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

Jiunn-Kai Huang and Jessy W. Grizzle, *Life Fellow, IEEE*

***Abstract*—We propose and experimentally demonstrate a reactive planning system for bipedal robots on unexplored, challenging terrain. The system includes: a multilayer local map for assessing traversability; an anytime omnidirectional control Lyapunov function for use with a rapidly exploring random tree star (RRT*) that generates a vector field for specifying motion between nodes; a subgoal finder when the final goal is outside of the current map; and a finite-state machine to handle high-level mission decisions. The system also includes a reactive thread that copes with robot deviations via a vector field, defined by a closed-loop feedback policy. The vector field provides real-time control commands to the robot's gait controller as a function of instantaneous robot pose. The system is evaluated on various challenging outdoor terrains and cluttered indoor scenes in both simulation and experiment on Cassie Blue, a bipedal robot with 20 degrees of freedom. All implementations are coded in C++ with the robot operating system and are available at https://github.com/UMich-BipedLab/CLF_reactive_planning_system.**

***Index Terms*—Autonomous robots, autonomous systems, legged locomotion, motion planning, path planning, robot sensing systems, robot motion, robot vision systems.**

I. 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 [1], [2], [3], [4], [5], [6], [7], [8], [9]. 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 nonsmooth motion of the actual robot, due to abrupt acceleration or heading changes when transitioning between waypoints/pathways.

This article 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 nonsmooth aspects of the planned motions arising from waypoints/pathways transitions are detrimental to stability of the overall system.

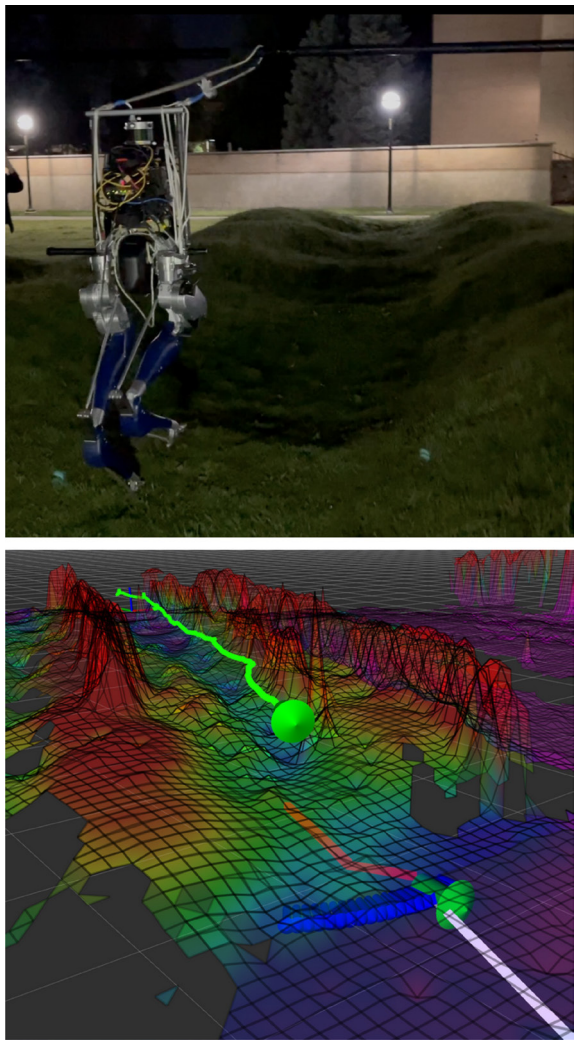


Fig. 1. In the top figure, Cassie Blue autonomously traverses the Wave Field via the proposed reactive planning system, comprised of a planning thread and a reactive thread. The planning thread involves a multi-layer local map to compute traversability, a subgoal finder, and an omnidirectional CLF RRT*. 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.

Manuscript received 15 August 2022; accepted 28 October 2022. Date of publication 6 January 2023; date of current version 7 June 2023. This work was supported by the Toyota Research Institute. The work of J. W. Grizzle was supported by NSF under Grant 1808051 and Grant 2118818. This paper was recommended for publication by Associate Editor M. Fallon and Editor E. Yoshida upon evaluation of the reviewers' comments. *(Corresponding author: Jiunn-Kai Huang.)*

The authors are with the Robotics Institute, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: bjhuang@umich.edu; grizzle@umich.edu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TRO.2022.3228713.

Digital Object Identifier 10.1109/TRO.2022.3228713

Several approaches have been developed to address the non-smooth aspects of paths produced by motion planning, such as reactive motion planning [10], [11], [12], [13], [14], [15], [16], [17], [18] and feedback motion planning [19], [20], [21]. 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 Park and Kuipers [20], [21], which proposed a control Lyapunov function (CLF) to realize reactive planning for a nonholonomic differential-drive wheeled robot. A CLF is a Lyapunov function for a closed-loop system, where at any given time instant, there exists a control input that renders the derivative of the Lyapunov function along the system dynamics negative definite. Hence, a CLF is associated with asymptotically approaching a goal; see Section III-A. While the underlying model of Park and Kuipers [20], [21] was designed for differential-drive robots, which is not directly applicable to a Cassie bipedal robot due to different dynamics and control laws, their basic concept is applicable; see Section III-B for a more detailed discussion. As part of our work, we design an appropriate CLF for robots capable of walking in any direction with 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 Park and Kuipers [20] and [21] 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. In addition, a rudimentary finite-state machine (FSM) is integrated to handle actions such as where to turn at intersections.

II. 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.

A. Sampling-Based Motion Planning

Rapidly exploring tree (RRT) [1] stands out for its low complexity and high efficiency in exploring unknown configuration spaces. Its asymptotically optimal version — RRT* [5] — 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 [1], [2], [3], [4], [5], [6], [7], [8], [9], [22], [23]. However, abrupt (nondifferentiable) 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* [24], [25], [26], [27], [28], [29] that utilize different smoothing techniques or steering functions.

Trajectory smoothing (B-spines, Dubins, or other parametric curves) is often designed independently of robot dynamics [30], [31], [32], 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.

B. Optimization-Based Planning and DARPA Subterranean Challenge

Point-wise in time optimization- and model-based algorithms, such as CLF paired with quadratic programs, (CLF-QP), or, when integrated with Control Barrier Functions, (CLF-CBF-QP) [33], [34], [35], [36] have been developed to provide low-level control for safety. The techniques can be used in tandem with the proposed CLF to avoid obstacles smoothly. Data-driven planning and control algorithms for safety–critical systems are combining machine-learning [37], [38], model predictive control (MPC), reinforcement learning [39], or belief-space learning [40].

Majumdar and Tedrake [41] proposed the region of attraction for time-varying systems to divide the path into several overlapping regions, similar to funnels [42], in which the system is invariant. To avoid numerical optimization for verifying the overlap of the regions and to allow generating stable trajectory from the funnels in the presence of disturbances, Tiseo et al. [43] presented a framework to combine the region of attraction and the stability properties of a fractal-impedance controller. The region of attraction is a function of a list of waypoints. They fit a force profile to the list of waypoints, so that the desired trajectory is an attractive set for an associated vector field. On the other hand, in our CLF reactive system, given a list of waypoints connecting a starting point and a subgoal, a vector field provided by the proposed CLF—designed specifically for bipedal robots—is utilized to connect two consecutive waypoints. Therefore, the full resulting trajectory composed of several CLF vector fields is easy for bipedal robots to follow. The CLF is also used in the pruning process for the RRT*, also ensuring that it favors trajectories that are the most compatible with the CLF, and hence the motions of the robot.

Uncertainty-aware planning through networked belief-aware perceptual autonomy (NeBula) [44] from the DARPA Subterranean Challenge (DARPA SubT) [45] probabilistically fuses various sensing modalities to allow the robot to create belief-aware local maps. The underlying planner is

stochastic traversability evaluation and planning where they build an uncertainty-aware 2.5D traversability map to solve an online receding horizon MPC problem. Miller et al. [46] presented a quadruped with higher levels of autonomy to explore a tunnel environment in the 2019 DARPA SubT, where a heuristic cost function is designed to penalize sidestepping, whereas we design a CLF to account for it. To achieve autonomous exploration, MARBLE [47] proposes graph and frontier-based path planning algorithms on four-wheeled and tracked ground robots complimented with multirotor platforms in which a reactive centering controller for wheeled robots is implemented to the the system avoids obstacles while navigating the graph. The GBPlanner proposed in [48] builds and uses a real-time topological map during subterranean exploration. An overview of ground robotics systems for underground environments is provided in [49]. The research challenges faced in such robotics exploration missions span the domains of communications, perception, simultaneous localization, and mapping (SLAM), planning, and control.

C. Reactive Planning

Reactive planning contributes another significant concept to the motion planning literature [10], [11], [12], [13], [14], [15], [16], [17], [18], 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 Section II-A 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 [19], [20], [21], 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 nonsmooth paths. The feedback motion planning in [19] 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, LQR, and Sum of Squares (SoS), whereas the feedback motion planning of [20], [21] uses a single CLF and varies the associated equilibrium to set subgoal poses.

The feedback motion planning of [20], [21] is the starting point for the work in this article. 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 Section III-B, the CLF in [20] and [21] is designed for differential-drive wheeled robots which must respect nonholonomic constraints associated with wheels. In Section III-B, 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 Section IV, 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 Section V-B. This allows our Cassie robot to navigate undulating terrain.

We note that sampling-based approaches exist that are more efficient than RRT*, such as bidirectional RRT*, informed RRT* [50], [51], and RRT*-AB [52], [53]. 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*[54], Bi-directional A*[55], or ANA*[56] are efficient on 2-D graphs but not efficient in continuous space.

D. Contributions

In particular, the present work has the following contributions:

1) We propose a novel 2-D smooth CLF with a closed-form solution to the feedback controller for omnidirectional robots. The 2-D 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. In addition, 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 [20], 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 multilayer 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 [57], which simulates 20 degrees of freedom (DoF) of Cassie in MATLAB Simmechanics on a 3-D terrain. 3) Last, the reactive planning system successfully allows Cassie Blue to complete several indoor and 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. 1.

The videos of the autonomy experiments can be found at [58]. 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 [59].

The remainder of this article is organized as follows. Section III constructs the new CLF for bipeds and omnidirectional robots. The omnidirectional CLF-RRT* is introduced in Section IV. Section V integrates all the above components as a reactive planning system. Simulated and experimental evaluations of the proposed reactive system is presented in Section VI. Finally, Section VII concludes this article.

III. Construction of a CLF

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.

A. Lyapunov and CLF

In this article, our goals and pseudogoals are chosen to be equilibrium points of the center of mass (CoM) 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 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 [60] for the formal definition. Consider $\mathcal{O}$ an open set about the origin of $\mathbb{R}^n$, and

$$\dot{x} = f(x, u) \tag{1}$$

a control system with $x \in \mathbb{R}^n$ is system state and $u \in \mathbb{R}^m$ control commands. A differentiable function $V: \mathcal{O} \to [0, \infty)$ is a CLF if (i) $V(x) = 0 \Rightarrow x = 0$ 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 $u = a(x)$ such that $\nabla V(x) f(x, a(x)) < 0$ then renders the origin asymptotically stable.

In [20], the output of their planner is a feedback function $u = a(x)$ rendering a particular CLF negative definite. When the CLF is associated with a goal or pseudogoal 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 article as well.

B. Redesign of CLF Proposed in the Literature

The 2-D CLF planner of [20], [21] has been designed for differentially driven nonholonomically constrained robots, whose dynamics and control laws are inappropriate for bipedal robots. Like most robot models, the work [20] 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 [61], [62], [63], [64], [65], 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 CoM 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 2-D CLF of [20], [21], 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. 2. This oscillation is undesirable as it can affect the robot's balance.

With a Dubins car model as used in [21] and [20], 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.

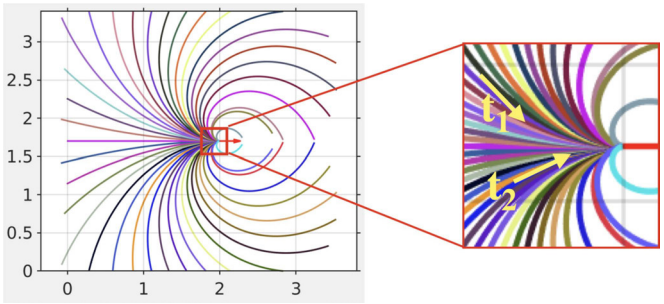


Fig. 2. Plots show paths in 2-D generated by the CLF of [20], [21] for Dubins cars. At each point, the tangent to a path (t_1 and t_2 in the blowup) is the heading angle for the robot. These paths clearly fail to account for a biped's ability to move laterally. Moreover, in practice, an underactuated 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.

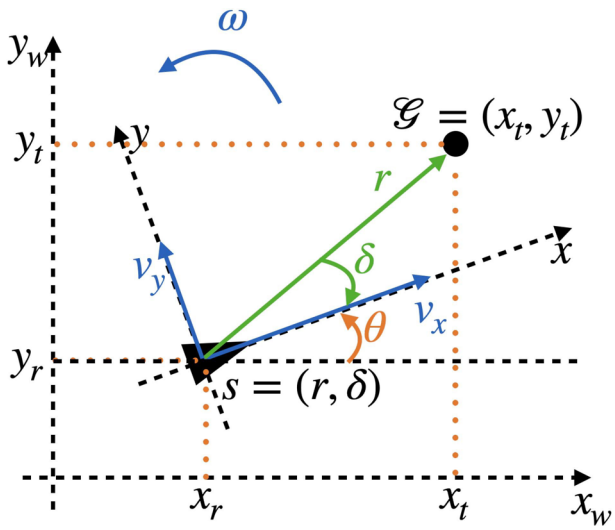


Fig. 3. Illustration of the robot pose-centric polar representation ($s = (r, \delta)$) for robot pose (x_r, y_r, θ) and target position $\mathcal{G} = (x_t, y_t)$. Here, r is the radial distance to the target and δ is the angle between the heading angle θ of the robot and the line of sight from the robot to the goal. The longitudinal velocity, lateral velocity, and angular velocity are v_x, v_y and ω, respectively.

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.

C. State Representation

As mentioned in Section III, 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 $\mathrm{p} = (x_r, y_r, \theta)$ 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*

$$s = \{(r, \delta) \mid r \in \mathbb{R}, \text{ and } \delta \in (-\pi, \pi]\} \tag{2}$$

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. 3.

Remark 1: Park and Kuipers [20], [21] 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.

D. Construction of CLF

The kinematics of an omnidirectional robot is defined as

$$\begin{pmatrix} \dot{r} \\ \dot{\delta} \end{pmatrix} = \begin{pmatrix} -\cos(\delta) & -\sin(\delta) \\ \frac{1}{r}\sin(\delta) & -\frac{1}{r}\cos(\delta) \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix} + \begin{pmatrix} 0 \\ \omega \end{pmatrix}. \tag{3}$$

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

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

is negative definite (and hence invertible) for all $r > 0$, the model (3) is over actuated for $r > 0$.

Remark 2: Observe that $\delta < \delta^1$ when the robot moves along the x-axis, as shown in Fig. 4(a). Therefore, $\frac{v_x}{r}\sin(\delta)$ is positive. Similarly, $\frac{v_y}{r}\cos(\delta)$ is negated because $\delta < \delta^1$ when the robot moves toward the y-axis, as shown in Fig. 4(b).

We next note that the change of control variables

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

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

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

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

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} := \begin{pmatrix} \cos(\delta) & r\sin(\delta) \\ \sin(\delta) & -r\cos(\delta) \end{pmatrix} \begin{pmatrix} v_r \\ v_\delta - \omega \end{pmatrix}. \tag{4}$$

As mentioned in Section III, lateral walking is more expensive than longitudinal walking because movement in the lateral hip joint is limited. A candidate CLF[1] $\mathcal{L}$, in terms of the robot's current pose and target (end) position, is defined as

$$\mathcal{L} = \frac{r^2 + \gamma^2 \sin^2(\beta\delta)}{2} \tag{5}$$

where γ is a weight on the orientation and the role of $\beta > 0$ will be described later. We next check that $\mathcal{L}$ is a CLF. The derivative

[1]In polar coordinate, the function $\mathcal{L}$ is positive definite in the sense that $\mathcal{L} = 0 \Rightarrow r = 0$, and when $r = 0$, the angle δ is arbitrary or undefined.

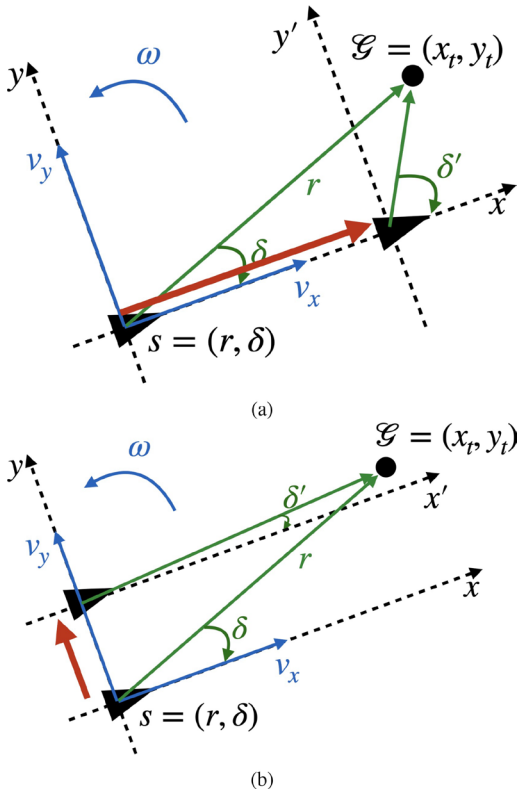


Fig. 4. This figure explains the signs in (3). On the top, δ increases when the robot moves parallel to the x-axis. Therefore, $\frac{v_x}{r}\sin\delta$ is positive. Similarly, $\frac{v_y}{r}\cos\delta$ is negated because δ decreases when the robot moves toward the y-axis.

of $\mathcal{L}$ is

$$\dot{\mathcal{L}} = \dot{r} + \frac{\beta\gamma^2}{2}\sin(2\beta\delta)\dot{\delta}$$
$$= r(-v_r) + \frac{\beta\gamma^2}{2}\sin(2\beta\delta)v_\delta. \quad (6)$$

The feedback

$$v_r = k_{r1}\frac{r}{k_{r2}+r}$$
$$v_\delta = -\frac{2}{\beta}k_{\delta 1}\frac{r}{k_{\delta 2}+r}\sin(2\beta\delta) \quad (7)$$

results in

$$\dot{\mathcal{L}} = -\frac{k_{r1}}{k_{r2}+r}r^2 - k_{\delta 1}\gamma^2\frac{r}{k_{\delta 2}+r}\sin^2(2\beta\delta) \quad (8)$$

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 2-D. Later, the cost function for the RRT*-based planner will be 3-D; see Section IV-B. Work in [62], [66], [67] shows how to adapt the local model to the terrain in such a way that the model (7) is always valid and hence the 2-D CLF is applicable.

Remark 3: From (7), it follows that $\dot{\delta} = v_\delta = 0$ for $2\beta\delta \in \{0, \pm\pi\}$. Therefore, the manifolds

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

are invariant for the closed-loop system. From (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. 6. In practice, a 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 (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.

E. Closed-Form Solution

Taking (4) as a constraint, we propose to select ω so as to keep v_y small (limit lateral walking) by optimizing

$$J = \min_{v_y,\omega} (v_y)^2 + a\omega^2. \quad (9)$$

The parameter $a > 0$ allows us to penalize aggressive yaw motions ω, as will be illustrated in Section III-F. Plugging in the constraint (4), (9) leads to

$$J = \min_{\omega} \{[\sin(\delta)v_r - r\cos(\delta)(v_\delta - \omega)]^2 + a\omega\}$$
$$= \min_{\omega} \{(\sin(\delta)v_r)^2 + [r\cos(\delta)(v_\delta - \omega)]$$
$$- 2\sin(\delta)v_r(r\cos(\delta)(v_\delta - \omega)) + a\omega^2\}.$$

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

$$\omega^* = \arg\min_{\omega} \{r^2\cos(\delta)(v_\delta - \omega)^2$$
$$+ 2rv_r\sin(\delta)\cos(\delta)\omega + a\omega^2\}$$

which implies that

$$\left(a + r^2\cos^2(\delta)\right)\omega^* + r\cos(\delta)[v_r\sin(\delta) - rv_\delta\cos(\delta)] = 0. \quad (10)$$

The final result is

$$\omega^* = \frac{r\cos(\delta)[rv_\delta\cos(\delta) - v_r\sin(\delta)]}{a + r^2\cos^2(\delta)} \quad (11)$$

and then

$$v_y^* = \frac{a(v_r\sin(\delta) - rv_\delta\cos(\delta))}{r^2\cos(\delta)^2 + a}$$
$$v_x^* = \frac{v_r\cos(\delta)r^2 + av_\delta\sin(\delta)r + av_r\cos(\delta)}{r^2\cos(\delta)^2 + a}. \quad (12)$$

F. Qualitative Analysis of the Closed-Loop Trajectories

The default parameters applied in this analysis are shown in Table I. Fig. 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

TABLE I
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

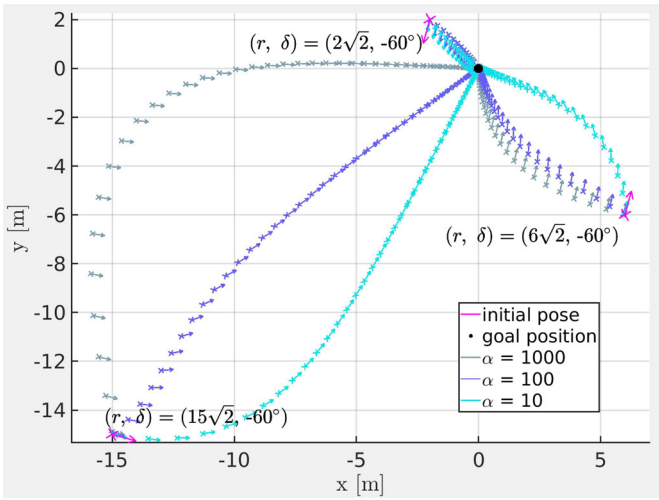


Fig. 5. Distance to the target and the penalty on yaw motion in (9) both affect closed-loop behavior arising from the CLF. The arrows indicate the robot's absolute heading. In each solution of the closed-loop system, the robot's heading relative to the target is initialized at $-60°$. When the robot 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.

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.

Fig. 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 I, 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.

IV. OMNIDIRECTIONAL CLF-RRT*

This section integrates the CLF proposed in Section III into the original RRT* algorithm. The resulting omnidirectional CLF RRT* provides feasible paths for (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.

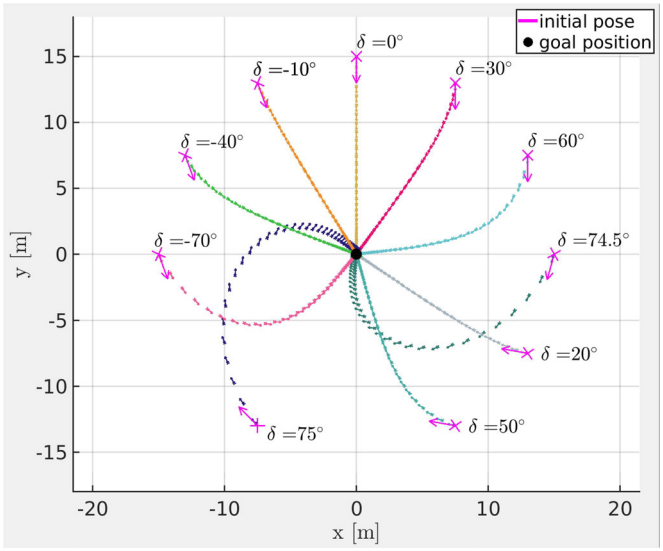


Fig. 6. This figure illustrated how the closed-loop trajectories generated by the CLF in (5) vary as a function the initial relative heading to the target, when starting at a fixed distance of $r = 15$ m, and $\alpha = 10$. The arrows indicate the robot's heading. As shown in Table I, we are using $\beta = 1.2$, 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. 5.

A. Standard RRT* Algorithm

The original RRT* [3] 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. In addition, 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.

B. 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 (5), which takes relative heading into account. Second, the steering/extending functions use the closed-loop trajectories generated by (12) to define paths between nodes. Third, because the cost (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 [20], [21]. Finally, when connecting, exploring, and rewiring the tree, additional terms are added to the cost (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 [4]. Let $X = \{(x, y, \theta) | x, y \in \mathbb{R}$ and $\theta \in (-\pi, \pi]\}$ be the configuration space and let X_{obs} denote the obstacle region, which together define the free region for walking $X_{free} = X_{obs}$. The omnidirectional CLF RRT* solves the optimal path planning problem by growing a tree $T = (V, E)$, where $V \in X_{free}$ is a vertex set of poses connected by edges E of feasible path segments. Briefly speaking,

Algorithm 1: $\mathsf{T} = (V, E) \leftarrow$ Omnidirectional CLF RRT*.

$\mathcal{T} \leftarrow$ InitializeTree();
$\mathcal{T} \leftarrow$ InsertNode($\emptyset, n_{\text{init}}, \mathcal{T}$);
for $i = 1$ **to** N **do**
  $n_{\text{rand}} \leftarrow$ Sample(i)
  $n_{\text{nearest}} \leftarrow$ Nearest($\mathcal{T}, n_{\text{rand}}$)
  $(n_{\text{new}}, \mathscr{T}') \leftarrow$ Extend($n_{\text{nearest}}, n_{\text{rand}}, \kappa$)
  if *ObstacleFree*($\mathscr{T}'$) **then**
    $\mathcal{N}_T \leftarrow$ NearTo($\mathcal{T}, n_{\text{new}}, |V|$)
    $n_{\min} \leftarrow$ ChooseParent($\mathcal{N}_T, n_{\text{nearest}}, n_{\text{new}}$)
    $\mathcal{T} \leftarrow$ InsertNode($n_{\min}, n_{\text{new}}, \mathcal{T}$)
    $\mathcal{N}_F \leftarrow$ NearFrom($\mathcal{T}, n_{\text{new}}, |V|$)
    $\mathcal{T} \leftarrow$ ReWire($\mathcal{T}, \mathcal{N}_F, n_{\min}, n_{\text{new}}$)
return $\mathcal{T}$

Algorithm 2: $n_{\text{parent}} \leftarrow$ ChooseParent($\mathsf{N}_T, n_{\text{nearest}}, n_{\text{new}}$).

$n_{\text{parent}} \leftarrow n_{\text{nearest}}$
$c_{\text{parent}} \leftarrow \text{Cost}(n_{\text{nearest}}) + \text{c}(n_{\text{nearest}}, n_{\text{new}})$
for $n_{near} \in \mathcal{N}_T$ **do**
  $\mathscr{T}' \leftarrow$Steer($n_{\text{near}}, n_{\text{new}}$)
  if *ObstacleFree*($\mathscr{T}'$) **then**
    $c' = \text{Cost}(n_{\text{near}}) + c(n_{\text{near}}, n_{\text{new}})$
    if $c' < Cost(n_{new})$ *and* $c' < c_{parent}$ **then**
      $n_{\text{parent}} \leftarrow n_{\text{near}}$
      $c_{\text{parent}} \leftarrow c'$
return n_{parent}

the proposed RRT* (see Algorithm 1) explores the configuration space by random sampling and extending nodes to grow the tree (explore the configuration space), just as in the classic RRT [1]. Considering nearby nodes of a sample to choose the best parent node and rewiring the graph guarantee asymptotic optimality (see Algorithm 2 and 3), as with the classic algorithm. As emphasized previously, a key difference lies in how the paths between vertices are generated.

1) Sampling: This step randomly samples a pose $n_{\text{rand}} = (x, y, \theta) \in \mathsf{X}_{\text{free}}$. 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 subgoal and a degree of goal biasing, we bias samples to be from a Gaussian distribution centered about the subgoal; see Section V-B. Furthermore, we limit the sampling space to a sector in front of the robot.

2) Distance: To account for the asymmetry in lateral versus longitudinal motions, as discussed in Section III-B, the distance $d(n_i, n_k)$ from node n_i to node n_k in the tree T is defined by (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.

Remark 4: As mentioned in Section III-D, 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:

$$d(n_i, n_k) = \mathfrak{L} + k_\delta \max(|\delta| - |U|, 0) \tag{13}$$

where $\mathfrak{L}$ is defined in (5), k_δ is a positive constant, and U corresponds to a repulsive point (i.e., $\pm\frac{\pi}{2}$) in Remark 3.

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

$$\mathbb{T}(\mathsf{P}, \boldsymbol{\mathscr{T}}) = \sum_{\forall x_t, y_t \in \mathscr{T}} C(x_t, y_t, z_r) \tag{14}$$

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 [68], [69], [70].

Remark 5: The planning system is designed so that any traversability index [71], [72] can be leveraged in the cost function on the local map to indicate the relative ease, difficulty, or safety of traversing a section of terrain. Therefore, the system can be readily adapted to different types of terrain.

4) Cost Between Nodes: Let $c(n_i, n_k)$ be the cost from n_i to n_k in the tree T, defined as

$$c(n_i, n_k) = d(n_i, n_k) + k_t \mathbb{T}(\mathsf{P}, \boldsymbol{\mathscr{T}}) \tag{15}$$

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 this article, $k_t = 1$.

5) Nearby Nodes: Due to the use of the CLF function, the distinction between near-to nodes N_T and near-from nodes N_F is necessary.

$$\mathsf{N}_T(n_i, \mathsf{T}, \mathsf{M}, m) := \{n \in V \mid d(n, n_i) \le L(m) \;\&\; |\mathbb{T}(n, \mathsf{P}) - \mathbb{T}(n_i, \mathsf{P})| \le T_k\} \tag{16}$$

where $|\cdot|$ is the absolute value, m is the number of nodes in the tree T, and $L(m) = \eta(\log(n)/n)^{(1/\xi)}$ with the constant η and dimension of space ξ (3 in our case) [6] and T_k is a positive constant. Similarly, the near-from nodes N_F are determined by

$$\mathsf{N}_F(n_i, \mathsf{T}, \mathsf{M}, m) := \{n \in V \mid d(n_i, n) \le L(m) \;\&\; |\mathbb{T}(n, \mathsf{P}) - \mathbb{T}(n_i, \mathsf{P})| \le T_k\}. \tag{17}$$

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

7) Steering and Extending: The steering function generates a path segment $\boldsymbol{\mathscr{T}}$ 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.

8) Parent Choosing and Graph Rewiring: Choosing the best parent node (see Algorithm 2) and rewiring the graph (see

[2]The path is generated from the CLF in Section III.

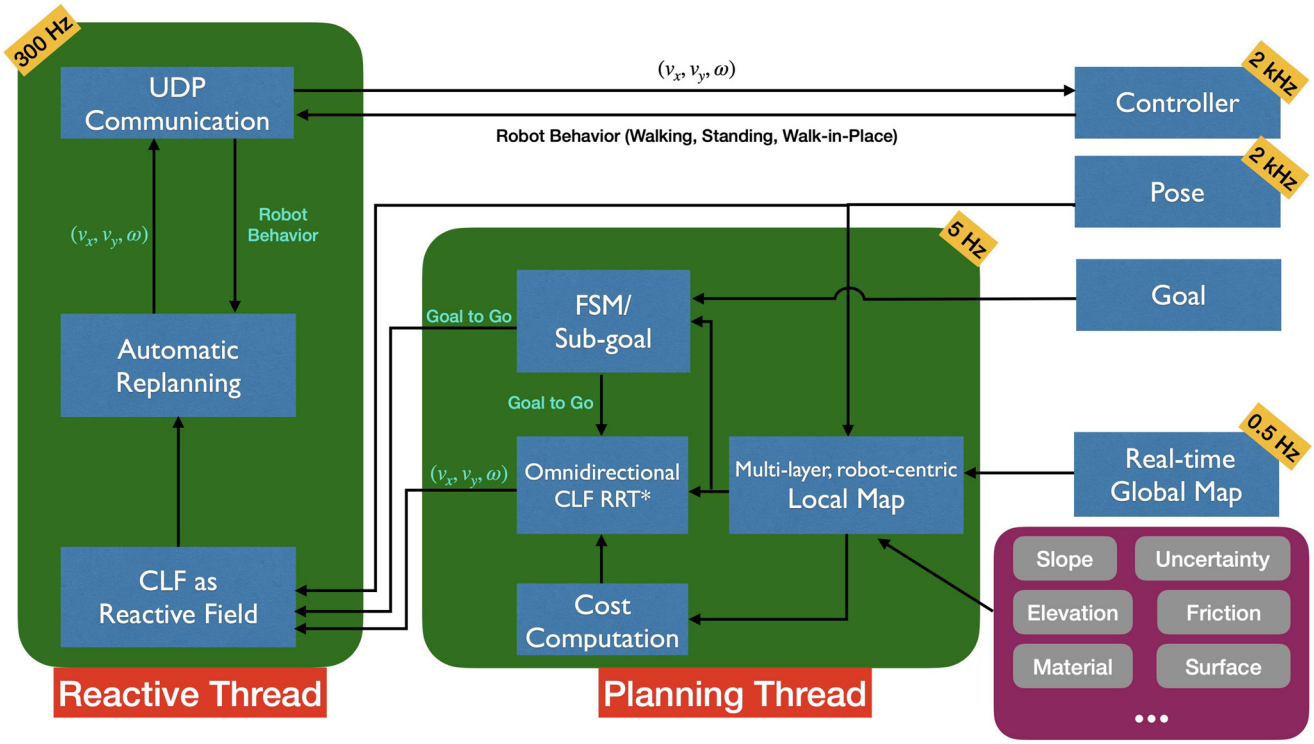


Fig. 7. This figure summarizes the proposed reactive planning system. The planning thread is built around RRT* and an omnidirectional CLF that is used to assign distances, define locally optimal path segments, search radius, and linking conditions for rewiring and choosing a parent. In addition, the planning thread contains a multilayer, robot-centric local map for computing traversability, a subgoal finder, and a FSM to choose subgoal locations guiding the robot to a distant goal. The terrain information extracted from the multilayer local map can be shared with a terrain-aware controller, such as [66]. Instead of a common waypoint-following or path-tracking strategy, the reactive thread copes with robot deviation while eliminating nonsmooth 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.

Algorithm 3: $\mathcal{T} \leftarrow \text{ReWire}(\mathcal{T}, \mathcal{N}_F, n_{\min}, n_{\text{new}})$.

```
for n_near ∈ N_F \ {n_min} do
    T' ← Steer(n_new, n_near)
    if ObstacleFree(T') and
       Cost(n_new) + c(n_new, n_near) < Cost(n_near) then
        T ← Re-Connect(n_new, n_near, T)
return T
```

Algorithm 3) 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 $n \in \mathcal{N}_T$ with smallest cost from the root to the node

$$n_{\text{parent}} = \arg\min_{n_{\text{near}} \in \mathcal{N}_T} \text{Cost}(n_{\text{near}}) + c(n_{\text{near}}, n_{\text{new}}). \quad (18)$$

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) Collision Check: This step verifies whether a path σ 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.

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.

V. 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 [10], [11], [12], [13], [14], [15], [16], [17], [18], [19]. A version based on the work of [20], [21] will be incorporated into our overall planning system. In addition, we take into account features in a local map.

A. 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 [19], [20], [21], we populate the configuration space with a discrete set of feedback control

laws that steer the robot from local chart about a subgoal to the subgoal itself. The collision free property is handled by the low-frequency planner at the current time. Others have used CBFs for this purpose [33], [34], [35], [36], [73], [74]. A 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 multithreading in C++ based on the ROS library [75]. One thread is for the planning and the other is for the reactive thread, as illustrated in Fig. 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 [68], [69], [70] based on collected LiDAR and/or camera data.

B. Planning Thread

The planning thread deals with short-range planning (less than 20 meters) at a frequency of 5–10 Hz. It includes a robot-centric local map, our omnidirectional CLF-RRT* algorithm of Section III, cost computation, a subgoal finder, and a FSM.

1) Robot-Centric Local Map and Cost Computation: Fig. 8(a) shows the robot-centric multilayer local map (highlighted area), which crops a submap 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 [76] to remove walkable area behind glass walls can be computed in this step, see Section VI-E. In addition, terrain information such as slopes, frictions, or staircase detection can be sent to a terrain-aware low-level controller [66]. The computations with the local map are efficient compared to processing the full map.

Remark 6: 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 [76].

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 (5) turns into real-time feedback laws for anytime replanning.

3) Subgoal Finder and FSM: Ideally, a global planner [77], [78], [79] is present to guide the robot to a distant goal, which may not be viewable at the time of mission start [78]. In relatively simple situations such as that shown in Figs. 8 and 9, it is sufficient to complete many of short-term missions by positioning a subgoal (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 subgoal finder is also used as a FSM to handle high-level missions such as making turn selections at intersections, or determining if there is an intersection; see Section VI-E. It is emphasized that the final path is connected by several subgoals determined locally by the subgoal finder. In the future, the sub-goal finder will be replaced with a global planner to achieve globally optimal paths.

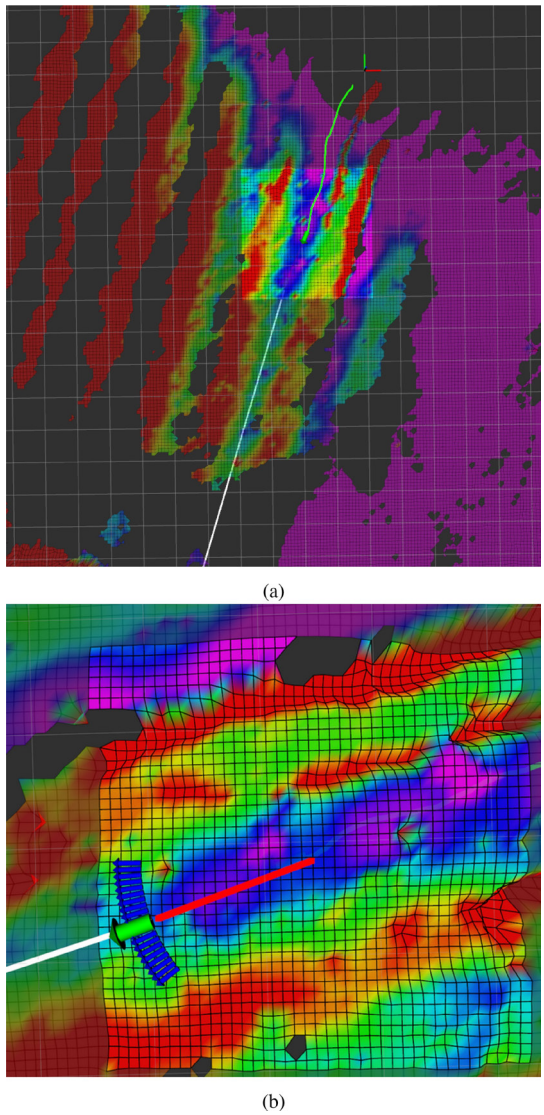


Fig. 8. Elevation map (colored by height) was built online while Cassie was autonomously traversing the Wave Field on the North Campus of the University of Michigan. The highlighted area is the smoothed, robot-centric local map. The blue arc and the green arrow (pointing from the red line to the white line) are the subgoal finder and the chosen sub-goal for the omnidirectional CLF RRT*, respectively. The red shows the locally optimal path.

Remark 7: A subgoal 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 subgoal is always observable because it is within the local map. Subgoals 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

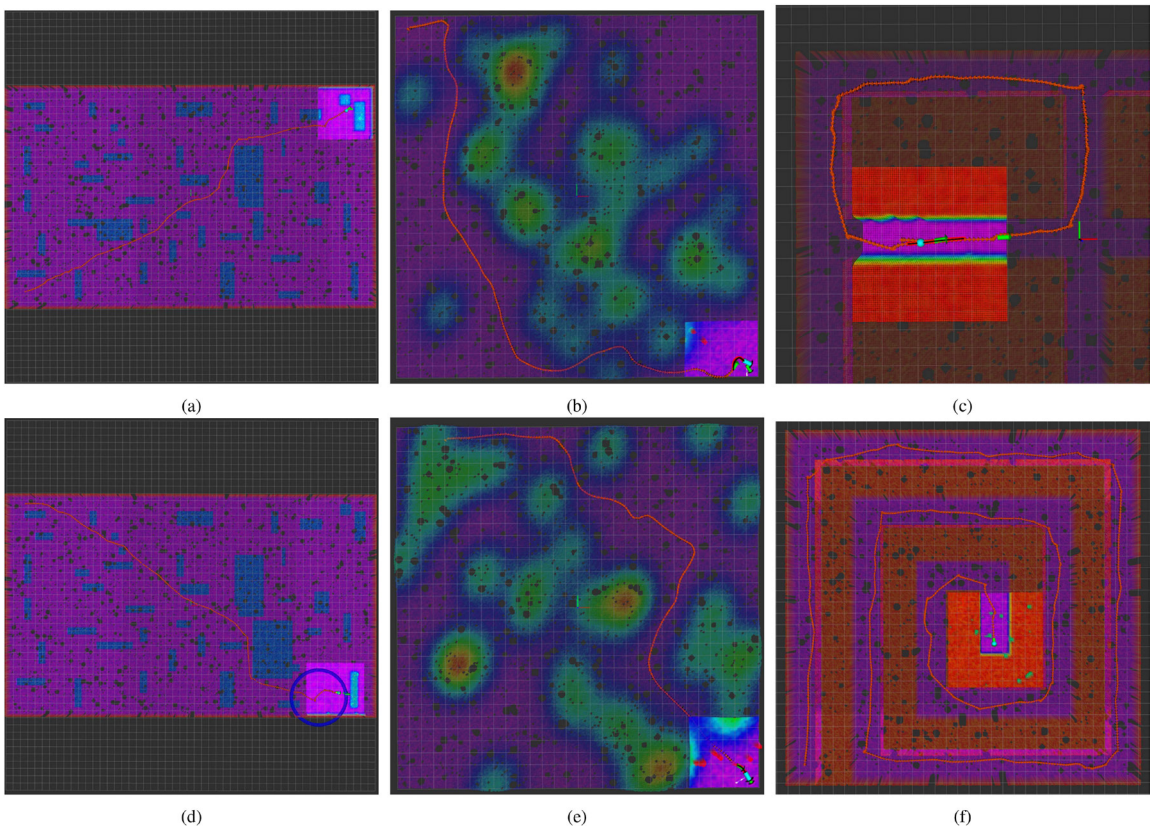


Fig. 9. Simulated scenes and results obtained with the proposed reactive planning system. On the left are cluttered indoor scenes with obstacles and holes, in the middle are noisy undulating outdoor terrains, and on the right, are high-level missions. Each grid in a map is $1_{\times}1$ meter. The simulated 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 $8_{\times}$ 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 [59]. All the figures are vector graphics, so one can enlarge in the browser for best viewing.

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 subgoals 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 [77], [78], [79].

Remark 8: 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 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 Section VI-F for more details.

C. Reactive Thread

The work in [10], [11], [12], [13], [14], [15], [16], [17], [18] 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 replanner.

In the reactive planner of [11], [13], 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 (5) to define alternative paths in the configuration space. In essence, our feedback functions (12) and (11) provide instantaneous replanning of the control commands for the omnidirectional model (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 CLF described in Section III 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 9: 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.

VI. 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. 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 [61], [62] 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; see our GitHub [59] for videos and more results.

Remark 10: The ALIP robot [61], [62] takes piece-wise constant inputs from the reactive planning system. Let g, H, τ be the gravity, the robot's CoM height, and the time interval of a swing phase, respectively. The motion of an ALIP robot on the x-axis is defined as

$$\begin{pmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{pmatrix} = \begin{pmatrix} \cosh(\xi) & \frac{1}{\rho}\sinh(\xi) \\ \rho\sinh(\xi) & \cosh(\xi) \end{pmatrix} \begin{pmatrix} x_k \\ \dot{x}_k \end{pmatrix} + \begin{pmatrix} 1-\cosh(\xi) \\ -\rho\sinh(\xi) \end{pmatrix} p_x \tag{19}$$

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 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 11: 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.

B. 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 User Datagram Protocol (UDP) from ROS [75] 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 via UDP. The planner system successfully took the simulated Cassie to the goal without falling, as shown in Fig. 10.

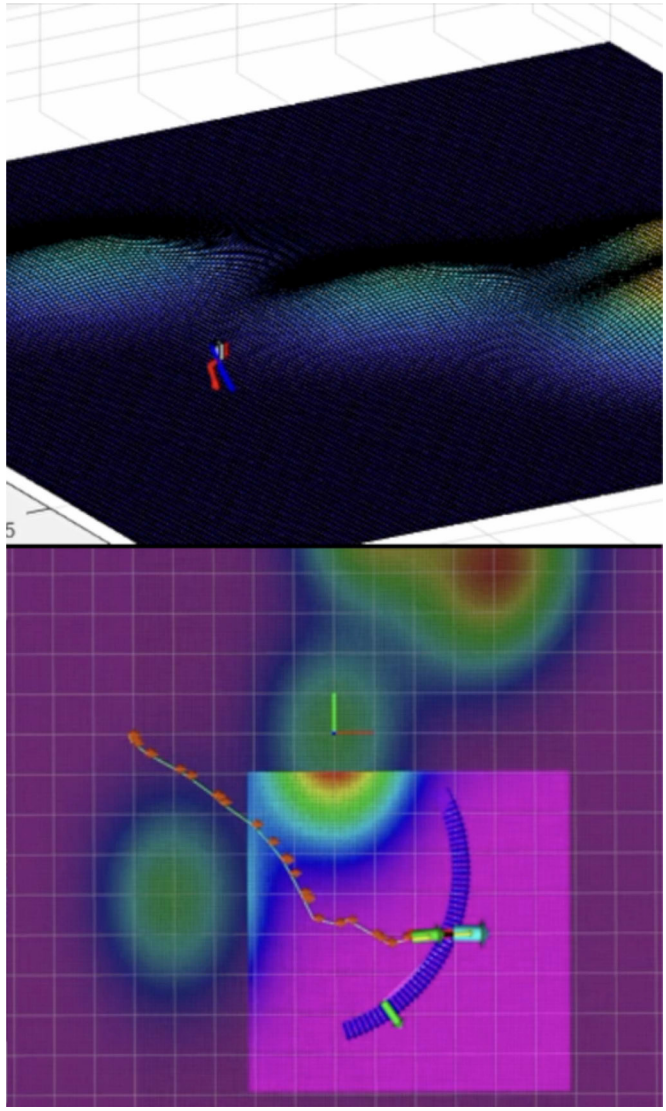
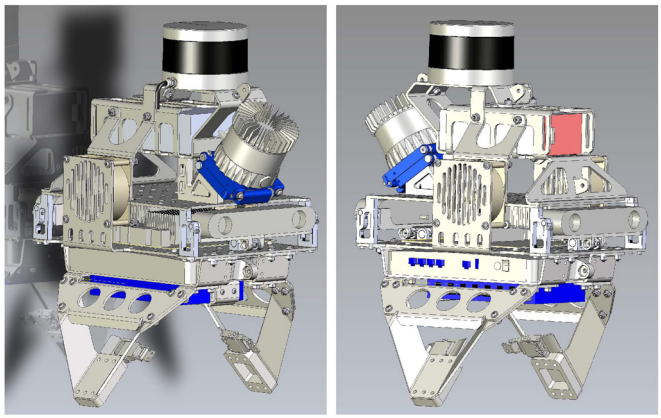


Fig. 10. Simulation of a C++-implementation of the reactive planner on full-dynamic model of Cassie, which accounts for all 20 degrees of freedom of the robot in Matlab-SimMechanics and includes a 3-D terrain model. The reactive planning system receives the pose of the simulated Cassie via User Datagram 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 [59].

C. Perception Suite Design and Hardware System Integration

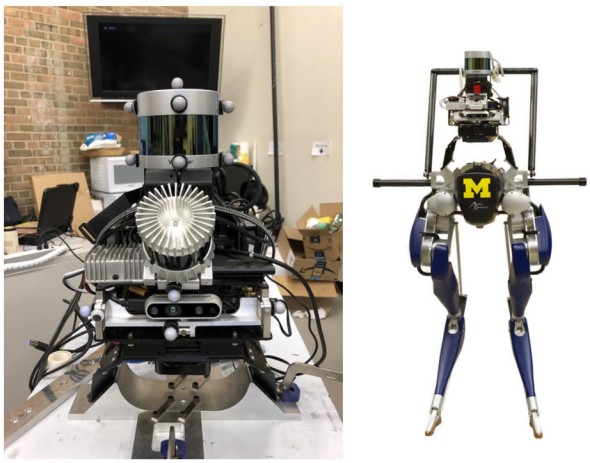
To allow the robot to perceive its surroundings under different lighting conditions and environments, we designed a perception suite that consists of an RGB-D camera (Intel RealSense™ D435) and a *32-Beam Velodyne ULTRA Puck LiDAR*. Two fans cool a Jetson AGX Xavier with Graphics Processing Unit (GPU). 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. Fig. 11 shows the design of the full sensor suite and the step files are available at [80].

The weight of the sensor suite, with batteries and everything included, is 8.5 Kg. We use an industrial-grade router and internet switch to ensure stable connections among the sensors,



(a) (b)

Fig. 11. Computer-aided design (CAD) of the sensor suite. The left shows the front view of the sensor suite and the right shows the back.



(a) (b)

Fig. 12. Left shows the sensor suite with different sensors, and the right shows the sensor suite mounted on Cassie Blue.

GPU, and the secondary computer on the Cassie robot. Fig. 12 shows all the sensors mounted on the perception suite.

D. Software System Integration for Real-Time Deployment

System integration is critical for real-time use. Fig. 13 shows the integrated system, distribution, and frequency of each computation. In particular, the sensor calibrations are performed via [81], [82], [83], [84], [85], [86], [87], [88]. The invariant Extended Kalman Filter (InEKF) [89], [90] is used to estimate the state of Cassie Blue at 2 kHz. Images are segmented via MobileNets [91] and a LiDAR point cloud is projected back to the segmented image to produce a 3-D segmented point cloud. The resulting point clouds are then utilized to build a multilayer map (MLM) [68], [69], [70], [92], [93]. The reactive planning system then crops the MLM around the robot position to create a local map and performs several operations to acquire extra information, as described in Section V-B. In addition, 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 subposes via the proposed CLF; see Section III and Section V-C. The control commands are then sent to Cassie Blue's gait controller [61], [94], [95] via UDP.

E. Full Autonomy Experiments With Cassie Blue

We conducted several indoor and outdoor full autonomy experiments with Cassie Blue. The running cost in (14) was selected as

$$C(x_t, y_t, z_r) := C_e(x_t, y_t) + 0.5C_s(x_t, y_t) + 0.3(C_e(x_t, y_t) - z_r) \tag{20}$$

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.

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 [96]; see Fig. 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. Fig. 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. 14(c). The planning system navigated Cassie Blue around a hump that protrudes into one of the valleys, as shown in Fig. 14(d). Fig. 15 shows the control commands sent to Cassie Blue. This experiment was presented in the Legged Robots Workshop at ICRA 2021; the video can be viewed at [97]. The video of the Wave Field experiment is uploaded and can be found at [58] and [59].

Remark 12: 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 [85] for more details about LiDAR properties.

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. 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] [99], 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

[3]We chose this clustering algorithm because the number of clusters is unknown. Therefore, algorithms like K-Means Clustering [98] cannot be used.

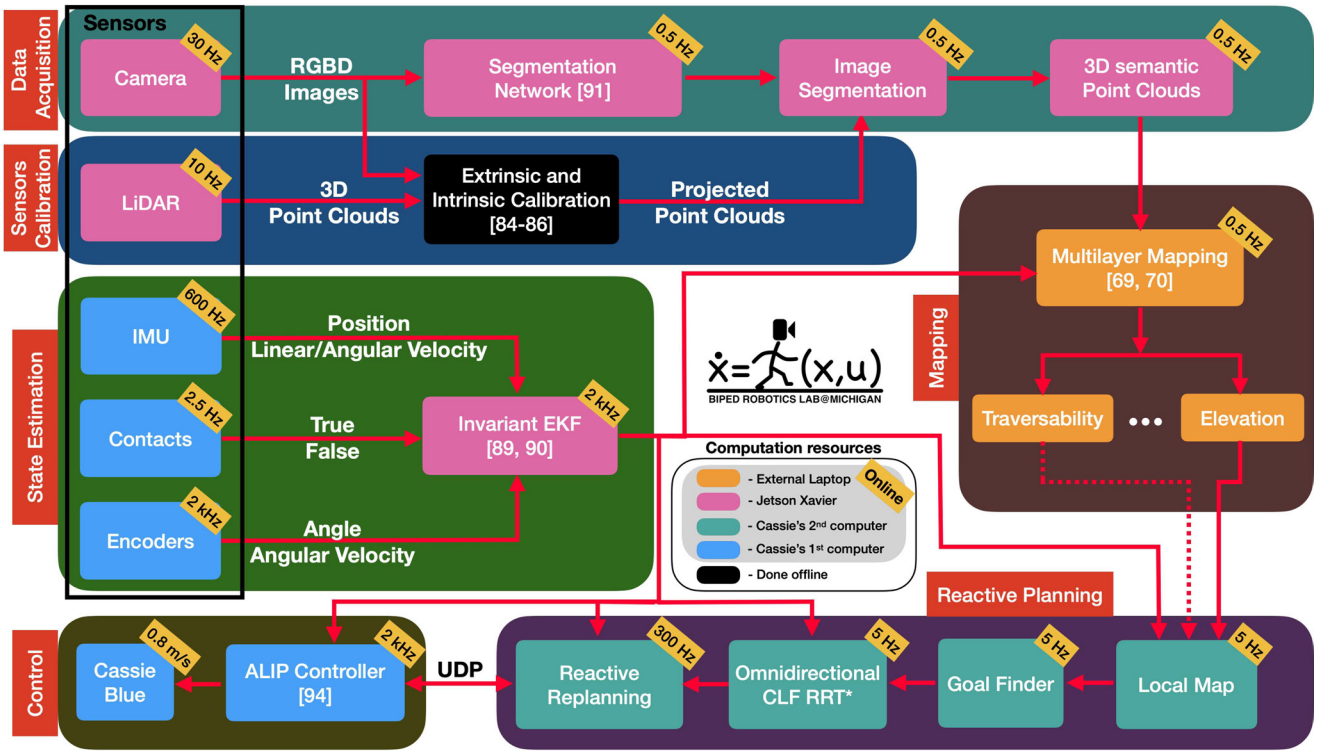


Fig. 13. Illustration of how the various processes in the overall autonomy system are distributed and their computation frequencies. The larger boxes indicate various modules such as data acquisition, planning, and control. The smaller boxes are colored according to the processor that runs them.

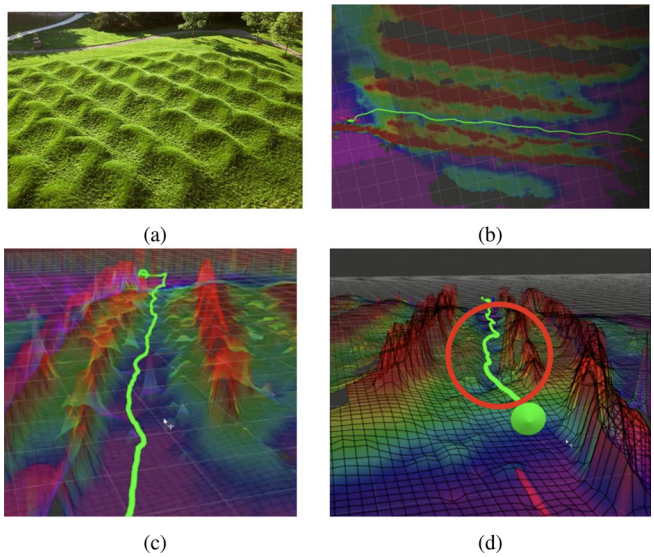


Fig. 14. Experimental results on the Wave Field. The top-left shows the experiment terrain, the Wave Field, on the North Campus of the University of Michigan. The top-right shows a bird's-eye view of the resulting trajectory from the reactive planning system. The bottom-left shows a back-view of the 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.

experiments without falling or colliding with obstacles. The total distance traveled was about 80 m. The experiment videos can be viewed at [100] and [59].

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 [76] 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 Section V-B. The proposed reactive planning system successfully guided Cassie Blue back to its initial position, as shown in Fig. 17. The total distance traveled was about 200 m. The experiment videos can be viewed at [101] and [59].

F. Experiment Discussion

In the two indoor experiments, Cassie exhibited a walk-and-stop motion. Where does it come from? As mentioned in Section V, the planning threading runs at 5 Hz. At the kth update, there will be an optimal path $\mathbf{P}_k$, 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 reset every 200 ms, and thus the robot was never allowed to evolve along the integral curves of the vector field.

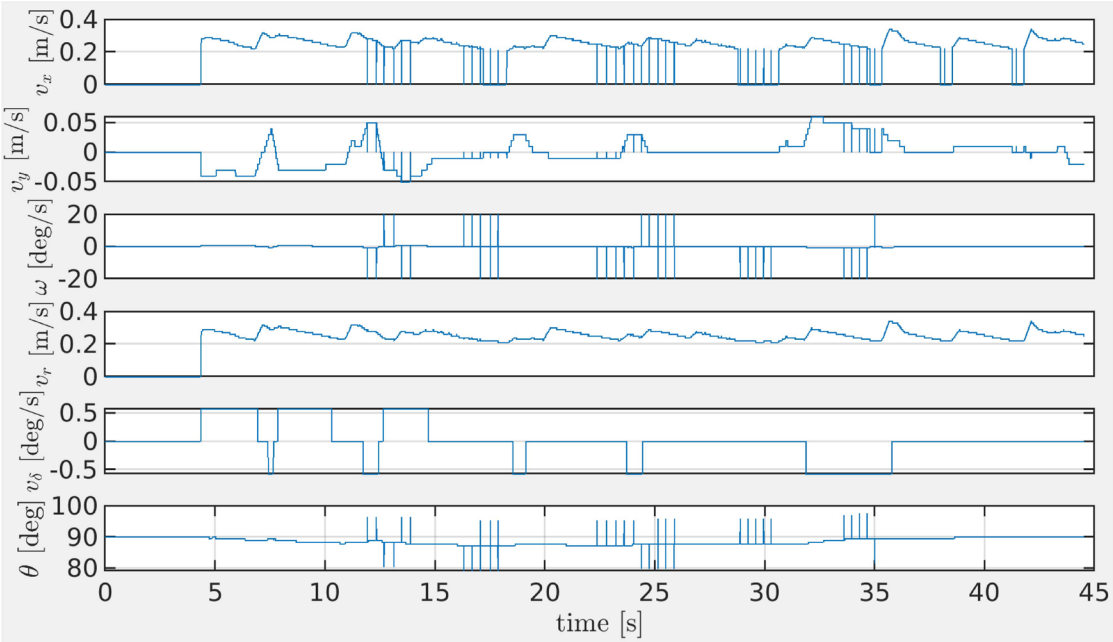


Fig. 15. Control commands sent to Cassie Blue. UDP packet drops show up as vertical lines. When these occur, the controller uses the previous value.

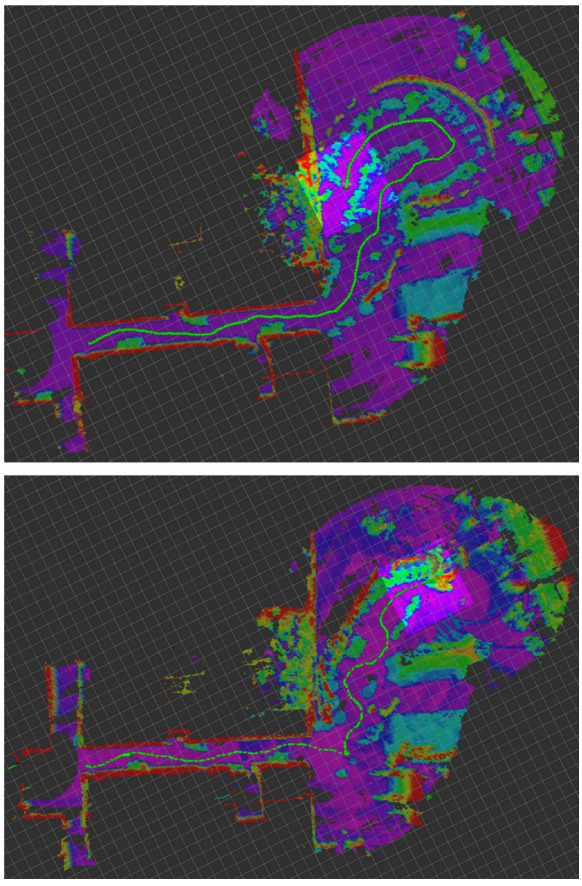


Fig. 16. Resulting trajectories on the first floor of the FRB3. The map colored by height was built online while Cassie was guided by the planning system. The green lines are the resulting trajectories and green patches in the map are tables and furniture considered obstacles.

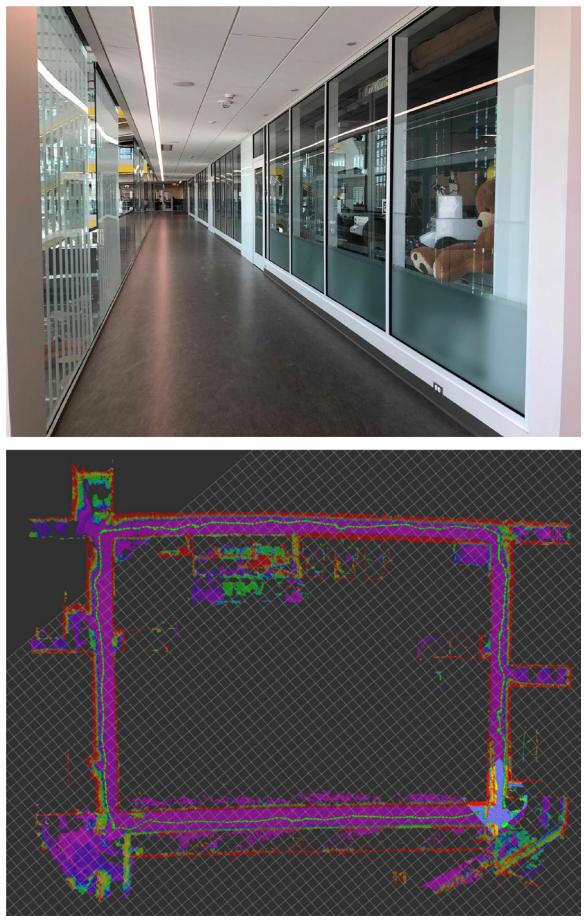


Fig. 17. Experimental results on the second floor of the FRB. The top shows glass walls, which lead to refection of LiDAR lasers and creating walkable area behind the wall. The bottom illustrates the resulting (200 m) trajectory produced by the planning system as Cassie Blue walks. It is remarked that the plot is based on pure odometry so no loop closure is performed [89].

The solution is straightforward: at the $(k+1)$-st planning update, we leave the first unreached way-pose fixed in the path $\mathbf{P}_k$ to ensure continuity. In addition, to fully utilize the optimal path from the previous update, we keep the current optimal path $\mathbf{P}_k$ as a branch and prune all the samples from the kth update. This provides a warm start for the $(k+1)$-st update, as long as the path $\mathbf{P}_k$ 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 [102] and [59].

VII. Conclusion

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 10 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 multilayer, robot-centric local map to compute traversability for challenging terrains, a subgoal finder, and a FSM to choose a subgoal 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 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 CBF [33], [34], [35], [36], [73], [74] with the CLF in the reactive thread to handle dynamic obstacles. In addition, the current local map is a 2.5D, multilayer 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 3-D is another interesting area for future research.

Acknowledgment

This article solely reflects the opinions and conclusions of its authors and not the funding entities. The authors would like to thank Lu Gan and Ray Zhang for their assistance in the development of the autonomy package used on Cassie Blue in these experiments and Yukai Gong and Dianhao Chen for the low-level gait controller used in Cassie. They would also like to thank Dianhao Chen, Jinze Liu, Jenny Tan, Dongmyeong Lee, Jianyang Tang, and Peter Wrobel, Minzhe Li, Lu Gan, Ray Zhang, Yukai Gong, and Oluwami Dosunmu-Ogunbi for their assistance in the experiments. The first author thanks to Jong Jin Park, Collin Johnson, Peter Gaskell, and Prof. Benjamin Kuipers for kindly providing insightful discussion for their work. The first author thanks Wonhui Kim for useful conversations.

References

[1] S. M. LaValle et al., "Rapidly-exploring random trees: A new tool for path planning," 1998.
[2] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, 2001.
[3] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.
[4] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 1478–1483.
[5] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robot. Sci. Syst. VI*, vol. 104, no. 2, 2010. [Online]. Available: http://www.roboticsproceedings.org/rss06/p34.pdf
[6] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *Proc. IEEE Conf. Decis. Control*, 2010, pp. 7681–7687.
[7] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "PQ-RRT*: An improved path planning algorithm for mobile robots," *Expert Syst. Appl.*, vol. 152, 2020, Art. no. 113425.
[8] L. Palmieri, S. Koenig, and K. O. Arras, "RRT-based nonholonomic motion planning using any-angle path biasing," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 2775–2781.
[9] J. Wang, M. Q.-H. Meng, and O. Khatib, "EB-RRT: Optimal motion planning for mobile robots," *IEEE Trans. Automat. Sci. Eng.*, vol. 17, no. 4, pp. 2063–2073, Oct. 2020.
[10] F. Golbol, M. M. Ankarali, and A. Saranli, "Rg-trees: Trajectory-free feedback motion planning using sparse random reference governor trees," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 6506–6511.
[11] O. Arslan and D. E. Koditschek, "Sensor-based reactive navigation in unknown convex sphere worlds," *Int. J. Robot. Res.*, vol. 38, no. 2–3, pp. 196–223, 2019.
[12] S. Paternain, D. E. Koditschek, and A. Ribeiro, "Navigation functions for convex potentials in a space with convex obstacles," *IEEE Trans. Autom. Control*, vol. 63, no. 9, pp. 2944–2959, Sep. 2017.
[13] O. Arslan and D. E. Koditschek, "Exact robot navigation using power diagrams," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1–8.
[14] D. E. Koditschek and E. Rimon, "Robot navigation functions on manifolds with boundary," *Adv. Appl. Math.*, vol. 11, no. 4, pp. 412–442, 1990.
[15] E. Rimon, "Exact robot navigation using artificial potential functions," Ph.D. dissertation, Yale Univ., New Haven, CT, USA, 1990.
[16] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, no. 5, pp. 1179–1187, Sep./Oct. 1989.
[17] D. Koditschek, "Exact robot navigation by means of potential functions: Some topological considerations," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1987, vol. 4, pp. 1–6.
[18] J. V. Gómez, A. Lumbier, S. Garrido, and L. Moreno, "Planning robot formations with fast marching square including uncertainty conditions," *Robot. Auton. Syst.*, vol. 61, no. 2, pp. 137–152, 2013.
[19] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *Int. J. Robot. Res.*, vol. 29, no. 8, pp. 1038–1052, 2010.
[20] J. J. Park and B. Kuipers, "A smooth control law for graceful motion of differential wheeled mobile robots in 2D environment," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 4896–4902.
[21] J. J. Park and B. Kuipers, "Feedback motion planning via non-holonomic RRT* for mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*., 2015, pp. 4035–4040.

[22] 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," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 883–921, 2015.

[23] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *Int. J. Robot. Res.*, vol. 35, no. 7, pp. 797–822, 2016.

[24] G. Vailland, V. Gouranton, and M. Babel, "Cubic bézier local path planner for non-holonomic feasible and comfortable path generation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 7894–7900.

[25] C. Lau and K. Byl, "Smooth RRT-connect: An extension of RRT-connect for practical use in robots," in *Proc. IEEE Int. Conf. Technol. Practical Robot Appl.*, 2015, pp. 1–7.

[26] W. G. Aguilar, S. Morales, H. Ruiz, and V. Abad, "RRT* GL based optimal path planning for real-time navigation of UAVs," in *Int. Work- Conf. Artif. Neural Netw.*, Cham, Switzerland: Springer, 2017, pp. 585–595.

[27] X. Lan and S. Di Cairano, "Continuous curvature path planning for semi-autonomous vehicle maneuvers using RRT," in *Proc. IEEE Eur. Control Conf.*, 2015, pp. 2360–2365.

[28] H.-T. L. Chiang and L. Tapia, "COLREG-RRT: An RRT-based COLREGS-compliant motion planner for surface vehicle navigation," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 2024–2031, Jul. 2018.

[29] T. T. Enevoldsen, C. Reinartz, and R. Galeazzi, "COLREGS-informed RRT* for collision avoidance of marine crafts," 2021, *arXiv:2103.14426*.

[30] V. Parque and T. Miyashita, "Smooth curve fitting of mobile robot trajectories using differential evolution," *IEEE Access*, vol. 8, pp. 82855–82866, 2020.

[31] A. Zdevsar and I. vSkrjanc, "Optimum velocity profile of multiple bernstein-bézier curves subject to constraints for mobile robots," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 5, pp. 1–23, 2018.

[32] T. Jusko and E. Stoll, *Scalable Trajectory Optimization Based on Bézier Curves*, Braunschweig, Germany: Deutsche Luft-und Raumfahrtkongress, 2016. [Online]. Available: https://www.researchgate.net/publication/327043649_Scalable_Trajectory_Optimization_based_on_Bezier_Curves

[33] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.

[34] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *Proc. IEEE Conf. Decis. Control*, 2014, pp. 6271–6278.

[35] Y. Chen, H. Peng, and J. Grizzle, "Obstacle avoidance for low-speed autonomous vehicles with barrier function," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 1, pp. 194–206, Jan. 2018.

[36] 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.

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

[38] 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 Trans. Robot.*, vol. 34, no. 6, pp. 1603–1622, Dec. 2018.

[39] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 908–919, 2017. [Online]. Available: https://dl.acm.org/doi/10.5555/3294771.3294858

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

[41] A. Majumdar and R. Tedrake, "Robust online motion planning with regions of finite time invariance," in *Algorithmic Foundations of Robotics X*. Berlin, Heidelberg: Springer, 2013, pp. 543–558.

[42] V. Vasilopoulos, G. Pavlakos, K. Schmeckpeper, K. Daniilidis, and D. E. Koditschek, "Reactive navigation in partially familiar planar environments using semantic perceptual feedback," *Int. J. Robot. Res.*, vol. 41, no. 1, pp. 85–126, 2022.

[43] C. Tiseo, V. Ivan, W. Merkt, I. Havoutis, M. Mistry, and S. Vijayakumar, "A passive navigation planning algorithm for collision-free control of mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 8223–8229.

[44] A. Agha et al., "Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge," 2021, *arXiv:2103.11470*.

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

[46] I. D. Miller et al., "Mine tunnel exploration using multiple quadrupedal robots," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2840–2847, Apr. 2020.

[47] M. T. Ohradzansky et al., "Multi-agent autonomy: Advancements and challenges in subterranean exploration," 2021, *arXiv:2110.04390*.

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

[49] D. Tardioli et al., "Ground robotics in tunnels: Keys and lessons learned after 10 years of research and experiments," *J. Field Robot.*, vol. 36, no. 6, pp. 1074–1101, 2019.

[50] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 2997–3004.

[51] 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. 19842–19852, 2020.

[52] I. Noreen, A. Khan, H. Ryu, N. L. Doh, and Z. Habib, "Optimal path planning in cluttered environment using RRT*-AB," *Intell. Serv. Robot.*, vol. 11, no. 1, pp. 41–52, 2018.

[53] 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, 2019, Art. no. 945. [Online]. Available: https://www.mdpi.com/2073-8994/11/7/945

[54] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968, doi: 10.1109/tssc.1968.300136.

[55] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A. search meets graph theory," in *Proc. 16th Annu. ACM-SIAM symp. Discrete algorithms*, vol. 5. Citeseer, Jan. 2005, pp. 156–165. [Online]. Available: https://dl.acm.org/doi/10.5555/1070432.1070455

[56] J. Van Den Berg, R. Shah, A. Huang, and K. Goldberg, "Any-time nonparametric a," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 105–111. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3680/3826

[57] Agility Robotics, "Cassie Simulators," 2018. [Online]. Available: http://www.agilityrobotics.com/sims/

[58] J. Huang, "Fully autonomous on the wave field 2021," 2021. [Online]. Available: https://youtu.be/gE3Y-2Q3gco

[59] 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_reactive_planning_system

[60] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, vol. 6. Springer, 2013.

[61] Y. Gong and J. Grizzle, "Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller," 2020, *arXiv:2008.10763*.

[62] S. Kajita and K. Tani, "Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1991, vol. 2, pp. 1405–1411.

[63] R. Blickhan, "The spring-mass model for running and hopping," *J. Biomech.*, vol. 22, no. 11, pp. 1217–1227, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021929089902248

[64] J. Grizzle, G. Abba, and F. Plestan, "Asymptotically stable walking for biped robots: Analysis via systems with impulse effects," *IEEE Trans. Autom. Control*, vol. 46, no. 1, pp. 51–64, Jan. 2001.

[65] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 295–302.

[66] 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," 2021, *arXiv:2109.14862*.

[67] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. Expanding Societal Role Robot. Next Millennium*, 2001, vol. 1, pp. 239–246.

[68] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *Proc. Int. Conf. Climbing Walking Robots*, 2014 pp. 433–440.

[69] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3019–3026, Oct. 2018.

[70] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 790–797, Apr. 2020.
[71] H. Lee and W. Chung, "A self-training approach-based traversability analysis for mobile robots in urban environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 3389–3394.
[72] T. Shan, J. Wang, B. Englot, and K. Doherty, "Bayesian generalized kernel inference for terrain traversability mapping," in *Proc. Conf. Robot. Learn.*, 2018, pp. 829–838.
[73] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, "3D dynamic walking on stepping stones with control barrier functions," in *Proc. IEEE Conf. Decis. Control*, 2016, pp. 827–834.
[74] 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*. Cham, Switzerland: Springer, 2020, pp. 384–399.
[75] M. Quigley et al., "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009. [Online]. Available: https://www.researchgate.net/publication/233881999_ROS_an_open-source_Robot_Operating_System
[76] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Syst. J.*, vol. 4, no. 1, pp. 25–30, 1965.
[77] 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.
[78] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart, "Topomap: Topological mapping and navigation based on visual slam maps," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 3818–3825.
[79] J. Janos, V. Vonasek, and R. Penicka, "Multi-goal path planning using multiple random trees," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 4201–4208, Apr. 2021.
[80] J. K. Huang and J. W. Grizzle, "Cassie torso design," 2021. [Online]. Available: https://github.com/UMich-BipedLab/torso_design_for_cassie
[81] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 4304–4311.
[82] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1280–1286.
[83] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1360–1367.
[84] J. Huang and J. W. Grizzle, "Improvements to target-based 3D LiDAR to camera calibration," *IEEE Access*, vol. 8, pp. 134101–134110, 2020.
[85] J. K. Huang, S. Wang, M. Ghaffari, and J. W. Grizzle, "LiDARTag: A real-time fiducial tag system for point clouds," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4875–4882, Jul. 2021.
[86] J.-K. Huang, C. Feng, M. Achar, M. Ghaffari, and J. W. Grizzle, "Global unifying intrinsic calibration for spinning and solid-state LiDARs," 2020, *arXiv:2012.03321*.
[87] J. K. Huang and J. W. Grizzle, "Extrinsic LiDAR camera calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/extrinsic_lidar_camera_calibration
[88] J. K. Huang, C. Feng, M. Achar, M. Ghaffari, and J. W. Grizzle, "Intrinsic LiDAR calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/LiDAR_intrinsic_calibration
[89] R. Hartley, M. G. Jadidi, J. Grizzle, and R. M. Eustice, "Contact-aided invariant extended Kalman filtering for legged robot state estimation," in *Proc. Robot.: Sci. Syst. Conf*, Pittsburgh, Pennsylvania, Jun. 2018. [Online]. Available: http://www.roboticsproceedings.org/rss14/p50.pdf
[90] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended Kalman filtering for robot state estimation," *Int. J. Robot. Res.*, vol. 39, no. 4, pp. 402–430, 2020.
[91] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[92] L. Gan et al., "Multi-task learning for scalable and dense multi-layer bayesian map inference," 2021, *arXiv:2106.14986*.
[93] L. Gan, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 8807–8814, Oct. 2022.
[94] Y. Gong and J. Grizzle, "Zero dynamics, pendulum models, and angular momentum in feedback control of bipedal locomotion," *J. Dyn. Syst., Meas., Control*, vol. 144, no. 12, 2021, Art. no. 121006.
[95] Y. Gong et al., "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *Proc. IEEE Amer. Control Conf.*, 2019, pp. 4559–4566.
[96] "The wave field on the north campus of the university of Michigan," https://arts.umich.edu/museums-cultural-attractions/wave-field/
[97] "ICRA 2021 Workshop on legged robots (towards real-world deployment of legged robots)," Accessed: Jun. 10, 2021. [Online]. Available: https://youtu.be/0Gg8BTs6HLY
[98] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.
[99] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.
[100] J. K. Huang et al., "Cassie autonomously navigates around obstacles," 2021. [Online]. Available: https://youtu.be/3HVJotA-w4Y
[101] J. K. Huang et al., "Cassie autonomously navigates in four long corridors (200 meters)," 2021. [Online]. Available: https://youtu.be/PT2mVaKTdT8
[102] J. K. Huang et al., "Cassie autonomous navigation: Smooth motion," 2021. [Online]. Available: https://youtu.be/nPGs4AWLLSg

Jiunn-Kai Huang received the Ph.D. degree in robotics from the University of Michigan, Ann Arbor, MI, USA, in 2022.

He was in charge of building a full stack of autonomy pipeline for the system integration of Cassie Blue, a bipedal robot with 20 degrees of freedom. His research focused on autonomy of bipedal robots, encompassing sensor fusion, pose estimation, and motion planning.

Jessy W. Grizzle (Life Fellow, IEEE) received the Ph.D. degree in electrical engineering from The University of Texas at Austin, Austin, TX, USA, in 1983. He is currently a Professor of robotics with the University of Michigan, where he holds the titles of the Elmer Gilbert Distinguished University Professor and the Jerry and Carol Levin Professor of Engineering.

He jointly holds 16 patents dealing with emissions reduction in passenger vehicles through improved control system design.

Prof. Grizzle is a Fellow IFAC. He was a recipient of the Paper of the Year Award from the IEEE Vehicular Technology Society in 1993, the George S. Axelby Award in 2002, the Control Systems Technology Award in 2003, the Bode Prize in 2012, the IEEE Transactions on Control Systems Technology Outstanding Paper Award in 2014, and the IEEE Transactions on Automation Science and Engineering, Googol Best New Application Paper Award in 2019. His work on bipedal locomotion has been the object of numerous plenary lectures and has been featured on CNN, ESPN, Discovery Channel, The Economist, Wired Magazine, Discover Magazine, Scientific American, and Popular Mechanics.