aggressive yaw motions ω, as will be illustrated in Sec. 9.3.6. Plugging in the constraint (9.4), (9.9) leads to

$$\begin{aligned} J &= \min_{\omega} \;\; \{[\sin(\delta)v_r - r\cos(\delta)(v_\delta - \omega)]^2 + \alpha\omega^2\} \\ &= \min_{\omega} \;\; \{(\sin(\delta)v_r)^2 + [r\cos(\delta)(v_\delta - \omega)]^2 \\ &\qquad - 2\sin(\delta)v_r(r\cos(\delta)(v_\delta - \omega)) + \alpha\omega^2\}. \end{aligned}$$

A few algebraic calculations and the dropping of "constant terms" lead to

$$\begin{aligned} \omega^* = \arg\min_{\omega} \;\; &\{r^2\cos^2(\delta)\,(v_\delta - \omega)^2 + \\ &\qquad 2rv_r\sin(\delta)\cos(\delta)\omega + \alpha\omega^2\}, \end{aligned}$$

which implies that

$$\left(\alpha + r^2\cos^2(\delta)\right)\omega^* + r\cos(\delta)\left[v_r\sin(\delta) - rv_\delta\cos(\delta)\right] = 0. \tag{9.10}$$

The final result is

$$\omega^* = \frac{r\cos(\delta)\left[rv_\delta\cos(\delta) - v_r\sin(\delta)\right]}{\alpha + r^2\cos^2(\delta)}, \tag{9.11}$$

and then

$$\begin{aligned} v_y^* &= \frac{\alpha\;(v_r\;\sin(\delta) - r\,v_\delta\;\cos(\delta))}{r^2\cos(\delta)^2 + \alpha} \\ v_x^* &= \frac{v_r\;\cos(\delta)\;r^2 + \alpha\,v_\delta\;\sin(\delta)\;r + \alpha\,v_r\;\cos(\delta)}{r^2\cos(\delta)^2 + \alpha}. \end{aligned} \tag{9.12}$$

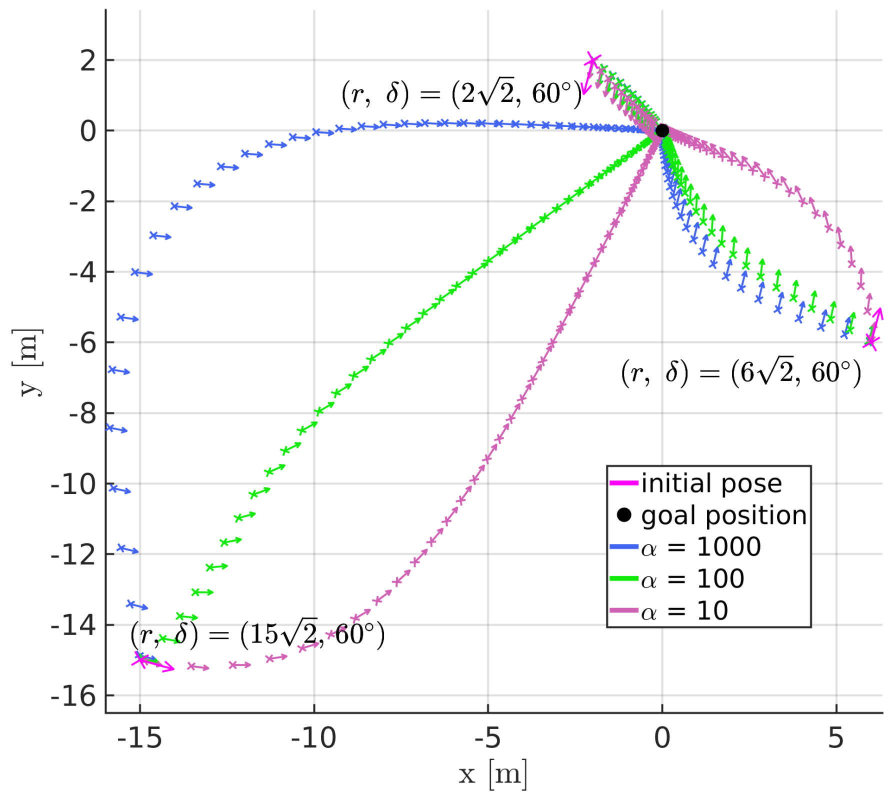


Figure 9.5: The distance to the target and the penalty on yaw motion in (9.9) both affect closed-loop behavior arising from the CLF. The arrows indicate the robot's absolute heading. In each solution of the closed-loop system, the robot's heading relative to the target is initialized at $60°$. When the robot is distant from the goal and the heading does not point toward the goal, it will align its relative heading to the target while approaching the goal. The level of alignment depends on the yaw motion penalty, α. On the other hand, when the robot is close to the goal, the closed-loop controller no longer adjusts the heading angle and employs a lateral motion to reach the goal.

9.3.6 Qualitative Analysis of the Closed-loop Trajectories

The default parameters applied in this analysis are shown in Table. 9.1. Figure 9.5 shows how the closed-loop trajectories vary as a function of heavy, medium, and light penalties on yaw motion, and three different initial distances from the target, with δ, the robot's heading relative to the target, fixed at $60°$. We observe that with $r = 2\sqrt{2}$, the robot walks laterally to achieve the goal for all values of the penalty on yaw motion. With $r = 15\sqrt{2}$ and $\alpha = 10$, the robot aligns its heading to the target while walking to reduce its lateral movement, whereas with $\alpha = 100$, it maintains its heading and combines lateral and longitudinal motion as needed to reach the goal.

Figure 9.6 shows how the closed-loop trajectories vary as a function the initial relative heading to the target, when starting at a fixed distance of $r = 15$ m, and $\alpha = 10$. As indicated in Table 9.1, we are using $\beta = 1.2$, which yields FoV of $\pm 75°$. For relative heading "errors" less than $40°$, the robot aligns quickly to the target and longitudinal walking dominates. If quicker zeroing of the heading error is desired, a smaller value of α could be used or the robot could turn in place before starting a new segment.

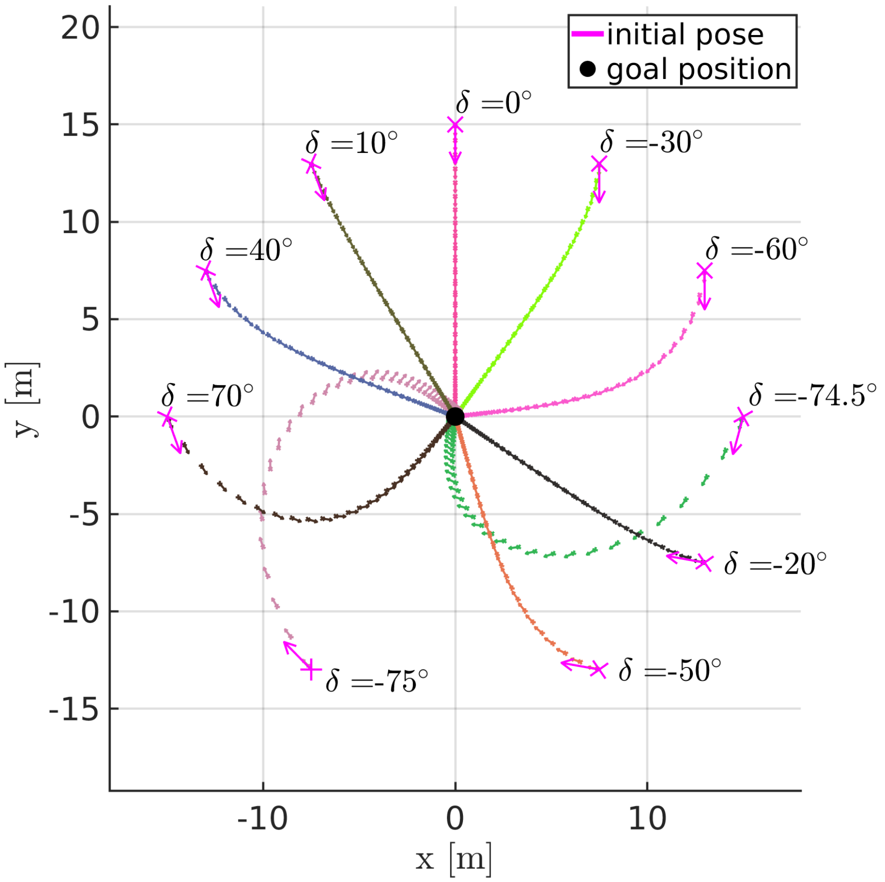


Figure 9.6: This figure illustrated how the closed-loop trajectories generated by the CLF in (9.5) vary as a function the initial relative heading to the target, when starting at a fixed distance of $r = 15$ m, and $\alpha = 10$. The arrows indicate the robot's heading. As shown in Table 9.1, we are using $\beta = 1.2$, which yields an FoV of $\pm 75°$. For relative heading "errors" less than $40°$, the robot aligns quickly to the target and longitudinal walking dominates. These motions should be compared to those in Fig. 9.5

9.4 Omnidirectional CLF-RRT*

This section integrates the CLF proposed in Sec. 9.3 into the original RRT* algorithm. The resulting omnidirectional CLF RRT* provides feasible paths for (9.3) while (i) accounting for relative heading, (ii) the asymmetry in roles of target position and current pose induced by the CLF, and (iii) the fact that walking laterally is more challenging than walking in the longitudinal direction for robots such as Cassie.

9.4.1 Standard RRT* Algorithm

The original RRT* [194] is a sampling-based, incremental planner with guaranteed asymptotic optimality. In configuration space, RRT* grows a tree where leaves are states connected by edges of linear path segments with the minimal cost. Additionally, RRT* considers nearby nodes of a sample to choose the best parent node and to rewire the graph if shorter path is possible to guarantee asymptotic optimality.

9.4.2 Omnidirectional CLF-RRT* Algorithm

The omnidirectional CLF-RRT* differs from the original RRT* in four aspects. First, the distance between two nodes is defined by the CLF in (9.5), which takes relative heading into account. Second, the steering/extending functions use the closed-loop trajectories generated by (9.12) to define paths between nodes. Third, because the cost (9.5) between two nodes i and j is not symmetric (i.e, a different cost is assigned if node i is the origin versus it is the target), a distinction must be made between near-to nodes and near-from nodes. The above three aspects are common to the CLF-RRT* variant introduced in [235, 236]. Finally, when connecting, exploring, and rewiring the tree, additional terms are added to the cost (9.5) to account for the relative ease or difficulty of traversing the path.

Our proposed RRT* modification is summarized below with notation that generally follows [195]. Let $\mathcal{X} = \{(x, y, \theta) \mid x, y \in \mathbb{R} \text{ and } \theta \in (-\pi, \pi]\}$ be the configuration space and let $\mathcal{X}_{\text{obs}}$ denote the obstacle region, which together define the free region for walking $\mathcal{X}_{\text{free}} = \mathcal{X} \backslash \mathcal{X}_{\text{obs}}$. The omnidirectional CLF RRT* solves the optimal path planning problem by growing a tree $\mathcal{T} = (V, E)$, where $V \in \mathcal{X}_{\text{free}}$ is a vertex set of poses connected by edges E of feasible path segments. Briefly speaking, the proposed RRT* (Algorithm 2) explores the configuration space by random sampling and extending nodes to grow the tree (explore the configuration space), just as in the classic RRT [193]. Considering nearby nodes of a sample to choose the best parent node and rewiring the graph guarantee asymptotic optimality (Algorithm 3 and 4), as with the classic algorithm. As emphasized previously, a key difference lies in how the paths between vertices are generated.

9.4.2.1 Sampling

This step randomly samples a pose $n_{\text{rand}} = (x, y, \theta) \in \mathcal{X}_{\text{free}}$. To facilitate faster convergence and to find better paths, we use several techniques such as sampling with a goal bias, limited search space, and Gaussian sampling. Specifically, given a sub-goal and a degree of goal biasing, we bias samples to be from a Gaussian distribution centered about the sub-goal; see Sec. 9.5.2. Furthermore, we limit the sampling space to a sector in front of the robot.

9.4.2.2 Distance

To account for the asymmetry in lateral vs longitudinal motions, as discussed in Sec. 9.3.2, the distance $d(n_i, n_k)$ from node n_i to node n_k in the tree $\mathcal{T}$ is defined by (9.5). Note that when computing the distance, n_i is a pose (x_i, y_i, θ_i) and the heading of n_k is ignored, meaning only its (x_k, y_k) values are used.

Algorithm 2: $\mathcal{T} = (V, E) \leftarrow$ Omnidirectional CLF RRT*

```
1  $\mathcal{T} \leftarrow$ InitializeTree();
2  $\mathcal{T} \leftarrow$ InsertNode($\varnothing, n_{\text{init}}, \mathcal{T}$);
3  **for** $i$=1 **to** $N$ **do**
4      $n_{\text{rand}} \leftarrow$ Sample($i$)
5      $n_{\text{nearest}} \leftarrow$ Nearest($\mathcal{T}, n_{\text{rand}}$)
6      $(n_{\text{new}}, \mathscr{T}') \leftarrow$ Extend($n_{\text{nearest}}, n_{\text{rand}}, \kappa$)
7      **if** ObstacleFree($\mathscr{T}'$) **then**
8          $\mathcal{N}_T \leftarrow$ NearTo($\mathcal{T}, n_{\text{new}}, |V|$)
9          $n_{\text{min}} \leftarrow$ ChooseParent($\mathcal{N}_T, n_{\text{nearest}}, n_{\text{new}}$)
10         $\mathcal{T} \leftarrow$ InsertNode($n_{\text{min}}, n_{\text{new}}, \mathcal{T}$)
11         $\mathcal{N}_F \leftarrow$ NearFrom($\mathcal{T}, n_{\text{new}}, |V|$)
12         $\mathcal{T} \leftarrow$ ReWire($\mathcal{T}, \mathcal{N}_F, n_{\text{min}}, n_{\text{new}}$)
13 **return** $\mathcal{T}$
```

Remark 34. *As mentioned in Sec. 9.3.4, the robot will rotate in place if the target point is outside the FoV. If rotating in place is laborious, one can also consider the following distance function:*

$$d(n_i, n_k) = \ell + k_\delta \max\left(|\delta| - |U|, 0\right), \tag{9.13}$$

where ℓ is defined in (9.5)*, k_δ is a positive constant, and U corresponds to a repulsive point (i.e., $\pm\frac{\pi}{2\beta}$) in Remark 33.*

9.4.2.3 Traversability of a path

Let $\mathcal{P} = (x_r, y_r, z_r, \theta)$ be the current robot pose and denote $\mathscr{T}(\mathcal{P}, n_i, n_j)$ the path[2] connecting n_i and n_j. Finally, let $\mathbb{T}(\mathcal{P}, \mathscr{T})$ be the cost of the path traversability, defined as a running cost along the trajectory of the robot, namely

$$\mathbb{T}(\mathcal{P}, \mathscr{T}) = \sum_{\forall x_t, y_t \in \mathscr{T}} C(x_t, y_t, z_r) \tag{9.14}$$

where (x_t, y_t) is the location of the robot at time t, $C(x_t, y_t, z_r)$ can depend upon elevation change with respect to the robot's current elevation, z_r, ground slope, friction coefficient, or other terrain characteristics provided by the mapping software [50, 217, 218].

Remark 35. *The planning system is designed so that any traversability index [275, 276] can be leveraged in the cost function on the local map to indicate the relative ease, difficulty, or safety of*

[2]The path is generated from the CLF in Sec. 9.3.

traversing a section of terrain. Therefore, the system can be readily adapted to different types of terrain.

9.4.2.4 Cost between Nodes

Let $c(n_i, n_k)$ be the cost from n_i to n_k in the tree $\mathcal{T}$, defined as

$$c(n_i, n_k) = d(n_i, n_k) + k_t \mathbb{T}(\mathcal{P}, \mathscr{T}), \tag{9.15}$$

where k_t trades terrain traversability versus distance. It sets how much more distance the overall mission is allowed to detour for a better traversable path. For all experiments conducted in the chapter, $k_t = 1$.

9.4.2.5 Nearby Nodes

Due to the use of the CLF function, the distinction between near-to nodes $\mathcal{N}_T$ and near-from nodes $\mathcal{N}_F$ is necessary.

$$\begin{aligned} \mathcal{N}_T(n_i, \mathcal{T}, \mathcal{M}, m) := \{n \in V \mid d(n, n_i) \leqslant L(m) \ \ \& \\ |\mathbb{T}(n, \mathcal{P}) - \mathbb{T}(n_i, \mathcal{P})| \leqslant T_k\}, \end{aligned} \tag{9.16}$$

where $|\cdot|$ is the absolute value, m is the number of nodes in the tree $\mathcal{T}$, and $L(m) = \eta \left(\log(n)/n\right)^{(1/\xi)}$ with the constant η and dimension of space ξ (3 in our case) [221] and T_k is a positive constant. Similarly, the near-from nodes $\mathcal{N}_F$ are determined by

$$\begin{aligned} \mathcal{N}_F(n_i, \mathcal{T}, \mathcal{M}, m) := \{n \in V \mid d(n_i, n) \leqslant L(m) \ \ \& \\ |\mathbb{T}(n, \mathcal{P}) - \mathbb{T}(n_i, \mathcal{P})| \leqslant T_k\}. \end{aligned} \tag{9.17}$$

9.4.2.6 Nearest Node

Given a node $n_i \in \mathcal{X}$, the tree $\mathcal{T}$, and the local map $\mathcal{M}$, the nearest node is any node $n_* \in \mathcal{T}$ in the tree where the cost from n_* to n_i is minimum.

9.4.2.7 Steering and Extending

The steering function generates a path segment $\mathscr{T}$ that starts from n_i and ends exactly at n_k. The extending function extends the path from n_i toward n_k until n_k is reached or the distance traveled is κ in which case it returns a new sample n_{new} at the end of the extension.

Algorithm 3: $n_{\text{parent}} \leftarrow \text{ChooseParent}(\mathcal{N}_T, n_{\text{nearest}}, n_{\text{new}})$

```
1  $n_{\text{parent}} \leftarrow n_{\text{nearest}}$
2  $c_{\text{parent}} \leftarrow \text{Cost}(n_{\text{nearest}}) + \text{c}(n_{\text{nearest}}, n_{\text{new}})$
3  for $n_{near} \in \mathcal{N}_T$ do
4      $\mathscr{T}' \leftarrow \text{Steer}(n_{\text{near}}, n_{\text{new}})$
5      if $ObstacleFree(\mathscr{T}')$ then
6          $c' = \text{Cost}(n_{\text{near}}) + c(n_{\text{near}}, n_{\text{new}})$
7          if $c' < Cost(n_{new})$ and $c' < c_{parent}$ then
8              $n_{\text{parent}} \leftarrow n_{\text{near}}$
9              $c_{\text{parent}} \leftarrow c'$
10 return $n_{\text{parent}}$
```

Algorithm 4: $\mathcal{T} \leftarrow \text{ReWire}(\mathcal{T}, \mathcal{N}_F, n_{\text{min}}, n_{\text{new}})$

```
1  for $n_{near} \in \mathcal{N}_F \backslash \{n_{min}\}$ do
2      $\mathscr{T}' \leftarrow \text{Steer}(n_{\text{new}}, n_{\text{near}})$
3      if $ObstacleFree(\mathscr{T}')$ and
4        $Cost(n_{new}) + c(n_{new}, n_{near}) < Cost(n_{near})$ then
5          $\mathcal{T} \leftarrow \text{Re-Connect}(n_{\text{new}}, n_{\text{near}}, \mathcal{T})$
6  return $\mathcal{T}$
```

9.4.2.8 Parent Choosing and Graph Rewiring

Choosing the best parent node (Algorithm 3) and rewiring the graph (Algorithm 4) guarantee asymptotic optimality. Let $\text{Cost}(n_i)$ be the cost from the root of the tree $\mathcal{T}$ to the node n_i. The parent n_{parent} of a node n_{new} is determined by finding a node $n_i \in \mathcal{N}_T$ with smallest cost from the root to the node:

$$n_{\text{parent}} = \underset{n_{\text{near}} \in \mathcal{N}_T}{\arg\min}\, \text{Cost}(n_{\text{near}}) + c(n_{\text{near}}, n_{\text{new}}). \tag{9.18}$$

After a parent node is chosen, nearby nodes $\mathcal{N}_F$ are rewired if shorter paths are found. In our experiments, we used the extending function for exploration, and the steering function to find the best parent node and to rewire the graph.

9.4.2.9 Collision Check

This step verifies whether a path $\mathscr{T}$ lies within the obstacle-free region of the configuration space. Note that additional constraints, such as curvature bounds and minimum clearance, can also be examined in this step.

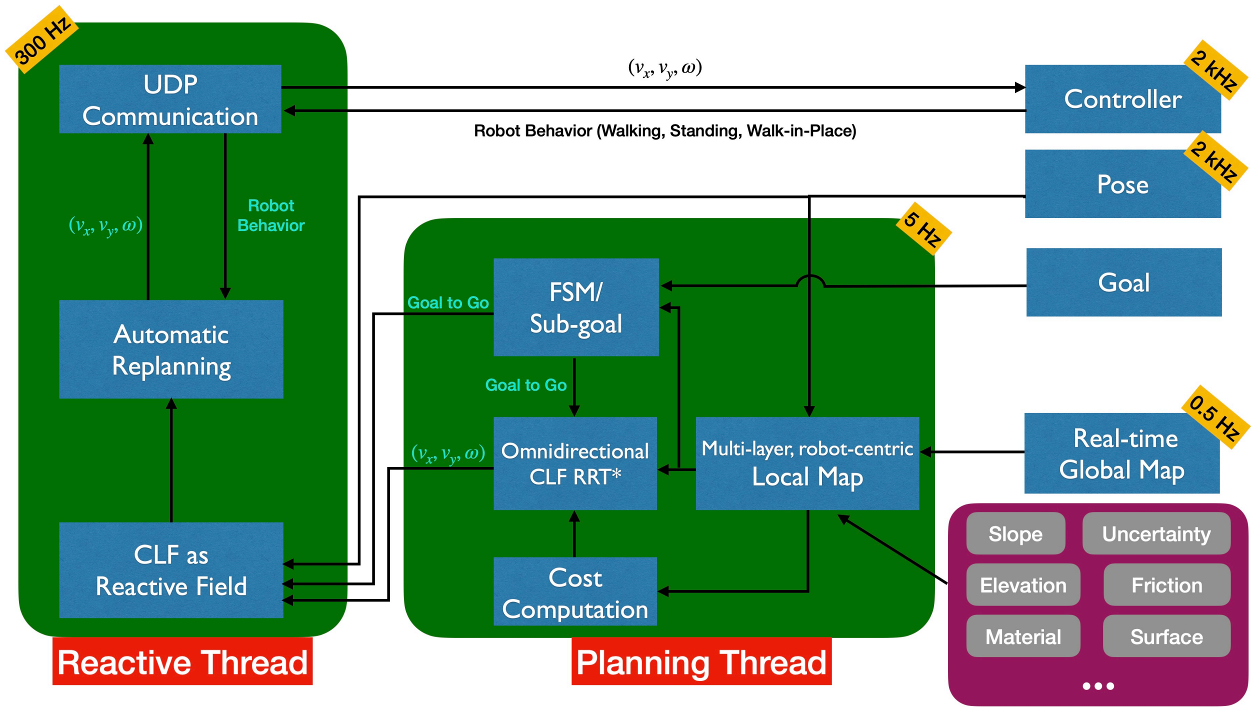


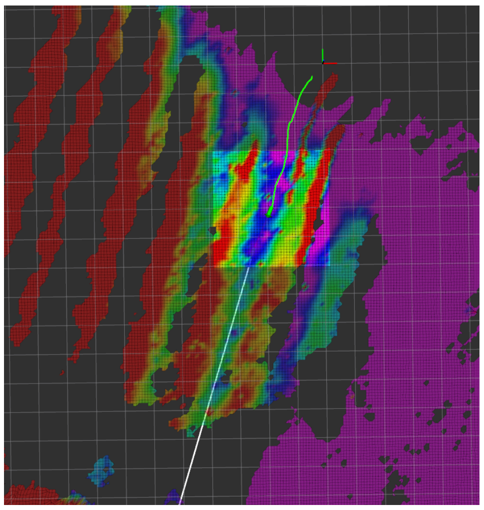
Figure 9.7: This figure summarizes the proposed reactive planning system. The planning thread is built around RRT* and an omnidirectional CLF that is used to assign distances, define locally optimal path segments, search radius, and linking conditions for re-wiring and choosing a parent. In addition, the planning thread contains a multi-layer, robot-centric local map for computing traversability, a sub-goal finder, and a Finite-State Machine (FSM) to choose sub-goal locations guiding the robot to a distant goal. The terrain information extracted from the multi-layer local map can be shared with a terrain-aware controller, such as [273]. Instead of a common waypoint-following or path-tracking strategy, the reactive thread copes with robot deviation while eliminating non-smooth motions via a vector field (defined by a closed-loop feedback policy arising from the CLF). The vector field provides real-time control commands to the robot's gait controller as a function of instantaneous robot pose.

9.4.2.10 Node Insertion

Given the current tree $\mathcal{T} = (V, E)$ and a node $v \in V$, this step inserts the node n to V and creates an edge e_{nv} from n to v.

9.5 Reactive Planning System

The previous section provides a sparse set of paths from a robot's initial location to a goal. The degree of optimality depends on how long the planning algorithm is run. A typical update rate may be 5 Hz for real-time applications. When the robot is perturbed off the nominal path, one is left with deciding how to reach the goal, say by tracking the nominal path with a PID controller. Important alternatives to this, called a high-frequency reactive planner or a feedback motion planner, were introduced in [225–234]. A version based on the work of [235, 236] will be incorporated into our overall planning system. In addition, we take into account features in a local map.


(a)

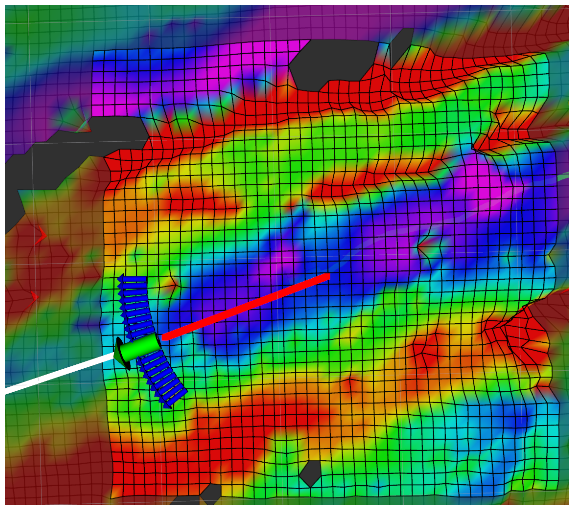

(b)

Figure 9.8: The elevation map (colored by height) was built online while Cassie was autonomously traversing the Wave Field on the North Campus of the University of Michigan. The highlighted area is the smoothed, robot-centric local map. The blue arc and the green arrow (pointing from the red line to the white line) are the sub-goal finder and the chosen sub-goal for the omnidirectional CLF RRT*, respectively. The red shows the locally optimal path.

9.5.1 Elements of the Overall Planning System

The overall objective of the planner system is to replace the commonly used waypoint-following or path-tracking strategies with a family of closed-loop feedback control laws that steer the robot along a sequence of collision-free sub-goals leading to the final goal. In simple terms, as in [234–236], we populate the configuration space with a discrete set of feedback control laws that steer the robot from local chart about a sub-goal to the sub-goal itself. The collision free property is handled by the low-frequency planner at the current time. Others have used Control Barrier Functions (CBFs) for this purpose [248–251, 277, 278]. A Finite-State Machine (FSM) is integrated into the low-frequency planner to handle high-level mission requirements such as turning left at every intersection. The planning system is implemented with multi-threading in C++ based on the ROS library [81]. One thread is for the planning and the other is for the reactive thread, as illustrated in Fig. 9.7.

The planner assumes the initial robot pose, a final goal, and real-time map building are provided. It is assumed that the initial robot pose and final goal are initialized in an otherwise featureless metric map, with the robot's initial pose as the origin. The featureless map is filled in by the real-time mapping package [50, 217, 218] based on collected LiDAR and/or camera data.

9.5.2 Planning Thread

The planning thread deals with short-range planning (less than 20 meters) at a frequency of 5 to 10 Hz. It includes a robot-centric local map, our omnidirectional CLF-RRT* algorithm of

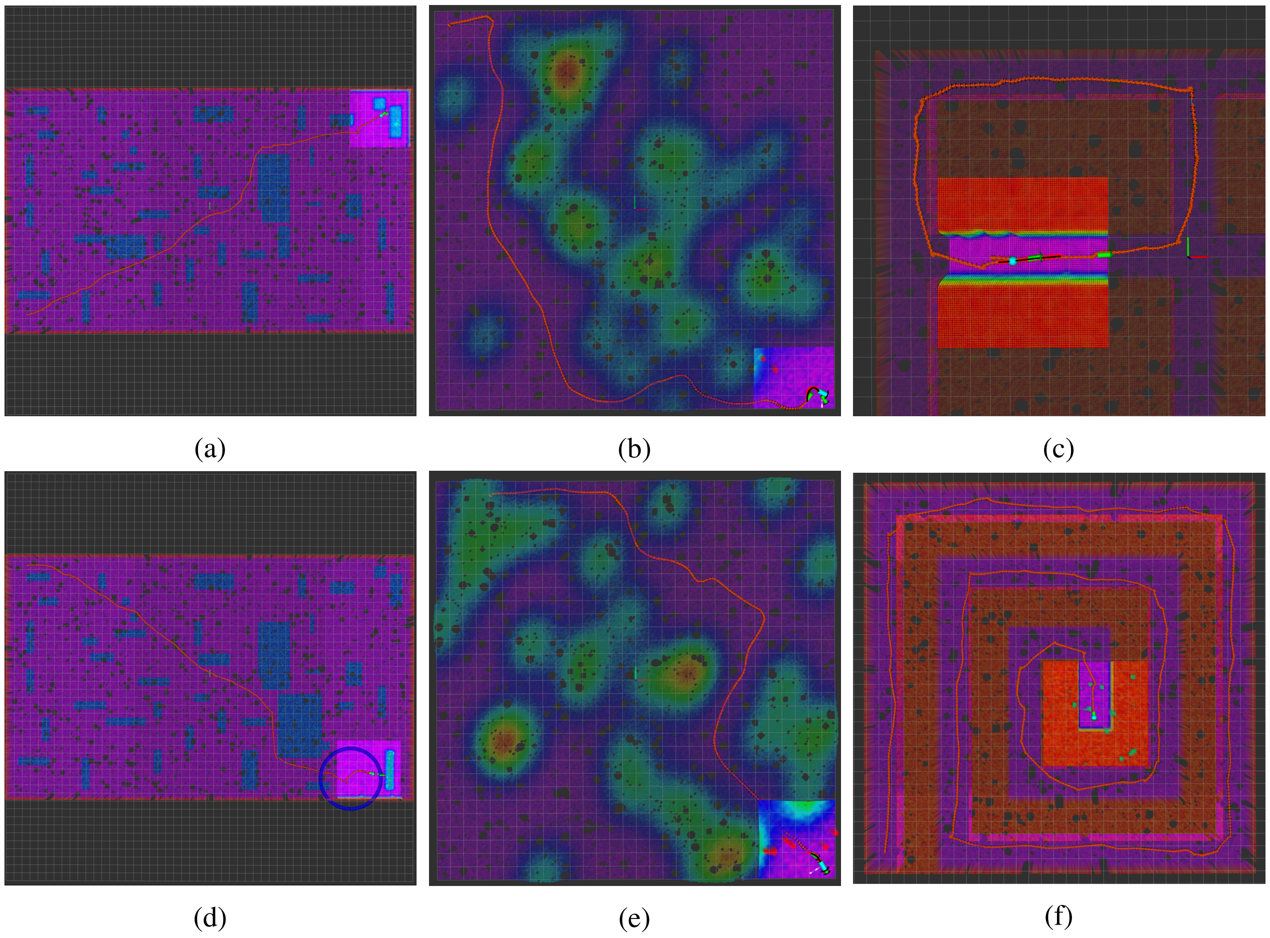


Figure 9.9: Simulated scenes and results obtained with the proposed reactive planning system. On the left are cluttered indoor scenes with obstacles and holes, in the middle are noisy undulating outdoor terrains, and on the right, are high-level missions. Each grid in a map is 1×1 meter. The simulated robot is based on the ALIP model and accepts piece-wise constant inputs at the beginning of each step, is used in all simulation. The robot's initial pose and position of the final goal were hand selected. The highlighted areas show the local maps being provided to the robot 8×8 for left and middle columns and 9×9 for the right column. In each case, the planner guided the robot to the goal. Animations of the simulations are available at [267]. All the figures are vector graphics, so one can enlarge in the browser for best viewing.

Section 9.3, cost computation, a sub-goal finder, and a finite-state machine.

9.5.2.1 Robot-centric Local Map and Cost Computation

Figure 9.8(a) shows the robot-centric multi-layer local map (highlighted area), which crops a sub-map centered around the robot's current position from the global map provided by the mapping algorithm. The local map computes additional useful information such as terrain slope (local gradient) which is useful for assigning cost. Moreover, other necessary operations for different experiment scenes such as applying the Bresenham algorithm [279] to remove walkable area behind glass walls can be computed in this step, see Sec. 9.6.5. Additionally, terrain information such as

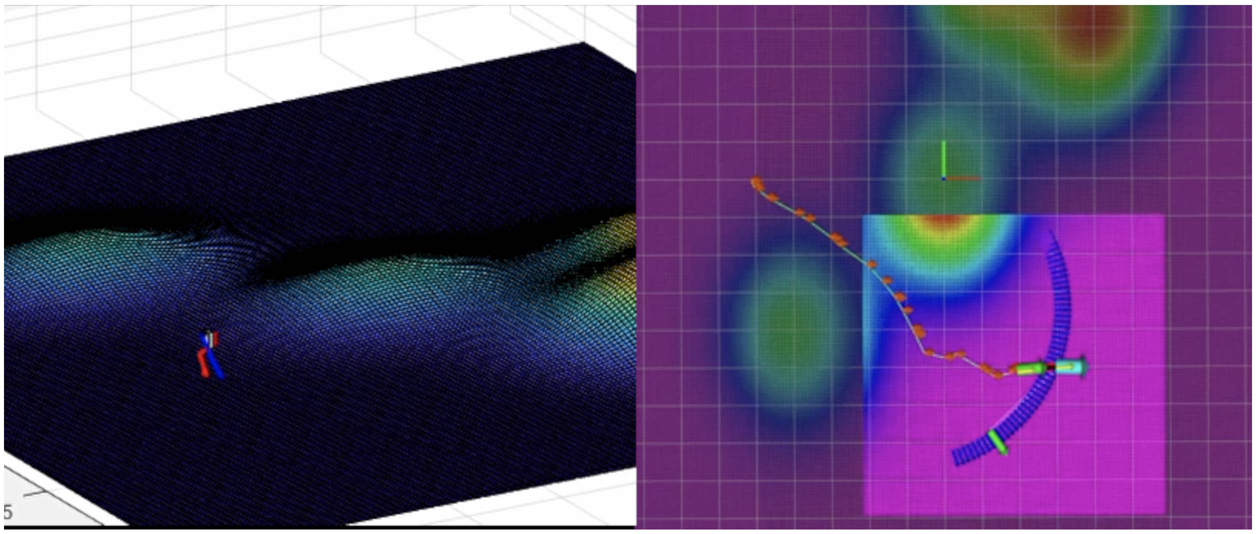


Figure 9.10: A simulation of a C++-implementation of the reactive planner on full-dynamic model of Cassie, which accounts for all 20 degrees of freedom of the robot in Matlab-SimMechanics and includes a 3D terrain model. The reactive planning system receives the pose of the simulated Cassie via User Datagram Protocol (UDP). Cassie's simulator receives and executes the resulting control commands via UDP. The planning system successfully takes the simulated Cassie to the goal without falling. An animation is available at [267].

slopes, frictions, or staircase detection can be sent to a terrain-aware low-level controller [273]. The computations with the local map are efficient compared to processing the full map.

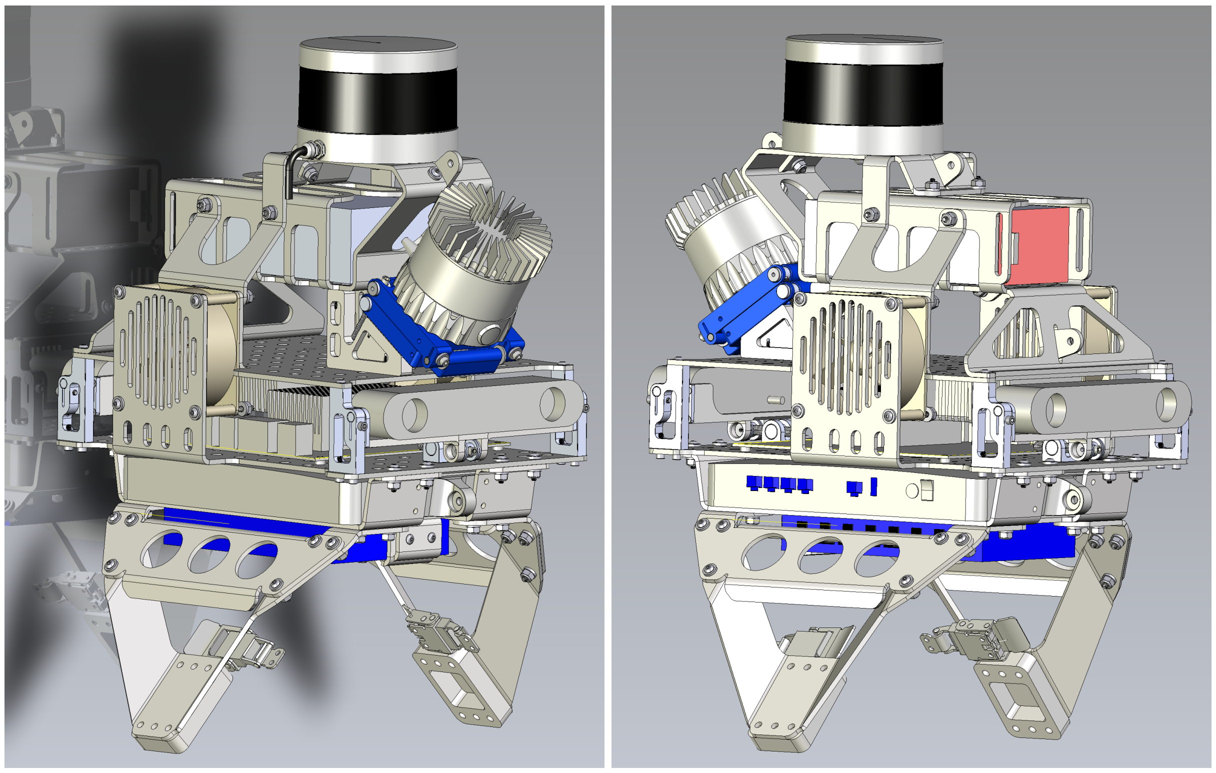
Remark 36. *In the experiment videos, it can be seen that we covered many of the glass walls with paper to prevent LiDAR penetration and the labeling of the space behind the glass walls as walkable. However, some LiDAR measurements still penetrate the glass (such as the bottom part of Fig. 17) and resulting in unwanted walkable regions behind the glass; these are removed by the Bresenham algorithm [279].*

9.5.2.2 Anytime Omnidirectional CLF-RRT* Planner

The anytime feature is a direct result of using RRT* as a planner. The algorithm can be queried at anytime to provide a suboptimal path comprised of wayposes, which the CLF (9.5) turns into real-time feedback laws for anytime replanning.

9.5.2.3 Sub-goal Finder and Finite-State Machine

Ideally, a global planner [280–282] is present to guide the robot to a distant goal, which may not be viewable at the time of mission start [281]. In relatively simple situations such as that shown in Fig. 9.8 and Fig. 9.9, it is sufficient to complete many of short-term missions by positioning a sub-goal (green arrow) at the lowest cost (cost-to-come + cost-to-goal) on an arc (blue arrows) to guide the robot to the final goal. This sub-goal finder is also used as a finite-state machine to handle



(a) (b)

Figure 9.11: The CAD of the sensor suite. The left shows the front view of the sensor suite and the right shows the back.

high-level missions such as making turn selections at intersections, or determining if there is an intersection; see Sec. 9.6.5. It is emphasized that the final path is connected by several sub-goals determined locally by the sub-goal finder. In the future, the sub-goal finder will be replaced with a global planner to achieve globally optimal paths.

Remark 37. *A sub-goal is essentially an intermediate goal with the lowest cost (cost-to-come and cost-to-goal) in the local map and is determined by the FSM. It is emphasized that the local map contains all the available information at that specific timestamp. Therefore, the sub-goal is always observable because it is within the local map. Sub-goals are needed for planning systems to reach a final goal, which might be* ***not observable*** *by the sensors on the robot from its current position and orientation. Piecing together trajectories from one sub-goal to another results in a trajectory from the initial position to the final destination. It is emphasized again that the sub-goals are determined by the FSM based on the current available information in the local map and therefore the overall trajectory may not be globally optimal. To achieve a globally optimal path, the FSM should eventually be replaced by a global path planner [280–282].*

Remark 38. *Although each vector field associated to a CLF is continuous (even smooth), switching among CLFs can induce discontinuity. It is important to keep unchanged the way-pose and CLF*

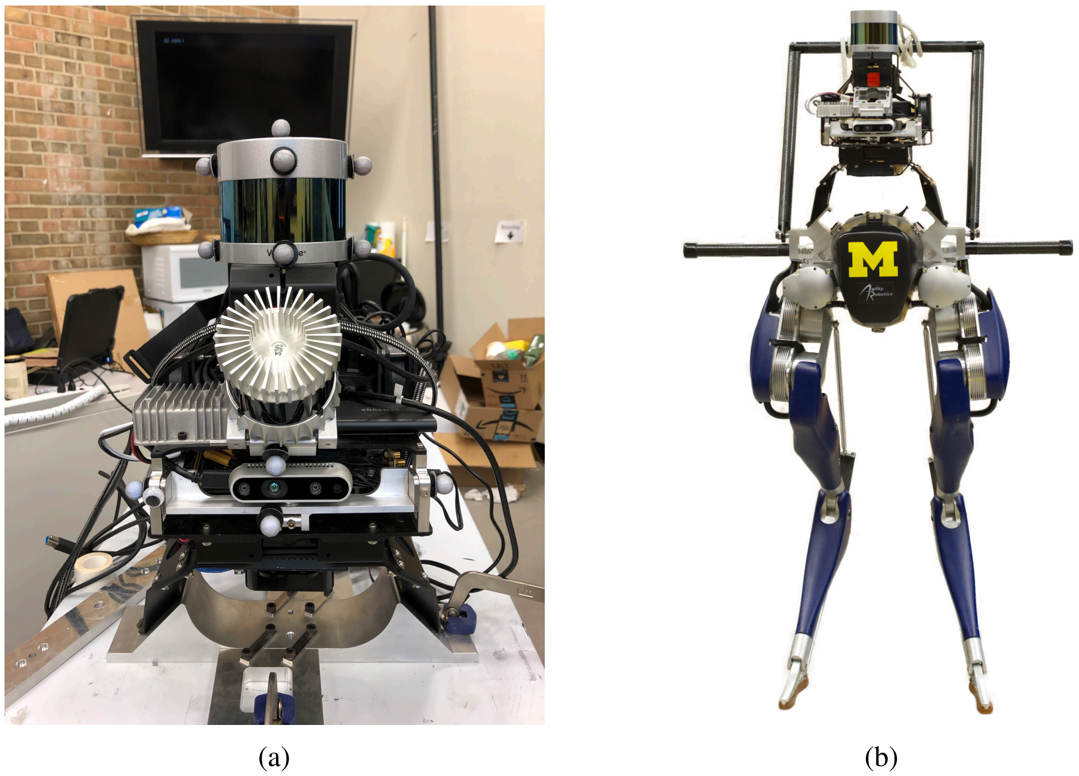


Figure 9.12: The left shows the sensor suite with different sensors, and the right shows the sensor suite mounted on Cassie Blue.

combination the robot is currently targeting so as to ensure continuity. Updates can be made to further way-pose and CLF pairs in the path, but not the current ones; see Sec. 9.6.6 for more details.

9.5.3 Reactive Thread

The work in [225–233] provided a significant alternative to the standard path tracking. Their *high frequency reactive planners* create a vector field on the configuration space whose integrals curves (i.e., solutions of the vector field) provide alternative paths to the goal. When the robot is perturbed, it immediately starts following the new path specified by the vector field, instead trying to asymptotically rejoin the original path. The vector field is in essence an instantaneous re-planner.

In the reactive planner of [226, 228], the vector field arises from the gradient of a potential function defined on the configuration space. Here, we use the solutions of the closed-loop system associated with the CLF in (9.5) to define alternative paths in the configuration space. In essence, our feedback functions (9.12) and (9.11) provide instantaneous re-planning of the control commands for the omnidirectional model (9.3). This reactive planner can be run at 300 Hz in real-time.

The reactive thread is a reactive planner, in which the motion of the robot is generated by a vector field that relies on a closed-loop feedback policy giving controller commands in real-time as a function of the instantaneous robot pose. In other words, the reactive planner utilizes the proposed

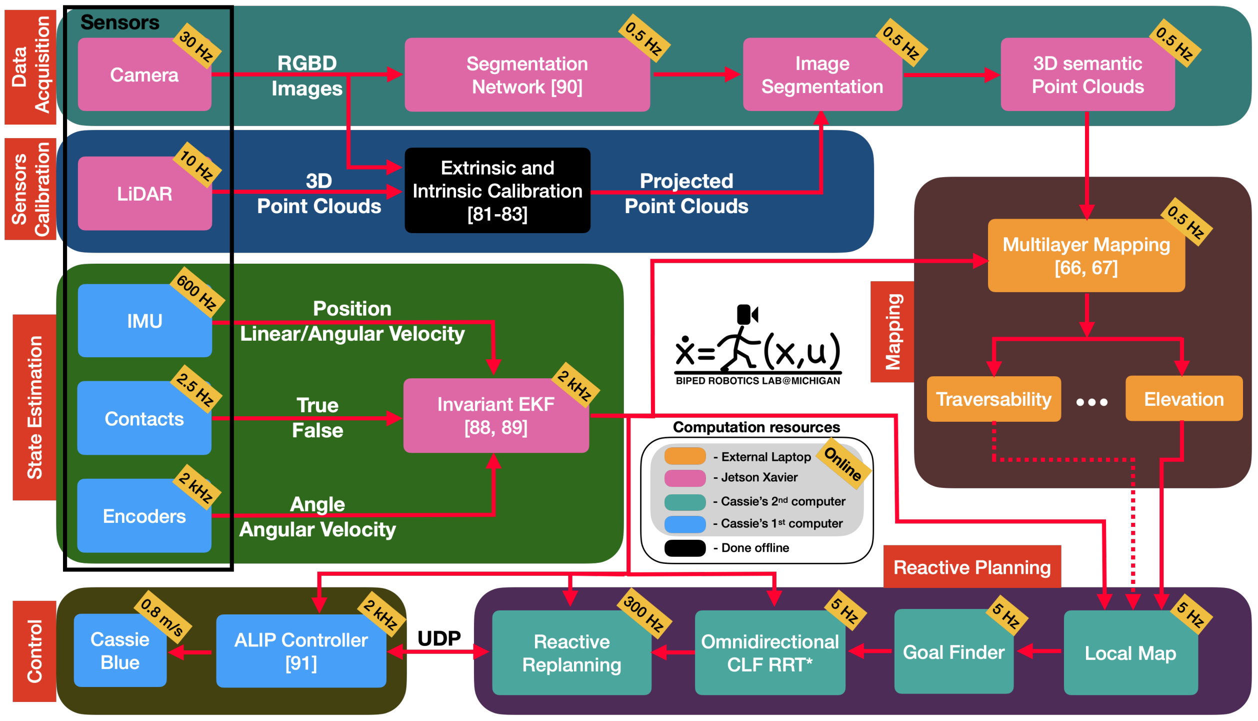


Figure 9.13: Illustration of how the various processes in the overall autonomy system are distributed and their computation frequencies. The larger boxes indicate various modules such as Data Acquisition, Planning, and Control. The smaller boxes are colored according to the processor that runs them.

CLF described in Sec. 9.3 to adjust controller commands automatically when the robot deviates from the optimal path. This thread steers the robot to the optimal path at 300 Hz.

Remark 39. *The "timing" of Cassie's foot placement is inherently event-driven and stochastic. Even though a step cycle may be planned for 300 ms, variations in the terrain and deviations of the robot's joints from nominal conditions result in foot-ground contact being a random variable, with a mean of roughly 300 ms. Running the reactive planner at anything over 100 Hz essentially allows that Cassie's gait controller, which runs at 2 kHz, is accepting the most up-to-date commands from the planner, even if a every other messages is lost over UDP transmission.*

9.6 Simulation and Experimental Results

The proposed reactive planning system integrates a local map, the omnidirectional CLF-RRT*, and fast replanning from the reactive thread. We performed three types of evaluation of the reactive planning system.

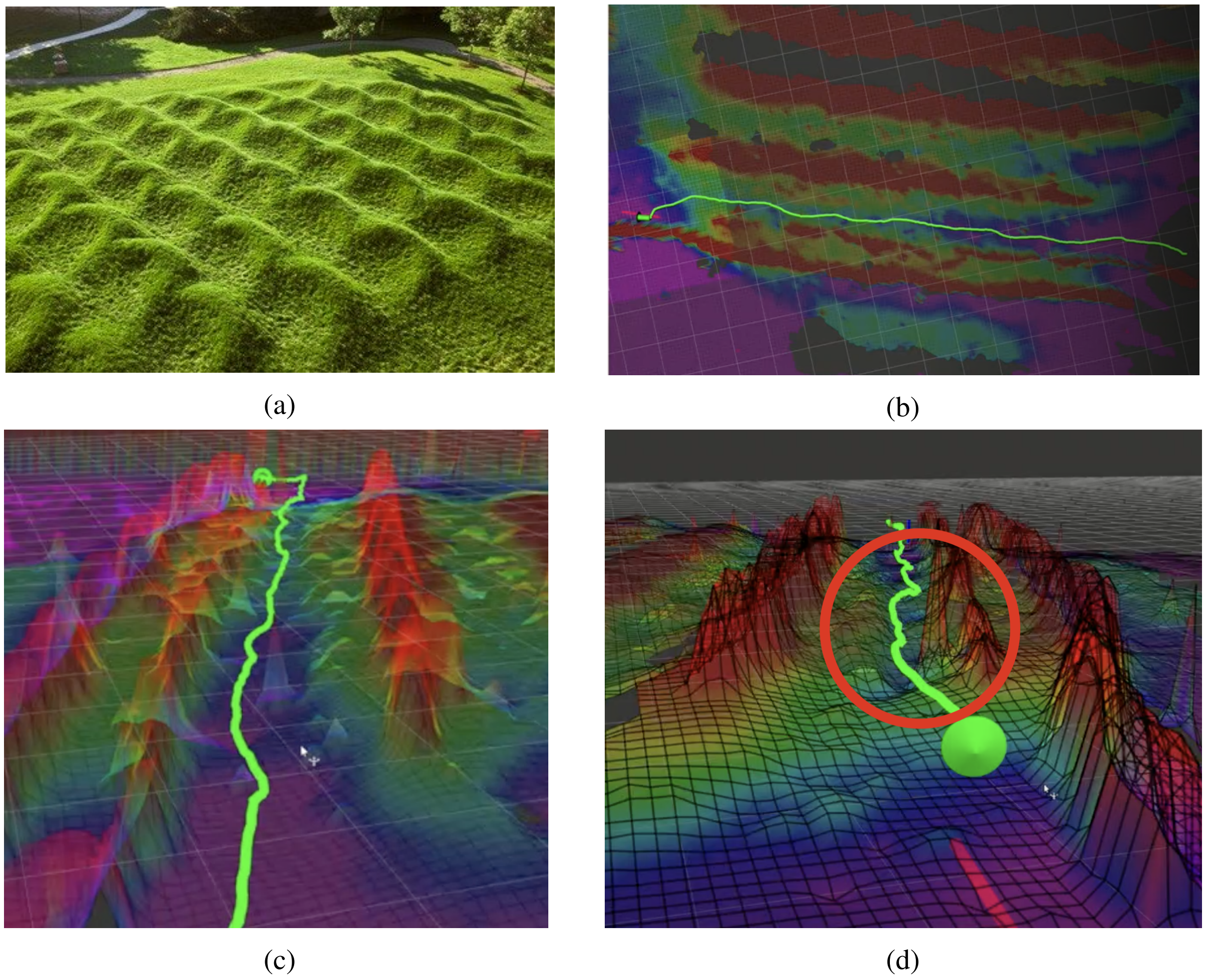

(a) (b) (c) (d)

Figure 9.14: Experimental results on the Wave Field. The top-left shows the experiment terrain, the Wave Field, on the North Campus of the University of Michigan. The top-right shows a bird's-eye view of the resulting trajectory from the reactive planning system. The bottom-left shows a back-view of the trajectory produced by the planning system as Cassie Blue walks in a valley (highly traversable area) of the Wave Field. The bottom-right demonstrates the planning system avoiding areas of higher cost. The red peaks are from the experimenters walking alongside Cassie.

9.6.1 Angular Linear Inverted Pendulum (ALIP) Robot with Simulated Challenging Outdoor Terrains and Indoor Cluttered Scenes

We first ran the reactive planning system on several synthetic environments, in which an ALIP robot model [216, 269] navigated several simulated noisy, patchy, challenging outdoor terrains as well as cluttered indoor scenes. The ALIP robot successfully reached all the goals in different scenes. We tested the system on more than 10 different environments, both indoor and outdoor with and without obstacles. Due to space limitations, we only show the results of six simulations in Fig 9.9; see our GitHub [267] for videos and more results.

Remark 40. *The ALIP robot [216, 269] takes piece-wise constant inputs from the reactive planning system. Let g, H, τ be the gravity, the robot's center of mass height, and the time interval of a swing*

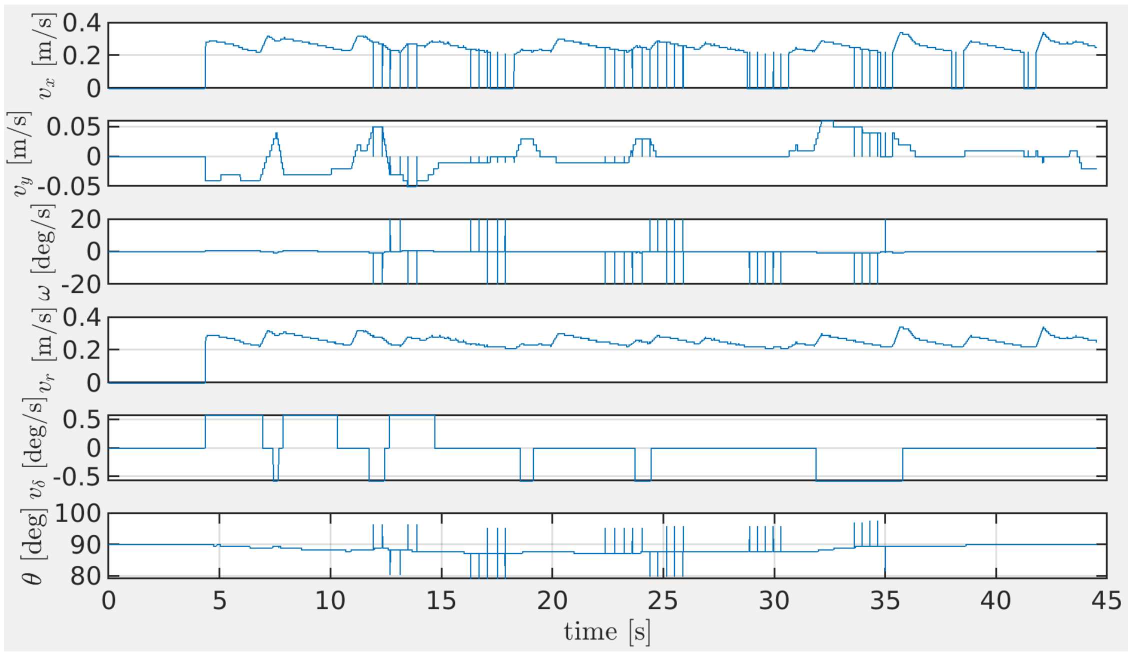


Figure 9.15: Control commands sent to Cassie Blue. UDP packet drops show up as vertical lines. When these occur, the controller uses the previous value.

phase, respectively. The motion of an ALIP robot on the x-axis is defined as

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} \cosh(\xi) & \frac{1}{\rho}\sinh(\xi) \\ \rho\sinh(\xi) & \cosh(\xi) \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} 1-\cosh(\xi) \\ -\rho\sinh(\xi) \end{bmatrix} p_x, \tag{9.19}$$

where x_k and $\dot{x}_k$ are the contact position and the contact velocity of the swing foot on the x-axis, p_x is the Center of Mass (CoM) on the x-axis of the robot, $\xi = \rho\tau$ and $\rho = \sqrt{g/H}$. Similarly, the motion of the robot on the y-axis can be defined.

Remark 41. *Even though a full global map is given in each simulation environment, only the information in the local map is given to the planning system at each timestamp. The path generated from omnidirectional RRT* is asymptotically optimal within the local map, for the given time window. It is emphasized that no global information is provided to the planner which is why the resulting trajectory from the initial point to the goal may not be the shortest path.*

9.6.2 Validation of Control Command Feasibility via a Whole-body Cassie Simulator

To ensure the control commands from the reactive planning system are feasible for Cassie-series bipedal robots, we sent the commands via UDP from ROS [81] C++ to Matlab-Simmechanics, which simulates a 20 DoF of Cassie, using footfalls on the specified terrain. The simulator then sent back the pose of the simulated Cassie robot to the planning system to plan for the optimal path

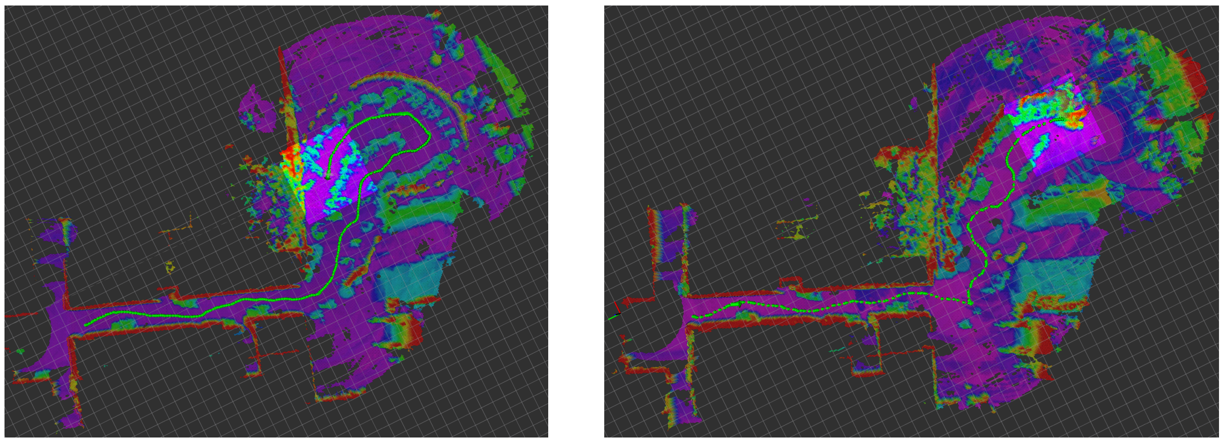


Figure 9.16: The resulting trajectories on the first floor of the Ford Robotics Building. The map colored by height was built online while Cassie was guided by the planning system. The green lines are the resulting trajectories and green patches in the map are tables and furniture considered obstacles.

via UDP. The planner system successfully took the simulated Cassie to the goal without falling, as shown in Fig. 9.10.

9.6.3 Perception Suite Design and Hardware System Integration

To allow the robot to perceive its surroundings under different lighting conditions and environments, we designed a perception suite that consists of an RGB-D camera (Intel RealSense™ D435) and a *32-Beam Velodyne ULTRA Puck LiDAR*. Two fans cool a Jetson AGX Xavier with GPU. A router, a USB hub, and an internet switch are utilized for communication from users and the robot to the perception suite. Finally, a 12-volt Lithium Polymer battery powers up all the sensors. Figure 9.11 shows the design of the full sensor suite and the step files are available at [45].

The weight of the sensor suite, with batteries and everything included, is 8.5 kilograms. We use an industrial-grade router and internet switch to ensure stable connections among the sensors, GPU, and the secondary computer on the Cassie robot. Figure 9.12 shows all the sensors mounted on the perception suite.

9.6.4 Software System Integration for Real-time Deployment

System integration is critical for real-time use. Figure 9.13 shows the integrated system, distribution, and frequency of each computation. In particular, the sensor calibrations are performed via [9, 21, 39, 40, 62, 86, 101, 211–213]. The Invariant Extended Kalman Filter (InEKF) [61, 214] is used to estimate the state of Cassie Blue at 2 kHz. Images are segmented via MobileNets [283] and a LiDAR point cloud is projected back to the segmented image to produce a 3D segmented point cloud. The resulting point clouds are then utilized to build a Multi-Layer Map (MLM) [50, 217, 218, 284, 285]. The reactive planning system then crops the MLM around the

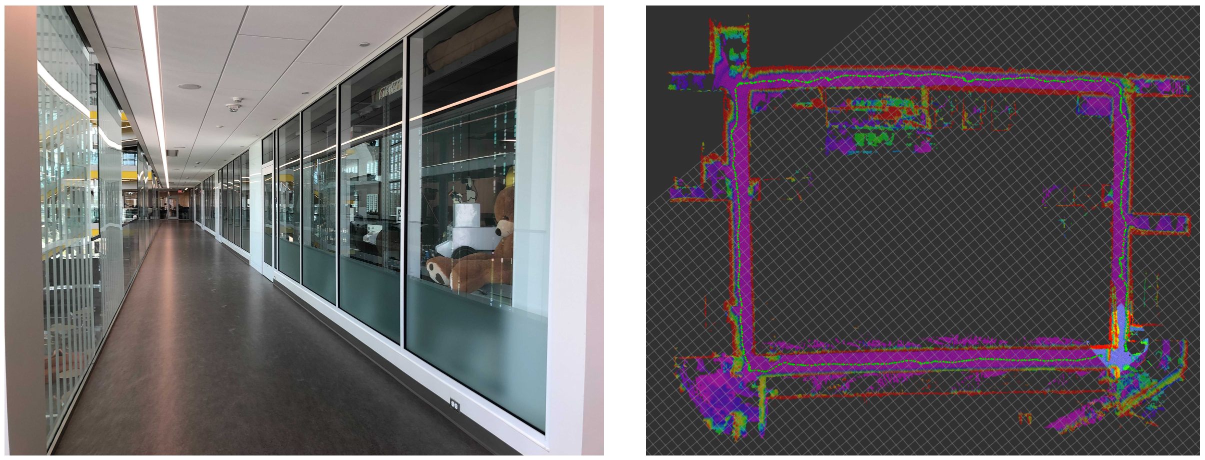


Figure 9.17: Experimental results on the second floor of the Ford Robotics Building. The top shows glass walls, which lead to refection of LiDAR lasers and creating walkable area behind the wall. The bottom illustrates the resulting (200 meter) trajectory produced by the planning system as Cassie Blue walks. It is remarked that the plot is based on pure odometry so no loop closure is performed [214].

robot position to create a local map and performs several operations to acquire extra information, as described in Sec. 9.5.2. Additionally, the reactive planner receives the robot poses from the InEKF at 300 Hz to adjust the control commands that guide the robot to the nominal sub-poses via the proposed CLF; see Sec. 9.3 and Sec. 9.5.3. The control commands are then sent to Cassie Blue's gait controller [51, 215, 216] via UDP.

9.6.5 Full Autonomy Experiments with Cassie Blue

We conducted several indoor and outdoor full autonomy experiments with Cassie Blue. The running cost in (9.14) was selected as

$$C(x_t, y_t, z_r) := C_e(x_t, y_t) + 0.5C_s(x_t, y_t) + 0.3(C_e(x_t, y_t) - z_r), \tag{9.20}$$

where $C_e(x_t, y_t), C_s(x_t, y_t)$ are the elevation and the magnitude of the gradient at a point (x_t, y_t), respectively.

9.6.5.1 The Wave Field

We achieved full autonomy with Cassie Blue on the Wave Field, located on the North campus of the University of Michigan, an earthen sculpture designed by Maya Lin [286]; see Fig. 9.14(a). The Wave Field consists of sinusoidal humps with a depth of approximately 1.5 m from the bottom of the valleys to the crest of the humps; there is a second sinusoidal pattern running orthogonal to the main pattern, which adds 25 cm ripples peak-to-peak even in the valleys. Figure 9.14(b) shows the top-view of the resulting trajectory of the reactive planning system. The planning system guided Cassie Blue to walk in the valley (the more traversable area), as shown in Fig. 9.14(c). The planning system navigated Cassie Blue around a hump that protrudes into one of the valleys, as shown in Fig. 9.14(d). Figure 9.15 shows the control commands sent to Cassie Blue. This experiment was presented in the Legged Robots Workshop at IEEE International Conference on Robotics and Automation (ICRA) 2021; the video can be viewed at [287]. The video of the Wave Field experiment is uploaded and can be found at [49] and [267].

Remark 42. *We conducted most of the experiments at night because there are fewer people walking around. Because of the intrinsic properties of LiDAR sensors, ambient lighting does not affect their measurements; see [39] for more details about LiDAR properties.*

9.6.5.2 Turn left at detected intersections of corridors and avoid obstacles

We conducted two experiments of this type on the first floor of the FRB at the University of Michigan. The experiments' scenes consist of corridors and an open area cluttered with tables and couches, which are considered as obstacles (height greater than 30 cm from the mapping package), as shown in Fig. 9.16. To detect the intersections of the corridors, we group walkable segments within a ring around Cassie Blue. Each walkable segment either links with an existing cluster or creates a new cluster via the single-linkage agglomerative hierarchical clustering algorithm[3] [133], where the linkage criteria is the Euclidean distance. If there are more than two clusters of walkable segments, we consider there exists an intersection. Subsequently, Cassie Blue makes a left turn at the detected intersection. After exiting the corridors, the robot reaches an open area cluttered with furniture and performs obstacle avoidance. Under the proposed reactive planning system, Cassie Blue completed the experiments without falling or colliding with obstacles. The total distance traveled was about 80 meters. The experiment videos can be viewed at [288] and [267].

9.6.5.3 Turn right at detected intersections of corridors and return to the initial position

This experiment was conducted on the second floor of the FRB and the experiment scene contains four long corridors with glass walls. Some of the LiDAR beams penetrated glass a certain points along the corridors, causing the mapping algorithm to consider area behind the glass walls as free and walkable. We applied the Bresenham line algorithm [279] to remove the walkable area behind the glass walls. The computation of the Bresenham algorithm is not expensive because it is only applied within the local map, mentioned in Sec. 9.5.2. The proposed reactive planning system successfully guided Cassie Blue back to its initial position, as shown in Fig. 9.17. The total distance traveled was about 200 meters. The experiment videos can be viewed at [289] and [267].

[3]We chose this clustering algorithm because the number of clusters is unknown. Therefore, algorithms like K-Means Clustering [134] cannot be used.

9.6.6 Experiment Discussion

In the two indoor experiments, Cassie exhibited a walk-and-stop motion. Where does it come from? As mentioned in Sec. 9.5, the planning threading runs at 5 Hz. At the k-th update, there will be an optimal path $\mathcal{P}_k$, comprised of a number of way-poses connected by CLFs. Although each vector field associated to a CLF is continuous (even smooth), switching among CLFs can induce discontinuity. This discontinuity induces Cassie's walk-and-stop motion seen in the videos of the indoor experiments. How? At each planning update, the entire tree was being discarded and a new one constructed. In particular, the closest way-pose to Cassie was being re-set every 200 ms, and thus the robot was never allowed to evolve along the integral curves of the vector field.

The solution is straightforward: at the $(k{+}1)$-st planning update, we leave the first unreached way-pose fixed in the path $\mathcal{P}_k$ to ensure continuity. Additionally, to fully utilize the optimal path from the previous update, we keep the current optimal path $\mathcal{P}_k$ as a branch and prune all the samples from the k-th update. This provides a warm start for the $(k{+}1)$-st update, as long as the path $\mathcal{P}_k$ is still valid and collision-free. If a dynamic obstacle has invalidated the path between the robot's current position and the first unreached way-pose, then the entire tree is discarded, as before. With these changes made, we conducted several additional experiments to confirm that it resolves the walk-and-stop movement. The experiments can be viewed at [290] and [267].

9.7 Conclusion and Future Work

We presented a novel reactive planning system that consists of a 5-Hz planning thread to guide a robot to a distant goal and a 300-Hz CLF-based reactive thread to cope with robot deviations. In simulation, we evaluated the reactive planning system on ten challenging outdoor terrains and cluttered indoor scenes. In experiments on Cassie Blue, a bipedal robot with 20 DoF, we performed fully autonomous navigation outdoors on sinusoidally varying terrain and indoors in cluttered hallways and an atrium.

The planning thread uses a multi-layer, robot-centric local map to compute traversability for challenging terrains, a sub-goal finder, and a finite-state machine to choose a sub-goal location as well as omnidirectional CLF-RRT* to find an asymptotically optimal path for Cassie to walk in a traversable area. The omnidirectional CLF-RRT* utilizes the newly proposed Control Lyapunov Function (CLF) as the steering function and the distance measure on the CLF manifold in the RRT* algorithm. Both the proposed CLF and the distance measure have a closed-form solution. The distance measure nicely accounts for the inherent "features" of Cassie-series robots, such as high-cost for lateral movement. The robot's motion in the reactive thread is generated by a vector field depending on a closed-loop feedback policy providing control commands to the robot in real-time as a function of instantaneous robot pose. In this manner, problems typically encountered by waypoint-following and pathway-tracking strategies when transitioning between waypoints or pathways (unsmooth motion, sudden turning, and abrupt acceleration) are resolved.

In the future, we shall combine control barrier functions [248–251, 277, 278] with the CLF in the reactive thread to handle dynamic obstacles. Additionally, the current local map is a 2.5D, multi-layer grid map with fixed resolution; it is also interesting to see how to efficiently represent a continuous local map. An obstacle in the local map is assigned simply by height that the robot cannot step over; how to robustly determine an object is an obstacle or not is also an interesting research. Furthermore, how to extend the CLF to 3D is another interesting area for future research.

Part IV

Conclusion and Future Directions

CHAPTER 10

Conclusion and Contributions

10.1 Summary of Dissertation and Contributions

In this dissertation, we have presented an autonomy system for bipedal robots. In brief, the proposed system enables the robot to

1. perceive surrounding information via the intrinsically and extrinsically calibrated sensor suite (Chapter 2-5),
2. estimate their poses in textureless and completely dark environments through LiDAR-based fiducial markers (Chapter 6),
3. leverage the optimal target shape and a global solver to estimate their poses when the target is 30 meters away (Chapter 7),
4. plan routes to multiple destinations and determine the visiting order of multiple destinations in large-complex graph-based maps (Chapter 8),
5. traverse through unexplored, unstructured environments and undulating terrains (Chapter 9).

We first introduce various types of sensors and the perception suite in Chapter 2. The extrinsic calibration algorithm to estimate the LiDAR to camera transformation is proposed in Chapter 3. The algorithm avoids all together the delicate task of identifying the edge points and parsing them into individual edges of the target. Chapter 4 introduces the global unifying LiDAR intrinsic calibration framework for both spinning LiDARs and solid-state LiDARs. The LiDAR intrinsic parameters are represented as the action of the seven-dimensional matrix Lie group, $\mathrm{Sim}(3)$ and formulated as a QCQP where Lagrangian duality relaxation is used. We mathematically prove that given four targets with appropriate orientations, the proposed model is well-constrained (i.e., a unique answer exists). We then show how to profitably apply efficient, globally convergent algorithms for $\mathrm{SE}(3)$ to determine a solution to our problem in $\mathrm{Sim}(3)$. The automatic calibration system is introduce in Chapter 5.

Next, the very first LiDAR-based fiducial marker system – LiDARTag is developed in Chapter 6. The LiDARTag system detects different sizes of targets in a full point cloud scan at 100 Hz. Furthermore, the system can be operated in a completely dark environment. In Chapter 7, we propose the concept of optimizing target shape to ameliorate problems caused by quantization uncertainty and sparsity of the LiDAR image of a target. The pose estimation problem is formulated so that an existing SDP global solver can be modified to globally and efficiently compute the target's pose. The optimal shape with the global solver achieves centimeter error in translation and a few degrees of error in rotation when the targets are at a distance of 30 meters and partially illuminated.

Later, we present informable multi-objective and multi-directional RRT* (IMOMD-RRT*) in Chapter 8. The system is an anytime iterative algorithm to concurrently solve the multi-objective path planning problem and determine the visiting order of destinations. The system is comprised of an anytime informable multi-objective and multi-directional RRT* algorithm to form a simple connected graph, and a proposed solver that consists of an enhanced cheapest insertion algorithm and a genetic algorithm to solve the Relaxed Traveling Salesman Problem (R-TSP) in polynomial time.

Finally, we propose a novel reactive planning system to smoothly approach the intermediate goals in Chapter 9. The reactive planning system consists of a 5-Hz planning thread to guide a robot to a distant goal and a 300-Hz Control-Lyapunov-Function-based (CLF-based) reactive thread to cope with robot deviations. This reactive planning system allows Cassie to autonomously avoid obstacles, complete high-level missions, and traverse sinusoidally varying terrains.

CHAPTER 11

Future Work

11.1 Extension of Autonomy System

This chapter outlines several ideas to extend the current autonomy stack and provides some preliminary results. In particular, we suggest the following:

1. In Chapter 3, we solve the extrinsic parameters via a local solver. Even though we achieve pixel-level reprojection error, it still has some limitation where a good initial guess has to be provided. We suggest leveraging the global solver mentioned in Chapter 4.5 to avoid incorrect local minimums due to a poor initial guess.

2. In Chapter 8, we develop an IMOMD-RRT* system for high-level robotics path planning on OpenStreetMap (OSM). However, an OSM is not available indoor. As a future work, we propose a light-weight graph-based topological map for robotics indoor navigation.

3. Using the graph-based topological map mentioned above, we suggest the robot localizing into the map and executing for high-level planning as a part of the future work.

4. We develop a CLF-based terrain planner in Chapter 9. The map and obstacles are updated at a lower rate (1-0.1 Hz), whereas the reactive threading is running at 300 Hz. It is an interesting future direction to represent obstacles via Control Barrier Functions (CBFs), and combine the CBFs with the CLF in the reactive thread to cope with dynamic obstacles at a higher rate.

11.2 Global Solver for Extrinsic Calibration of LiDAR and Camera

In this chapter, we leverage the global solver mentioned in Chapter 4.5.3 to avoid incorrect local minimums due to poor initial guess. We first build a LiDAR and camera simulator. Next, we adopt two approaches to re-project 2D camera corners to 3D camera vertices in the camera frame so that 3D point-to-point correspondences can be formed. Finally, we utilize the global solver to estimate extrinsic parameters. We demonstrate preliminary results to show the potential of this idea.

11.2.1 LiDAR and Camera Simulators

To simulate 3D points on a planar target, we generate rays from the simulated LiDAR sensor, and define a “target” point as the point at which the ray intersects the target. After locating the exact LiDAR returns on the target, based on the LiDAR type, we then add uncertainty to the returns and report them as measured data, as shown in Fig. 11.1.

The camera is modeled as a pin-hole camera. The camera simulator takes 3D points measured in the world frame and projects the points onto the 2D image plane. Figure 11.2(a) shows the configuration of a LiDAR and a camera. The camera image of the target is shown in Fig. 11.2(b).

Remark 43. *The LiDAR simulator first finds all intersection points with the (infinite) plane defined by the target. To determine if a point on the plane is within the boundary of the target polygon is a well-known problem in computer graphics, called the PIP problem [98, 99, 291–295]. The Winding Number algorithm [100] is implemented in this simulator. If the winding number of a point is not zero, the point lies inside the boundary; otherwise it is outside.*

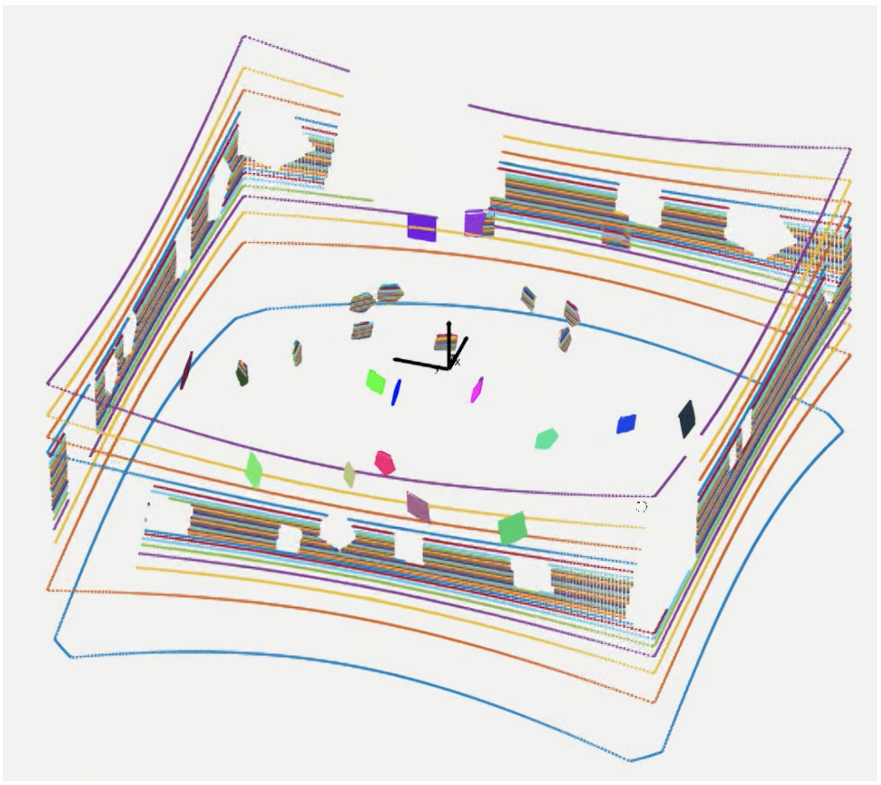
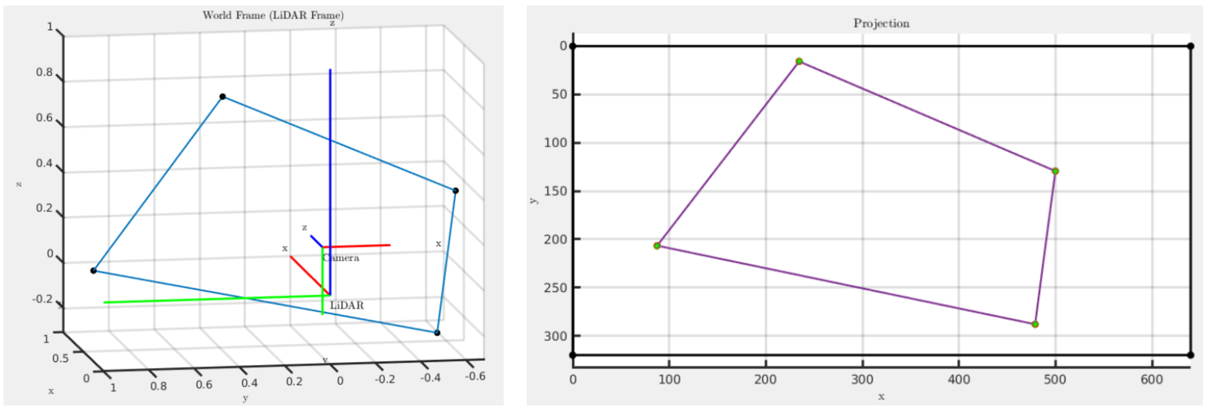


Figure 11.1: Illustration of the LiDAR simulator in the scene with planar objects. The outer boundary is the LiDAR measurements on the walls.



(a) (b)

Figure 11.2: (a) shows the configuration of a LiDAR-camera pair. (b) demonstrates the image of the target.

11.2.2 3D Points Triangulation from 2D Image Features with Geometry Constraints

This section provides two methods to compute 3D target vertices from 2D image corners in the image coordinate.

11.2.2.1 Direct Solution by Grunert Algorithm

To estimate the 3D target vertices ${}_C X_i$ from the 2D image corners ${}_C Y_i$, we implemented the direct solution by the Grunert Algorithm [296]. Let K be the intrinsic matrix of the camera [155] defined as

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{11.1}$$

Let $\mathcal{V} = \{v_i\}_{i=1}^4$ be the set of vectors that emanate from the camera origin O to ${}_C X_i$, as shown in Fig. 11.3. We can then compute v_i by

$$v_i = \mathbf{K}^{-1} {}_C Y_i'; \tag{11.2}$$

see [70, Eq. (1.14)] for more details. By the law of cosines, we also have

$$\begin{aligned} d^2 &= s_1^2 + s_2^2 - 2 s_1 s_2 \frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|} \\ d^2 &= s_2^2 + s_3^2 - 2 s_2 s_3 \frac{v_2 \cdot v_3}{\|v_2\| \|v_3\|} \\ d^2 &= s_3^2 + s_4^2 - 2 s_3 s_4 \frac{v_3 \cdot v_4}{\|v_3\| \|v_4\|} \\ d^2 &= s_4^2 + s_1^2 - 2 s_4 s_1 \frac{v_4 \cdot v_1}{\|v_4\| \|v_1\|} \end{aligned}, \tag{11.3}$$

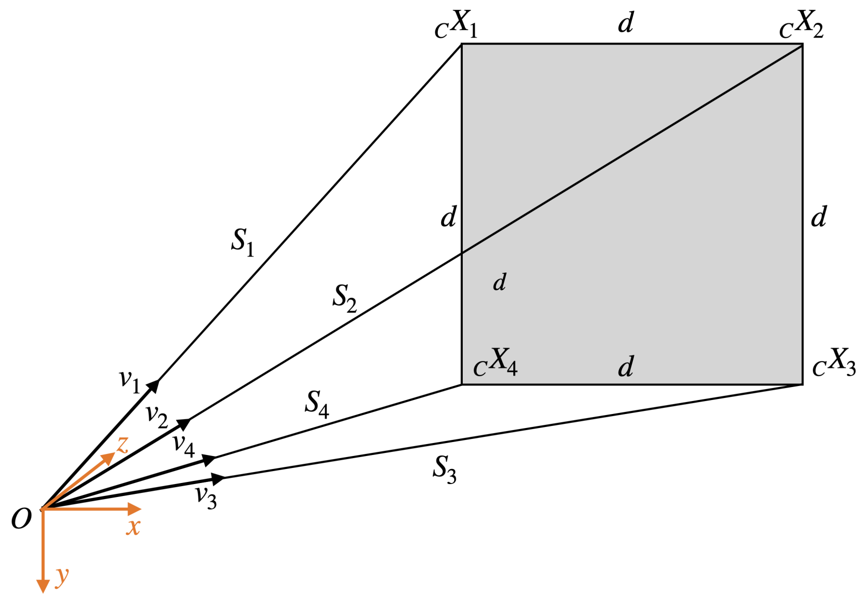


Figure 11.3: The geometry of the four target points and camera origin.

where s_i are unknown depth values and d is the target size. From (11.2) and (11.3), we can determine $\{s_i\}_{i=1}^4$. Finally, the target vertices ${}_CX_i$ in the camera coordinate system are determined by

$$ {}_CX_i = s_i \frac{v_i}{\|v_i\|}. \tag{11.4}$$

However, the Grunert Algorithm [296] does not consider geometry constraints where the target dimension and the target planar geometry are known. In particular, the noise in the intrinsic matrix $\mathbf{K}$ will lead to non-planar target vertices ${}_CX_i$ in the camera frame. We propose an optimization problem that takes geometry constraints into account while estimating the target vertices in the next section.

11.2.2.2 Optimization for Target Vertices with Geometry Constraints

We employ the L_1-inspired method, mentioned in Chapter 3.2.2. Similarly, let H_C^T be the rigid-body transformation from the template in the camera frame with vertices $\{{}_C\overline{X}_i\}_{i=1}^4$, onto the actual camera vertices $\{{}_C\widetilde{X}_i\}_{i=1}^4$, namely,

$$ {}_C\widetilde{X}_i = \begin{bmatrix} \widetilde{x}_i \\ \widetilde{y}_i \\ \widetilde{z}_i \end{bmatrix} = H_T^C \overline{X}_i, \quad 1 \leqslant i \leqslant 4. \tag{11.5}$$

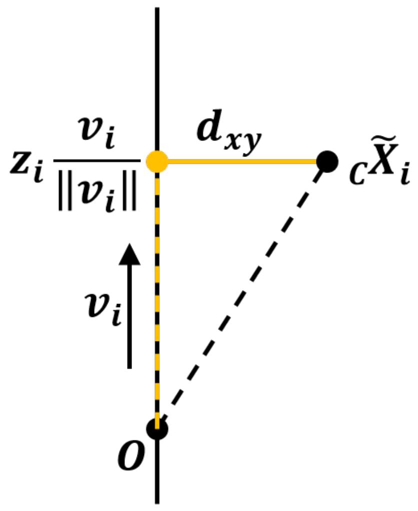


Figure 11.4: Illustration of the distance between rays and estimated target vertices parallel to the image plane.

The optimization problem is defined as

$$\begin{aligned}\mathbf{H}_T^{C*} &:= \underset{R_T^C, t_T^C}{\operatorname{argmin}} \sum_{i=1}^{4n} d_{xy}^2(v_i, \mathbf{H}_T^C \bar{\mathbf{X}}_\mathrm{i}') \\ &= \underset{R_T^C, t_T^C}{\operatorname{argmin}} \sum_{i=1}^{4} \left\| \tilde{z}_i \frac{v_i}{\|v_i\|} -_C \tilde{\mathbf{X}}_i \right\|^2,\end{aligned} \tag{11.6}$$

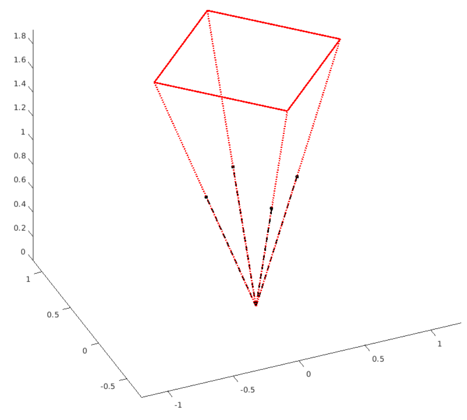
where v_i are rays emanating from the camera origin O, defined in (11.2), and d_{xy} is the distance between rays and target vertices parallel to the image plane, as shown in Fig. 11.4. Finally, the estimated target vertices ${}_CX_i^*$ in the camera frame is determined by

$${}_CX_i^* = \mathbf{H}_T^{C*} \bar{X}_i, \quad 1 \leqslant i \leqslant 4. \tag{11.7}$$

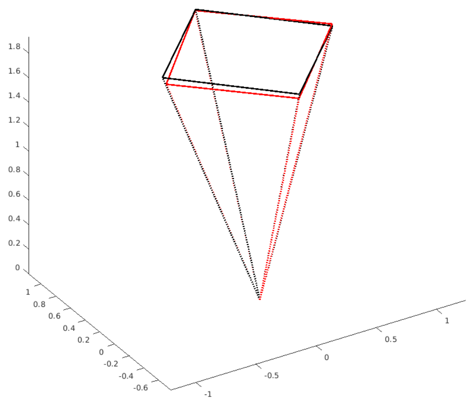
11.2.3 Globally Optimal Solution of Extrinsic Parameters via Point-to-Point Correspondences

Given the correspondences of the camera vertices and LiDAR vertices, we seek the transformation H_C^L that minimizes the point-to-point distance:

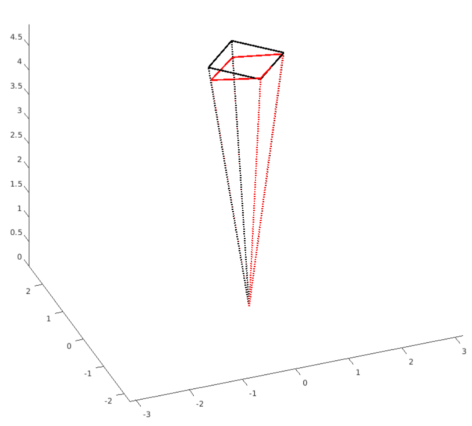
$$j_i(H_C^L;\ {}_C\overline{X}_i, {}_L\overline{X}_i) = \|H_C^L \circ {}_C\overline{X}_i - {}_L\overline{X}_i\|_2^2. \tag{11.8}$$



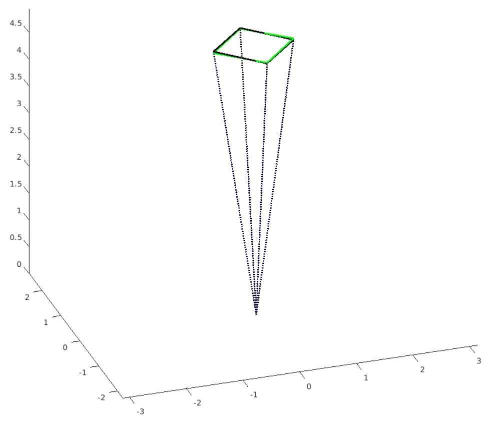
(a)



(b)



(c)



(d)

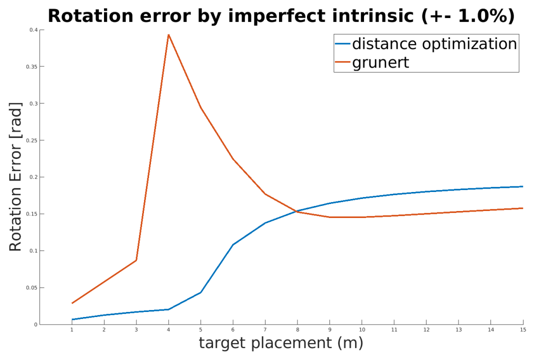
Figure 11.5: (a) shows the result of vertices estimation with perfect intrinsic parameters. (b) and (c) show the results of target vertices estimation determined by Grunert method with imperfect intrinsic parameters($\pm 1.0\%$ of noise). (d) shows the result of our optimization method with the same imperfect intrinsic parameters. The black indicates the ground truth, the red is estimation from the Grunert algorithm, and the green is estimation results from the L_1-inspired method.

The total fitting error is defined as

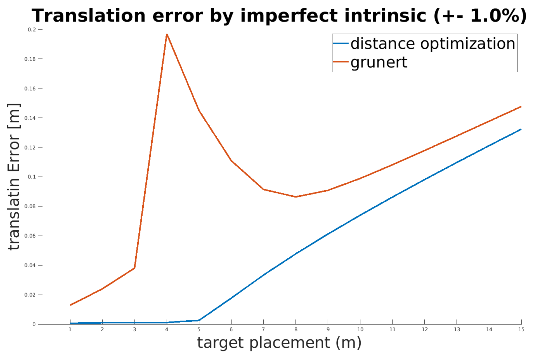
$$J(H_C^L;\ {}_C\overline{X}_i, {}_L\overline{X}_i) = \underset{R_L^C, t_L^C}{\arg\min} \sum_{i=1}^{N} j_i(H_C^L;\ {}_C\overline{X}_i, {}_L\overline{X}_i), \tag{11.9}$$

where N is the number of correspondences.

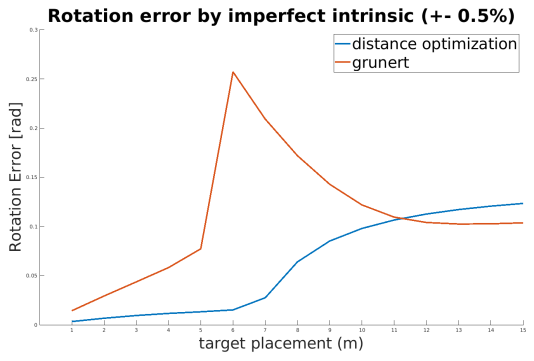
To minimize (11.9), we adopt techniques that were used to globally solve 3D registration or 3D SLAM [1, 82–84] where the problem is formulated as a QCQP, and the Lagrangian dual relaxation is used. The relaxed problem becomes a SDP and convex. The problem can thus be solved globally



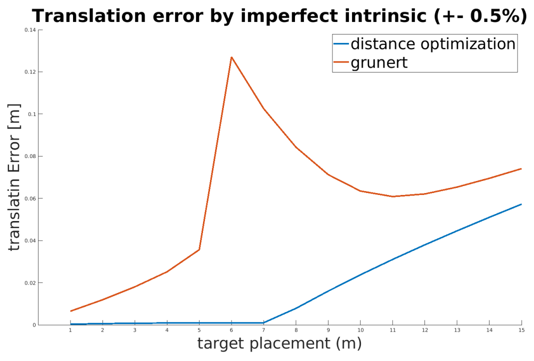
(a)



(b)



(c)



(d)

Figure 11.6: The translation and rotation error are computed from (11.10). (a) and (b) are the estimation results with $\pm 1\%$ noise applied to the ground truth intrinsic parameters. (c) and (d) are the estimation results with $\pm 0.5\%$ noise applied to the ground truth intrinsic parameters.

and efficiently by off-the-shelf specialized solvers [85]. As shown in [84], the dual relaxation is empirically always tight (the duality gap is zero). Therefore, the transformation H_C^L between the camera and the LiDAR can be globally obtain.

11.2.4 Simulation Results

The result of vertex estimation is evaluated on translation e_t in $\mathbb{R}^3$ and rotation e_r on $\mathrm{SO}(3)$, separately. In particular, e_t and e_r are computed by

$$e_t := \|t - \tilde{t}\| \quad \text{and} \quad e_r := \|\mathrm{Log}(R\tilde{R}^\mathsf{T})\|, \tag{11.10}$$

where $\| \cdot \|$ is the Euclidean norm, $\tilde{\cdot}$ is the estimated quantity, R and t are the ground truth rotation and translation, respectively, and $\mathrm{Log}(\cdot)$ is the logarithm map in the Lie group $\mathrm{SO}(3)$.

11.2.4.1 3D Camera Vertices from Grunert Algorithm and L_1-inspired Optimization

We first apply random noise ($\pm 1\%$) to the ground truth intrinsic camera parameters. Figure 11.5 shows qualitative results of the estimation from Grunert algorithm and the proposed L_1-inspired optimization. The quantitative results are shown in Fig. 11.6.

11.3 Light-Weight Topological Map and Localization for Robotics Navigation

In Chapter 8, we develop an IMOMD-RRT* system for high-level robotics path planning on OpenStreetMap (OSM) [173]. However, OSM is not available indoor. Therefore, building such a topological (topometric) map for indoor environments is critical for robot to localize into the topometric map and perform autonomous navigation. We demonstrate preliminary results in this chapter.

11.3.1 Construction of Topological Map via Voronoi Graph

The proposed algorithm takes an occupancy grid map converted from an elevation map to construct a Voronoi graph and then to build a topological map. A grid in the elevation map is considered as an obstacle if the height difference between the grid and the nearest robot pose is larger than a certain threshold. After an occupancy map is built, the proposed algorithm uses it to construct a Voronoi graph. Next, the topological map is built from the Voronoi graph.

The construction of discretized Voronoi diagram from an occupancy map is inspired from [297]. If a free cell in the occupancy map has more than one nearest occupied cell, it is called a Voronoi point. The distance between a Voronoi point and its corresponding obstacle cells is defined as clearance. Next, topological regions in the graph is divided by connecting Voronoi points that have local minimum clearance. For each topological region r_i, a convex hull is constructed h_i. If r_i is connected with r_j, the potential new region by merging r_i and r_j is $r_{i,j}$. The score of merging the two topological regions r_i and r_j is defined as

$$s_{i,j} := \frac{\text{area}(r_{i,j})}{\text{area}(h_{i,j})}. \tag{11.11}$$

Each time, the pair with the minimum score is merged until the minimum merging score is larger than a threshold. To represent the safe paths, the Voronoi point with the largest clearance in a region will be connected with the Voronoi points used to divide the region. To represent the obstacles, the obstacle boundary with only one or two connections are merged until the least square error is larger than the threshold. Algorithm 5 summarizes the overall process. The resulting Voronoi graph and topological map of the second floor of FRB is illustrated in Fig. 11.7 and Fig. 11.8, respectively.

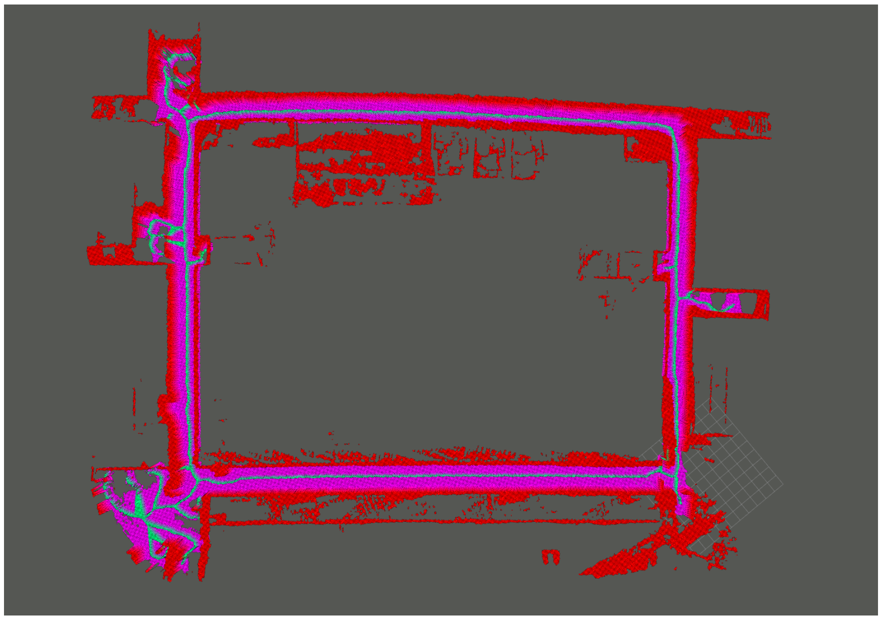


Figure 11.7: Voronoi graph of second floor of FRB. The green and the red indicates the Voronoi points and obstacles, respectively. The pink is free area.

The size of the resulting map is reduced from a 10 GB ROS bagfile to 16 KB (200 doubles) text file. Additionally, the proposed algorithm takes 25 seconds to build the topological map from a 10-min ROS bagfile on a typical modern laptop.

11.3.2 Localization into Topological Map

As mentioned in the last section, we introduce a means to build a light-weight topometric map via Voronoi graphs. Another type of topometric map is OpenStreetMap (OSM). Both types are graph-based representation containing vertices and edges. Edges connecting vertices are topological line boundaries such as walls or road curbs. These line features could be readily obtained from

Algorithm 5: Voronoi Graph Construction

```
1  for $n$ in free grids do
2      $n_{nearest} \leftarrow$ Nearest($n$)
3      $d_{clearence} \leftarrow$ dist($n$, $n_{nearest}$)
4      $O \leftarrow$ Obstacle($n$, $d_{clearence}$, $d_{clearence} + \sqrt{2}$)
5      $n_{basis} \leftarrow$ RemoveTightlyConnect($O$)
6      if size($n_{basis}$)$> 1$ then
7          $V$.append($n$)
8  $G \leftarrow$ DivideGroups($V$)
9  $C \leftarrow$ Localminimum($G$)
10 $C \leftarrow$ MergeCells($C$)
```

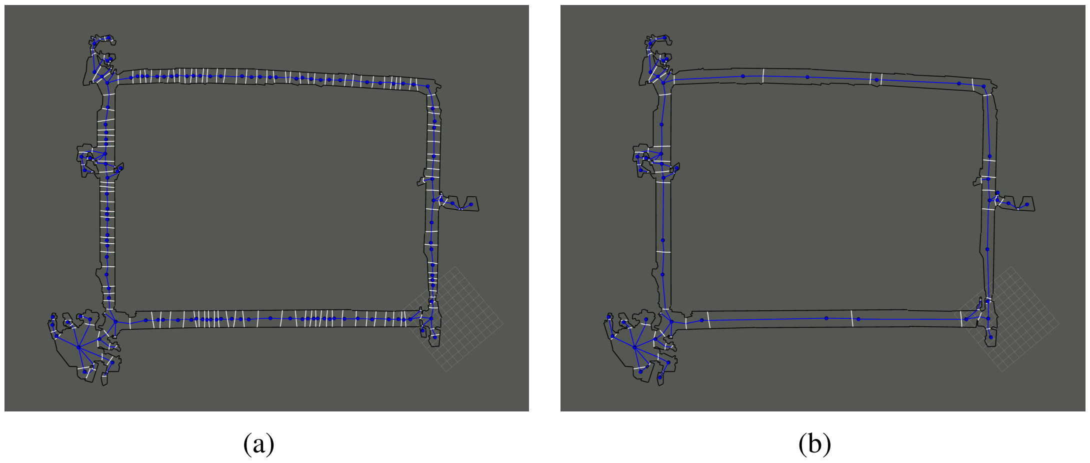


Figure 11.8: (a) shows the topological built from Fig. 11.7. (b) shows the topological map after merging. The black, blue, and white lines are walls, walkable routes, and region boundaries, respectively.

LiDAR sensors or camera images. We have two sets of line features, one from the topological maps and the other from the sensor measurements. Lines can also be parameterized as two points. Therefore, we have two sets of point clouds.

We propose to leverage the particle filter [298] for localization where each particle is a potential location of the robot. The weight of a particle is computed via Reproducing Kernel Hilbert Space (RKHS) mentioned in Chapter 6.6. We construct a continuous function in an inner product space for a set of point clouds [105]. Finally, we compute the inner product of the two estimated continuous functions. The largest inner product is the pose of the robot location.

APPENDIX A

Global Unifying Intrinsic Calibration on Lie Group for Spinning LiDAR and Solid-state LiDAR

A.1 Proof of Uniqueness of Similarity Transformation

This proof provides a guideline to place targets such that an answer to the optimization problem is unique. In particular, under Assumption N and Assumption B, the optimization problem has only one unique answer. As stated in Sec. 4.3, the intrinsic calibration parameters are modeled as an element of the similarity Lie group ($\mathrm{Sim}(3)$). An element of this group in matrix from is

$$\mathbf{H} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathrm{Sim}(3), \tag{A.1}$$

where $\mathbf{R} \in \mathrm{SO}(3), \mathbf{t} \in \mathbb{R}^3$ and $s \in \mathbb{R}^+$.

A.1.1 Mathematical Definitions and Preliminaries

Let $[\mathcal{S}]$ and $[\mathcal{S}]^\perp$ be the span and its orthogonal complement of $\mathcal{S} \subset \mathbb{R}^3$, respectively[1]. We denote the union of intersection of K spans as $([\mathcal{S}_1] + [\mathcal{S}_2] + \cdots + [\mathcal{S}_K]) = \cap_{i=1}^K [\mathcal{S}_i]$. Let $\mathbf{P} \subset \mathbb{R}^3$ be a plane. $\mathbf{P}$ modeled the set traced out by a single ring of a perfectly calibrated spinning LiDAR. Let $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ denote the canonical basis for $\mathbb{R}^3$. Without loss of generality, we assume $\mathbf{P} = [\mathbf{e}_3]^\perp = \{\mathbf{e}_1, \mathbf{e}_2\}$. Therefore, $\forall s > 0, \mathbf{R} \in \mathrm{SO}(3)$,

$$(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 0 \iff s = 1, \mathbf{R} = \mathbf{I}.^2$$

A.1.2 Assumptions

Consider four targets with unit normal vectors $\mathbf{n}_i$ and let $\mathbf{p}_{0,i}$ be a point on the i-th target.

[1] $\because \mathcal{S} \subset \mathbb{R}^3$, $[\mathcal{S}] = \mathbb{R}^3 \iff [\mathcal{S}]^\perp = 0$

[2] $(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = [(s\mathbf{R} - \mathbf{I})\mathbf{e}_1] + [(s\mathbf{R} - \mathbf{I})\mathbf{e}_2)]$

Assumption N (Normal Vectors)
All sets of three distinct vectors from $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \boldsymbol{n}_3, \boldsymbol{n}_4, \boldsymbol{e}_3\}$ *are linearly independent.*

For $1 \leqslant i \leqslant 4$, the plane defined by the i-th target is $\mathbf{V}_i := \mathbf{p}_{0,i} + [\mathbf{n}_i]^{\perp}$. Under Assumption N, some facts are listed below without proofs:

(a) For each $i \neq j \in \{1, 2, 3, 4\}$, $\mathbf{p}_{ij} := \mathbf{P} \cap \mathbf{V}_i \cap \mathbf{V}_j$ exists and is unique. There are $\binom{4}{2}$ intersection points.

(b) $\mathbf{V}_i \cap \mathbf{V}_j = \mathbf{p}_{ij} + [\mathbf{n}_i, \mathbf{n}_j]^{\perp}$.

(c) $\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j$ if, and only if,

$$(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + \mathbf{t} \in [\mathbf{n}_i, \mathbf{n}_j]^{\perp}.$$

(d) Because $\dim[\mathbf{n}_i, \mathbf{n}_j]^{\perp} = 1$, $(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + \mathbf{t} \neq 0$ if, and only if, $[(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + \mathbf{t}] = [\mathbf{n}_i, \mathbf{n}_j]^{\perp}$.

(e) Let $\{\mathbf{p}_a, \mathbf{p}_b\}$ be a basis for $\mathbf{P}$. Then $(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 0$ if, and only if, $(s\mathbf{R} - \mathbf{I})(\mathbf{p}_a) = 0$ and $(s\mathbf{R} - \mathbf{I})(\mathbf{p}_b) = 0$.

Assumption B (Basis Vectors)
Given the six intersection points, any two form a set of basis for the ring plane. There are $\binom{6}{2}$ *sets of basis. Subsets of the basis' are linearly independent.*

(a) $\{\boldsymbol{p}_{12}, \boldsymbol{p}_{13}\}$, $\{\boldsymbol{p}_{13}, \boldsymbol{p}_{14}\}$, $\{\boldsymbol{p}_{14}, \boldsymbol{p}_{12}\}$,

(b) $\{\boldsymbol{p}_{12}, \boldsymbol{p}_{23}\}$, $\{\boldsymbol{p}_{23}, \boldsymbol{p}_{24}\}$, $\{\boldsymbol{p}_{24}, \boldsymbol{p}_{12}\}$,

(c) $\{\boldsymbol{p}_{13}, \boldsymbol{p}_{23}\}$, $\{\boldsymbol{p}_{23}, \boldsymbol{p}_{34}\}$, $\{\boldsymbol{p}_{34}, \boldsymbol{p}_{13}\}$,

(d) $\{\boldsymbol{p}_{14}, \boldsymbol{p}_{24}\}$, $\{\boldsymbol{p}_{24}, \boldsymbol{p}_{34}\}$, $\{\boldsymbol{p}_{34}, \boldsymbol{p}_{14}\}$,

(e) $\{\boldsymbol{p}_{14}, \boldsymbol{p}_{23}\}$

A.1.3 A Complete Proof

Theorem 1. *Assume that Assumptions N and B hold and let* $\boldsymbol{H} \in \mathrm{Sim}(3)$. *If for each* $i \neq j \in \{1, 2, 3, 4\}$, $\boldsymbol{H} \cdot \boldsymbol{p}_{ij} \in \boldsymbol{V}_i \cap \boldsymbol{V}_j$, *then*

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{A.2}$$

Proof. The proof is by exhaustion on the dimension of $(s\mathbf{R} - \mathbf{I})(\mathbf{P})$. We noticed that $\mathbf{H} \cdot \mathbf{p}_{ij} = s\mathbf{R}\mathbf{p}_{ij} + \mathbf{t}$, and therefore

$$\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j \iff (s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + \mathbf{t} \in [\mathbf{n}_i, \mathbf{n}_j]^{\perp}. \tag{A.3}$$

- **Case 1:** $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 0$.

  Then $\mathbf{t} \in [\mathbf{n}_i, \mathbf{n}_j]^{\perp}$ for all $i \neq j$. Hence, $\mathbf{t} \in [\mathbf{n}_1, \mathbf{n}_2]^{\perp} \cap [\mathbf{n}_2, \mathbf{n}_3]^{\perp} = 0$, which implies that $\mathbf{t} = 0$. Therefore, by **Simple Fact 1**, we have finished. We next show that $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) > 0$ and $\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j$ for all $i \neq j$ lead to contradictions.

- **Case 2:** $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 1$

  (a) Suppose $\mathbf{t} \notin (s\mathbf{R} - \mathbf{I})(\mathbf{P})$. Then, for any $\mathbf{p}_{ij}$, $(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + \mathbf{t} \neq 0$. Hence,

$$\begin{aligned}
[\mathbf{n}_1, \mathbf{n}_2]^{\perp} &= [(s\mathbf{R} - \mathbf{I})(\mathbf{p}_{12}) + \mathbf{t}] \\
[\mathbf{n}_2, \mathbf{n}_3]^{\perp} &= [(s\mathbf{R} - \mathbf{I})(\mathbf{p}_{23}) + \mathbf{t}] \\
[\mathbf{n}_3, \mathbf{n}_4]^{\perp} &= [(s\mathbf{R} - \mathbf{I})(\mathbf{p}_{34}) + \mathbf{t}].
\end{aligned}$$

  Because $[\mathbf{n}_1, \mathbf{n}_2] \cap [\mathbf{n}_2, \mathbf{n}_3] \cap [\mathbf{n}_3, \mathbf{n}_4] = 0$, we know that $[\mathbf{n}_1, \mathbf{n}_2]^{\perp} + [\mathbf{n}_2, \mathbf{n}_3]^{\perp} + [\mathbf{n}_3, \mathbf{n}_4]^{\perp} = \mathbb{R}^3$. We deduce that $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 2$, which is a contradiction.

  (b) Suppose $\mathbf{t} \in (s\mathbf{R} - \mathbf{I})(\mathbf{P})$ and thus there exists $\mathbf{p}_t \in \mathbf{P}$ such that $\mathbf{t} = (s\mathbf{R} - \mathbf{I})(\mathbf{p}_t)$. The condition (A.3) can therefore be written as

$$\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j \iff (s\mathbf{R} - \mathbf{I})(\mathbf{p}_{ij} + \mathbf{p}_t) \perp \{\mathbf{n}_i, \mathbf{n}_j\}. \tag{A.4}$$

  Because $\{\mathbf{p}_{12}, \mathbf{p}_{13}\}$, $\{\mathbf{p}_{13}, \mathbf{p}_{14}\}$, $\{\mathbf{p}_{14}, \mathbf{p}_{12}\}$ are bases for $\mathbf{P}$, we conclude that

$$[\mathbf{p}_{12} + \mathbf{p}_t, \mathbf{p}_{13} + \mathbf{p}_t, \mathbf{p}_{14} + \mathbf{p}_t] = \mathbf{P}. \tag{A.5}$$

  Applying (A.4) to this set of vectors, we deduce that

$$(s\mathbf{R} - \mathbf{I})(\mathbf{P}) \perp \mathbf{n}_1.$$

  Applying the same reasoning to $\{\mathbf{p}_{12}, \mathbf{p}_{23}\}$, $\{\mathbf{p}_{23}, \mathbf{p}_{24}\}$, $\{\mathbf{p}_{24}, \mathbf{p}_{12}\}$, we deduce that $(s\mathbf{R} - \mathbf{I})(\mathbf{P}) \perp \mathbf{n}_2$. Similarly, we have $(s\mathbf{R} - \mathbf{I})(\mathbf{P}) \perp \mathbf{n}_3$ and thus $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 0$.

- **Case 3:** $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 2$

We rewrite (A.3) as

$$\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j \iff -\mathbf{t} \in (s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + [\mathbf{n}_i, \mathbf{n}_j]^{\perp}. \tag{A.6}$$

Hence, if for all $i \neq j \in \{1,2,3,4\}$, $\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j$, then the lines $(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + [\mathbf{n}_i, \mathbf{n}_j]^{\perp}$ in $\mathbb{R}^3$ must have a common point of intersection, namely $-\mathbf{t}$ and hence

$$\bigcap_{i \neq j \in \{1,2,3,4\}} \{(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + [\mathbf{n}_i, \mathbf{n}_j]^{\perp}\} \neq \varnothing. \tag{A.7}$$

Remark 44. *When $(s\boldsymbol{R} - \boldsymbol{I})(\boldsymbol{P}) = 0$, the intersections in* (A.7) *are non-empty; indeed, the equation reduces to*

$$\bigcap_{i \neq j \in \{1,2,3,4\}} [\boldsymbol{n}_i, \boldsymbol{n}_j]^{\perp} = \left(\sum_{i \neq j} [\boldsymbol{n}_i, \boldsymbol{n}_j] \right)^{\perp} = (\mathbb{R}^3)^{\perp} = 0,$$

and hence $\boldsymbol{t} = 0$.

In the remainder of the proof, we show that when the intersection is non-empty, it contradicts $\dim\,(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 2$. We do this by examining the intersections in (A.7) pairwise to arrive at a set of necessary conditions for $\mathbf{H} \cdot \mathbf{p}_{ij} \in \mathbf{V}_i \cap \mathbf{V}_j$ for $i \neq j$, and then use the necessary conditions to complete the proof. We note that for $ij \neq kl$,

$$\begin{aligned} &\{(s\mathbf{R} - \mathbf{I})\mathbf{p}_{ij} + [\mathbf{n}_i, \mathbf{n}_j]^{\perp}\} \cap \{(s\mathbf{R} - \mathbf{I})\mathbf{p}_{kl} + [\mathbf{n}_k, \mathbf{n}_l]^{\perp}\} \neq \varnothing \\ \iff\ &(s\mathbf{R} - \mathbf{I})(\mathbf{p}_{ij} - \mathbf{p}_{kl}) \in [\mathbf{n}_i, \mathbf{n}_j]^{\perp} + [\mathbf{n}_k, \mathbf{n}_l]^{\perp}. \end{aligned}$$

The indices are a bit easier to keep track of if we set

$$\begin{aligned} \mathbf{q}_1 &:= (s\mathbf{R} - \mathbf{I})(\mathbf{p}_{12}),\ \mathbf{U}_1 := [\mathbf{n}_1, \mathbf{n}_2]^{\perp} \\ \mathbf{q}_2 &:= (s\mathbf{R} - \mathbf{I})(\mathbf{p}_{13}),\ \mathbf{U}_2 := [\mathbf{n}_1, \mathbf{n}_3]^{\perp} \\ \mathbf{q}_3 &:= (s\mathbf{R} - \mathbf{I})(\mathbf{p}_{23}),\ \mathbf{U}_3 := [\mathbf{n}_1, \mathbf{n}_4]^{\perp} \\ \mathbf{q}_4 &:= (s\mathbf{R} - \mathbf{I})(\mathbf{p}_{14}),\ \mathbf{U}_4 := [\mathbf{n}_2, \mathbf{n}_3]^{\perp}, \end{aligned}$$

where $\{\mathbf{U}_k | k = 1, \cdots, 4\}$ denote the indicated one-dimensional subspaces. Then, for each $i \neq j \in \{1,2,3,4\}$, $\mathbf{U}_i \cap \mathbf{U}_j = 0$, and we have $\mathbf{U}_1 \oplus \mathbf{U}_2 \oplus \mathbf{U}_3 = \mathbb{R}^3$. Let $\mathbf{u}_i$ be a basis for $\mathbf{U}_i$, so that $\mathbf{U}_i = [\mathbf{u}_i]$, and write

$$\mathbf{u}_4 = \alpha\mathbf{u}_1 + \beta\mathbf{u}_2 + \gamma\mathbf{u}_3.$$

Claim 1

Each of the coefficients α, β, γ is non-zero.

Proof. Suppose $\alpha = 0$. Then $\mathbf{U}_4 \subset \mathbf{U}_2 + \mathbf{U}_3$, that is,

$$[\mathbf{n}_2, \mathbf{n}_3]^\perp \subset [\mathbf{n}_1, \mathbf{n}_3]^\perp + [\mathbf{n}_1, \mathbf{n}_4]^\perp$$

But this is equivalent to

$$[\mathbf{n}_1] = [\mathbf{n}_1, \mathbf{n}_3] \cap [\mathbf{n}_1, \mathbf{n}_4] \subset [\mathbf{n}_2, \mathbf{n}_3],$$

and hence $\{\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3\}$ is not linearly independent, contradicting Assumption N. The same argument holds for the other coefficients. ∎

Claim 2

A necessary condition for

$$\bigcap_{i=1}^{4} \{\boldsymbol{q}_i + \boldsymbol{U}_i\} \neq \varnothing \tag{A.8}$$

is that there exist real numbers c_1, c_2, c_3, c_4 such that

$$\begin{aligned}
\Delta\boldsymbol{q}_{12} &:= \boldsymbol{q}_1 - \boldsymbol{q}_2 = c_1\boldsymbol{u}_1 + c_2\boldsymbol{u}_2 \\
\Delta\boldsymbol{q}_{13} &:= \boldsymbol{q}_1 - \boldsymbol{q}_3 = c_1\boldsymbol{u}_1 + c_3\boldsymbol{u}_3 \\
\Delta\boldsymbol{q}_{14} &:= \boldsymbol{q}_1 - \boldsymbol{q}_4 = c_1\boldsymbol{u}_1 + c_4\boldsymbol{u}_4 \\
\Delta\boldsymbol{q}_{23} &:= \boldsymbol{q}_2 - \boldsymbol{q}_3 = c_3\boldsymbol{u}_3 - c_2\boldsymbol{u}_2 \\
\Delta\boldsymbol{q}_{24} &:= \boldsymbol{q}_2 - \boldsymbol{q}_4 = c_4\boldsymbol{u}_4 - c_2\boldsymbol{u}_2 \\
\Delta\boldsymbol{q}_{34} &:= \boldsymbol{q}_3 - \boldsymbol{q}_4 = c_3\boldsymbol{u}_3 - c_4\boldsymbol{u}_4
\end{aligned} \tag{A.9}$$

Proof. Each row of (A.9) corresponds to a condition of the form $\mathbf{q}_i - \mathbf{q}_j \in \mathbf{U}_i \oplus \mathbf{U}_j$. The proof proceeds by expressing each of the six rows in (A.9) with distinct coefficients (12 in total), and then writing down three necessary compatibility conditions,

$$\begin{aligned}
\Delta\mathbf{q}_{12} - \Delta\mathbf{q}_{13} + \Delta\mathbf{q}_{23} &= 0 \\
\Delta\mathbf{q}_{12} - \Delta\mathbf{q}_{14} + \Delta\mathbf{q}_{24} &= 0 \\
\Delta\mathbf{q}_{14} - \Delta\mathbf{q}_{13} + \Delta\mathbf{q}_{34} &= 0.
\end{aligned} \tag{A.10}$$

Because $-\mathbf{t}$ is in the intersection of the lines in (A.8), the resulting linear equations must have a solution, and indeed direct computation shows that the set of solutions can be parameterized as given in the claim. ∎

The next step is to note that $[\Delta\mathbf{q}_{12}, \cdots, \Delta\mathbf{q}_{34}] \subset (s\mathbf{R} - \mathbf{I})(\mathbf{P})$, and hence its dimension must be less than three. Additional straightforward calculations show that

$$\dim\ [\Delta\mathbf{q}_{12}, \Delta\mathbf{q}_{13}, \Delta\mathbf{q}_{34}] = \text{rank} \begin{bmatrix} c_1 & c_1 & -\alpha c_4 \\ c_2 & 0 & c_2 - \beta c_4 \\ 0 & c_3 & -\gamma c_4 \end{bmatrix}.$$

In light of Claim 1, the rank is less than three if, and only if, any two coefficients of $\{c_1, c_2, c_3, c_4\}$ are zero. But if this is the case, then at least one row of (A.9) must be zero. Each row of (A.9), however, has the form $(s\mathbf{R} - \mathbf{I})(\mathbf{p}_a - \mathbf{p}_b)$, where $\{\mathbf{p}_a, \mathbf{p}_b\}$ is a basis for **P**, and thus it cannot be the case that $\dim(s\mathbf{R} - \mathbf{I})(\mathbf{P}) = 2$. This completes the proof.

■

A.2 Lagrangian Duality Relaxation for P2P Distance on $\mathrm{SE}(3)$

At the k-th iteration, the scaling parameter is determined (see Sec. 4.5) and the rest of the parameters are $\mathrm{SE}(3)$. To solve the remaining parameters, we adopt techniques that were used to solve 3D registration or 3D SLAM [1, 82–84]. We summarize below for completeness. The action of $\mathrm{SE}(3)$ on $\mathbb{R}^3$ can be rewritten as:

$$\mathbf{H} \cdot \mathbf{x} = \mathbf{R}\mathbf{x} + \mathbf{v} = \begin{bmatrix} \mathbf{x}^\top \otimes I_3 & I_3 \end{bmatrix} \underbrace{\begin{bmatrix} \mathrm{vec}(\mathbf{R}) \\ \mathbf{v} \end{bmatrix}}_{\tau}, \tag{A.11}$$

where $\otimes$ and $\mathrm{vec}(\cdot)$ are the Kronecker product [299] and the vectorization operation [300], respectively.

A.2.1 P2P Distance Reformulation and Quadratic Formulation

The P2P distance (4.2) can be equivalently reformulated into a quadratic form:

$$\begin{aligned}\sum_{i=1}^{M} J_i &:= \sum_{i=1}^{M} |(\mathbf{H} \cdot \mathbf{x}_i - \mathbf{p}_{0,t})^\top (\mathbf{n}_t \mathbf{n}_t^\top)(\mathbf{H} \cdot \mathbf{x}_i \mathbf{p}_{0,t})| \\ &= \sum_{i=1}^{M} \begin{bmatrix} \tau \\ 1 \end{bmatrix}^\top N_i^\top (\mathbf{n}_t \mathbf{n}_t^\top) N_i \begin{bmatrix} \tau \\ 1 \end{bmatrix} = \tilde{\tau}^\top W_t \tilde{\tau},\end{aligned} \tag{A.12}$$

where $N_i = \begin{bmatrix} \mathbf{x}_i \otimes I_3 | -\mathbf{p}_{0,t} \end{bmatrix}$ and $W_t = \sum_i^M N_i^\top (\mathbf{n}_t \mathbf{n}_t^\top) N_i$. After rearranging (A.12), the resulting problem in (4.3) becomes

$$\begin{aligned}&\min_{H \in \mathrm{SE}(3)} \sum_{t=1}^{T} \begin{bmatrix} \mathrm{vec}(\mathbf{R}) \\ 1 \\ \mathbf{v} \end{bmatrix}^\top \tilde{W}_t' \begin{bmatrix} \mathrm{vec}(\mathbf{R}) \\ 1 \\ \mathbf{v} \end{bmatrix} \\ &= \min_{H \in \mathrm{SE}(3)} \sum_{t=1}^{T} \begin{bmatrix} \tilde{\mathbf{r}} \\ \mathbf{v} \end{bmatrix}^\top \begin{bmatrix} \tilde{W}_{\tilde{\mathbf{r}},\tilde{\mathbf{r}}} & \tilde{W}_{\tilde{\mathbf{r}},\mathbf{v}} \\ \tilde{W}_{\mathbf{v},\tilde{\mathbf{r}}} & W_{\mathbf{v},\mathbf{v}} \end{bmatrix}_t \begin{bmatrix} \tilde{\mathbf{r}} \\ \mathbf{v} \end{bmatrix} \\ &= \min_{H \in \mathrm{SE}(3)} \tilde{\mathbf{r}}^\top \tilde{W}_{\tilde{\mathbf{r}},\tilde{\mathbf{r}}} \tilde{\mathbf{r}} + 2\mathbf{v}^\top \tilde{W}_{\mathbf{v},\tilde{\mathbf{r}}} \tilde{\mathbf{r}} + \mathbf{v}^\top W_{\mathbf{v},\mathbf{v}} \mathbf{v}\end{aligned} \tag{A.13}$$

where $\tilde{\mathbf{r}} = \begin{bmatrix} \mathrm{vec}(\mathbf{R})^\top & 1 \end{bmatrix}^\top$. We introduce the Lagrangian multipliers. Due to $R \in \mathrm{SO}(3)$ constraints, the derivative with respect to $\mathbf{v}$ is zero: $\partial \mathrm{L}(H, \lambda)/\partial \mathbf{v} = 0$, which leads to

$$\mathbf{v}^* = -(W_{\mathbf{v},\mathbf{v}})^{-1} \tilde{W}_{\mathbf{v},\tilde{\mathbf{r}}} \tilde{\mathbf{r}}. \tag{A.14}$$

By substituting (A.14) into (A.13), we have

$$\min_{\mathbf{R}\in SO(3)} \tilde{\mathbf{r}}^\top \tilde{\mathbf{Q}} \tilde{\mathbf{r}}, \tag{A.15}$$

where $\tilde{\mathbf{Q}}$ is the Schur complement of $\tilde{W}_t$ and equal to $\tilde{W}_{\tilde{\mathbf{r}},\tilde{\mathbf{r}}} - \tilde{W}_{\tilde{\mathbf{r}},\mathbf{v}} W^{-1}_{\mathbf{v},\mathbf{v}} \tilde{W}_{\mathbf{v},\tilde{\mathbf{r}}}$.

A.2.2 Primal Problem and Its Dual

From [82–84], we re-define (A.15) to an *equivalent, homogeneous, strengthened* primal problem:

$$\begin{aligned} \min_R f(\tilde{\mathbf{q}}) = \min_R \tilde{\mathbf{q}}^\top \tilde{\mathbf{Q}} \tilde{\mathbf{q}},\ \tilde{\mathbf{q}} = \begin{bmatrix} \mathrm{vec}(\mathbf{R})^\top & y \end{bmatrix}^\top \\ \text{s.t. } R^\top R = y^2 I_3 \\ RR^\top = y^2 I_3 \\ R^{(i)} \times R^{(j)} = y R^{(k)},\ i,j,k = \text{permute}\{1,2,3\} \\ y^2 = 1. \end{aligned} \tag{A.16}$$

The primal problem (A.16) is a QCQP and the corresponding dual problem is defined as

$$\mathrm{L}(\tilde{\mathbf{q}}, \boldsymbol{\lambda}) = \gamma + \tilde{\mathbf{q}}^\top (\tilde{\mathbf{Q}} + \tilde{\mathbf{P}}(\boldsymbol{\lambda}))\tilde{\mathbf{q}} = \gamma + \tilde{\mathbf{q}}^\top \mathbf{Z} \tilde{\mathbf{q}}, \tag{A.17}$$

where **P** is the penalization matrix [84]. The Lagrangian relaxation is an unconstrained problem and has a closed-form solution:

$$g(\boldsymbol{\lambda}) = \min_{\tilde{\mathbf{q}}} \mathrm{L}(\tilde{\mathbf{q}}, \boldsymbol{\lambda}) = \min_{\tilde{\mathbf{q}}} \gamma + \tilde{\mathbf{q}}^\top \mathbf{Z} \tilde{\mathbf{q}} \tag{A.18}$$

$$= \begin{cases} \gamma, \text{ if } \mathbf{Z} \geq 0 \\ -\infty, \text{ otherwise.} \end{cases} \tag{A.19}$$

Therefore, the maximization of the dual problem (A.18) is a SDP:

$$g^* = \max_{\boldsymbol{\lambda}} \gamma,\ \text{s.t. } \mathbf{Z}(\boldsymbol{\lambda}) \geq 0 \tag{A.20}$$

This problem is convex and can be solved globally by off-the-shelf specialized solvers [85]. It is shown in [84] that the relaxation is empirically always tight (the duality gap is zero).

A.3 Observations on Global Convergence and Convexity

In this section, we provide numerical results that suggest the proposed method is also applicable to 3D registration problems and may achieve global convergence. Furthermore, we provide additional figures concerning about the potential convexity of (4.20).

A.3.1 Toward Global Convergence of the Proposed Method

We show that the proposed method in the experimental data collected by the *32-Beam Velodyne ULTRA Puck LiDAR* reduces the P2P distance by 68.6%, as shown in Fig. 4.8. In addition, we illustrate the proposed algorithm can be used in 3D registration problems (point-to-point, point-to-line, point-to-plane) by scaling the simulation (named Random) and experimental data (named RubikCube and SpaceStation) in [1, 84]. The experiment setup from [1] is shown in Fig. A.1 and the results of the registration problems are shown in Fig. A.2.

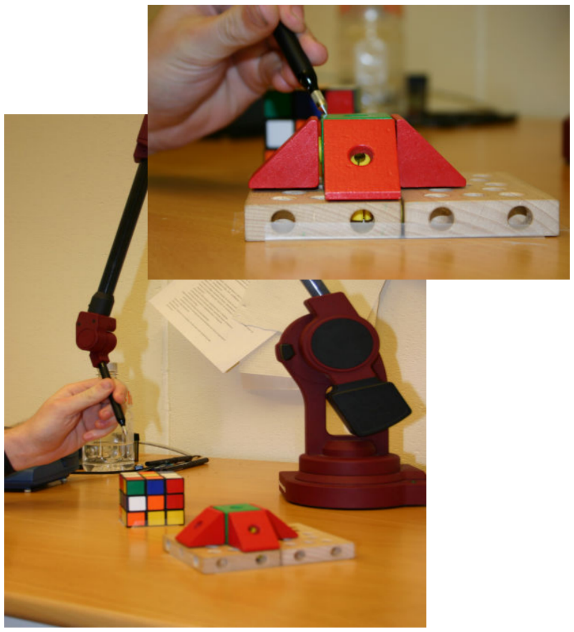


Figure A.1: This figure is taken from [1] to illustrate the experimental setup.

A.3.2 Potential Convexity of $f(s)$ and Bisection Method

Theoretically, a dense search on the scaling s should be performed on a range of interest. However, we have empirically observed that (4.20) is convex for all of our data sets. Figure A.3(a) shows $f(s)$ is convex for all the sets of the calibration parameters for our LiDAR intrinsic calibration datasets. Similarly, Fig. A.3(b) shows the convex property of $f(s)$ for the simulation and

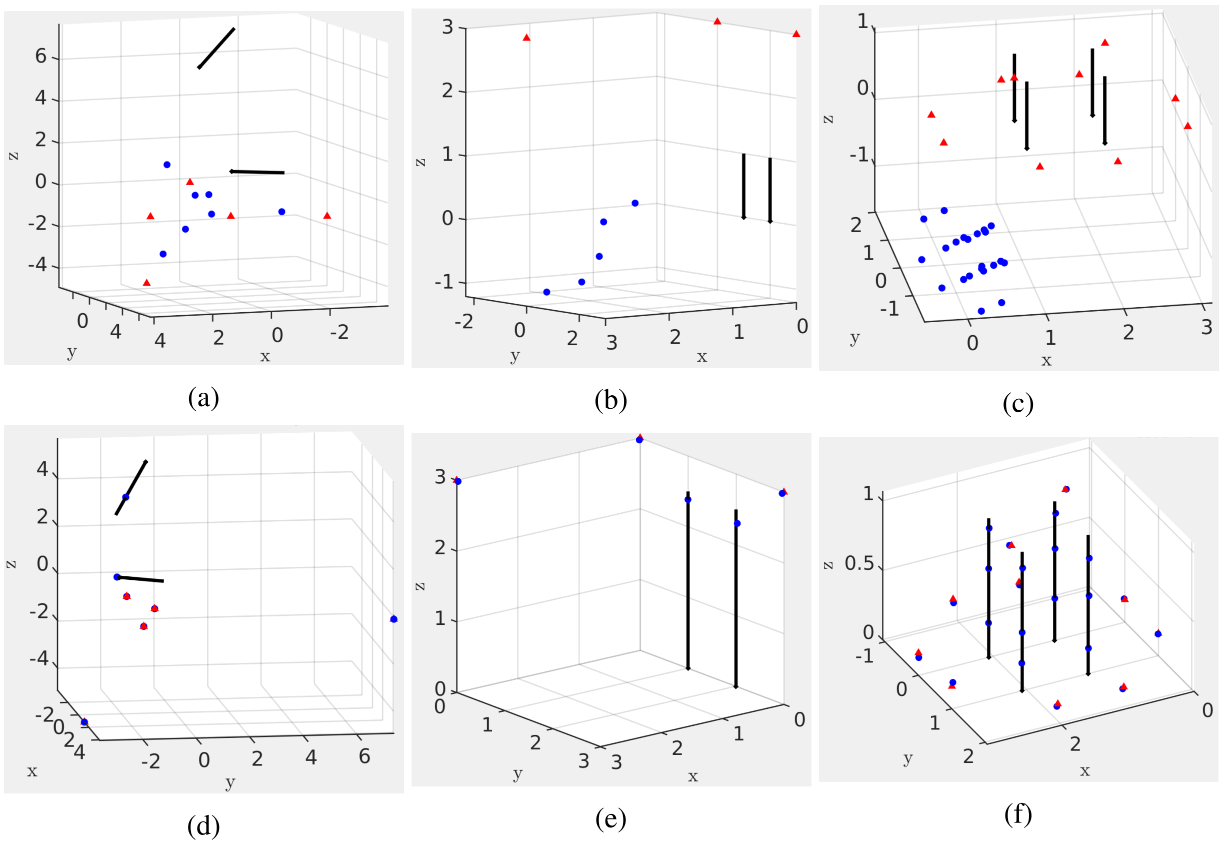


Figure A.2: This figure shows the proposed method that can be used in 3D registration problems. The correspondences are point-to-point (marked in blue and red), point-to-line (marked in blue and black), and point-to-plane (not shown to keep the figure readable). The top and bottom row show the initial status and the results of the registration problems, respectively. The simulation (Random) with five point-to-point and two point-to-line correspondences are shown in the left column. The result of the experimental data (RubikCube) with three point-to-point and two point-to-plane correspondences are shown in the second column. The last column shows the results of the experimental data (SpaceStation) with ten point-to-point and 12 point-to-line correspondences.

experimental data in [1, 84]. We, therefore, utilize the bisection method to determine the scaling s. It is emphasized that the convexity of (4.20) is unknown at this time.

Let ${}^{L}_{k}s$ and ${}^{U}_{k}s$ be the lower and upper bounds of the scaling parameters at the k-th iteration. Let ${}_{k+1}s = ({}^{U}_{k}s + {}^{L}_{k}s)/2$ be the update for $(k+1)$-th iteration. The updates of the scaling parameters are defined as:

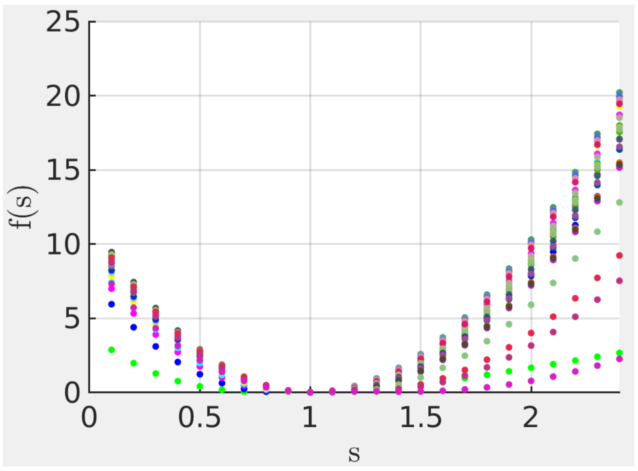
$$
\begin{aligned}
{}^{L}_{k+1}s &= \begin{cases} {}_{k+1}s, & \text{if } \nabla f({}_{k+1}s) < -\epsilon \\ {}^{L}_{k}s, & \text{otherwise} \end{cases} \\
{}^{U}_{k+1}s &= \begin{cases} {}_{k+1}s, & \text{if } \nabla f({}_{k+1}s) > \epsilon \\ {}^{U}_{k}s, & \text{otherwise} \end{cases} \quad ,
\end{aligned}
\tag{A.21}
$$

where the symmetric difference is used to approximate $\nabla f({}_{k+1}s)$:

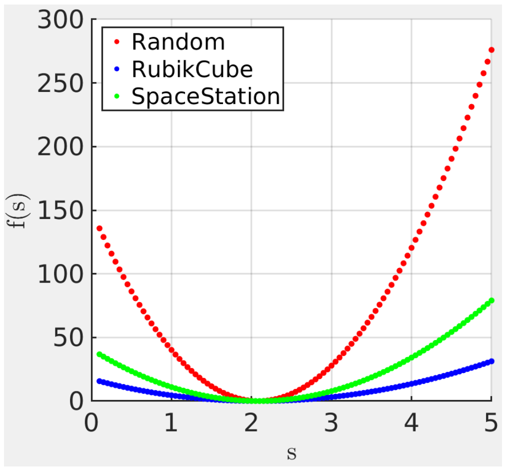
$$\nabla f({}_{k+1}s) \approx \frac{f({}_{k+1}s + h) - f({}_{k+1}s - h)}{2h}, \tag{A.22}$$

where $h > 0$ is a small value (taken as 10^{-3}).

Remark 45. *To obtain more accurate results,* $\boldsymbol{R}({}_{k+1}s+h)$, $\boldsymbol{v}({}_{k+1}s+h)$, $\boldsymbol{R}({}_{k+1}s-h)$ *and* $\boldsymbol{v}({}_{k+1}s-h)$ *should be computed separately.*



(a)



(b)

Figure A.3: This figure shows f vs s for the calibration parameters. The left figure shows the calibration parameters for all the rings, for f defined in (4.20). The right figure shows f for the 3D registration problems (point-to-point, point-to-line, point-to-plane). We suspect that the convex shape seen in the plot is in general true. Ongoing work is seeking a proof.

APPENDIX B

Informable Multi-Objective and Multi-Directional RRT* System for Robot Path Planning

B.1 ECI-Gen TSP Solver Benchmarking

The proposed ECI-Gen TSP solver is an enhanced version of the cheapest insertion algorithm [178], which comprises of a set of actions: **1)** in-sequence insertion, $\lambda_{\text{in-sequence}}$, which is the regular cheapest insertion; **2)** in-place insertion, $\lambda_{\text{in-place}}$, to allow the algorithm to revisit existing nodes; and **3)** swapping insertion, $\lambda_{\text{swapping}}$, which is inspired by genetic algorithms. Finally, sequence refinement is performed at the end of the algorithm. The ECI-Gen solver is able solve graphs that are complete or incomplete, and graphs with or without triangularity constraints.

To benchmark the ECI-Gen solver, we randomly generated 1000 complete and incomplete graphs with order of 5 to 30 [1]. For each graph, the edge connecting two vertices is either a straight line or a random distance larger than the Euclidean distance. Consequently, the former obeys the triangular inequality whereas the latter does not.

For a complete graph consisting of K vertices (graph order of K), there are $K!$ possible combinations of orders. A brute force algorithm [207] that permutes all the possible solutions is implemented as a baseline for graph order $K \leqslant 13$. Solving the graph size of 14 takes about an hour (1000 hours for 1000 graphs) on a modern computer and therefore it is intractable to solve the order of large graphs by brute force. The brute force algorithm guarantees the optimal path for each graph. For graph orders larger than 14, we compare the results to OR-Tools [301], an open source TSP solver developed by Google.

For each Korder of graphs, we evaluate the EGI-Gen TSP solver by the following metrics: average path cost $\bar{\Theta}_{\text{method}}$, elapsed time, ratio of better solution ρ^K_{better}, and ratio of feasible solutions ρ^K_{feasible}. For graph order $K \leqslant 13$, we have optimal solutions provided by the brute force algorithm; we further compute ratio of path cost ρ^K_{mean}, ratio of optimal solutions ρ^K_{optimal}, ratio of path cost of worst case ρ^K_{worst}, and standard deviation of inverse of path cost ratio ρ^K_{std}. In particular, they are computed as

[1]The order of a graph is the cardinality of its vertex set.

$$\bar{\Theta}_{\text{method}} = \frac{1}{N}\sum_{i=1}^{N} {}^{i}\Theta_{\text{method}}, \tag{B.1}$$

$$\rho_{\text{better}}^{K} = \frac{1}{N}\sum_{i=1}^{N} I\big[{}^{i}\Theta_{\text{ECI-Gen}} < {}^{i}\Theta_{\text{Baseline}}\big], \tag{B.2}$$

$$\rho_{\text{feasible}}^{K} = \frac{1}{N}\sum_{i=1}^{N} I[f_i \in \mathcal{F}] = \frac{1}{N}\|\mathcal{F}\|, \tag{B.3}$$

$$\rho_{\text{mean}}^{K} = \frac{\sum_{i=1}^{N} {}^{i}\Theta_{\text{Optimal}}}{\sum_{i=1}^{N} {}^{i}\Theta_{\text{method}}} = \frac{\bar{\Theta}_{\text{Optimal}}}{\bar{\Theta}_{\text{method}}}, \tag{B.4}$$

$$\rho_{\text{optimal}}^{K} = \frac{1}{N}\sum_{i=1}^{N} I\big[{}^{i}\Theta_{\text{Optimal}} == {}^{i}\Theta_{\text{method}}\big], \tag{B.5}$$

$$\rho_{\text{worst}}^{K} = \max\left(\frac{{}^{i}\Theta_{\text{method}}}{{}^{i}\Theta_{\text{Optimal}}}\Big|_{i=1}^{N}\right), \tag{B.6}$$

$$\rho_{\text{std}}^{K} = \texttt{std}\Big(\frac{1}{\rho_{\text{mean}}^{K}}\Big) = \texttt{std}\Big(\frac{{}^{i}\Theta_{\text{method}}}{{}^{i}\Theta_{\text{Optimal}}}\Big), \tag{B.7}$$

where $\|\mathcal{F}\|$ is the cardinality of the set of feasible solutions $\mathcal{F}$, N is the number of graphs ($N = 1000$), K is the order of graphs, and I is the indicator function returning one if the condition holds; otherwise zero.

Figure B.1 and B.2 show the solver evaluated on complete graphs with triangularity constraints. The results of complete graphs without triangularity constraints are shown in Fig. B.3 and B.4. The OR-Tools failed to solve incomplete graphs with triangularity constraints as shown in Fig. B.5 and B.6. Finally, we show the OR-Tools also failed to solve incomplete graphs without triangularity constraints in Fig. B.7 and B.8. We demonstrate that the proposed ECI-Gen TSP solver is able to solve broader sets of problems and provide better solutions with comparable computation speed.

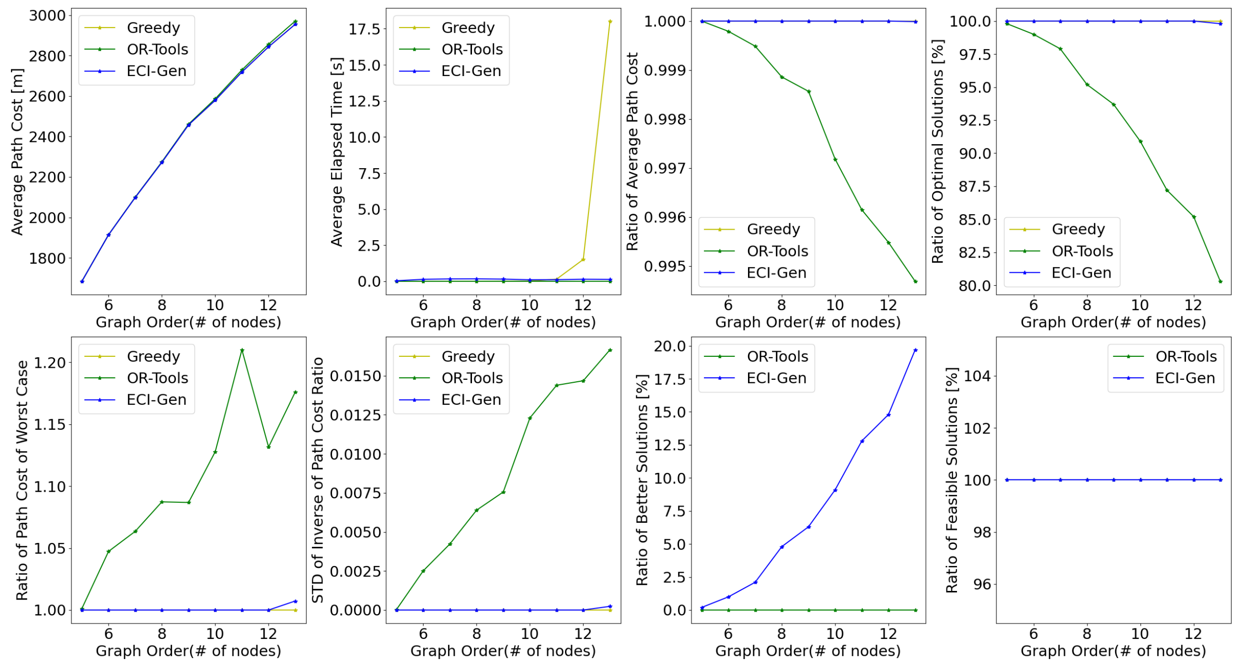


Figure B.1: The validation of the ECI-Gen solver on small complete graphs ($5 \leqslant K \leqslant 13$) with triangular inequality constraints.

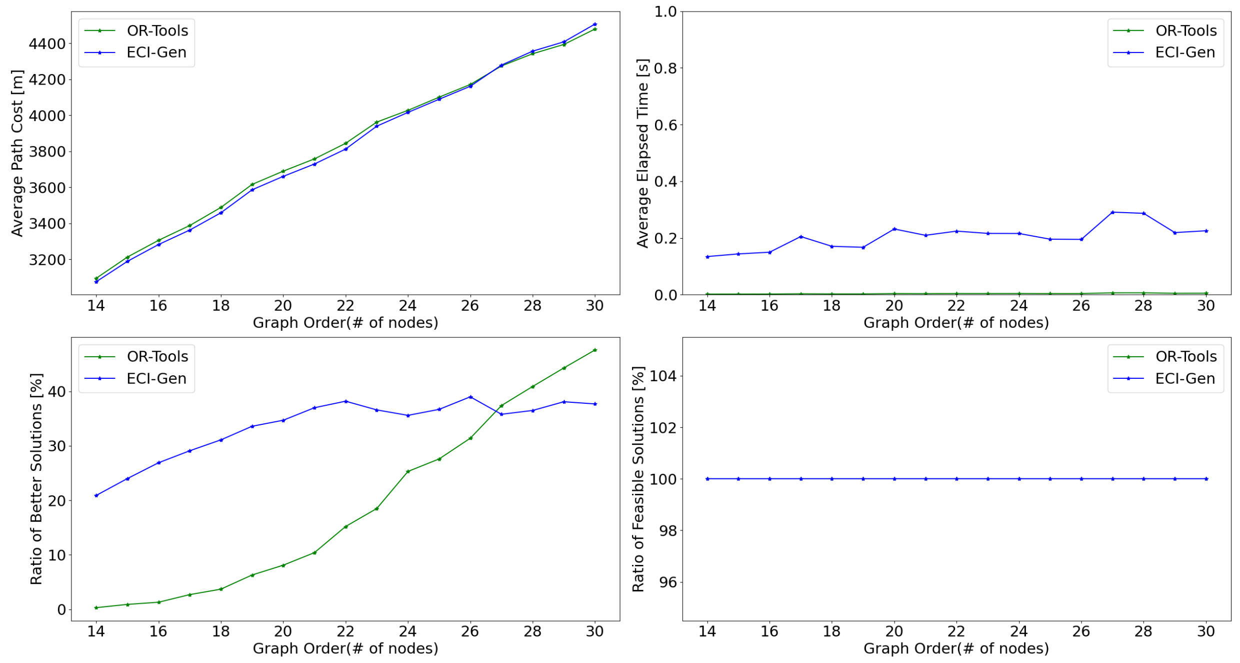


Figure B.2: The validation of the ECI-Gen solver on large complete graphs ($14 \leqslant K \leqslant 30$) with triangular inequality constraints.

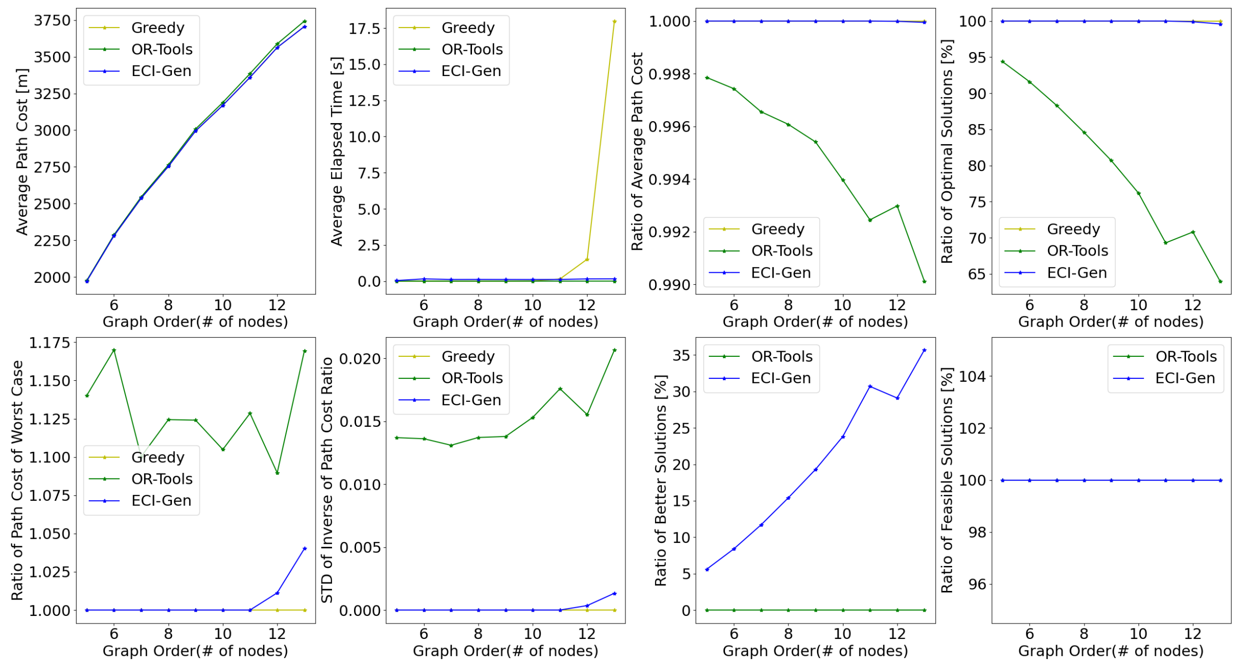


Figure B.3: The validation of the ECI-Gen TSP solver on small complete graphs ($5 \leqslant K \leqslant 13$) without triangular inequality constraints.

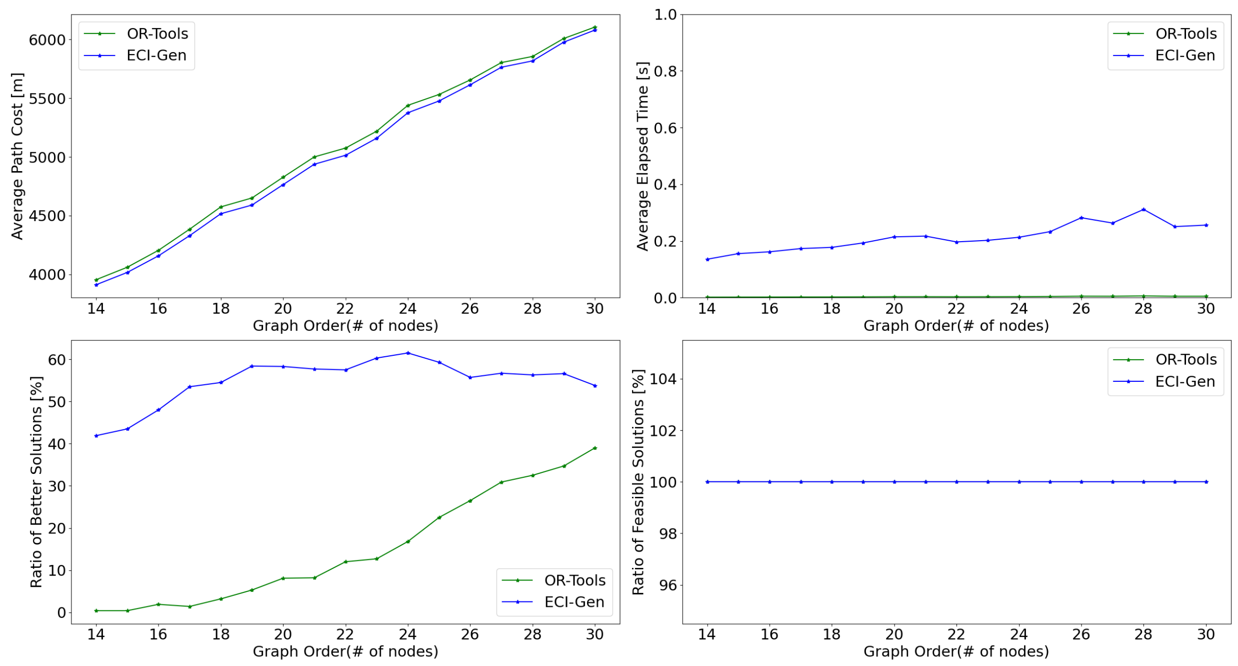


Figure B.4: The validation of the ECI-Gen TSP solver on large complete graphs ($14 \leqslant K \leqslant 30$) without triangular inequality constraints.

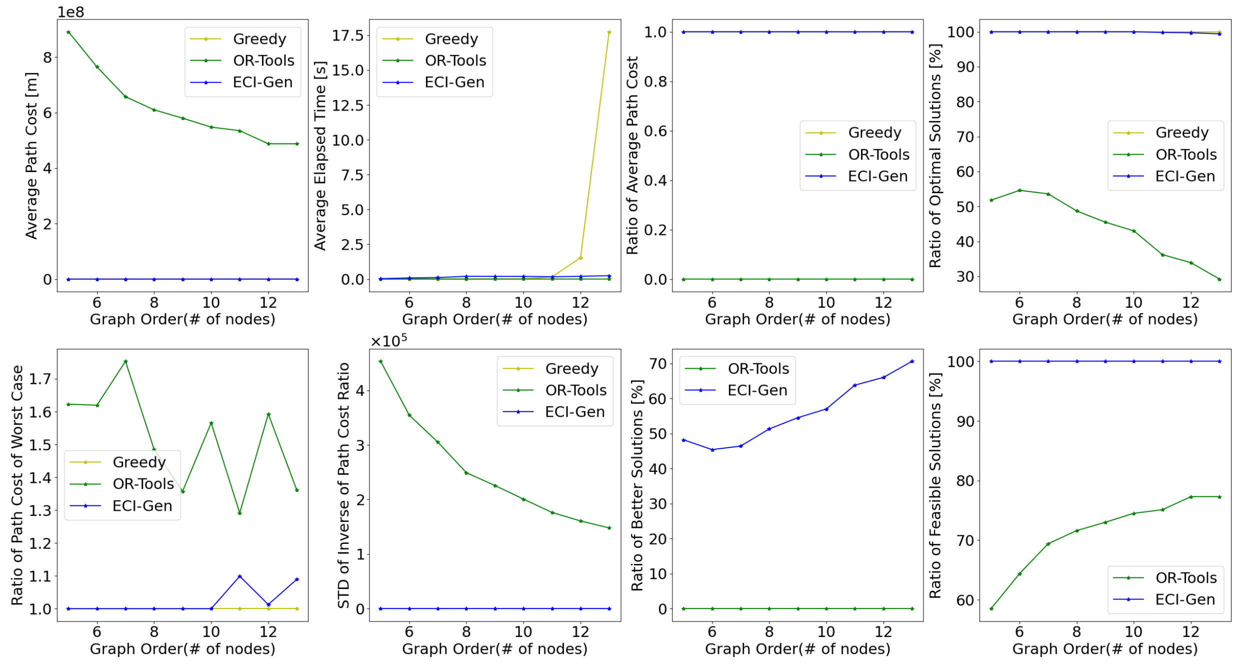


Figure B.5: The validation of the ECI-Gen TSP solver on small incomplete graphs ($5 \leqslant K \leqslant 13$) with triangular inequality constraints.

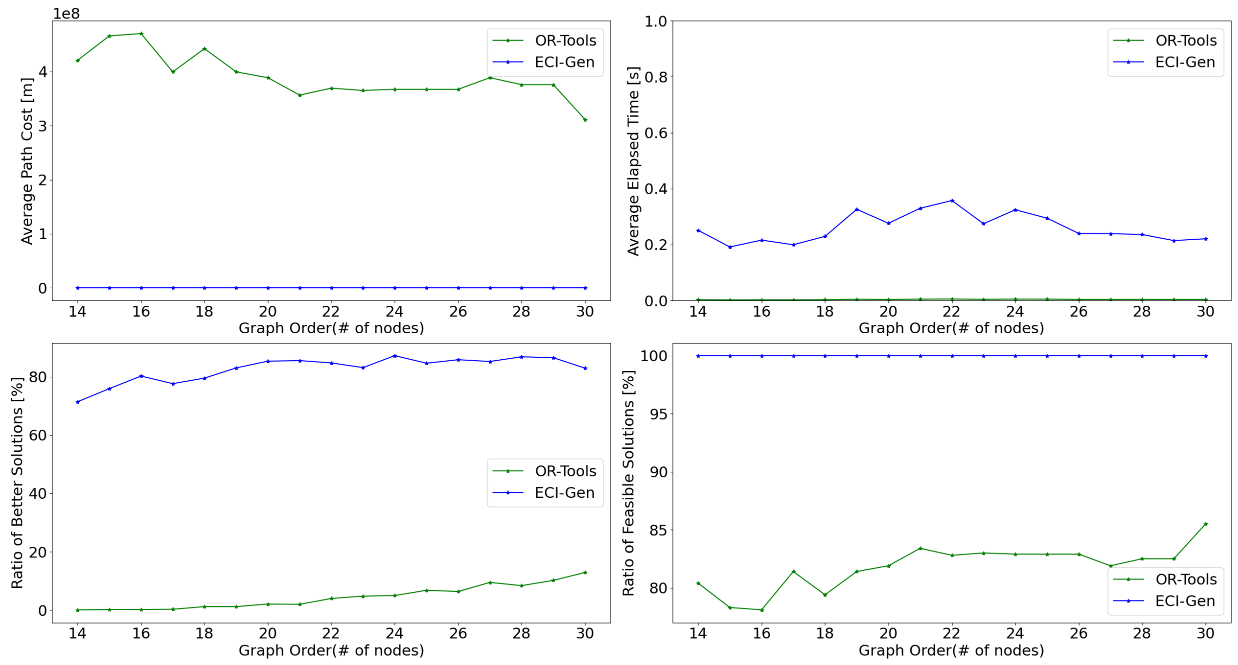


Figure B.6: The validation of the ECI-Gen TSP solver on large incomplete graphs ($14 \leqslant K \leqslant 30$) with triangular inequality constraints.

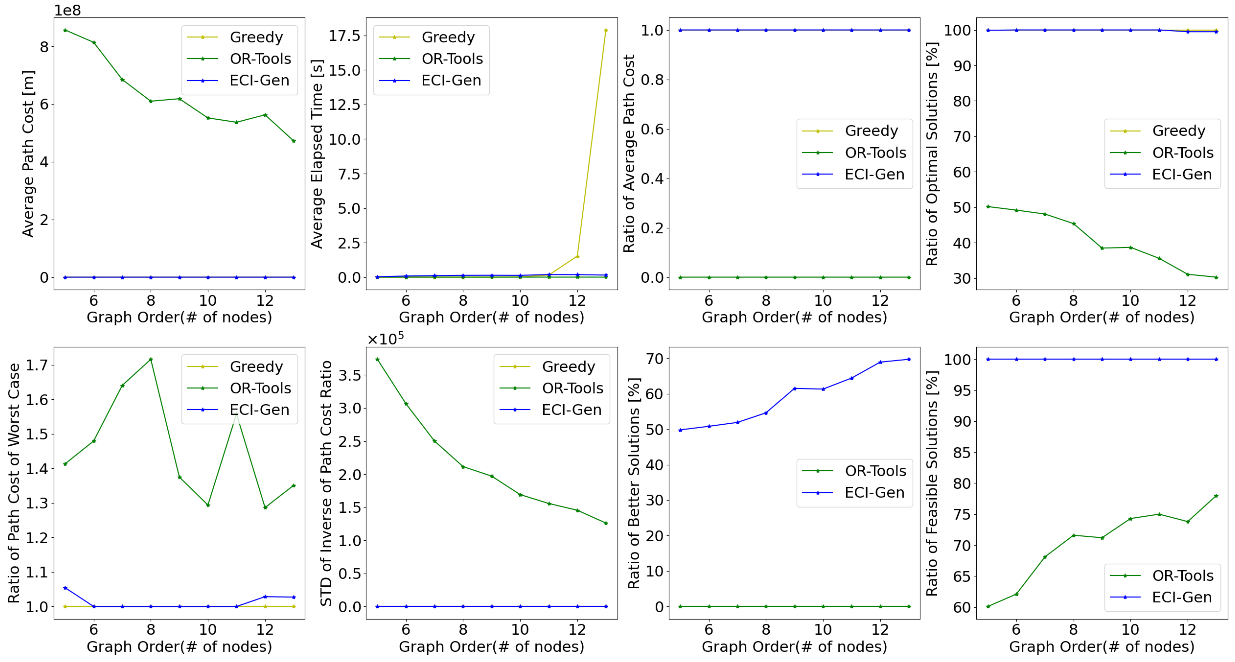


Figure B.7: The validation of the ECI-Gen TSP solver on small incomplete graphs ($5 \leqslant K \leqslant 13$) without triangular inequality constraints.

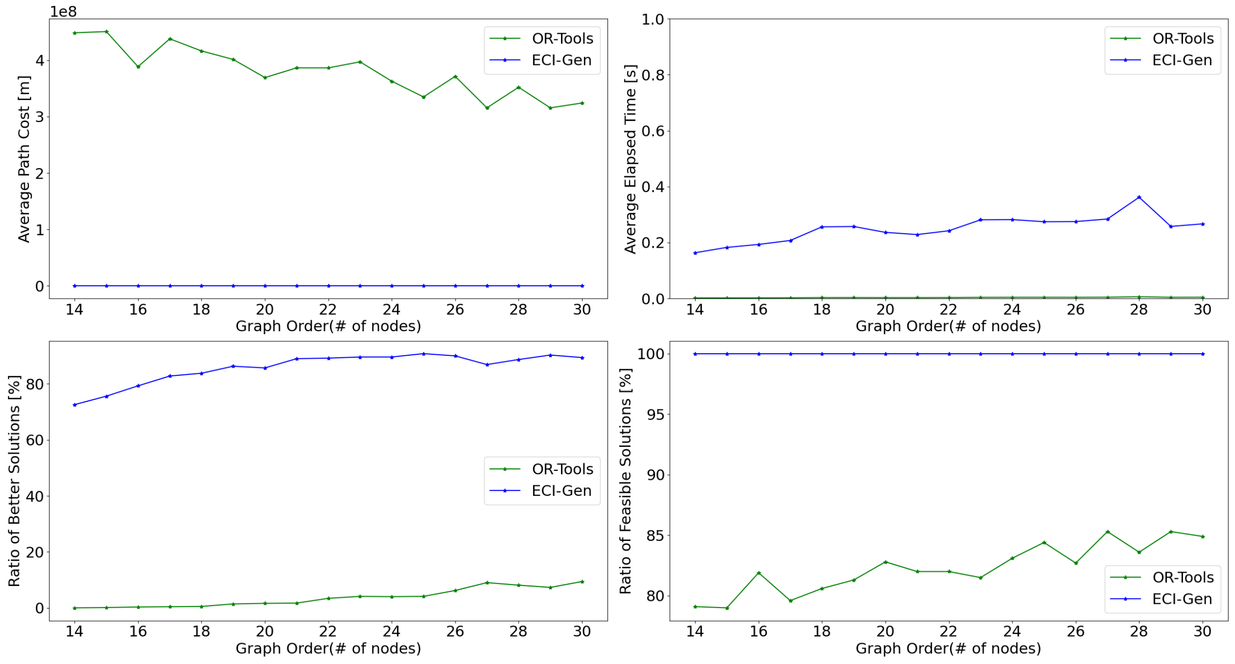


Figure B.8: The validation of the ECI-Gen TSP solver on large incomplete graphs ($14 \leqslant K \leqslant 30$) without triangular inequality constraints.

BIBLIOGRAPHY

[1] C. Olsson and A. Eriksson, "Solving quadratically constrained geometrical problems using lagrangian duality," in *Proceedings of the International Conference Pattern Recognition*. IEEE, 2008, pp. 1–5.

[2] R. Carroll, "Early land vertebrates," *Nature*, vol. 418, no. 6893, pp. 35–36, 2002.

[3] A. J. Wendruff, L. E. Babcock, C. S. Wirkner, J. Kluessendorf, and D. G. Mikulic, "A silurian ancestral scorpion with fossilised internal anatomy illustrating a pathway to arachnid terrestrialisation," *Scientific reports*, vol. 10, no. 1, pp. 1–6, 2020.

[4] K. Steudel, "Limb morphology, bipedal gait, and the energetics of hominid locomotion," *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists*, vol. 99, no. 2, pp. 345–355, 1996.

[5] A. Agrawal, O. Harib, A. Hereid, S. Finet, M. Masselin, L. Praly, A. D. Ames, K. Sreenath, and J. W. Grizzle, "First steps towards translating hzd control of bipedal robots to decentralized control of exoskeletons," *IEEE Access*, vol. 5, pp. 9919–9934, 2017.

[6] O. Harib, A. Hereid, A. Agrawal, T. Gurriet, S. Finet, G. Boeris, A. Duburcq, M. E. Mungai, M. Masselin, A. D. Ames, *et al.*, "Feedback control of an exoskeleton for paraplegics: Toward robustly stable, hands-free dynamic walking," *IEEE Control Systems Magazine*, vol. 38, no. 6, pp. 61–87, 2018.

[7] R. D. Gregg, T. Lenzi, L. J. Hargrove, and J. W. Sensinger, "Virtual constraint control of a powered prosthetic leg: From simulation to experiments with transfemoral amputees," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1455–1471, 2014.

[8] H. Zhao, J. Horn, J. Reher, V. Paredes, and A. D. Ames, "A hybrid systems and optimization-based control approach to realizing multi-contact locomotion on transfemoral prostheses," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2015, pp. 1607–1612.

[9] J. Huang and J. W. Grizzle, "Improvements to Target-Based 3D LiDAR to Camera Calibration," *IEEE Access*, vol. 8, pp. 134 101–134 110, 2020.

[10] Q. Liao, Z. Chen, Y. Liu, Z. Wang, and M. Liu, "Extrinsic calibration of LiDAR and camera with polygon," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 200–205.

[11] L. Zhou, Z. Li, and M. Kaess, “Automatic extrinsic calibration of a camera and a 3D LiDAR using line and plane correspondences,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018, pp. 5562–5569.

[12] X. Gong, Y. Lin, and J. Liu, “3D LiDAR-camera extrinsic calibration using an arbitrary trihedron,” *Sensors*, vol. 13, no. 2, pp. 1902–1918, 2013.

[13] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, “LiDAR-camera calibration using 3D-3D point correspondences,” *arXiv preprint arXiv:1705.09785*, 2017.

[14] S. Verma, J. S. Berrio, S. Worrall, and E. Nebot, “Automatic extrinsic calibration between a camera and a 3D LiDAR using 3D point and plane correspondences,” *arXiv preprint arXiv:1904.12433*, 2019.

[15] J. Jiao, Q. Liao, Y. Zhu, T. Liu, Y. Yu, R. Fan, L. Wang, and M. Liu, “A novel dual-LiDAR calibration algorithm using planar surfaces,” *arXiv preprint arXiv:1904.12116*, 2019.

[16] E.-S. Kim and S.-Y. Park, “Extrinsic calibration of a camera-LiDAR multi sensor system using a planar chessboard,” in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2019, pp. 89–91.

[17] C. Guindel, J. Beltrán, D. Martín, and F. García, “Automatic extrinsic calibration for LiDAR-stereo vehicle sensor setups,” in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.

[18] A.-I. García-Moreno, J.-J. Gonzalez-Barbosa, F.-J. Ornelas-Rodriguez, J. B. Hurtado-Ramos, and M.-N. Primo-Fuentes, “LiDAR and panoramic camera extrinsic calibration approach using a pattern plane,” in *Mexican Conference on Pattern Recognition*. Springer, 2013, pp. 104–113.

[19] S. Mishra, G. Pandey, and S. Saripalli, “Extrinsic calibration of a 3d-lidar and a camera,” *arXiv preprint arXiv:2003.01213*, 2020.

[20] B. Xue, J. Jiao, Y. Zhu, L. Zheng, D. Han, M. Liu, and R. Fan, “Automatic calibration of dual-lidars using two poles stickered with retro-reflective tape,” *arXiv preprint arXiv:1911.00635*, 2019.

[21] J.-K. Huang, C. Feng, M. Achar, M. Ghaffari, and J. W. Grizzle, “Global Unifying Intrinsic Calibration for Spinning and Solid-State LiDARs,” *arXiv preprint arXiv:2012.03321*, 2020.

[22] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis, “3d lidar–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization,” *International Journal of Robotics Research*, vol. 31, no. 4, pp. 452–467, 2012.

[23] G. Pandey, J. McBride, S. Savarese, and R. Eustice, “Extrinsic calibration of a 3d laser scanner and an omnidirectional camera,” *IFAC Proceedings Volumes*, vol. 43, no. 16, pp. 336–341, 2010.

[24] C. Glennie and D. D. Lichti, "Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning," *Remote sensing*, vol. 2, no. 6, pp. 1610–1624, 2010.

[25] H. Nouira, J.-E. Deschaud, and F. Goulette, "Point cloud refinement with a target-free intrinsic calibration of a mobile multi-beam lidar system," 2016.

[26] T. Chan, D. D. Lichti, and D. Belton, "Temporal analysis and automatic calibration of the velodyne hdl-32e lidar system," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci*, vol. 2, pp. 61–66, 2013.

[27] G. Atanacio-Jiménez, J.-J. González-Barbosa, J. B. Hurtado-Ramos, F. J. Ornelas-Rodríguez, H. Jiménez-Hernández, T. García-Ramirez, and R. González-Barbosa, "Lidar velodyne hdl-64e calibration using pattern planes," *International Journal of Advanced Robotic Systems*, vol. 8, no. 5, p. 59, 2011.

[28] N. Muhammad and S. Lacroix, "Calibration of a rotating multi-beam LIDAR," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 5648–5653.

[29] R. Bergelt, O. Khan, and W. Hardt, "Improving the intrinsic calibration of a velodyne lidar sensor," in *IEEE Sensors Journal*. IEEE, 2017, pp. 1–3.

[30] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3400–3407.

[31] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 4193–4198.

[32] D. Wagner and D. Schmalstieg, "Artoolkit on the pocketpc platform," in *IEEE International Augmented Reality Toolkit Workshop*. IEEE, 2003, pp. 14–15.

[33] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2005, pp. 590–596.

[34] M. Krogius, A. Haggenmiller, and E. Olson, "Flexible layouts for fiducial tags," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1898–1903.

[35] J. DeGol, T. Bretl, and D. Hoiem, "Chromatag: a colored marker and fast detection algorithm," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1472–1481.

[36] B. Atcheson, F. Heide, and W. Heidrich, "Caltag: High precision fiducial markers for camera calibration." in *VMV*, vol. 10. Citeseer, 2010, pp. 41–48.

[37] M. Fiala, "Comparing artag and artoolkit plus fiducial marker systems," in *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*. IEEE, 2005, pp. 6–pp.

[38] B. Wang, "Lftag: A scalable visual fiducial system with low spatial frequency," *arXiv preprint arXiv:2006.00842*, 2020.

[39] J. K. Huang, S. Wang, M. Ghaffari, and J. W. Grizzle, "LiDARTag: A Real-Time Fiducial Tag System for Point Clouds," *IEEE Robotics and Automation Letters*, pp. 1–1, 2021.

[40] Jiunn-Kai Huang, Shoutian Wang, Maani Ghaffari, and Jessy W. Grizzle, "LiDARTag: A real-time fiducial tag using point clouds," *arXiv preprint arXiv:1908.10349*, 2020.

[41] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2016, pp. 1271–1278.

[42] A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2019.

[43] J.-K. Huang, W. Clark, and J. W. Grizzle, "Optimal target shape for lidar pose estimation," *arXiv preprint arXiv:2109.01181*, 2021.

[44] J.-K. Huang and J. W. Grizzle, "Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain," *arXiv preprint arXiv:2108.06699*, 2021.

[45] J.K. Huang and Jessy W. Grizzle, "Cassie Torso Design," 2021. [Online]. Available: https://github.com/UMich-BipedLab/torso_design_for_cassie

[46] J. Huang, "Cassie Blue walks Around the Wavefield," https://youtu.be/LhFC45jweFMc, 2019.

[47] ——, "Cassie vs Spin: A True Story," https://youtu.be/f-FvcHOQXPc, 2019.

[48] ——, "Halloween Edition: Cassie Autonomy," HalloweenEdition:CassieAutonomy, 2019.

[49] ——. (2021) Fully Autonomous on the Wave Field 2021. https://youtube.com/playlist?list=PLFe0SMV3hBCAYMpChjW_-LQ71PNfieWAe.

[50] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 790–797, April 2020.

[51] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *Proceedings of the American Control Conference*. IEEE, 2019, pp. 4559–4566.

[52] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic targetless extrinsic calibration of a 3D LiDAR and camera by maximizing mutual information," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2012.

[53] Z. Taylor and J. Nieto, "Automatic calibration of LiDAR and camera images using normalized mutual information," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013.

[54] J. Jeong, Y. Cho, and A. Kim, "The road is enough! extrinsic calibration of non-overlapping stereo camera and LiDAR using road information," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2831–2838, 2019.

[55] J. Jiang, P. Xue, S. Chen, Z. Liu, X. Zhang, and N. Zheng, "Line feature based extrinsic calibration of LiDAR and camera," in *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 2018, pp. 1–6.

[56] S. Erke, D. Bin, N. Yiming, X. Liang, and Z. Qi, "A fast calibration approach for onboard lidar-camera systems," *International Journal of Advanced Robotic Systems*, vol. 17, no. 2, p. 1729881420909606, 2020.

[57] W. Zhen, Y. Hu, J. Liu, and S. Scherer, "A joint optimization approach of LiDAR-camera fusion for accurate dense 3D reconstructions," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3585–3592, 2019.

[58] A.-S. Vaida and S. Nedevschi, "Automatic extrinsic calibration of lidar and monocular camera images," in *2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2019, pp. 117–124.

[59] P. An, T. Ma, K. Yu, B. Fang, J. Zhang, W. Fu, and J. Ma, "Geometric calibration for lidar-camera system fusing 3d-2d and 3d-3d point correspondences," *Optics Express*, vol. 28, no. 2, pp. 2122–2141, 2020.

[60] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International Journal of Computer Vision*, vol. 81, no. 2, p. 155, 2009.

[61] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[62] J.K. Huang and Jessy W. Grizzle, "Extrinsic LiDAR Camera Calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/extrinsic_lidar_camera_calibration

[63] C. G. Harris, M. Stephens, *et al.*, "A combined corner and edge detector." in *Alvey vision conference*, vol. 15, no. 50. Citeseer, 1988, pp. 10–5244.

[64] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443.

[65] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 105–119, 2008.

[66] J. Canny, "A computational approach to edge detection," in *Readings in computer vision*. Elsevier, 1987, pp. 184–203.

[67] J. S. Lim, "Two-dimensional signal and image processing," *Englewood Cliffs, NJ, Prentice Hall, 1990, 710 p.*, 1990.

[68] O. R. Vincent, O. Folorunso, *et al.*, "A descriptive algorithm for sobel image edge detection," in *Proceedings of Informing Science & IT Education Conference (InSITE)*, vol. 40. Informing Science Institute California, 2009, pp. 97–107.

[69] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision second edition*. Cambridge University Press, 2000.

[70] D. A. Forsyth and J. Ponce, *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[71] R. Ochilbek, "A new approach (extra vertex) and generalization of shoelace algorithm usage in convex polygon (point-in-polygon)," in *2018 14th International Conference on Electronics Computer and Computation (ICECCO)*. IEEE, pp. 206–212.

[72] B. Braden, "The surveyor's area formula," *The College Mathematics Journal*, vol. 17, no. 4, pp. 326–337, 1986.

[73] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Information Processing Letters*, vol. 1, pp. 132–133, 1972.

[74] A. M. Andrew, "Another efficient algorithm for convex hulls in two dimensions," *Information Processing Letters*, vol. 9, no. 5, pp. 216–219, 1979.

[75] "Autonomous Navigation and 3D Semantic Mapping on Bipedal Robot Cassie Blue," https://youtu.be/N8THn5YGxPw, accessed: 2020-01-31.

[76] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[77] D. Liebowitz and A. Zisserman, "Metric rectification for perspective images of planes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1998, pp. 482–488.

[78] G. P. Stein, "Accurate internal camera calibration using rotation, with analysis of sources of error," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 1995, pp. 230–236.

[79] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 473–480.

[80] Itseez, "Open source computer vision library," https://github.com/itseez/opencv, 2015.

[81] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009.

[82] R. Tron, D. M. Rosen, and L. Carlone, "On the inclusion of determinant constraints in lagrangian duality for 3d slam," in *Proceedings of the Robotics: Science and Systems Conference*, 2015.

[83] L. Carlone, D. M. Rosen, G. Calafiore, J. J. Leonard, and F. Dellaert, "Lagrangian duality in 3d slam: Verification techniques and optimal solutions," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2015, pp. 125–132.

[84] J. Briales and J. Gonzalez-Jimenez, "Convex global 3d registration with lagrangian duality," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4960–4969.

[85] M. Grant and S. Boyd, "Cvx: Matlab software for disciplined convex programming, version 2.1," 2014.

[86] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, "Intrinsic LiDAR Calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/LiDAR_intrinsic_calibration

[87] J.K. Huang, C Feng and Jessy W. Grizzle, "LiDAR Simulator Package," 2020. [Online]. Available: https://github.com/UMich-BipedLab/lidar_simulator

[88] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, "Global Sim3 Solver," 2020. [Online]. Available: https://github.com/UMich-BipedLab/global_sim3_solver

[89] A. Frederiksen and S. Hartmann, "Lidar device including at least one diffuser element," Feb. 6 2020, uS Patent App. 16/525,010.

[90] C. V. Poulton, M. J. Byrd, P. Russo, E. Timurdogan, M. Khandaker, D. Vermeulen, and M. R. Watts, "Long-range lidar and free-space data communication with high-performance optical phased arrays," *IEEE Journal of Quantum Electronics*, vol. 25, no. 5, pp. 1–8, 2019.

[91] C. V. Poulton, P. Russo, E. Timurdogan, M. Whitson, M. J. Byrd, E. Hosseini, B. Moss, Z. Su, D. Vermeulen, and M. R. Watts, "High-performance integrated optical phased arrays for chip-scale beam steering and lidar," in *CLEO: Applications and Technology*. Optical Society of America, 2018, pp. ATu3R–2.

[92] H. W. Yoo, N. Druml, D. Brunner, C. Schwarzl, T. Thurner, M. Hennecke, and G. Schitter, "Mems-based lidar for autonomous driving," *e & i Elektrotechnik und Informationstechnik*, vol. 135, no. 6, pp. 408–415, 2018.

[93] G. Pelz and N. Elbel, "Matrix light source and detector device for solid-state lidar," Feb. 6 2020, uS Patent App. 16/052,862.

[94] N. Druml, I. Maksymova, T. Thurner, D. van Lierop, M. Hennecke, and A. Foroutan, “1d mems micro-scanning lidar,” in *Conference on Sensor Device Technologies and Applications (SENSORDEVICES)*, vol. 9, 2018.

[95] X. Lee and C. Wang, “Optical design for uniform scanning in mems-based 3d imaging lidar,” *Applied optics*, vol. 54, no. 9, pp. 2219–2223, 2015.

[96] D. Wang, C. Watkins, and H. Xie, “Mems mirrors for lidar: A review,” *Micromachines*, vol. 11, no. 5, p. 456, 2020.

[97] P. García-Gómez, S. Royo, N. Rodrigo, and J. R. Casas, “Geometric model and calibration method for a solid-state lidar,” *Sensors*, vol. 20, no. 10, p. 2898, 2020.

[98] M. M. Moscato, L. Titolo, M. A. Feliú, and C. A. Muñoz, “Provably correct floating-point implementation of a point-in-polygon algorithm,” in *International Symposium on Formal Methods*. Springer, 2019, pp. 21–37.

[99] M. El-Salamony and A. Guaily, “Enhanced modified-polygon method for point-in-polygon problem,” in *Recent Advances in Engineering Mathematics and Physics*. Springer, 2020, pp. 47–61.

[100] D. Alciatore and R. Miranda, “A winding number and point-in-polygon algorithm,” *Glaxo Virtual Anatomy Project Research Report, Department of Mechanical Engineering, Colorado State University*, 1995.

[101] J.K. Huang, Shoutian Wang, Maani Ghaffari, and Jessy W. Grizzle, “LiDARTag ROS Package,” 2020. [Online]. Available: https://github.com/UMich-BipedLab/LiDARTag

[102] J.K. Huang, C Feng and Jessy W. Grizzle, “AprilTag ROS Package,” 2020. [Online]. Available: https://github.com/UMich-BipedLab/AprilTag_ROS

[103] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, “Automatic Extrinsic LiDAR Camera Calibration,” 2020. [Online]. Available: https://github.com/UMich-BipedLab/automatic_lidar_camera_calibration

[104] J. DeGol, T. Bretl, and D. Hoiem, “Improved structure from motion using fiducial marker matching,” in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 273–288.

[105] M. Ghaffari, W. Clark, A. Bloch, R. M. Eustice, and J. W. Grizzle, “Continuous direct sparse visual odometry from RGB-D images,” in *Proceedings of the Robotics: Science and Systems Conference*, Freiburg, Germany, June 2019.

[106] M. Klopschitz and D. Schmalstieg, “Automatic reconstruction of wide-area fiducial marker models,” in *IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 2007, pp. 71–74.

[107] M. Velas, M. Spanel, Z. Materna, and A. Herout, “Calibration of rgb camera with velodyne lidar,” 2014.

[108] Y. Park, S. Yun, C. S. Won, K. Cho, K. Um, and S. Sim, "Calibration between color camera and 3d lidar instruments with a polygonal planar board," *Sensors*, vol. 14, no. 3, pp. 5333–5353, 2014.

[109] A. Trachtenbert, "Computational methods in coding theory," Master's thesis, University of Illinois at Urbana-Champaign, 1996.

[110] F. Bergamasco, A. Albarelli, E. Rodola, and A. Torsello, "Rune-tag: A high accuracy fiducial marker with strong occlusion resilience," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 113–120.

[111] L. Calvet, P. Gurdjos, C. Griwodz, and S. Gasparini, "Detection and accurate localization of circular fiducials under highly challenging conditions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 562–570.

[112] O. Grinchuk, V. Lebedev, and V. Lempitsky, "Learnable visual markers," in *Proceedings of the Advances in Neural Information Processing Systems Conference*, 2016, pp. 4143–4151.

[113] D. Hu, D. DeTone, and T. Malisiewicz, "Deep charuco: Dark charuco marker pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8436–8444.

[114] S. Song and J. Xiao, "Sliding shapes for 3d object detection in depth images," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 634–651.

[115] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2017, pp. 1355–1361.

[116] S. Song and J. Xiao, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[117] ——, "Deep sliding shapes for amodal 3d object detection in rgb-d images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 808–816.

[118] B. Li, "3D fully convolutional network for vehicle detection in point cloud," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2017, pp. 1513–1518.

[119] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 391–405.

[120] K. E. Van de Sande, J. R. Uijlings, T. Gevers, A. W. Smeulders, *et al.*, "Segmentation as selective search for object recognition." in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 1, no. 2, 2011, p. 7.

[121] J. Carreira and C. Sminchisescu, “Cpmc: Automatic object segmentation using constrained parametric min-cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1312–1328, 2011.

[122] J. Li, S. Luo, Z. Zhu, H. Dai, A. S. Krylov, Y. Ding, and L. Shao, “3D IoU-Net: IoU guided 3D object detector for point clouds,” *arXiv preprint arXiv:2004.04962*, 2020.

[123] S. Shi, X. Wang, and H. Li, “Pointrcnn: 3D object proposal generation and detection from point cloud,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 770–779.

[124] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” *International Journal of Robotics Research*, 2013.

[125] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, “Semantic3d net: A new large scale point cloud classification benchmark,” in *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2017, pp. 91–98.

[126] W. Kim, M. S. Ramanagopal, C. Barto, M.-Y. Yu, K. Rosaen, N. Goumas, R. Vasudevan, and M. Johnson-Roberson, “Pedx: Benchmark dataset for metric 3-d pose estimation of pedestrians in complex urban intersections,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1940–1947, 2019.

[127] Velodyne Lidar, “Velodyne Ultra Puck: VLP-32C User Manual,” 2019. [Online]. Available: https://icave2.cse.buffalo.edu/resources/sensor-modeling/VLP32CManual.pdf

[128] L. Naimark and E. Foxlin, “Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker,” in *Proceedings of the International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 2002, p. 27.

[129] J. Rekimoto and Y. Ayatsuka, “Cybercode: designing augmented reality environments with visual tags,” in *Proceedings of the Designing augmented reality environments*. ACM, 2000, pp. 1–10.

[130] Y. Cho, J. Lee, and U. Neumann, “A multi-ring color fiducial system and an intensity-invariant detection method for scalable fiducial-tracking augmented reality,” in *In IWAR*. Citeseer, 1998.

[131] W. Clark, M. Ghaffari, and A. Bloch, “Nonparametric continuous sensor registration,” *arXiv preprint arXiv:2001.04286*, 2020.

[132] T. Shan and B. Englot, “LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018, pp. 4758–4765.

[133] S. C. Johnson, “Hierarchical clustering schemes,” *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[134] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[135] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[136] S. G. Johnson, "The nlopt nonlinear-optimization package," 2014. [Online]. Available: https://github.com/stevengj/nlopt

[137] J. C. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, no. 1, pp. 33–51, 1975.

[138] A. L. Price, N. J. Patterson, R. M. Plenge, M. E. Weinblatt, N. A. Shadick, and D. Reich, "Principal components analysis corrects for stratification in genome-wide association studies," *Nature genetics*, vol. 38, no. 8, p. 904, 2006.

[139] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*. MIT press, 2006, vol. 1.

[140] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *International conference on computational learning theory*. Springer, 2001, pp. 416–426.

[141] G. Wahba, *Spline models for observational data*. SIAM, 1990.

[142] G. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *Journal of mathematical analysis and applications*, vol. 33, no. 1, pp. 82–95, 1971.

[143] "M-air at the university of michigan, ann arbor," 2018. [Online]. Available: https://robotics.umich.edu/about/mair/

[144] K. Svanberg, "A class of globally convergent optimization methods based on conservative convex separable approximations," *SIAM journal on optimization*, vol. 12, no. 2, pp. 555–573, 2002.

[145] ——, "The method of moving asymptotes: a new method for structural optimization," *International journal for numerical methods in engineering*, vol. 24, no. 2, pp. 359–373, 1987.

[146] D. Q. Huynh, "Metrics for 3D rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[147] D. Padua, Ed., *TBB (Intel Threading Building Blocks)*. Boston, MA: Springer US, 2011, pp. 2029–2029. [Online]. Available: https://doi.org/10.1007/978-0-387-09766-4_2080

[148] J. L. Blanco and P. K. Rai, "nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees," https://github.com/jlblancoc/nanoflann, 2014.

[149] J.K. Huang and Jessy W. Grizzle, "Optimal Shape Design for LiDAR," 2021. [Online]. Available: https://github.com/UMich-BipedLab/optimal_shape_generation

[150] ——, “Global Pose Estimation using Optimal Shape,” 2021. [Online]. Available: https://github.com/UMich-BipedLab/optimal_shape_global_pose_estimation

[151] B. Muralikrishnan, P. Rachakonda, V. Lee, M. Shilling, D. Sawyer, G. Cheok, and L. Cournoyer, “Relative range error evaluation of terrestrial laser scanners using a plate, a sphere, and a novel dual-sphere-plate target,” *Measurement*, vol. 111, pp. 60–68, 2017.

[152] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[153] J. N. Cederberg, *A course in modern geometries*. Springer Science & Business Media, 2013.

[154] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[155] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[156] D. Lenton, “Projective transformations in 2d,” 2019. [Online]. Available: https://mc.ai/part-ii-projective-transformations-in-2d/

[157] J. Faigl and G. A. Hollinger, “Unifying multi-goal path planning for autonomous data collection,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2937–2942.

[158] J. McMahon and E. Plaku, “Autonomous underwater vehicle mine countermeasures mission planning via the physical traveling salesman problem,” in *OCEANS 2015-MTS/IEEE Washington*. IEEE, 2015, pp. 1–5.

[159] X. Zhuo, M. Liu, Y. Wei, G. Yu, F. Qu, and R. Sun, “Auv-aided energy-efficient data collection in underwater acoustic sensor networks,” *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 010–10 022, 2020.

[160] H. Hu, K. Xiong, G. Qu, Q. Ni, P. Fan, and K. B. Letaief, “Aoi-minimal trajectory planning and data collection in uav-assisted wireless powered iot networks,” *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 1211–1223, 2020.

[161] M. Samir, S. Sharafeddine, C. M. Assi, T. M. Nguyen, and A. Ghrayeb, “Uav trajectory planning for data collection from time-constrained iot devices,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 34–46, 2019.

[162] M. Jünger, G. Reinelt, and G. Rinaldi, “The traveling salesman problem,” *Handbooks in operations research and management science*, vol. 7, pp. 225–330, 1995.

[163] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook, *The traveling salesman problem*. Princeton university press, 2011.

[164] C. Ma, R. He, and W. Zhang, “Path optimization of taxi carpooling,” *PLoS One*, vol. 13, no. 8, p. e0203221, 2018.

[165] H. Huang, D. Bucher, J. Kissling, R. Weibel, and M. Raubal, “Multimodal route planning with public transport and carpooling,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3513–3525, 2018.

[166] A. O. Al-Abbasi, A. Ghosh, and V. Aggarwal, “Deeppool: Distributed model-free algorithm for ride-sharing using deep reinforcement learning,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4714–4727, 2019.

[167] S.-C. Huang, M.-K. Jiau, and Y.-P. Liu, “An ant path-oriented carpooling allocation approach to optimize the carpool service problem with time windows,” *IEEE Systems Journal*, vol. 13, no. 1, pp. 994–1005, 2018.

[168] Y. Duan, T. Mosharraf, J. Wu, and H. Zheng, “Optimizing carpool scheduling algorithm through partition merging,” in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[169] M. Tamannaei and I. Irandoost, “Carpooling problem: A new mathematical model, branch-and-bound, and heuristic beam search algorithm,” *Journal of Intelligent Transportation Systems*, vol. 23, no. 3, pp. 203–215, 2019.

[170] S. Hulagu and H. B. Celikoglu, “An electric vehicle routing problem with intermediate nodes for shuttle fleets,” *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[171] H. K. Suman and N. B. Bolia, “Improvement in direct bus services through route planning,” *Transport Policy*, vol. 81, pp. 263–274, 2019.

[172] Y. Lyu, C.-Y. Chow, V. C. Lee, J. K. Ng, Y. Li, and J. Zeng, “Cb-planner: A bus line planning framework for customized bus systems,” *Transportation Research Part C: Emerging Technologies*, vol. 101, pp. 233–253, 2019.

[173] OpenStreetMap contributors, “Planet dump retrieved from https://planet.osm.org ,” https://www.openstreetmap.org, 2017.

[174] J. Janos, V. Vonásek, and R. Penicka, “Multi-goal path planning using multiple random trees,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4201–4208, 2021.

[175] D. Devaurs, T. Siméon, and J. Cortés, “A multi-tree extension of the transition-based rrt: Application to ordering-and-pathfinding problems in continuous cost spaces,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2991–2996.

[176] V. Vonásek and R. Penicka, “Space-filling forest for multi-goal path planning,” in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 1587–1590.

[177] B. Englot and F. Hover, “Multi-goal feasible path planning using ant colony optimization,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2255–2260.

[178] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis, II, "An analysis of several heuristics for the traveling salesman problem," *SIAM journal on computing*, vol. 6, no. 3, pp. 563–581, 1977.

[179] J.-Y. Potvin, "Genetic algorithms for the traveling salesman problem," *Annals of Operations Research*, vol. 63, no. 3, pp. 337–370, 1996.

[180] C. Moon, J. Kim, G. Choi, and Y. Seo, "An efficient genetic algorithm for the traveling salesman problem with precedence constraints," *European Journal of Operational Research*, vol. 140, no. 3, pp. 606–617, 2002.

[181] H. Braun, "On solving travelling salesman problems by genetic algorithms," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1990, pp. 129–133.

[182] Z. H. Ahmed, "Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator," *International Journal of Biometrics & Bioinformatics (IJBB)*, vol. 3, no. 6, p. 96, 2010.

[183] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory." in *SODA*, vol. 5. Citeseer, 2005, pp. 156–165.

[184] J. Van Den Berg, R. Shah, A. Huang, and K. Goldberg, "Anytime nonparametric a," in *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[185] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. [Online]. Available: https://doi.org/10.1109/tssc.1968.300136

[186] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[187] M. Likhachev and A. Stentz, "R* search," *University of Pennsylvania*, 2008.

[188] M. Likhachev, G. J. Gordon, and S. Thrun, "Ara*: Anytime a* with provable bounds on sub-optimality," *Advances in neural information processing systems*, vol. 16, 2003.

[189] D. Harabor and A. Grastien, "The jps pathfinding system," in *International Symposium on Combinatorial Search*, vol. 3, no. 1, 2012.

[190] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning aˆ*," *Artificial Intelligence*, vol. 155, no. 1-2, pp. 93–146, 2004.

[191] A. Stentz, "Optimal and efficient path planning for partially known environments," in *Intelligent unmanned ground vehicles*. Springer, 1997, pp. 203–220.

[192] S. Koenig and M. Likhachev, "Dˆ* lite," *Aaai/iaai*, vol. 15, pp. 476–483, 2002.

[193] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[194] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[195] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the rrt*," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011, pp. 1478–1483.

[196] S. Morgan and M. S. Branicky, "Sampling-based planning for discrete spaces," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 2. IEEE, 2004, pp. 1938–1945.

[197] M. S. Branicky, M. M. Curtiss, J. A. Levine, and S. B. Morgan, "Rrts for nonlinear, discrete, and hybrid planning and control," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 1. IEEE, 2003, pp. 657–663.

[198] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete rrt for exploration of implicit roadmaps in multi-robot motion planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 501–513, 2016.

[199] R. Shome, K. Solovey, A. Dobson, D. Halperin, and K. E. Bekris, "drrt*: Scalable and informed asymptotically-optimal multi-robot motion planning," *Autonomous Robots*, vol. 44, no. 3, pp. 443–467, 2020.

[200] M. D. Simoni, E. Kutanoglu, and C. G. Claudel, "Optimization and analysis of a robot-assisted last mile delivery system," *Transportation Research Part E: Logistics and Transportation Review*, vol. 142, p. 102049, 2020.

[201] S. Naccache, J.-F. Côté, and L. C. Coelho, "The multi-pickup and delivery problem with time windows," *European Journal of Operational Research*, vol. 269, no. 1, pp. 353–362, 2018.

[202] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takác, "Reinforcement learning for solving the vehicle routing problem," *Advances in neural information processing systems*, vol. 31, 2018.

[203] E. H.-C. Lu and Y.-W. Yang, "A hybrid route planning approach for logistics with pickup and delivery," *Expert Systems with Applications*, vol. 118, pp. 482–492, 2019.

[204] J. A. Bondy, U. S. R. Murty, *et al.*, *Graph theory with applications*. Macmillan London, 1976, vol. 290.

[205] A. P. Punnen, "The traveling salesman problem: Applications, formulations and variations," in *The traveling salesman problem and its variations*. Springer, 2007, pp. 1–28.

[206] D. B. West *et al.*, *Introduction to graph theory*. Prentice hall Upper Saddle River, 2001, vol. 2.

[207] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[208] Z. Galil and G. F. Italiano, "Data structures and algorithms for disjoint set union problems," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 319–344, 1991.

[209] G. Van Brummelen, "Heavenly mathematics," in *Heavenly Mathematics*. Princeton University Press, 2012.

[210] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[211] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2016, pp. 4304–4311.

[212] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[213] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1360–1367.

[214] R. Hartley, M. G. Jadidi, J. Grizzle, and R. M. Eustice, "Contact-aided invariant extended Kalman filtering for legged robot state estimation," in *Proceedings of the Robotics: Science and Systems Conference*, Pittsburgh, Pennsylvania, June 2018.

[215] Y. Gong and J. Grizzle, "Zero dynamics, pendulum models, and angular momentum in feedback control of bipedal locomotion," 2021.

[216] ——, "Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller," *arXiv preprint arXiv:2008.10763*, 2020.

[217] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 3019–3026, 2018.

[218] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *International Conference on Climbing and Walking Robots (CLAWAR)*, 2014.

[219] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[220] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI*, vol. 104, no. 2, 2010.

[221] ——, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2010, pp. 7681–7687.

[222] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "Pq-rrt*: An improved path planning algorithm for mobile robots," *Expert Systems with Applications*, vol. 152, p. 113425, 2020.

[223] L. Palmieri, S. Koenig, and K. O. Arras, “Rrt-based nonholonomic motion planning using any-angle path biasing,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2016, pp. 2775–2781.

[224] J. Wang, M. Q.-H. Meng, and O. Khatib, “Eb-rrt: Optimal motion planning for mobile robots,” *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 2063–2073, 2020.

[225] F. Golbol, M. M. Ankarali, and A. Saranli, “Rg-trees: trajectory-free feedback motion planning using sparse random reference governor trees,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018, pp. 6506–6511.

[226] O. Arslan and D. E. Koditschek, “Sensor-based reactive navigation in unknown convex sphere worlds,” *International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 196–223, 2019.

[227] S. Paternain, D. E. Koditschek, and A. Ribeiro, “Navigation functions for convex potentials in a space with convex obstacles,” *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 2944–2959, 2017.

[228] O. Arslan and D. E. Koditschek, “Exact robot navigation using power diagrams,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2016, pp. 1–8.

[229] D. E. Koditschek and E. Rimon, “Robot navigation functions on manifolds with boundary,” *Advances in applied mathematics*, vol. 11, no. 4, pp. 412–442, 1990.

[230] E. Rimon, “Exact robot navigation using artificial potential functions,” Ph.D. dissertation, Yale University, 1990.

[231] J. Borenstein and Y. Koren, “Real-time obstacle avoidance for fast mobile robots,” *IEEE Transactions on systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989.

[232] D. Koditschek, “Exact robot navigation by means of potential functions: Some topological considerations,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4. IEEE, 1987, pp. 1–6.

[233] J. V. Gómez, A. Lumbier, S. Garrido, and L. Moreno, “Planning robot formations with fast marching square including uncertainty conditions,” *Robotics and Autonomous Systems*, vol. 61, no. 2, pp. 137–152, 2013.

[234] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, “Lqr-trees: Feedback motion planning via sums-of-squares verification,” *International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[235] J. J. Park and B. Kuipers, “A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011, pp. 4896–4902.

[236] ——, “Feedback motion planning via non-holonomic rrt* for mobile robots,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 4035–4040.

[237] L. Janson, E. Schmerling, A. Clark, and M. Pavone, “Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions,” *International Journal of Robotics Research*, vol. 34, no. 7, pp. 883–921, 2015.

[238] M. Otte and E. Frazzoli, “Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning,” *International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[239] G. Vailland, V. Gouranton, and M. Babel, “Cubic bézier local path planner for non-holonomic feasible and comfortable path generation,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2021.

[240] C. Lau and K. Byl, “Smooth rrt-connect: An extension of rrt-connect for practical use in robots,” in *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, 2015, pp. 1–7.

[241] W. G. Aguilar, S. Morales, H. Ruiz, and V. Abad, “Rrt* gl based optimal path planning for real-time navigation of uavs,” in *International Work-Conference on Artificial Neural Networks*. Springer, 2017, pp. 585–595.

[242] X. Lan and S. Di Cairano, “Continuous curvature path planning for semi-autonomous vehicle maneuvers using rrt,” in *2015 European Control Conference (ECC)*, 2015, pp. 2360–2365.

[243] H.-T. L. Chiang and L. Tapia, “Colreg-rrt: An rrt-based colregs-compliant motion planner for surface vehicle navigation,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2024–2031, 2018.

[244] T. T. Enevoldsen, C. Reinartz, and R. Galeazzi, “Colregs-informed rrt* for collision avoidance of marine crafts,” *arXiv preprint arXiv:2103.14426*, 2021.

[245] V. Parque and T. Miyashita, “Smooth curve fitting of mobile robot trajectories using differential evolution,” *IEEE Access*, vol. 8, pp. 82 855–82 866, 2020.

[246] A. Zdevsar and I. vSkrjanc, “Optimum velocity profile of multiple bernstein-bézier curves subject to constraints for mobile robots,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 9, no. 5, pp. 1–23, 2018.

[247] T. Jusko and E. Stoll, *Scalable Trajectory Optimization Based on Bézier Curves*. Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV, 2016.

[248] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[249] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 6271–6278.

[250] Y. Chen, H. Peng, and J. Grizzle, "Obstacle avoidance for low-speed autonomous vehicles with barrier function," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 194–206, 2018.

[251] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Robustness of control barrier functions for safety critical control," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 54–61, 2015.

[252] N. Wagener, C.-A. Cheng, J. Sacks, and B. Boots, "An online learning approach to model predictive control," *arXiv preprint arXiv:1902.08967*, 2019.

[253] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[254] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017.

[255] S.-K. Kim, R. Thakker, and A.-A. Agha-Mohammadi, "Bi-directional value learning for risk-aware planning under uncertainty," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2493–2500, 2019.

[256] A. Agha, K. Otsu, B. Morrell, D. D. Fan, R. Thakker, A. Santamaria-Navarro, S.-K. Kim, A. Bouman, X. Lei, J. Edlund, *et al.*, "Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge," *arXiv preprint arXiv:2103.11470*, 2021.

[257] "Darpa subterranean challenge." [Online]. Available: https://www.subtchallenge.com

[258] I. D. Miller, F. Cladera, A. Cowley, S. S. Shivakumar, E. S. Lee, L. Jarin-Lipschitz, A. Bhat, N. Rodrigues, A. Zhou, A. Cohen, *et al.*, "Mine tunnel exploration using multiple quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2840–2847, 2020.

[259] M. T. Ohradzansky, E. R. Rush, D. G. Riley, A. B. Mills, S. Ahmad, S. McGuire, H. Biggie, K. Harlow, M. J. Miles, E. W. Frew, *et al.*, "Multi-agent autonomy: Advancements and challenges in subterranean exploration," *arXiv preprint arXiv:2110.04390*, 2021.

[260] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1363–1388, 2020.

[261] D. Tardioli, L. Riazuelo, D. Sicignano, C. Rizzo, F. Lera, J. L. Villarroel, and L. Montano, "Ground robotics in tunnels: Keys and lessons learned after 10 years of research and experiments," *Journal of Field Robotics*, vol. 36, no. 6, pp. 1074–1101, 2019.

[262] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, “Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2997–3004.

[263] R. Mashayekhi, M. Y. I. Idris, M. H. Anisi, I. Ahmedy, and I. Ali, “Informed rrt*-connect: An asymptotically optimal single-query path planning method,” *IEEE Access*, vol. 8, pp. 19 842–19 852, 2020.

[264] I. Noreen, A. Khan, H. Ryu, N. L. Doh, and Z. Habib, “Optimal path planning in cluttered environment using rrt*-ab,” *Intelligent Service Robotics*, vol. 11, no. 1, pp. 41–52, 2018.

[265] I. Noreen, A. Khan, K. Asghar, and Z. Habib, “A path-planning performance comparison of rrt*-ab with mea* in a 2-dimensional environment,” *Symmetry*, vol. 11, no. 7, p. 945, 2019.

[266] A. Robotics, “Cassie Simulators,” http://www.agilityrobotics.com/sims/, 2018.

[267] J.K. Huang and Jessy W. Grizzle, “omni-directional CLF Reactive Planning System for tough terrains,” 2020. [Online]. Available: https://github.com/UMich-BipedLab/CLF_motion_planning

[268] E. D. Sontag, *Mathematical control theory: deterministic finite dimensional systems*. Springer Science & Business Media, 2013, vol. 6.

[269] S. Kajita and K. Tani, “Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1991, pp. 1405–1411 vol.2.

[270] R. Blickhan, “The spring-mass model for running and hopping,” *Journal of Biomechanics*, vol. 22, no. 11, pp. 1217–1227, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021929089902248

[271] J. Grizzle, G. Abba, and F. Plestan, “Asymptotically stable walking for biped robots: analysis via systems with impulse effects,” *IEEE Transactions on Automatic Control*, vol. 46, no. 1, pp. 51–64, 2001.

[272] H. Dai, A. Valenzuela, and R. Tedrake, “Whole-body motion planning with centroidal dynamics and full kinematics,” in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 295–302.

[273] G. Gibson, O. Dosunmu-Ogunbi, Y. Gong, and J. Grizzle, “Terrain-aware foot placement for bipedal locomotion combining model predictive control, virtual constraints, and the alip,” *arXiv preprint arXiv:2109.14862*, 2021.

[274] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, “The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation,” in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 1, 2001, pp. 239–246 vol.1.

[275] H. Lee and W. Chung, “A self-training approach-based traversability analysis for mobile robots in urban environments,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3389–3394.

[276] T. Shan, J. Wang, B. Englot, and K. Doherty, “Bayesian generalized kernel inference for terrain traversability mapping,” in *Conference on Robot Learning*. PMLR, 2018, pp. 829–838.

[277] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, “3d dynamic walking on stepping stones with control barrier functions,” in *Proceedings of the IEEE Conference on Decision and Control*, 2016, pp. 827–834.

[278] Q. Nguyen, X. Da, J. Grizzle, and K. Sreenath, “Dynamic walking on stepping stones with gait library and control barrier functions,” in *Algorithmic Foundations of Robotics XII*. Springer, 2020, pp. 384–399.

[279] J. E. Bresenham, “Algorithm for computer control of a digital plotter,” *IBM Systems journal*, vol. 4, no. 1, pp. 25–30, 1965.

[280] J.-K. Huang, Y. Tan, D. Lee, V. R. Desaraju, and J. W. Grizzle, “Informable multi-objective and multi-directional rrt* system for robot path planning,” 2022.

[281] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart, “Topomap: Topological mapping and navigation based on visual slam maps,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3818–3825.

[282] J. Janos, V. Vonasek, and R. Penicka, “Multi-goal path planning using multiple random trees,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4201–4208, apr 2021.

[283] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[284] L. Gan, Y. Kim, J. W. Grizzle, J. M. Walls, A. Kim, R. M. Eustice, and M. Ghaffari, “Multi-task learning for scalable and dense multi-layer bayesian map inference,” *arXiv preprint arXiv:2106.14986*, 2021.

[285] L. Gan, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, “Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8807–8814, 2022.

[286] “The Wave Field on the North Campus of the University of Michigan,” https://arts.umich.edu/museums-cultural-attractions/wave-field/.

[287] “ICRA 2021 Workshop on Legged Robots (Towards Real-World Deployment of Legged Robots),” https://youtu.be/0Gg8BTs6HLY, accessed: 2021-06-10.

[288] J.K. Huang, Dianhao Chen, Jinze Liu, Yingwen Tan, Dongmyeong Lee, Jianyang Tang, Peter Wrobel, and Jessy W. Grizzle. (2021) Cassie Autonomously Navigates around Obstacles. https://youtu.be/3HVJotA-w4Y.

[289] ——. (2021) Cassie Autonomously Navigates in Four Long Corridors (200 meters). https://youtu.be/PT2mVaKTdT8.

[290] ——. (2021) Cassie Autonomous Navigation: Smooth Motion. https://youtu.be/nPGs4AWLLSg.

[291] E. Haines, "Point in polygon strategies." *Graphics Gems*, vol. 4, pp. 24–46, 1994.

[292] K. Hormann and A. Agathos, "The point in polygon problem for arbitrary polygons," *Computational geometry*, vol. 20, no. 3, pp. 131–144, 2001.

[293] J. Li, W. Wang, and E. Wu, "Point-in-polygon tests by convex decomposition," *Computers & Graphics*, vol. 31, no. 4, pp. 636–648, 2007.

[294] G. N. Kumar and M. Bangi, "An extension to winding number and point-in-polygon algorithm," *IFAC-PapersOnLine*, vol. 51, no. 1, pp. 548–553, 2018.

[295] J. Zhang and S. You, "Speeding up large-scale point-in-polygon test based spatial join on gpus," in *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*, 2012, pp. 23–32.

[296] J. Grunert, "Das pothenotiscfie problem in erweiterter gestalt nebst ber seine anwendungen in der geodsie," *Grunerts Archiv fr Mathematik und Physik*, vol. 1, pp. 238–248, 1841.

[297] M. Liu, F. Colas, L. Oth, and R. Siegwart, "Incremental topological segmentation for semi-structured environments using discretized gvg," *Autonomous Robots*, vol. 38, no. 2, pp. 143–160, 2015.

[298] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proceedings-Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 2–7, 1999.

[299] J. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE Transactions on Circuits and Systems*, vol. 25, no. 9, pp. 772–781, 1978.

[300] Paul L. Fackler, "Notes on Matrix Calculus," 2005. [Online]. Available: https://media.gradebuddy.com/documents/1897145/1ad5e235-824d-4bbf-81d6-ffd2040e37ec.pdf

[301] Google, "Google optimization tools (a.k.a., or-tools) is an open-source, fast and portable software suite for solving combinatorial optimization problems," https://developers.google.com/optimization, 2022.