Informed Sampling-Based Planning to Enable Legged Robots to Safely Negotiate Permeable Obstacles

Yiyu Chen
Research Assistant
Dynamics robotics and Control Lab
Department of Aerospace and Mechanical Engineering
University of Southern California
Email: yiyuc@usc.edu *†

Lingchen Lian
Research Assistant
Dynamics robotics and Control Lab
Department of Computer Science
University of Southern California
Email: lianl@usc.edu

Yu-Hsiu Hsieh
Research Assistant
Dynamics robotics and Control Lab
Department of Computer Science
University of Southern California
Email: yuhsiuhs@usc.edu

Quan Nguyen
Assistant Professor
Dynamics robotics and Control Lab
Department of Aerospace and Mechanical Engineering
University of Southern California
Email: quann@usc.edu

Satyandra K. Gupta
Professor
Center for Advanced Manufacturing
Department of Aerospace and Mechanical Engineering
University of Southern California
Email: skgupta@usc.edu

Legged robots have a unique capability of traversing rough terrains and negotiating cluttered environments. Recent control development of legged robots has enabled robust locomotion on rough terrains. However, such approaches mainly focus on maintaining balance for the robot body. In this work, we are interested in leveraging the whole body of the robot to pass through a permeable obstacle (e.g., a small confined opening) with height, width, and terrain constraints. This paper presents a planning framework for legged robots manipulating its body and legs to perform collision-free locomotion through a permeable obstacle. The planner incorporates quadrupedal gait constraint, biasing scheme, and safety margin for the simultaneous body and foothold motion planning. We perform informed sampling for the body poses and swing foot position based on the gait constraint while ensuring stability and collision avoidance. The footholds are planned based on the terrain and the contact constraint. We also integrate the planner with robot control to execute the planned trajectory successfully. We validated our approach in high-fidelity simulation and hardware experiments on the Unitree A1 robot navigating through different representative permeable obstacles.

1 INTRODUCTION

Legged robots have been highlighted as promising machines for locomotion over complex terrains [1, 2, 3, 4, 5, 6, 7, 8, 9]. Quadruped robots, in particular, are proven to serve as a multipurpose robot platform that can be easily adapted to suit all challenging environments. They are superior to wheeled robots in terms of their loco-

*This work is supported by USC Viterbi School of Engineering.
†The authors are with the University of Southern California, Los Angeles, CA 90089.

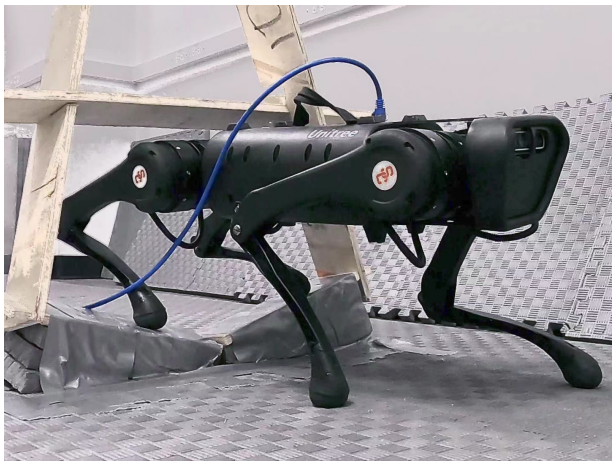


Fig. 1. Unitree A1 robot traversing small a confined opening with our proposed planning framework. Support video: https://youtu.be/xpARKN_LWwk

motion performance. Moreover, the increase in mobility of quadruped robots allows them to overcome obstacles, such as steps and stairs, with statically and dynamically stable gaits. Most control development for legged robots [10, 11] focuses on robust locomotion on rough terrains. However, many situations require machines to go through a small opening on a damaged surface or collapsed building due to an earthquake. This challenge requires developments for locomotion control and motion planning for the entire robot body to allow the robot to navigate through confined spaces.

In this paper, we seek to address the problem of quadruped robots negotiating permeable obstacles (e.g., an irregular-shaped opening in the wall). These obstacles are commonplace in search rescue missions, and rescue dogs are often trained for such scenarios. However, when legged robots encounter this kind of obstacle, common motion planning approaches would ignore such possibility of the robot going through a confined opening but treat it as an obstacle to avoid. This would mean that a portion of a building might become inaccessible to the robot in a time-critical context. Therefore, We present a sampling-based motion planning framework for the legged robot to leverage its entire body to negotiate permeable obstacles, such as an irregular-shaped hole on a wall surface. We specifically address permeable obstacles with height, width, and terrain constraints, requiring the robot to plan a short path to traverse through the opening.

Our approach samples the 6D body pose and foot placement for every walking step simultaneously, utilizing the full-body motion of the robot to go through a confined obstacle while obeying kinematic, stability, collision-free and contact constraints.

The framework also accounts for the model difference between planning and execution by introducing safety margin to the planner. This is a general approach that can adapt to any quadruped robot model, and we validated our framework on the Unitree A1 robot in high-fidelity simulation and hardware experiments.

The main contributions of the paper are as follows:

1. We proposed a motion planning framework for legged robots to negotiate permeable obstacles with width, height, and terrain constraints by planning a short trajectory to traverse through it. It allows simultaneous body and foothold motion planning based on a preset quadrupedal walking gait while enforcing kinematic and stability constraints and accounting for model difference with a safety margin.
2. The proposed framework utilizes the perception data for the planner to select appropriate sample biasing, which accelerates the search significantly.
3. We validated our approach on the Unitree A1 robot using multiple scenarios.

The rest of the paper is organized as follows. Sec. 2 discusses related work. Sec. 3 presents our integrated motion planning and control framework for legged robots navigating confined openings. Numerical and experimental validation are presented in Sec. 4.

2 RELATED WORK

There has been extensive research in the field of robot motion planning. For example, [12, 13] utilize trajectory optimization to ensure collision avoidance while reaching the goal state. Sampling-based methods are also widely used for motion planning.

Many variants of Probabilistic Roadmaps (PRMs) [14], and Rapidly-exploring Random Trees (RRTs) have been developed such as [15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 28, 29, 30] and have sub-optimality bounds.Sampling based methods is futher combined with learning [31] to ensure safety and improve sampling efficiency. However, for legged robots, the problem becomes very challenging due to the complexity of the system, including high degrees of freedom, high dimensional models due to the switching between different gaits or contact modes.

To plan for contacts, the work by Bretl [32] points out two fundamental issues: 1) planning path for the base of the robot and 2) planning a sequence of configurations along the path. The key issue is to handle the computational complexity when accounting for possible contacts and potential paths simultaneously. This

work introduces an effective algorithm to handle situations such as climbing. Following this idea, many works [33, 34, 35, 36, 37, 38, 39, 40, 41, 42] decouple these two problems to reduce the complexity.

In these works, the robot's base trajectory is generated independently with its foot trajectories and contact sequences. Since each task only solves a subset of the motion planning problem, such approaches typically are computationally fast. However, such a hierarchical planning framework could limit the feasible motion of the robot navigating complex terrains or openings because the foot locations play a crucial role in specifying the feasible range of the body motion and collision-induced constraints.

On the other hand, optimization is also commonly used for this motion planning problem [43, 44, 45, 46, 47, 48, 46, 49, 50, 51, 52, 53], while they primarily focus on selecting optimal footholds over rough terrain while ignoring collision constraints. On the other hand, trajectory optimization can also be used to plan for both of body motion and footholds simultaneously [54, 55, 56, 57]. However, these approaches would face local-minima problems with complex obstacles, in our case, permeable obstacles with irregular-shaped openings. However, most of these works focus on footstep planning without considering obstacles in the environment. Some recent developments also account for obstacles with height and width constraints as presented in [58] and [59]. They all employ a hierarchical motion planning framework that separates the planning for footholds and body and thus would constraint feasible motion due to the nature of this approach.

3 APPROACH

3.1 Overview

This section presents a novel framework that integrates perception, planning, and control for a quadruped robot to negotiate permeable obstacles, which typically consist of height limit, width limit, and complex terrain affecting foot placement. Typical planning approaches for mobile robots would consider almost any objects as obstacles and try to find a feasible path to avoid these obstacles. In this scenario, collision-free paths can be found using path planning approaches [60, 31].

However, when navigating in complex and cluttered environments, quadruped robots could encounter some obstacles that are actually feasible for the robot to negotiate and go through the obstacle (e.g., obstacles with certain heights, a low window, or a sizable hole on a broken wall) and it only requires a short trajectory to be planned for the robot to traverse through it. In this scenario, conventional path planning approaches such as A* or RRT can navigate the robot to approach the obstacle directly head-on to the permeable obstacle. Then, given enough knowledge of the obstacle and the terrain around it, we introduce a framework that allows legged robots to leverage the whole-body motion to navigate through the complex permeable obstacles if feasible.

Fig. 2 presents the system diagram of our approach. We first consider the problem of motion planning for the 18 DOF system. Sampling-based planning methods are generally used for high-degree of freedom systems. Similar to RRT, a search tree is created to grow from the initial node to the goal. However, an approach based on purely random sampling does not guarantee that solution exists and usually requires a significant computational time when sampling nodes for the search tree. To address this issue, the idea behind our approach is to select a proper subspace inside the configuration space to facilitate the search to address the body motion planning and foot motion planning in a coupled manner. To plan the robot trajectory to navigate through permeable obstacles, we characterize the environment to account for the obstacle and the terrain. This characterization helps to 1) reduce the size of the configuration space and 2) plan foot holds for the robot. In addition, the planner utilizes information such as the gait schedule and constraints from the control framework to 1) reduce the size of sampling space and 2) impose proper constraints so that the controller is able to execute the planned motion.

3.2 Perception

An obstacle can be considered permeable if there are one or more openings on the obstacle, which allows the robot to move through, for example, the space under a low desk, an opening on a wall, etc. However, conventional perception/mapping algorithms typically ignore such confined openings and treat the entire structure as one impassable obstacle. Therefore, we develop obstacle parameterization to extract useful information from the environment point cloud. We also extract a 2.5D height map [61] of the terrain and a voxel map of the obstacle.

The 2.5D height map outputs the terrain height given x and y coordinates, which the planner then utilizes to impose contact constraints when planning for foot placement. To generate the height map, we selected all points with at most 0.2m height from the ground because the robot can't lift its feet higher than this limit. Then the segmented environment pointcloud is discretized into square cells with a length of 1 cm. The terrain height map is generated based on the z-height of each corresponding grid cell. In the presence of a height constraint, as in Fig. 7(b) and Fig. 7(c), the terrain height map can extract the terrain

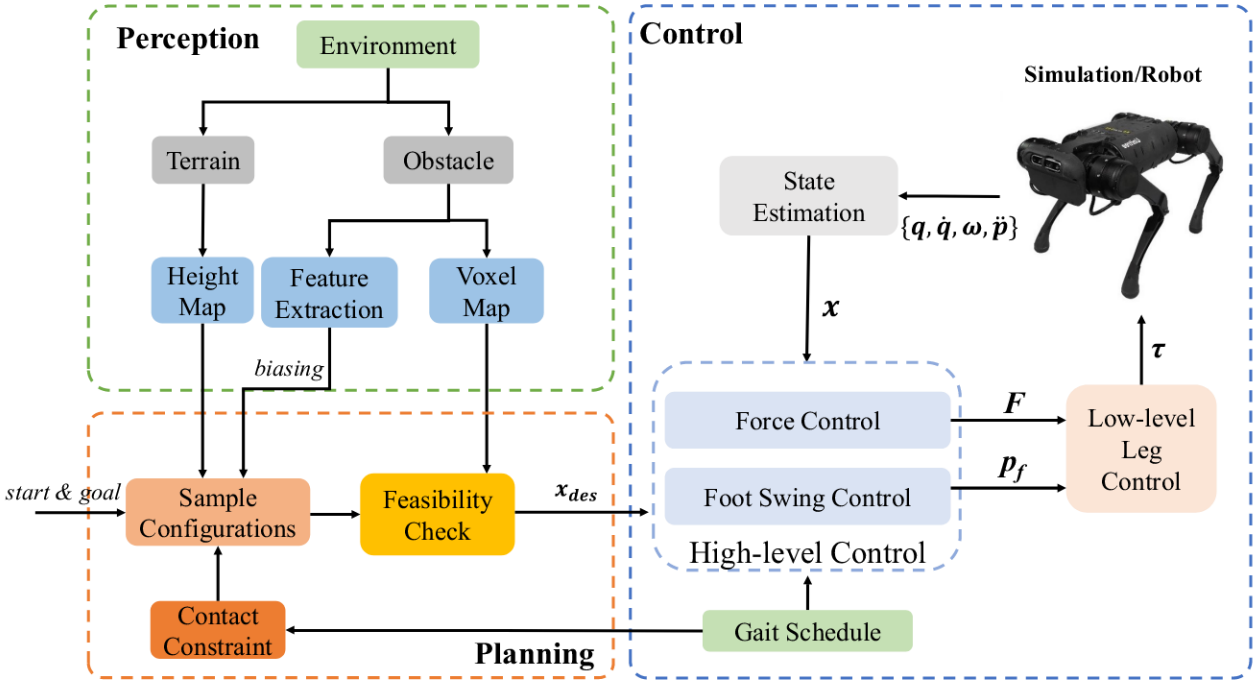


Fig. 2. Block diagram of the proposed planning framework.

information beneath the height limit for footholds selection. The voxel map of the obstacle serves as the collision model of the obstacle while providing data for a feature extraction that will be described below. It is extracted from the pointcloud of the obstacle and used for collision check.

Since the sampling-based itself cannot guarantee that solution exists to support the planning algorithm, we develop a feature extraction algorithm to extract parameters of the obstacle for the planner to (1) decide whether the obstacle is permeable and (2) adopt a proper biasing scheme to improve the efficiency of the algorithm. We utilized the voxel map of the obstacle and sliced it into multiple vertical planes with the body x-axis of the robot's initial configuration as plane normal to extract critical features of the obstacle opening, such as the width clearance and height clearance for each plane. These planes are 1 cm apart to discretize the permeable obstacle. After slicing, the opening in the obstacle is projected onto each sliced plane so that the algorithm uses edge detection to find the boundary of the opening and extract the height and width clearance of each slice. From the collection of slices, the algorithm is able to determine the minimum height and width clearance h_{max}, w_{max} of any arbitrarily shaped opening. Based on the minimum height and width of the robot H_{body}, W_{body}, the planner will decide whether the robot can go through the obstacle. For example, suppose the obstacle clearance is smaller than the minimum dimension of the robot. In that case, the planner won't proceed and will instead switch to a conventional path planning algorithm to find an alternative path for the robot to travel.

3.3 Planning

This section presents our sampling-based planning algorithm that exploits all feasible motions of a quadruped robot by simultaneously planning for the Center of mass (COM) position, base orientation, and foot motion. The planning algorithm also needs to consider gait constraints, collision-free constraints, and stability constraints.

For the execution of the planned trajectory, we also incorporate safety margin in certain constraints to account for model difference between the real robot and the mathematical model used for planning, along with the control accuracy.

Due to the complexity of the problem, applying a uniform-sampling based approach will imply a significant solving time, limiting the framework from real-time applications. Therefore, to speed up the search, our approach also uses a biasing scheme for sampling based on the parameterization of the obstacle. We also imple-

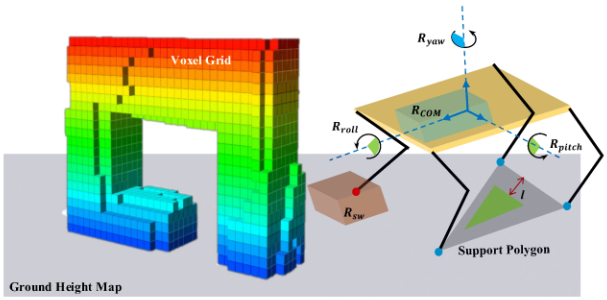


Fig. 3. **Sampling Space Overview.** Sampling space of the robot. The Voxel grid represents the obstacle along with a 2.5D ground height map. R_{COM} represents COM position sampling space; R_{sw} represents swing foot (red color) sampling space; $R_{roll}, R_{pitch}, R_{yaw}$ represents the orientation sampling space of the robot body. The support polygon is defined based on the stance feet (blue color) and l represents the safety margin needed for the support polygon to ensure stability.

mented a shortcut heuristic to smooth the planned COM and foot trajectories for execution. Details about these developments in our planning framework are presented as follows.

3.3.1 *State Representation*

A typical representation of the robot's state consists of the COM position, base orientation, and all the joint angles $\boldsymbol{q}$. We can also represent the robot state with the base states (position and orientation) and all the foot positions in the world frame. Although we can map between them with forward or inverse kinematics, it is hard to apply certain constraints with the joint angle representation. For the foot position representation, the range of foot position and biasing are always constant in the world frame, so we can easily sample the foot position within the range. However, due to the body motion, these constraints are time-varying if we sample with the joint angle representation. It's also hard to apply the contact constraints with the joint angle representation. For example, when the robot's foot stays on the ground, its position in the world frame won't change, but this leg's joint angles will vary due to the robot's base motion. For stance legs, the foot position representation doesn't need to change as the contact foot remains at the same location in the world frame. In contrast, the joint angle representation needs to be re-sampled because the base motion would change the joint configurations, and there is no guarantee that the foot is still in contact with the ground in the new configuration. On the other hand, when the foot is switching from swing to stance, with the foot position representation, we only need to set the z-height of the foot to ground height, while it's still hard to apply the contact condition with the joint angle representation. Therefore, it is more straightforward and convenient to sample the foot locations in the world frame directly, and such representation helps us to apply biasing schemes, range of motion constraints, and contact constraints for the sampling.

Therefore, the algorithm defines the robot state as discrete-time variables including center of mass (COM) position, base orientation, and foot positions for the four legs denoted as $\boldsymbol{x_i} = \left(\boldsymbol{P}_{COM}, \boldsymbol{\Theta}_{COM}, \boldsymbol{P}^w_{foot}\right)^T$, where $\boldsymbol{P}_{COM} \in \mathbb{R}^3$ represents the center of mass (COM) position, $\boldsymbol{\Theta}_{COM} \in \mathbb{R}^3$ is the Euler angle representation of the base orientation, and $\boldsymbol{P}^w_{foot} \in \mathbb{R}^{12}$ represents the position of the robot's four feet, all in the world frame. This state representation is sufficient to characterize any configuration for the robot in a 3D space. It can also be used to derive the 12 joint angles q, and foot positions w.r.t the hip frame $\boldsymbol{P}^h_{foot}$, which are used to ensure kinematic constraints and generate the collision model.

Legged motion requires a distinction between swing and stance phases. While legs during stance exert forces to the environment, legs during swing don't interact with the environment. Specifically, there is no need to sample stance foot positions due to the contact constraints, and only the swing foot positions are needed to be considered in the search. Therefore, the benefits of this state representation are the simplification in enforcing different contact modes or gaits while at the same time reducing the dimension of the search space.

The sampled state is derived from the robot state, and denoted as $\boldsymbol{x_{Sample}} = \left(\boldsymbol{P}_{COM}, \boldsymbol{\Theta}_{COM}, \boldsymbol{P}^w_{Swingfoot}\right)^T$, where $\boldsymbol{P}^w_{Swingfoot} \in \mathbb{R}^{3n}$, with n being the number of the swing leg at the instance of sampling. Therefore, to specify if a leg is in a swing or stance phase, we also need gait parameterization in our planning framework.

3.3.2 *Informed Sampling-based Search*

This framework uses an informed sampling-based search approach to search for feasible motion towards the goal state while accounting for contact and stability constraints and collision avoidance. In brief, we utilize the robot's gait constraint (explained in Sec. 3.3.3) to reduce the dimension of the sampling space. We then employ a biasing scheme based on obstacle parameters and grow the search tree from existing nodes in the tree to ensure the gait constraint.

As mentioned in Section 3.2, the perception algorithm extracts the obstacle clearance and compares it with the minimum dimensions of the robot. Therefore,

we assume there are such configurations that the robot can traverse the obstacle given enough clearance. The sampling function accounts for the contact constraints and the biasing scheme to generate possible configurations. A feasibility check (explained in Sec. 3.3.6) is then performed to ensure the sampled configurations satisfy other constraints such as kinematic and collision-free constraints(explained in Sec. 3.3.4).

A proper biasing scheme can facilitate the sampling-based search because a smaller search space can avoid more infeasible configurations to be sampled. The default biasing scheme presented in Table 1 can still be useful to deal with simpler obstacles (i.e., a single step or a height limit).

Incorporating the biasing scheme improves the solving time of the planning algorithm significantly for complex obstacles. Details about this improvement will be discussed in Section 4.

Based on the minimum clearance of the obstacle h_{max}, w_{max} and the dimension of the robot L_{body}, W_{body}

our adaptive basing scheme extracts the sampling space for the roll, pitch, y, and z as follows:

$$\begin{aligned} \theta_{limit} &= sin^{-1}(h_{max}/L_{body}) \\ \phi_{limit} &= sin^{-1}(w_{max}/w_{body}) \\ y_{limit} &= w_{max} - w_{body} sin(\phi)] \\ z_{limit} &= h_{max} \end{aligned} \tag{1}$$

These limits represent the sampling space for the robot and the algorithm employs this informed-sampling scheme instead of the default biasing scheme in Table 1 to facilitate the search algorithm.Therefore, we have pitch sampling space $\theta \in [-\theta_{limit}, \theta_{limit}]$, pitch sampling space $\phi \in [-\phi_{limit}, \phi_{limit}]$, y sampling space $\boldsymbol{P}_{COM,y} \in [-y_{limit}, y_{limit}]$ and z sampling space $\boldsymbol{P}_{COM,z} \in [0.1, z_{limit}]$. Based on these schemes and constraints, the algorithm creates a search tree that grows from the initial node to the goal state as presented in Algorithm 1. Unlike typical sampling-based searches, our algorithm enforces the gait constraint so that the sampling can only be done in the vicinity of the robot's current position and grow gradually toward the goal state. Thanks to this nature, the algorithm doesn't have to grow from every node greedily but from nodes with zero degrees in the tree (i.e., nodes closer to the goal but have no child node) to search for the next feasible configuration of the robot.

Algorithm 1 Search Algorithm

$T \leftarrow tree(N_{init})$
Set Biasing Scheme
while $!PathFound$ **do**
  for $i = 1$ to size(T) **do**
    if Depth($T[i]$) > criteria **then**
      N_{new} = Sample($T[i]$)
    end if
    if check_feasibility(N_{New}) **then**
      T.addNode(N_{New})
      Update criteria /* if degree of the tree increments*/
    end if
    if $||N_{new} - N_{goal}|| < \rho_{max}$ **then**
      $PathFound = true$
      break
    end if
  end for
end while
$PATH = T.Traceback()$
return SmoothPath(PATH)

Algorithm 2 explains the sampling function based on the sampling space presented in Fig. 3. The algorithm samples the quadruped robot configuration in its workspace, where it can reach with one footstep while obeying constraints of gait, contact, and stability. Given the gait parameterization, the algorithm doesn't have to plan for contact sequence but plan the entire motion step by step based on the gait parameterization, which increments every node.

When the swing foot switches from swing to stance, the sampling function enforces contact constraints based on the terrain information to ensure that the swing foot will land on the ground.

Algorithm 2 Sample(N_{input})

Input: Parent Node N_{input}
$P^w_{COM} = P^w_{COM,input} + P^w_{rand}$ /*in world frame*/
$\Theta_{new} = \Theta_{rand}$
Increment gait parameter and find swing leg ID
for $i : 1 - 4$ **do**
  if Swing Leg **then**
    $P^w_{Swingfoot} = P^w_{Swingfoot,input} + R(\Theta_{new})P^h_{rand}$
    if Enforce Contact **then**
      $P^w_{Swingfoot,z} = z_{terrain}$
    end if
  else if Stance leg **then**
    $P_{Stancefoot} = P^{input}_{Stancefoot}$
  end if
end for
$N_{new} \leftarrow P^w_{COM}, \Theta, P^w_{Swingfoot}, P^w_{Stancefoot}$
Update q
Update P^h_{foot}
UpdateCollisionModel
return N_{new}

3.3.3 *Gait Parameterization for Contact Sequence*

Gait schedule is commonly used for quadruped robot locomotion control [11], which can effectively achieve

Table 1. Default Biasing Scheme

Parameter	Value	Parameter	Value	Parameter	Value
$\boldsymbol{P}_{COM,x}$	$[-0.05, 0.2]m$	$roll(\phi)$	$\pm 0.5rad$	$\boldsymbol{P}_{Swingfoot,x}$	$[0, 0.2]m$
$\boldsymbol{P}_{COM,y}$	$\pm 0.1m$	$pitch(\theta)$	$\pm 0.5rad$	$\boldsymbol{P}_{Swingfoot,y}$	$\pm 0.1m$
$\boldsymbol{P}_{COM,z}$	$[0.1, 0.3]m$	$yaw(\psi)$	$\pm 0.5rad$	$\boldsymbol{P}_{Swingfoot,z}$	$[-0.3, -0.1]m$

a wide range of dynamic motion. A predefined gait sequence also helps to reduce the complexity of the planning framework significantly. Therefore, our work leverages the methodology of using a fixed gait schedule in legged robot control.

Instead of planning for a gait sequence, we leverage the periodic gait schedule commonly used for quadruped robot control. Our planner can adapt to any kind of gaits such as trotting, bounding, and galloping. When planning the robot to go through a permeable obstacle, a static walking gait is selected so that the robot can slowly leverage its whole-body motion to pass through confined openings.

Based on the gait schedule, the planning framework will perform sampling for swing legs only and keep stance legs at the same position during the stance phase. To extract both the contact switching and the swing trajectory of each foot, our approach discretizes the gait schedule as presented in Fig. 4. The algorithm specifies one time/phase-based gait cycle into $n+1$ time-independent state variables, which represent contact states for the robot legs at those instances. As the gait schedule iterates over time, the gait variable increments with the growth of the search tree (mentioned in Section 3.3.2), and similar to the gait cycle, it repeats itself when one gait cycle finishes.

The gait parameterization is embedded with the state representation so that the planner can switch each leg between stance and swing with the growth of the search tree. We can formulate the primary node for the search tree for our sampling-based approach based on the state and gait parameterization.

3.3.4 *Collision Model*

The collision model is critical to ensure safety. Going through a confined space requires an accurate collision model that doesn't enforce conservative assumptions. However, the complexity of the collision model also affects the computational time of the planning framework. In addition, the obstacle is represented by a voxel map which consists of hundreds or thousands of small collision spheres. Therefore, the robot's collision model must account for all its links while minimizing the number of collision boxes to facilitate the collision check. From the state representation, the planner generates the corresponding collision model for the robot as presented in Fig. 5. For fast computing, we utilize a spherical representation of every link of the robot based on the dimension of each link. This collision model accurately represents the robot's state without making any conservative assumption that may restrict feasible motions. Meanwhile, the collision model should have the flexibility to account for foot placement onto the obstacle. Therefore, our collision model removes the collision sphere for the stance feet while including a collision sphere for the swing feet to ensure collision avoidance. The collision model has to account for foot contacts to allow the robot to step on the obstacle.

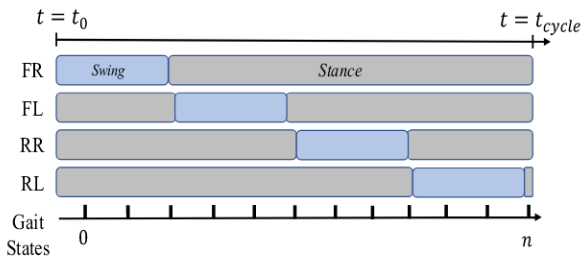


Fig. 4. **Gait Parameterization.** Parametrize a static walking gait into $n+1$ time-independent state variables, which corresponds to a specific contact state defined by the gait schedule.

3.3.5 *Foothold Selection & Obstacle negotiation*

The foothold selection has to account for two aspects of the problem: 1) Obstacle negotiation and 2) stability constraint. While some motion planning frameworks deterministically select foothold selection, our planner samples the swing foot position and the base position and orientation with the biasing scheme that drives the robot to move forward. As long as the sampled configuration is feasible in terms of kinematic, stability, and collision constraints, the footholds are determined for the sampled configuration without using any heuristic or other optimization techniques. Each swing foot will be sampled

for 2 configurations before it switches to stance. When the foot is in air, it doesn't affect the robot's motion because it has no contact force, while our algorithm makes sure it is collision-free. The foothold selection happens when the swing foot switches from swing to stance. The planner still samples the swing foot position while enforcing the contact constraint by forcing the z-position of the foot as the ground height. This allows the foot to swing forward or backward and land on the ground. Thanks to the 2.5D height map the framework obtained from the environment, the planner already accounts for any complex terrain, so it only needs to force the swing foot onto the ground and then check the feasibility of that specific configuration. Intuitively, the robot should be able to either step on or step over some obstacles. As presented in Fig. 2, the height map is used only during sampling for foot placement while the voxel grid is only used for feasibility check. In other words, the footholds are selected when sampling each node without considering the collision-free constraints. Since our planner utilizes the terrain height map, the footholds are sampled along with the base configuration according to the contact constraints. So, there can be configurations that satisfy the contact constraints but violating other constraints such as kinematic constraints or collision-free constraints. Therefore, we can set the unreachable region in the heightmap to null, so that the sampled configuration won't be feasible for kinematics constraints if the foothold is sampled in this region. On the other hand, when contact is enforced at a feasible location and satisfy kinematic constraints, the configuration is feasible as long as it can pass the collision check.

3.3.6 Feasibility Check

The algorithm performs a linear interpolation first for the input node and its parent to ensure feasible motion and safety. It makes sure the sampled node is feasible, and the path from its parent to the node is feasible. The function checks the robot's kinematic constraints for every interpolated node, such as joint limits and foot motion limits. It also guarantees that the COM position always lies in the support polygon to maintain balance for the robot. In addition, this function performs collision checks for the robot and obstacles to ensure safety.

3.3.7 Path Smoothing & Safety Margin

To execute the planned path with our controller, directly applying the resulting path from the sampling-based search algorithm is not adequate. Therefore, a basic shortcut heuristic [62] is implemented to remove unnecessary motions in the trajectory. In addition, the shortcut algorithm accounts for the contacts while removing unnecessary motion for both the body and the foot. Due to the difference in the robot model between the planning framework and simulation/experiment, we also implemented safety margins by enforcing slightly more conservative stability constraints and collision models. For the support polygon, we added an offset l as presented in Fig. 3 so that the COM always lies in the shrunken support polygon. We've also added small offsets of the obstacle and robot collision model to ensure the safety of the entire motion.

3.4 Control

In this Section, we present our proposed control framework used to execute the trajectory generated by the planner. The goal of the controller is to guarantee tracking performance for the robot's body and foot trajectories while keeping the robot balanced. Therefore, we utilize the combination between force-based control and Cartesian PD control to enforce these objectives in our approach. Our control architecture consists of modules as shown in Fig. 2, including high-level controller, low-level controller, state estimation, and gait scheduler. The gait scheduler sets up gait timing to switch each leg between swing and stance along with the reference trajectory. The high-level controller switch between different control algorithms for swing and stance legs based on the planned trajectory and gait timing. The low-level leg control converts command generated by high-level control into joint torques and sends them to the robot.

While a leg is in swing, Cartesian PD control is used to track the desired swing foot trajectory. Stance control leverages the combination between the QP force control [2] and Cartesian PD control. The force-based balancing controller is formulated as a quadratic program (QP) based on a simplified centroidal dynamics of the robot [6, 63]. This model implies a linear relationship between the linear acceleration $\ddot{p}_c$, angular acceleration $\dot{\omega}_b$ of the robot body, and the foot forces $\boldsymbol{F} = (\boldsymbol{F}_1^T, \boldsymbol{F}_2^T, \boldsymbol{F}_3^T, \boldsymbol{F}_4^T)^T$ acting on each of the robot foot. The linear model is derived as:

$$\underbrace{\begin{bmatrix} \mathbf{I}_3 & \dots & \mathbf{I}_3 \\ [\boldsymbol{p}_1 - \boldsymbol{p}_c]\times & \dots & [\boldsymbol{p}_4 - \boldsymbol{p}_c]\times \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{F} = \underbrace{\begin{bmatrix} m(\ddot{\boldsymbol{p}}_c + \boldsymbol{g}) \\ \boldsymbol{I}_G \dot{\boldsymbol{\omega}}_b \end{bmatrix}}_{\boldsymbol{b}}, \quad (2)$$

where m and $\boldsymbol{I}_G$ are the robot's total mass and rotational inertia, $\boldsymbol{g}$ is the gravity vector and $\boldsymbol{p}_i, i \in \{1, 2, 3, 4\}$ are the positions of the feet. The term $[\boldsymbol{p}_i - \boldsymbol{p}_c]\times$ is the skew-symmetric matrix representing the cross product $(\boldsymbol{p}_i - \boldsymbol{p}_c) \times \boldsymbol{F}_i$.

Then, the controller is able to drive the approximate dynamics to the corresponding desired dynamics. The QP-based controller can also enforce essential physical constraints such as input saturation, contact constraints and the friction constraints. Since the model (2) is linear, the controller can be solved by quadratic program (QP) [64] in real-time of 1 kHz.

On the other hand, the framework also requires tracking for foot position. Therefore, a Cartesian PD controller for foot position is used to compute joint torques to track the desired position for each foot:

$$\boldsymbol{\tau} = J^T[\boldsymbol{K}_{p,p}(\boldsymbol{p}_d - \boldsymbol{p}) + \boldsymbol{K}_{d,p}(\boldsymbol{v}_d - \boldsymbol{v})], \qquad (3)$$

where J is the leg Jacobian, $\boldsymbol{p}_d$ and $\boldsymbol{v}_d$ are desired foot position and velocity in hip frame, $\boldsymbol{p}$ and $\boldsymbol{v}$ are actual foot position and velocity in hip frame, $\boldsymbol{K}_{p,p}$ and $\boldsymbol{K}_{d,p}$ are the diagonal matrices of the proportional and derivative gains for foot position in Cartesian coordinate.

4 RESULTS AND DISCUSSION

In this work, we used the Unitree A1 robot to validate our approach. To validate our approach, we formulated several test cases of permeable obstacles with width/height constraints and complex terrains at the obstacle. First, we use three simple cases to compare different control approaches, safety margin, and obstacle negotiation: 1) flat ground, 2) 5-cm step, and 3) 10-cm step. Then, we validate our approach on three more complex cases. We formulated a 30-cm width limitation, the robot's nominal stance width, for case 1. Then we combined width and height constraints along with discrete terrain for case 2, a narrow window with 20-cm minimum clearance in height and 36-cm clearance in width, along with discrete 10-cm step. Finally, we formulated an arbitrarily shaped opening with more complex terrain for case 3, a small opening with an arbitrary shape combined with discrete sloped terrain.

4.1 Simulation

In simulation, we successfully validated our planning and control approaches and implemented our approach for our test cases as presented in Fig. 6 and our support video and our test video. To emphasize the contribution of our approach, we present the following comparisons.

4.1.1 Control Scheme

To illustrate the effectiveness of the proposed control approach as described in Section 3.4, we compare here different control schemes in executing the planned trajectory: (1) joint PD control, (2) foot Cartesian PD control, and (3) QP controller combined with Cartesian PD control. In simulation, control schemes (1) and (2) fail to balance the robot on flat ground after a few steps, while our proposed control scheme (3) works to track both the COM and foot motion. Since the joint PD control and Cartesian PD control do not consider the COM position and body orientation, they fail to balance the robot. Our proposed controller uses force-based balancing control, which considers feedback control in the body position and orientation. Therefore, the controller can ensure the execution of the planned trajectory.

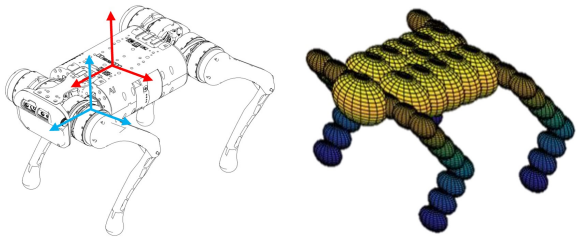
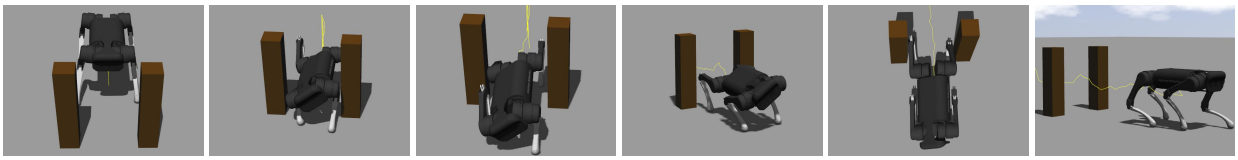


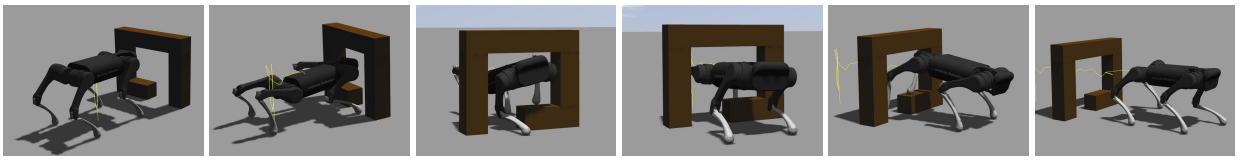
Fig. 5. **Robot Model.** The Unitree A1 robot and its corresponding collision Model. The red frame indicates the body frame of the robot while the blue frame represents the hip frame of the front left leg of the robot

4.1.2 Safety Margin

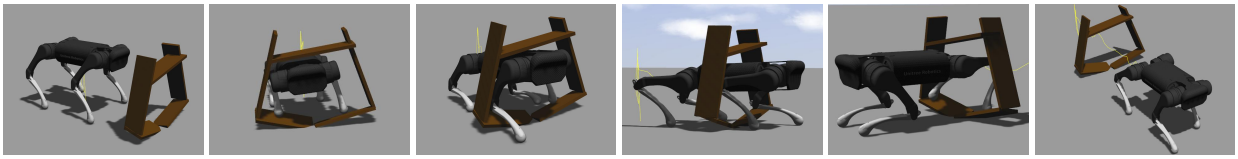
As discussed above, due to model difference and underactuation, it is challenging for the controller to achieve high accuracy in realizing the planned trajectory on the robot. Therefore, to guarantee the satisfaction of stability and collision-free constraints, we consider the control accuracy by adding safety margins within the planning framework. Here, we compare the performance with and without using the safety margin presented in Table 2. We planned 10 trajectories for each case and executed them in the simulation to compare the success rate. We use the same controller setup to compare the performance of applying safety margins for support polygon and collision check. For flat ground, the controller fails to balance the robot 5 out of 10 trajectories without the margin for support polygon, while it can achieve 100% success rate with safety margins in the planning. The safety margin becomes very important when the robot is planned to step on or step over the obstacle. For the 5-cm and 10-cm step cases, the front or rear legs collide with the obstacle or fail to step onto the obstacle without safety margins. Such issues in the execution are resolved with a proper safety



(a) Case 1



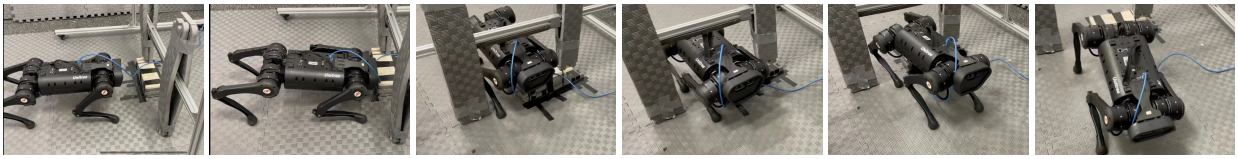
(b) Case 2



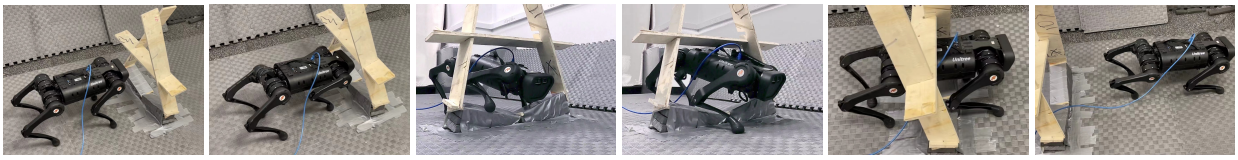
(c) Case 3

Fig. 6. **Simulation Result Highlight.** The planning framework deals with: Case 1) $30 - cm$ width limit; Case 2) Combined height $\&$ width limit along with discrete terrain; Case 3) Irregularly shaped opening with discrete sloped terrain.

(a) Case 1



(b) Case 2



(c) Case 3

Fig. 7. **Experiment Result Highlight.** The planning framework deals with: Case 1) $30 - cm$ width limit; Case 2) Combined height $\&$ width limit along with discrete terrain; Case 3) Irregularly shaped opening with discrete sloped terrain.

margin, and we can achieve a 100% success rate for these simulation cases.

4.1.3 Obstacle Negotiation

Since the planner has the flexibility of deciding whether to step on the obstacle, We also validated our approach with different obstacle sizes as presented in Table 3. For a small obstacle, the planner can either force

Table 2. Success rate of applying safety margin in Simulation

	w/o Safety Margin	w/ Safety Margin
Flat Ground	5/10	10/10
5cm Step	4/10	10/10
10cm Step	0/10	10/10

the robot to step over the obstacle or allow it to step on the obstacle. With larger and higher obstacles that do not allow the robot to step over, the planning approach automatically enforces a solution that asks the robot to step onto the obstacle.

4.1.4 Biasing Scheme

First, we validated our biasing scheme by comparing the computation time required for the planning. The planning algorithm was run with an AMD Ryzen9 5900X CPU for all cases 30 times with default biasing and obstacle-based biasing. As presented in Table 4 are the runtime results, and we compare the performance with and without using the proposed biasing scheme based on the obstacle feature. Although the runtime varies for different cases, our proposed biasing scheme significantly improves the runtime for the planning. Case 1 takes a longer runtime because finding a stable configuration is harder than other cases when the robot has a large roll angle. Every planned trajectory includes more than 20 steps so that the runtime is less than 1 second per step for most cases.

4.2 Experiments on Physical Platform

As shown in the support video, the robot would fail in the simulation without proper control scheme and safety margin. Therefore, we only tried with our proposed control scheme and safety margin to avoid damage to our hardware. With our proposed method, we validated our approach on hardware in terms of obstacle negotiation and biasing scheme. We replicate the test cases we have in simulation in real life as presented in Fig. 7. As presented in Table 5, we planned 10 different trajectories with our framework and tested them in simulation and robot hardware. The average time it takes for the robot to traverse the 3 test cases is approximately 20 seconds because the initial and final positions are set in the same position for all 3 cases. We keep the same control parameters for our experiments. We have a smaller success rate in hardware experiments as the robot's feet would be stuck at the obstacle with discrete steps. The reason is that we only use the Kalman filter for robot state estimation based on the IMU reading and all joint encoders, which inevitably has some drift as it runs. The inaccurate state estimation results in errors in 1) the COM trajectory tracking and 2) the foot position tracking. Since we convert from the world frame to the hip frame to control the foot position based on (3), the foot position estimation is critical for the controller to track the desired trajectory. Therefore, the experiment for case 1 has a higher success rate because the robot doesn't need to step on the obstacle. However, when the robot is planned to step on the obstacle, as in case 2 and case 3,

the drift in the state estimation would result in inaccurate foot trajectory commands and tracking, which often leads to failure to step on the obstacle. By integrating better state estimation, we can minimize this error so that the controller can track the desired base and foot trajectories with higher accuracy.

5 CONCLUSIONS

In this paper, we presented a novel context-based informed sampling-based approach for legged robots while integrating planning and control. We demonstrated its capability to simultaneously plan for robot COM and foot motions to traverse a complex permeable obstacle. We use a context-based biasing scheme to accelerate the search significantly. Such scheme allows the planner to explore feasible motion towards the final goal more efficiently. We successfully execute the planned trajectory with appropriate safety margins with the proposed control approach of combining force-based balance control and Cartesian control. With the proposed framework, the robot is able to traverse complex permeable obstacles.

Given the promising results of this framework, there are several directions of investigation to be pursued. While our method formulates a solution to the motion planning for legged robots, the framework will need to be further developed for online planning and execution by improving the sampling efficiency because the current sampling scheme still suffers from low sampling efficiency. Although the framework is able to deal with more complex environment data from the perception algorithm, the low sampling efficiency limits its performance to deal with such kind of scenarios. Also, we will utilize better localization with lidar or vision to improve the tracking performance of the controller as the current framework only uses Kalman filter for state estimation. To handle errors in the execution, we will incorporate reactive planning to make the entire motion more robust and then we can extend the framework for more complex obstacles, terrains, and more dynamic movements.

Table 3. Comparison of Different Obstacle negotiation strategies in Simulation

Height	Length	Stepping On?	Success Rate	Runtime
$5cm$	$10cm$	Yes	5/5	$1.47s$
$5cm$	$10cm$	No	5/5	$1.58s$
$10cm$	$10cm$	Yes	5/5	$1.65s$
$10cm$	$10cm$	No	5/5	$1.73s$
$10cm$	$15cm$	Yes	5/5	$1.79s$
$10cm$	$15cm$	No	N/A	N/A

Table 4. Average runtime

	w/o Biasing	w/ Biasing
Flat Ground	$1.27s$	$1.04s$
Case 1	$93.73s$	$32.82s$
Case 2	$109.71s$	$17.79s$
Case 3	$47.69s$	$10.05s$

Table 5. Successful trials for simulation and hardware experiments

	Case 1	Case 2	Case 3
Simulation	10/10	10/10	10/10
Hardware	10/10	7/10	6/10

REFERENCES

[1] Krotkov, E., Hackett, D., Jackel, L., Perschbacher, M. P. J., Strauss, J., Pratt, G., and Orlowski, C., 2017, "The darpa robotics challenge finals: Results and perspectives.," In Journal of Field Robotics, Vol. 34(2), pp. 229–240.

[2] Bledt, G., Powell, M. J., Katz, B., Di Carlo, J., Wensing, P. M., and Kim, S., 2018, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2245–2252.

[3] Katz, B., Carlo, J. D., and Kim, S., 2019, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," In 2019 International Conference on Robotics and Automation (ICRA), pp. 6295–6301.

[4] Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C. D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., and Hoepflinger, M., 2016, "ANYmal - a highly mobile and dynamic quadrupedal robot," In IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 38–44.

[5] Semini, C., Tsagarakis, N. G., Guglielmino, E., Focchi, M., Cannella, F., and Caldwell, D. G., 2011, "Design of hyq – a hydraulically and electrically actuated quadruped robot," Vol. 225, pp. 831–849.

[6] Focchi, M., del Prete, A., Havoutis, I., Featherstone, R., Caldwell, D. G., and Semini, C., 2017, "High-slope terrain locomotion for torque-controlled quadruped robots," Vol. 41, pp. 259–272.

[7] Nguyen, Q., and Sreenath, K., 2015, "L1 adaptive control for bipedal robots with control lyapunov function based quadratic programs," In 2015 American Control Conference (ACC), IEEE, pp. 862–867.

[8] Nguyen, Q., Da, X., Grizzle, J. W., and Sreenath, K., 2020, "Dynamic walking on stepping stones with gait library and control barrier functions," In Algorithmic Foundations of Robotics XII, Springer, pp. 384–399.

[9] Nguyen, Q., Agrawal, A., Martin, W., Geyer, H., and Sreenath, K., 2018, "Dynamic bipedal locomotion over stochastic discrete terrain," Vol. 37, SAGE Publications Sage UK: London, England, pp. 1537–1553.

[10] Raibert, M., Blankespoor, K., Nelson, G., and Playter, R., 2008, "Bigdog, the rough-terrain quadruped robot," *IFAC Proceedings Volumes,* **41**(2), pp. 10822–10825.

[11] Di Carlo, J., Wensing, P. M., Katz, B., Bledt, G., and Kim, S., 2018, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 1–9.

[12] Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., Pan, J., Patil, S., Goldberg, K., and Abbeel, P., 2014, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research,* **33**(9), pp. 1251–1270.

[13] Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., and Srinivasa, S. S., 2013, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research,* **32**(9-10), pp. 1164–1193.

[14] Kavraki, L. E., Kolountzakis, M. N., and Latombe, J.-C., 1998, "Analysis of probabilistic roadmaps for path planning," *IEEE Transactions on Robotics and automation,* **14**(1), pp. 166–171.

[15] Kim, B., Um, T. T., Suh, C., and Park, F. C., 2016, "Tangent bundle rrt: A randomized algorithm for constrained motion planning," *Robotica,* **34**(1), pp. 202–225.

[16] Berenson, D., Siméon, T., and Srinivasa, S. S., 2011, "Addressing cost-space chasms in manipulation planning," In 2011 IEEE International Conference on Robotics and Automation, IEEE, pp. 4561–4568.

[17] Jaillet, L., Cortés, J., and Siméon, T., 2010, "Sampling-based path planning on configuration-space costmaps," *IEEE Transactions on Robotics,* **26**(4), pp. 635–646.

[18] Karaman, S., and Frazzoli, E., 2011, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research,* **30**(7), pp. 846–894.

[19] Bry, A., and Roy, N., 2011, "Rapidly-exploring random belief trees for motion planning under uncertainty," In 2011 IEEE international conference on robotics and automation, IEEE, pp. 723–730.

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

[21] Karaman, S., and Frazzoli, E., 2010, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI,* **104**(2).

[22] Karaman, S., and Frazzoli, E., 2010, "Optimal kinodynamic motion planning using incremental sampling-based methods," In 49th IEEE conference on decision and control (CDC), IEEE, pp. 7681–7687.

[23] Xiang, S., Gao, H., Liu, Z., and Gosselin, C., 2020, "Dynamic Point-To-Point Trajectory Planning for Three Degrees-of-Freedom Cable-Suspended Parallel Robots Using Rapidly Exploring Random Tree Search," *Journal of Mechanisms and Robotics,* **12**(4), 03 041007.

[24] Zucker, M., Kuffner, J., and Bagnell, J. A., 2008, "Adaptive workspace biasing for sampling-based planners," In 2008 IEEE International Conference on Robotics and Automation, pp. 3757–3762.

[25] Tahirovic, A., and Ferizbegovic, M., 2018, "Rapidly-exploring random vines (rrv) for motion planning in configuration spaces with narrow passages," In 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 7055–7062.

[26] Ademovic, A., and Lacevic, B., 2016, "Path planning for robotic manipulators using expanded bubbles of free c-space," In 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 77–82.

[27] McMahon, T., Thomas, S., and Amato, N. M., 2018, "Sampling-based motion planning with reachable volumes for high-degree-of-freedom manipulators," *The International Journal of Robotics Research,* **37**(7), pp. 779–817.

[28] Guo, Z., Tachi, T., and Yu, H., 2021, "Folding Process Planning of Rigid Origami Using the Explicit Expression and Rapidly Exploring Random Tree Method," *Journal of Mechanisms and Robotics,* **14**(1), 07 011003.

[29] Venkatesan, V., Seymour, J., and Cappelleri, D. J., 2018, "Micro-Assembly Sequence and Path Planning Using Subassemblies," *Journal of Mechanisms and Robotics,* **10**(6), 10 061015.

[30] Kennedy, Monroe, I., Thakur, D., Ani Hsieh, M., Bhattacharya, S., and Kumar, V., 2018, "Optimal Paths for Polygonal Robots in SE(2)," *Journal of Mechanisms and Robotics,* **10**(2), 02 021005.

[31] Kim, Y., Kim, C., and Hwangbo, J., 2022, "Learning forward dynamics model and informed trajectory sampler for safe quadruped navigation," *arXiv preprint arXiv:2204.08647*.

[32] Bretl, T., 2006, "Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem," *The International Journal of Robotics Research,* **25**(4), pp. 317–342.

[33] Escande, A., Kheddar, A., Miossec, S., and Garsault, S., 2009, "Planning support contact-points for acyclic motions and experiments on hrp-2," In Experimental Robotics, Springer, pp. 293–302.

[34] Tonneau, S., Mansard, N., Park, C., Manocha, D., Multon, F., and Pettré, J., 2018, "A reachability-based planner for sequences of acyclic contacts in cluttered environments," In *Robotics Research*. Springer, pp. 287–303.

[35] Hauser, K., Bretl, T., Harada, K., and Latombe, J.-C., 2008, "Using motion primitives in probabilistic sample-based planning for humanoid robots," In *Algorithmic foundation of robotics VII*. Springer, pp. 507–522.

[36] Grey, M. X., Ames, A. D., and Liu, C. K., 2017, "Footstep and motion planning in semi-unstructured environments using randomized possibility graphs," In 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 4747–4753.

[37] LaValle, S. M., and James J. Kuffner, J., 2001, "Randomized kinodynamic planning," *The International Journal of Robotics Research,* **20**(5), pp. 378–400.

[38] Geisert, M., Yates, T., Orgen, A., Fernbach, P., and Havoutis, I., 2019, "Contact planning for the anymal quadruped robot using an acyclic reachability-based planner," In Towards Autonomous Robotic Systems, K. Althoefer, J. Konstantinova, and K. Zhang, eds., Springer International Publishing, pp. 275–287.

[39] Fankhauser, P., Dario Bellicoso, C., Gehring, C., Dubé, R., Gawel, A., and Hutter, M., 2016, "Free gait — an architecture for the versatile control of legged robots," In 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pp. 1052–1058.

[40] Hildebrandt, A.-C., Klischat, M., Wahrmann, D., Wittmann, R., Sygulla, F., Seiwald, P., Rixen, D., and Buschmann, T., 2017, "Real-time path planning in unknown environments for bipedal robots," *IEEE Robotics and Automation Letters,* **2**(4), pp. 1856–1863.

[41] Perrin, N., Stasse, O., Baudouin, L., Lamiraux, F., and Yoshida, E., 2011, "Fast humanoid robot collision-free footstep planning using swept volume approximations," *IEEE Transactions on Robotics,* **28**(2), pp. 427–439.

[42] Zucker, M., Ratliff, N., Stolle, M., Chestnutt, J., Bagnell, J. A., Atkeson, C. G., and Kuffner, J., 2011, "Optimization and learning for rough terrain legged locomotion," *The International Journal of Robotics Research,* **30**(2), pp. 175–191.

[43] Chignoli, M., and Kim, S., 2021, "Online trajectory optimization for dynamic aerial motions of a quadruped robot," In 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 7693–7699.

[44] Cebe, O., Tiseo, C., Xin, G., Lin, H.-c., Smith, J., and Mistry, M., 2021, "Online dynamic trajectory optimization and control for a quadruped robot," In 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 12773–12779.

[45] Huang, S., and Zhang, X., 2021, "Biologically inspired planning and optimization of foot trajectory of a quadruped robot," In International Conference on Intelligent Robotics and Applications, Springer, pp. 192–203.

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

[47] Mordatch, I., Todorov, E., and Popović, Z., 2012, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG),* **31**(4), pp. 1–8.

[48] Posa, M., Cantu, C., and Tedrake, R., 2014, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research,* **33**(1), pp. 69–81.

[49] Mastalli, C., Focchi, M., Havoutis, I., Radulescu, A., Calinon, S., Buchli, J., Caldwell, D. G., and Semini, C., 2017, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," In 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 1096–1103.

[50] Tian, Y., and Gao, F., 2018, "Efficient motion generation for a six-legged robot walking on irregular terrain via integrated foothold selection and optimization-based whole-body planning," *Robotica,* **36**(3), p. 333–352.

[51] Hereid, A., Cousineau, E. A., Hubicki, C. M., and Ames, A. D., 2016, "3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics," In 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1447–1454.

[52] Powell, M. J., Zhao, H., and Ames, A. D., 2012, "Motion primitives for human-inspired bipedal robotic locomotion: walking and stair climbing," In 2012 IEEE International Conference on Robotics and Automation, pp. 543–549.

[53] Yunt, K., and Glocker, C., 2006, "Trajectory optimization of mechanical hybrid systems using sumt," In 9th IEEE International Workshop on Advanced Motion Control, 2006., IEEE, pp. 665–671.

[54] Winkler, A. W., Farshidian, F., Pardo, D., Neunert, M., and Buchli, J., 2017, "Fast trajectory optimization for legged robots using vertex-based zmp constraints," *IEEE Robotics and Automation Letters,* **2**(4), pp. 2201–2208.

[55] Naveau, M., Kudruss, M., Stasse, O., Kirches, C., Mombaur, K., and Souères, P., 2017, "A reactive walking pattern generator based on nonlinear model predictive control," *IEEE Robotics and Automation Letters,* **2**(1), pp. 10–17.

[56] Winkler, A. W., Bellicoso, D. C., Hutter, M., and Buchli, J., 2018, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters (RA-L),* **3**, July, pp. 1560–1567.

[57] Winkler, A. W., Farshidian, F., Neunert, M., Pardo, D., and Buchli, J., 2017, "Online walking motion and foothold optimization for quadruped locomotion," In 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 5308–5313.

[58] Li, Z., Zeng, J., Chen, S., and Sreenath, K., 2021, Vision-aided autonomous navigation of bipedal robots in height-constrained environments.

[59] Buchanan, R., Wellhausen, L., Bjelonic, M., Bandyopadhyay, T., Kottege, N., and Hutter, M., 2021, "Perceptive whole-body planning for multilegged robots in confined spaces," *Journal of Field Robotics,* **38**(1), pp. 68–84.

[60] Wang, P., Zhou, X., Zhao, Q., Wu, J., and Zhu, Q., 2021, "Search-based kinodynamic motion planning for omnidirectional quadruped robots," In 2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 823–829.

[61] Fankhauser, P., and Hutter, M., 2016, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," In *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, A. Koubaa, ed. Springer, ch. 5.

[62] Baginski, B., 1997, "Efficient motion planning in high dimensional spaces: The parallelized zˆ 3-method," In Proc. of 6th International Workshop on Robotics in the Alpe-Adria-Danube Region RAAD'97.

[63] Stephens, B., and Atkeson, C., 2010, "Push recovery by stepping for humanoid robots with force controlled joints," In IEEE-RAS International Conference on Humanoid Robots, pp. 52–59.

[64] Gehring, C., Coros, S., Hutter, M., Bloesch, M., Hoepflinger, M., and Siegwart, R., 2013, "Control of dynamic gaits for a quadrupedal robot," In IEEE International Conference on Robotics and Automation (ICRA), pp. 3287–3292.