

Steppability-informed Quadrupedal Contact Planning through Deep Visual Search Heuristics

Max Asselmeier*, Ye Zhao, and Patricio A. Vela

Institute for Robotics and Intelligent Machines, Georgia Institute of Technology,
Atlanta GA 30308, USA,
`mass@gatech.edu`

Abstract. In this work, we introduce a method for predicting environment *steppability* – the ability of a legged robot platform to place a foothold at a particular location in the local environment – in the image space. This novel environment representation captures this critical geometric property of the local terrain while allowing us to exploit the computational benefits of sensing and planning in the image space. We adapt a primitive shapes-based synthetic data generation scheme to create geometrically rich and diverse simulation scenes and extract ground truth semantic information in order to train a steppability model. We then integrate this steppability model into an existing interleaved graph search and trajectory optimization-based footstep planner to demonstrate how this steppability paradigm can inform footstep planning in complex, unknown environments. We analyze the steppability model performance to demonstrate its validity, and we deploy the perception-informed footstep planner both in offline and online settings to experimentally verify planning performance.

Keywords: quadruped, contact planning, steppability, graph search, trajectory optimization

1 Introduction

Motion planning in off-road, complex environments necessitates a model of the local terrain. For wheeled systems, these terrain representations have been carefully studied through the aid of programs such as the DARPA Learning Applied to Ground Vehicles (LAGR) [1] program and the U.S. Army Research Lab Robotics Collaborative Technology Alliance (RCTA) [2]. As a result, high performance image space-based detection and planning methods have been developed that enable reliable real world deployment [3, 4].

However, the same can not be said for legged robot systems. For a legged platform, the morphological affordances are different than wheeled platforms.

* The work of Max Asselmeier is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2039655. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

As a result, the environment representations should be different as well. Despite this, commonly used environment models for legged systems fail to fully exploit the capabilities of the platform. In certain cases, the representation itself can be limiting through its need for expensive processing [5]. In other situations, the representation can fail to account for the unique affordances that legged systems provide such as stepping over small obstacles or jumping [6].

In this study, we explore how to embed the geometric characteristic of the local terrain’s ability to support a stable foothold — a property which we term *steppability* — within the efficient perception space. In order to do so, we adapt prior techniques for generating synthetic data in order to train a semantic segmentation model to predict this steppability property. Then, we integrate this notion of steppability into our search and optimization-based contact planner in the form of a visual search heuristic to enable online, reactive planning.

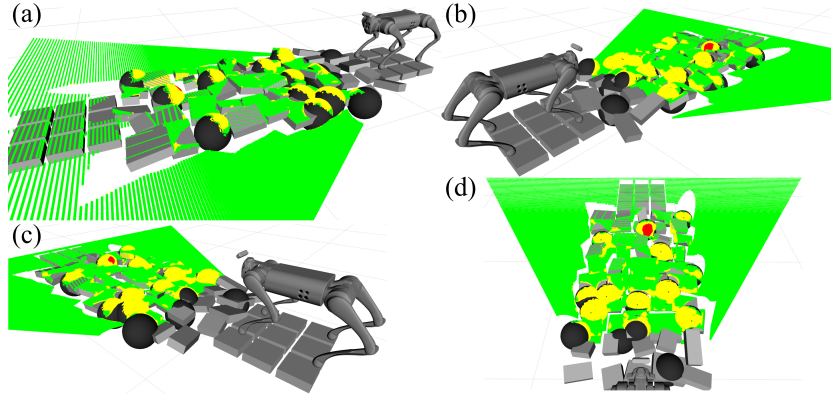


Fig. 1: Visualization of predicted steppability labels overlaid on irregular stepping stones. Green regions are *steppable*, meaning we can plan footholds at that location, yellow regions are *passable*, meaning that we can not plan footholds at that location but can plan swing trajectories over it, and red regions are *non-passable*, meaning that we can not plan footholds at that location and we also can not plan swing trajectories over it. All four images above show the same environment with (a) - (c) providing side views and (d) providing a top view.

In summary, our main contributions are:

1. Introduction of a novel image space-based legged environment representation of steppability
2. Adaptation of a primitive shapes-based synthetic data generation technique to the task of learning steppability
3. Integration of the learned steppability model into an interleaved search and optimization foothold planner
4. Experimental validation of the learned steppability representation through a perception-informed foothold planner in simulation

2 Related Works

2.1 Traversability for legged platforms

For the task of legged locomotion, it is important to capture several aspects of the local environment, one being what much of the literature refers to as traversability. Terrain traversability can be viewed as a continuous score assigned to a subsection of the local environment that reflects the quality of a potential foothold in that location. Traversability is often calculated as a function of terrain properties such as gradients [6, 7], height discontinuities [8], and curvature [9]. Traversability has also been learned through neural networks [5, 10].

This metric is also typically stored in a 2D cost map that is either searched [11, 12] or optimized [13] over when it comes to planning. While such a representation simplifies downstream decision making, converting raw sensor data such as point clouds or depth images into a local Cartesian frame grid requires several pre-processing steps including clustering, projection, and normal estimation which can often require GPU acceleration to run online. Such approaches can also lose environment information regarding overhanging obstacles due to the 2.5D height map representation.

Some approaches do leverage the image space for terrain processing, but purely for the task of terrain class identification [14]. This motivates a deeper investigation into image space-based representations for legged platforms.

2.2 Steppability for legged platforms

While the continuous scoring of terrain is an important metric for footstep planning, the discrete decomposition of the local environment into *steppable* and *non-steppable* regions is also relevant. The level of granularity can differ across approaches, with some just making a binary classification between these two labels [10, 15] while others further decompose the environment into gait- or behavior-specific regions such as jumping [16]. While the label scheme in [16] is similar to the one presented in this proposed work, all methods discussed here maintain a robot-centric 2D grid-based representation. Another common approach is to decompose the environment into a set of polygons [17–20] that represent support regions for footholds. Such a representation fits nicely into optimization frameworks for footstep planning.

The representations shown here play essential roles in impressive locomotion tasks, but the requisite processing and computation can bottleneck the autonomy pipeline and prohibit deployment.

2.3 Planning in Perception Space

A perception and planning paradigm that is of key importance to this proposed work is that of Planning in Perception Space (PiPS) [21]. A perception-space approach to planning eschews pre-processing steps for perception data and instead acts directly on the raw data representations that an external sensor provides.

In the case of laser or Lidar scans, this means keeping the range data in an egocentric reference frame and recasting the task of local perception-informed navigation as a robot-centric decision-making process.

To date, PiPS has provided strong benefits for fast depth space collision-checking [21], collision-free local robot navigation [22–24], and task and motion planning [25, 26]. The PiPS approach has also proven effective for end-to-end learning for footstep planning [27, 28]. However, these methods ultimately eschew any intermediate environment representation which can make planning results harder to interpret and reason about.

While PiPS can facilitate certain local robot processes, the perception space has its drawbacks as well. The rich nearby depth information is offset by sparse environment information at further distances which limits the spatio-temporal horizons in which planning in the perception space can be performed.

3 Methodology

3.1 Dataset Generation


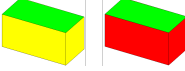
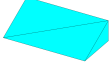
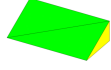



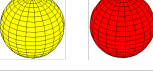

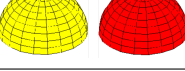








First, we will detail how we generate the simulation scenes that we use to extract the synthetic steppability data. Here, we adapt a primitive shapes-based technique that was previously used for manipulation tasks [29, 30]. In this work, the authors hypothesize that the geometry of graspable objects can be decomposed into a set of primitive shapes where each primitive shape class has a particular family of effective grasps. The ground truth labels of these objects can then be ascertained through the color of the primitives within the simulation scene. We perform a similar process here, but instead utilize class labels concerning steppability. The synthetic scenes used for training data are assembled according to a key set of design parameters that we detail now. Example scene data including depth images, ground truth labels, and model predictions is shown in Figure 3.

Primitive Shape Classes We use nine primitive shape classes to build scenes: *Cuboid*, *Cylinder*, *Ramp*, *Sphere*, *Semisphere*, *Pipe*, *Pole*, *Tube*, and *Floor*. The Cuboid and Ramp classes are parameterized by length l , width w , and height h , the Cylinder class is parameterized by x -dimensional radius r_x , y -dimensional radius r_y , and height h , the Sphere and Hemisphere classes are parameterized by radius r , and the Pipe, Pole, and Tube classes are parameterized by length l and radius r . The floor class is non-parametric in that all of its instances are $4\text{ m} \times 4\text{ m}$. Visualizations of each shape class along with design parameter ranges are shown in Table 1.

Primitive Shape Steppability Policies Each primitive shape class is assigned a mesh-based steppability policy that defines which faces of the shapes should be assigned which labels. The three labels used are:

- **Steppable (Green):** can support a stable foothold

Table 1: Primitive Shape Classes and Visualizations

Primitive Shape Class	Geometry Visualizations	Label Visualizations	Parameter Range (unit: m)
Cuboid			$l \in [0.2, 1.0]$ $w \in [0.1, 0.50]$ $h \in [0.05, 0.25]$
Ramp			$l \in [0.2, 1.0]$ $w \in [0.1, 0.50]$ $h \in [0.05, 0.25]$
Cylinder			$r_x \in [0.10, 0.50]$ $r_y \in [0.10, 0.50]$ $h \in [0.05, 0.25]$
Sphere			$r \in [0.025, 0.05]$
Semisphere			$r \in [0.025, 0.05]$
Pipe			$l \in [0.10, 0.50]$ $r \in [0.025, 0.05]$
Pole			$l \in [0.10, 0.50]$ $r \in [0.025, 0.05]$
Tube			$l \in [0.50, 1.0]$ $r \in [0.025, 0.05]$
Floor			N/A

- **Passable (Yellow):** can not support a stable foothold, but can be stepped over by a foot swing trajectory
- **Non-passable (Red):** can not support a stable foothold and can not be stepped over by a foot swing trajectory

For Cuboids, Ramps, and Cylinders, the top face is labeled as steppable due to its flat horizontal geometry. If the height of the primitive exceeds a maximum swing height for the robot leg, in this work defined as $h_{\max} = 0.10$ m, then all vertical faces are labeled as non-passable. Otherwise, the vertical faces are labeled as passable. For Spheres and Semispheres, the entire primitive is labeled as non-passable if the diameter and radius respectively exceed h_{\max} . Otherwise, the entire primitive is labeled as passable. For Pipes, Poles, and Tubes, the entire primitive is labeled as non-passable if the z -dimension of its pose exceeds h_{\max} . Otherwise, the entire primitive is labeled as passable. Lastly, floors are labeled as completely steppable.

Primitive Shape Pose The six-dimensional pose of each primitive shape can be set according to desired scene attributes. For instance, scenes can be set to feature clusters of primitive shapes localized within a particular region of the scene, the primitives can be scattered all throughout the scene, or the poses can be overwritten to accept manually defined entries if the user wants to create more contrived scenes that include structures such as staircases or stepping stones. The z -dimension of all shapes is restricted so that all shapes are placed on support surfaces, and the orientations of Cuboids, Ramps, and Cylinders are restricted to ensure that the face labeled as steppable remains as the top face in the scene.

Camera pose The six-dimensional pose of the camera placed within each simulation scene can also be parameterized to emulate expected real-world circumstances for onboard sensing. Given that our desired application is quadrupedal locomotion, we set the camera at a height of $z = 0.325$ m and pitch it downwards by 30° to approximate the pose of the depth camera attached to our quadrupedal hardware platform. To capture multiple frames of a single scene, the camera is set to follow a prescribed trajectory that approximates how a quadruped’s torso would move through a real world environment. Gaussian noise is also applied to all six pose dimensions to capture the jitter that the camera would experience during locomotion in the real world.

Scene Environment Lastly, the overall synthetic environment that the primitive shapes are placed within can also be controlled. In this work, we randomly select between indoor environments that include walls and a ceiling and outdoor environments that instead include an infinite horizon.

3.2 Model Training

For model training, we use the off-the-shelf DeepLabV3+ [31] model from the Detectron2 deep learning library [32]. To construct the dataset, 600 scenes were generated with 5 frames each, making for a total of 3,000 images. In total, dataset generation took roughly 12 hours. The generated data was put into 80%/10%/10% splits of 2,400, 300, and 300 images for training, validation, and test subsets respectively. Model training took 65 minutes on an Intel Xeon W-2223 CPU at 3.60GHz with an Nvidia T1000 GPU. Plots of training loss and intersection-over-union can be seen in Figure 2.

3.3 Steppability-informed Contact Planning

Now, we detail how the proposed steppability model can be used to inform footstep planning. We provide a brief overview of our existing contact planner here, and more details can be found at [33].

We incorporate this steppability model into our existing interleaved graph search and trajectory optimization-based footstep planner. To plan a discrete sequence of footholds, we perform a search over a mode transition graph $\mathcal{G} =$

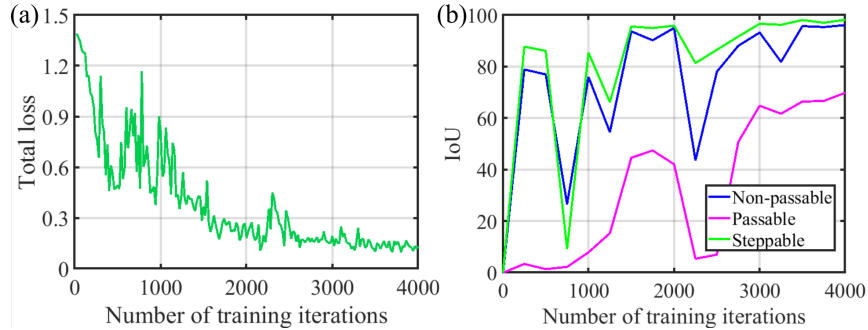


Fig. 2: Results of training. (a) Total training loss over the training iterations. (b) Intersection-over-union (IoU) of all classes over the training iterations.

(\mathcal{V}, \mathcal{E}) [34]. Each vertex $v \in \mathcal{V}$ represents a partial stance in which a proper subset of the quadruped’s feet are in contact with a unique combination of steppable objects in the environment. Each vertex v represents a mode family Ξ . Within each mode family, there is an infinite set of modes $\xi \in \Xi$ where each mode ξ represents a particular set of positions along the steppable objects where contact is made. A set of continuously varying coparameters χ parameterize these contact positions.

Each edge $e \in \mathcal{E}$ represents a transition between two partial stances which itself is a full stance in which all feet are in contact. For a transition between source mode $\xi_i = \langle \Xi_i, \chi_i \rangle$ and destination mode $\xi_{i+1} = \langle \Xi_{i+1}, \chi_{i+1} \rangle$, the graph edge $e = (\xi_i, \xi_{i+1})$ is assigned the weight

$$\begin{aligned}
 \Delta c(\xi_i, \xi_{i+1}) = & \\
 & \mathcal{D}^{\Xi_i, \Xi_{i+1}}(\chi_i, \chi_{i+1}) + \\
 & d_{\text{CoM}}(\xi_i, \xi_{i+1}) + \\
 & d_{\tau}(\xi_i, \xi_{i+1}) + \\
 & d_{\text{step}}(\xi_i, \xi_{i+1})
 \end{aligned} \tag{1}$$

where the distribution $\mathcal{D}^{\Xi_i, \Xi_{i+1}}(\chi_i, \chi_{i+1})$ captures the kinodynamically-aware cost of transitioning from ξ_i to ξ_{i+1} , $d_{\text{CoM}}(\xi_i, \xi_{i+1})$ is the Euclidean distance between nominal CoM positions for ξ_i and ξ_{i+1} , $d_{\tau}(\xi_i, \xi_{i+1})$ is the deviation of nominal CoM positions for ξ_i and ξ_{i+1} from a guiding torso path, and $d_{\text{step}}(\xi_i, \xi_{i+1})$ is a proposed steppability-informed weight. All terms are weighted by positive scalars to assign relative importance.

This graph search returns a discrete sequence of footholds that are then passed to a whole body trajectory optimization (TO) program to generate a full trajectory. The optimal cost values obtained from this TO program are then used to update the experience-based distribution $\mathcal{D}^{\Xi_i, \Xi_{i+1}}(\chi_i, \chi_{i+1})$ which is obtained offline.

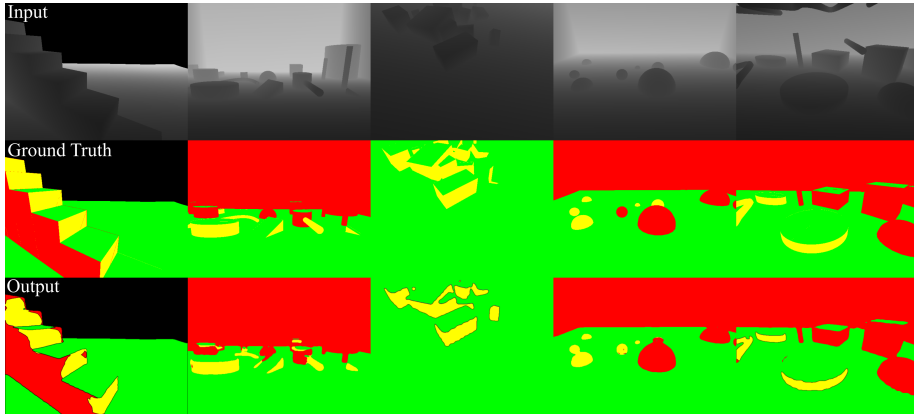


Fig. 3: Example outputs of learned steppability model. Top row: input depth images to the model. Middle row: ground truth steppability labels of the corresponding column’s input depth image. Bottom row: model outputs for the corresponding column’s input depth image.

3.4 Steppability heuristic

The mode transition graph is constructed under the assumption that we have access to the poses of the objects in the environment that we want to plan footholds on, but we assume that the poses of obstacles are unknown. We then rely on this perception-informed steppability term for reactive obstacle avoidance.

Prior to triggering the mode transition graph search, the learned model is queried to obtain the current steppability mask $\mathcal{I}_{\text{step}}$. This mask contains steppability labels for the current view of the environment. Then, during the graph search, when a new edge is visited, the stance foothold positions of the edge’s transition can be projected into the steppability mask to ascertain information regarding the quality of the candidate foothold positions.

From the graph edge e , we extract the world frame positions $\mathbf{p}_c^l \in \mathbb{R}^3 \quad \forall l \in [1, 4]$ of the center of each of the four transition footholds. To determine the correct pixel to query for its steppability label, we then perform pinhole camera projection

$$\begin{bmatrix} x_c^l \\ y_c^l \\ w \end{bmatrix} = \mathbf{K} \cdot [\mathbf{R}_{CW}^t \quad \mathbf{t}_{CW}^t] \cdot \begin{bmatrix} \mathbf{p}_c^l \\ 1 \end{bmatrix}, \quad (2)$$

where $[x_c^l \ y_c^l \ w]^T$ are homogeneous image coordinates, $\mathbf{K} \in \mathbb{R}^{3 \times 3}$ is the intrinsic camera calibration matrix, and $\mathbf{R}_{CW}^t \in \mathbb{R}^{3 \times 3}$, $\mathbf{t}_{CW}^t \in \mathbb{R}^3$ are the rotation and translation from the world frame to the camera frame at time t . Finally, the homogeneous image coordinates of the candidate footholds are obtained through the simple conversion

$$\mathbf{x}_c^l := \begin{bmatrix} x_c^l/w \\ y_c^l/w \end{bmatrix}. \quad (3)$$

Based on the steppability label returned for the foothold pixel \mathbf{x}_c^l , we assign a weight of

$$d_{\text{step}}^l(\xi_i, \xi_{i+1}) = \begin{cases} 1000, & \text{if } \mathcal{I}_{\text{step}}(\mathbf{x}_c^l) \rightarrow \text{non-passable} \\ 100, & \text{if } \mathcal{I}_{\text{step}}(\mathbf{x}_c^l) \rightarrow \text{passable} \\ 5, & \text{if } \mathbf{x}_c^l \text{ not in frame} \\ 1, & \text{if } \mathcal{I}_{\text{step}}(\mathbf{x}_c^l) \rightarrow \text{steppable} \end{cases}. \quad (4)$$

Given that we do not wish to plan footholds on such regions, we could discard any footholds with non-passable or passable labels. However, this could potentially eliminate feasible footholds that are simply obscured by passable or non-passable features in the environment. Therefore, we opt to administer higher penalties for such labels but still allow for these footholds to be selected in the event that they are essential to reaching the goal. Similarly, we allow for footholds to be planned outside of the current camera view.

Lastly, the total steppability weighting term is taken as the sum across all footholds

$$d_{\text{step}}(\xi_i, \xi_{i+1}) = \sum_l d_{\text{step}}^l(\xi_i, \xi_{i+1}). \quad (5)$$

To account for the finite size of the quadruped’s feet, we perform an inflation step modeled off of [14] where we additionally check the steppability labels of world frame positions that are offset from \mathbf{p}_c^l in the x - and y -direction by the radius of the foot and add these to the total steppability weight.

3.5 Trajectory Optimization

Our lower-level TO problem solves over the robot state $\mathbf{x} = [\mathbf{h}, \mathbf{q}_b, \mathbf{q}_j]$, the robot input $\mathbf{u} = [\mathbf{f}, \mathbf{v}_j]$, and the coparameters $\boldsymbol{\chi}_{i+1}$ of the destination mode family. The TO formulation for generating a whole-body trajectory to transition from $\xi_i = \langle \Xi_i, \boldsymbol{\chi}_i \rangle$ to $\xi_{i+1} = \langle \Xi_{i+1}, \boldsymbol{\chi}_{i+1} \rangle$ is written as:

$$\min_{\mathbf{x}, \mathbf{u}, \boldsymbol{\chi}_{i+1}} \|\mathbf{x}[N] - \mathbf{x}^{\text{des}}[N]\|_{Q_f}^2 + \sum_{k=0}^{N-1} \left(\|\mathbf{x}[k] - \mathbf{x}^{\text{des}}[k]\|_Q^2 + \|\mathbf{u}[k]\|_R^2 \right)$$

subject to

$$\text{(Mode } i) \quad F^{\xi_i}(\mathbf{q}[k]) = \mathbf{0} \quad (6a)$$

$$\text{(Mode } i + 1) \quad F^{\xi_{i+1}}(\mathbf{q}[N], \boldsymbol{\chi}_{i+1}) = \mathbf{0} \quad (6b)$$

$$\text{(Dynamics)} \quad \dot{\mathbf{x}}[k] = f_{\text{cent}}(\mathbf{x}[k], \mathbf{u}[k]) \quad (6c)$$

$$\text{(Friction)} \quad \mathbf{f}_l[k] \in \mathcal{F}_l(\mu, \mathbf{q}) \quad \forall l \in \mathcal{C}_i \quad (6d)$$

$$\mathbf{f}_l[k] = \mathbf{0} \quad \forall l \notin \mathcal{C}_i \quad (6e)$$

$$\text{(Collision)} \quad g(\mathbf{q}[k]) \geq 0 \quad \forall k \in [0, N] \quad (6f)$$

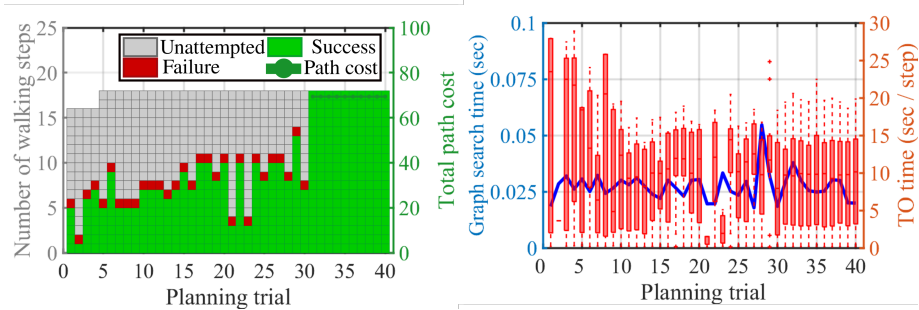
where f_{cent} are the centroidal dynamics described in [33] and [35], $\mathcal{F}_l(\mu, \mathbf{q})$ represents the friction cone which depends on the friction coefficient μ and robot pose \mathbf{q} , and \mathcal{C}_i represents the set of stance feet for ξ_i .

4 Experimental Results

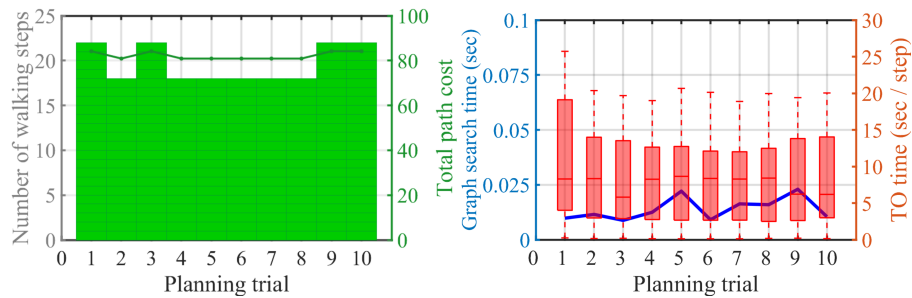
4.1 Offline experience accumulation for irregular terrain traversal

For our first experiment, we demonstrate the benefits of integrating the proposed steppability-based search heuristic into our contact planner. To do so, we perform a series of offline planning trials both with and without the steppability term active in the graph search. Each individual planning trial is comprised of a single graph search over candidate footholds and a sequence of trajectory optimization subproblems aimed to synthesize swing trajectories between footholds. If an intermediate subproblem fails, then the trial is terminated early, remaining subproblems are left unattempted, and the experience-based cost distributions along the traversed graph edges are updated.

Offline trials are run in the environment shown in Figures 1 and 5, characterized by irregularly posed planar regions along with spherical obstacles which we aim to avoid planning footsteps on while attempting to reach the desired goal region (shown as a transparent green sphere in Figure 5).



(a) Results for experience accumulation without steppability heuristic.



(b) Results for experience accumulation with steppability heuristic.

Fig. 4: Offline planning results for Section 4.1. On the left side, results of all planning trials run (a) without and (b) with the steppability heuristic active in the graph search. On the right side, graph search and trajectory optimization solve times for each trial are shown.

Results for the offline trials are shown in Figure 4. Left plots showcase the results of intermediate trajectory optimization subproblems along with total path costs for the planning trials that successfully reach the goal region. Right plots showcase the computation times of both the graph searches and trajectory optimization subproblems associated with each trial.

Without the steppability term (Figure 4a), the planner must ascertain the obstacle locations solely through the experience heuristic. This can be viewed as a form of proprioceptive sensing where the graph search only comes to avoid the obstacles through the high costs of the offending trajectories informing the graph’s corresponding edge weights. In this setup, the planning framework requires 30 offline planning trials to discover successful contact sequences.

With the steppability term (Figure 4b), the planner has a weighting term on the graph edges that can act as a form of exteroceptive sensing. Based on the current view of the environment, this steppability term guides the graph search away from footholds that overlap with obstacles. The impact of this term can be seen in that the planner can immediately identify successful contact sequences, requiring no offline planning in order to successfully reach the goal. This indicates that the steppability term greatly expedites offline experience accumulation process by providing a form of proactive foot-level collision avoidance at no cost to graph search timing.

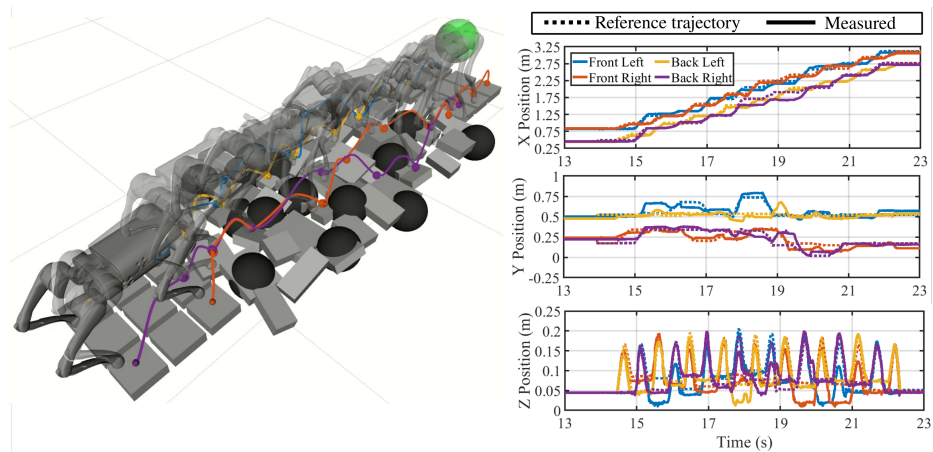


Fig. 5: Online tracking performance of offline reference trajectory in simulation.

While graph search times across all planning trials were minimal, never exceeding a tenth of a second, the solve times for the swing trajectory subproblems remain prohibitive, capable of taking more than 20 seconds for a single contact transition. These significant solve times are largely in part due to the highly non-linear collision avoidance constraint. Only obstacles sufficiently close to the robot

are activated in this constraint for any given subproblem, leading to maximal solve times in more dense environments, as is to be expected.

Lastly, Figure 5 showcases the online tracking performance of the reference trajectory received from the contact planner. For this task, we use a model predictive control (MPC)-based whole body controller (WBC) modeled off of [36]. Overall, the tracking performance in simulation is suitable for online deployment, and the reference trajectory successfully guides the platform the other side of the stepping stones layout. The Z Position figure includes moments where the robot’s end effectors slip off of the intended planar regions, evidenced by their Z-coordinate dropping below 0.05 m. However, these slips were not catastrophic and the trajectory tracking was still successful.

4.2 Regular terrain traversal within a reactive navigation framework

While the environment in Section 4.1 that the proposed contact planning framework is deployed in might be small, the depth images used as inputs to our steppability model are only capable of providing dense depth information within a range of 1 – 3m. Therefore, we do not want to rely on this perception data over large spatio-temporal scales in the first place. This can be observed in Figure 1a where the distance between consecutive points grows as the points get further away from the robot and camera. Additionally, Section 4.1 demonstrates that the lengthy trajectory optimization solve times prevent us from running our planner in a purely online fashion, and these computational trends only worsen as the size of the environment in which planning is performed increases. This motivates the utilization of a reactive hierarchical navigation framework in which planning is frequently performed on the constantly updating local environment. In this section, we detail the integration of our contact planner into a navigation framework as previously described and experimentally demonstrate the benefits of planning within such a framework.

We utilize an event-based navigation framework in which certain environmental triggers, namely when the robot has reached the boundary of its previously perceived local environment, send a re-plan request to our contact planner. An example of this process is visualized in Figure 6 where we perform planning over regular terrain, exemplified by a flight of stairs. In frame (a), the initial planning request is sent to the contact planner and a whole body trajectory is synthesized within the local portion of the environment. Note that the global goal region, depicted as a transparent green sphere, lies beyond the local environment in frame (a), necessitating future re-planning to reach it. Frames (a)-(f) show the online tracking of the initially generated trajectory. At frame (g), tracking has been completed and more of the environment has been revealed, including a tall pillar that must be avoided. A re-planning request is triggered and frames (g)-(i) show the online tracking of the new trajectory, allowing the robot to reach the global goal region.

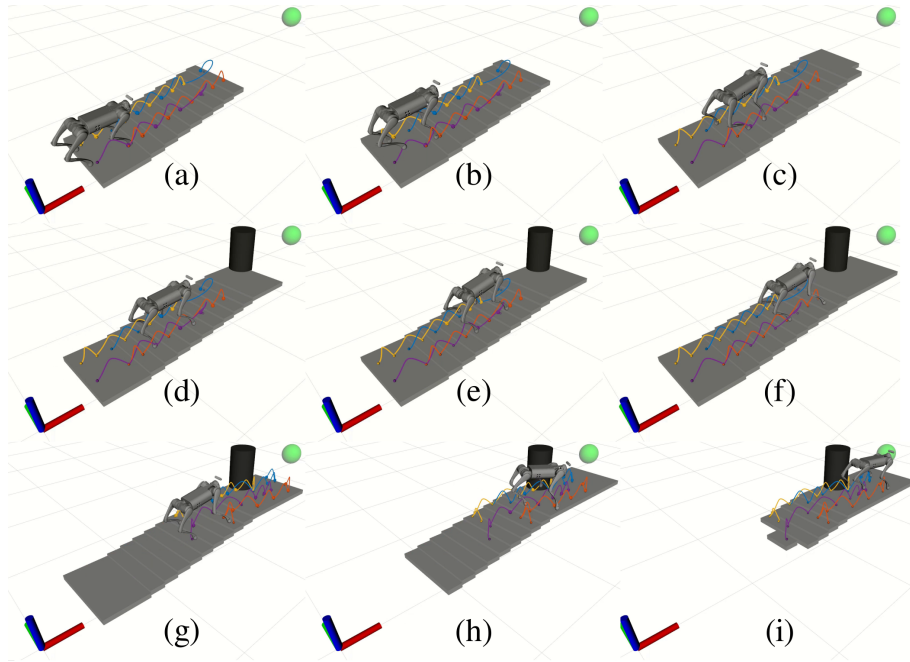


Fig. 6: Contact planning performed over regular terrain (stairs) within an online navigation framework. An inertial coordinate frame is shown at the bottom left of each image, and the global goal region is depicted as a transparent green sphere. During the tracking of the initially generated trajectory, shown in frames (a)-(f), the remaining portion of the environment is revealed to the planner, allowing it to re-plan at frame (g) in order to reach the global goal region.

4.3 Sloped terrain traversal through disturbance-based re-planning

Section 4.2 showcased the ability to re-plan in the event of successfully planning through the robot’s local environment. In this section, we introduce another key event that requires re-planning, online disturbance recovery, and demonstrate how our novel steppability heuristic facilitates in the detection of these disturbances.

In frames (a) and (b) of Figure 7, we perform planning in a sloped local environment. However, in frame (c), during trajectory tracking, we spawn a new obstacle in the path of the robot. Given that our contact planner can not perform fully online navigation, it is critical that we have a means to validate the trajectory that is currently being tracked against the potentially changing local environment. During trajectory tracking, we project upcoming footholds into the current steppability mask to check if the foothold is still steppable. If an upcoming foothold is deemed not to be steppable, a re-planning request is triggered. Frame (c) shows that when this new obstacle is spawned, the current steppability mask labels the region is non-passable, and the previously planned

foothold that now overlaps with this obstacle is deemed infeasible. Trajectory tracking is subsequently stopped at frame (d) to ensure that the robot does not collide with the new obstacle, and a new trajectory is generated at frame (e) that accounts for the new obstacle. Then, this new trajectory is tracked during frames (e)-(i), allowing the robot to finally reach the global goal region.

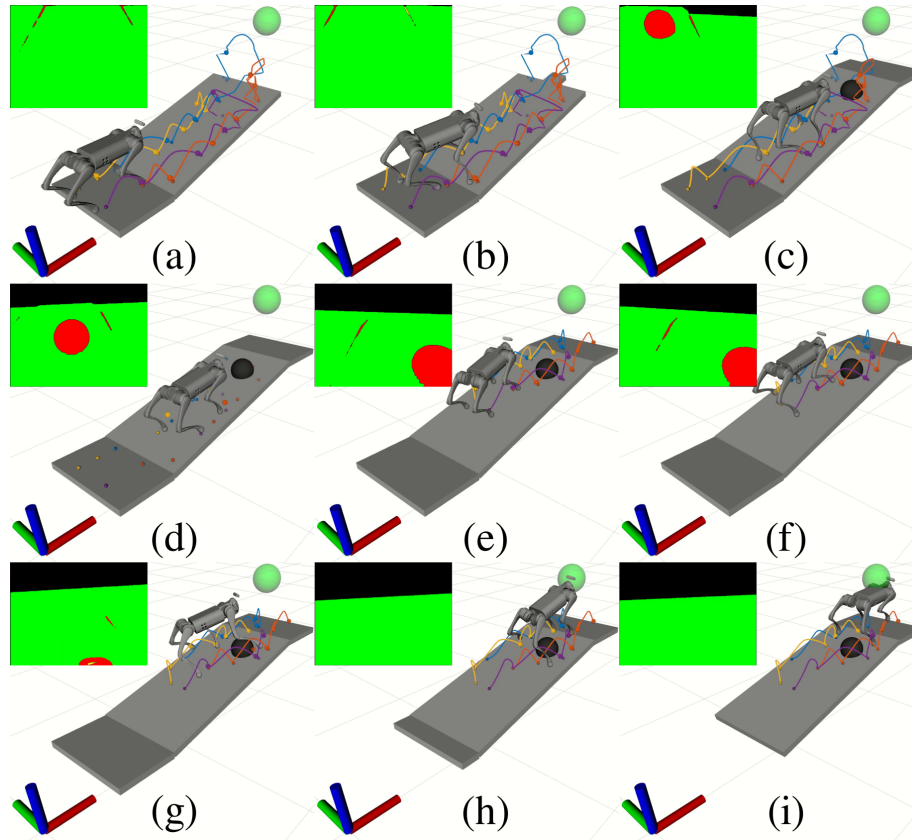


Fig. 7: Contact planning performed in the presence of an online disturbance in the form of a newly spawned obstacle. An inertial coordinate frame is shown at the bottom left of each image, the accompanying stepability mask at each frame is shown at the top left of each image, and the global goal region is depicted as a transparent green sphere. When the new obstacle is spawned in frame (c), the stepability mask identifies it and labels it as non-passable, triggering a re-planning request in response to the changed environment.

5 Conclusion

In this work, we introduce a novel image space-based method of predicting the steppability properties of the local environment for footstep planning. We adapt primitive shapes-based techniques from the domain of dexterous manipulation in order to efficiently generate diverse synthetic data for training a semantic segmentation model to perform steppability prediction, and we deploy this steppability model in our existing interleaved search and optimization contact planner in the form of a deep visual search heuristic. Through offline planning trials and online reactive navigation, we demonstrate how this steppability heuristic allows for expedited offline experience accumulation, proactive collision avoidance, and reactive recovery from disturbances.

In the future, we aim to validate the performance of our perception-informed contact planner on an integrated real-world quadrupedal system. To do so, we plan on generating synthetic data from higher fidelity simulation environments and bridging the sim-to-real gap by incorporating a bi-directional domain alignment process that merges simulation and real world data distributions by both corrupting simulation data and denoising real world data. We also seek to benchmark our steppability model against classical point cloud processing-based representations to better understand the speed and accuracy trade-off that our learned steppability model provides.

References

1. L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan, “The DARPA LAGR program: Goals, challenges, methodology, and phase I results,” *Journal of Field Robotics*, vol. 23, no. 11-12, pp. 945–973, 2006.
2. A. Rankin, M. Bajracharya, A. Huertas, A. Howard, B. Moghaddam, S. Brennan, A. Ansar, B. Tang, M. Turmon, and L. Matthies, “Stereo-vision-based perception capabilities developed during the Robotics Collaborative Technology Alliances program,” G. R. Gerhart, D. W. Gage, and C. M. Shoemaker, Eds., Orlando, Florida, Apr. 2010.
3. J. Seo, S. Sim, and I. Shim, “Learning Off-Road Terrain Traversability With Self-Supervisions Only,” *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4617–4624, Aug. 2023.
4. G. Vecchio, S. Palazzo, D. C. Guastella, D. Giordano, G. Muscato, and C. Spampinato, “Terrain traversability prediction through self-supervised learning and unsupervised domain adaptation on synthetic data,” *Autonomous Robots*, vol. 48, no. 2, p. 4, Mar. 2024.
5. T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, “Elevation Mapping for Locomotion and Navigation using GPU,” Apr. 2022.
6. A. Dixit, D. D. Fan, K. Otsu, S. Dey, A.-A. Agha-Mohammadi, and J. W. Burdick, “STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation; Results from the DARPA Subterranean Challenge,” *Field Robotics*, vol. 4, no. 1, pp. 182–210, Jan. 2024.
7. I. D. Miller, F. Cladera, A. Cowley, S. S. Shivakumar, E. S. Lee, L. Jarin-Lipschitz, A. Bhat, N. Rodrigues, A. Zhou, A. Cohen, A. Kulkarni, J. Laney, C. J. Taylor,

- and V. Kumar, “Mine Tunnel Exploration using Multiple Quadrupedal Robots,” Feb. 2020.
8. K. Stepanas, J. Williams, E. Hernández, F. Ruetz, and T. Hines, “OHM: GPU Based Occupancy Map Generation,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 078–11 085, Oct. 2022.
 9. H. Biggie, E. R. Rush, D. G. Riley, S. Ahmad, M. T. Ohradzansky, K. Harlow, M. J. Miles, D. Torres, S. McGuire, E. W. Frew, C. Heckman, and J. S. Humbert, “Flexible Supervised Autonomy for Exploration in Subterranean Environments,” *Field Robotics*, vol. 3, no. 1, pp. 125–189, Jan. 2023.
 10. L. Wellhausen and M. Hutter, “Rough Terrain Navigation for Legged Robots using Reachability Planning and Template Learning,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2021.
 11. A. Agha, K. Otsu, B. Morrell, D. D. Fan, R. Thakker, A. Santamaria-Navarro, S.-K. Kim, A. Bouman, X. Lei, J. Edlund, M. F. Ginting, K. Ebadi, M. Anderson, T. Pailevanian, E. Terry, M. Wolf, A. Tagliabue, T. S. Vaquero, M. Palieri, S. Tepsuporn, Y. Chang, A. Kalantari, F. Chavez, B. Lopez, N. Funabiki, G. Miles, T. Touma, A. Buscicchio, J. Tordesillas, N. Alatur, J. Nash, W. Walsh, S. Jung, H. Lee, C. Kanellakis, J. Mayo, S. Harper, M. Kaufmann, A. Dixit, G. Correa, C. Lee, J. Gao, G. Merewether, J. Maldonado-Contreras, G. Salhotra, M. S. Da Silva, B. Ramtoula, Y. Kubo, S. Fakoorian, A. Hatteland, T. Kim, T. Bartlett, A. Stephens, L. Kim, C. Bergh, E. Heiden, T. Lew, A. Cauligi, T. Heywood, A. Kramer, H. A. Leopold, C. Choi, S. Daftry, O. Toupet, I. Wee, A. Thakur, M. Feras, G. Beltrame, G. Nikolakopoulos, D. Shim, L. Carlone, and J. Burdick, “NeBula: Quest for Robotic Autonomy in Challenging Environments; TEAM CoSTAR at the DARPA Subterranean Challenge,” Oct. 2021, arXiv:2103.11470.
 12. N. Kottege, J. Williams, B. Tidd, F. Talbot, R. Steindl, M. Cox, D. Frousheger, T. Hines, A. Pitt, B. Tam, B. Wood, L. Hanson, K. L. Surdo, T. Molnar, M. Wildie, K. Stepanas, G. Catt, L. Tychsen-Smith, D. Penfold, L. Overs, M. Ramezani, K. Khosoussi, F. Kendoul, G. Wagner, D. Palmer, J. Manderson, C. Medek, M. O’Brien, S. Chen, and R. C. Arkin, “Heterogeneous robot teams with unified perception and autonomy: How Team CSIRO Data61 tied for the top score at the DARPA Subterranean Challenge,” Feb. 2023.
 13. F. Jenelten, T. Miki, A. E. Vijayan, M. Bjelonic, and M. Hutter, “Perceptive Locomotion in Rough Terrain – Online Foothold Optimization,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5370–5376, Oct. 2020.
 14. L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, “Where Should I Walk? Predicting Terrain Properties From Images Via Self-Supervised Learning,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1509–1516, Apr. 2019.
 15. P. Karkowski and M. Bennewitz, “Prediction Maps for Real-Time 3D Footstep Planning in Dynamic Environments,” in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 2517–2523.
 16. D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim, “Vision Aided Dynamic Exploration of Unstructured Terrain with a Small-Scale Quadruped Robot,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 2464–2470.
 17. R. J. Griffin, G. Wiedebach, S. McCrory, S. Bertrand, I. Lee, and J. Pratt, “Footstep Planning for Autonomous Walking Over Rough Terrain,” in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, Oct. 2019, pp. 9–16.

18. S. Bertrand, I. Lee, B. Mishra, D. Calvert, J. Pratt, and R. Griffin, “Detecting Usable Planar Regions for Legged Robot Locomotion,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA: IEEE, Oct. 2020, pp. 4736–4742.
19. R. Buchanan, L. Wellhausen, M. Bjelonic, T. Bandyopadhyay, N. Kottege, and M. Hutter, “Perceptive whole-body planning for multilegged robots in confined spaces,” *Journal of Field Robotics*, vol. 38, no. 1, pp. 68–84, 2021.
20. R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, “Perceptive Locomotion through Nonlinear Model Predictive Control,” in *arXiv*, Aug. 2022.
21. J. S. Smith and P. Vela, “PiPS: Planning in perception space,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 6204–6209.
22. R. Xu, S. Feng, and P. A. Vela, “Potential Gap: A Gap-Informed Reactive Policy for Safe Hierarchical Navigation,” *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8325–8332, 2021.
23. J. S. Smith, S. Feng, F. Lyu, and P. A. Vela, “Real-Time Egocentric Navigation Using 3D Sensing,” in *Machine Vision and Navigation*, 2020, pp. 431–484.
24. J. S. Smith, R. Xu, and P. Vela, “egoTEB: Egocentric, Perception Space Navigation Using Timed-Elastic-Bands,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 2703–2709.
25. D. Driess, J.-S. Ha, and M. Toussaint, “Deep Visual Reasoning: Learning to Predict Action Sequences for Task and Motion Planning from an Initial Scene Image,” in *Robotics: Science and Systems XVI*. Robotics: Science and Systems Foundation, Jul. 2020.
26. D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint, “Deep Visual Heuristics: Learning Feasibility of Mixed-Integer Programs for Manipulation Planning,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 9563–9569.
27. M. Sorokin, J. Tan, C. K. Liu, and S. Ha, “Learning to Navigate Sidewalks in Outdoor Environments,” Sep. 2021.
28. A. Agarwal, A. Kumar, J. Malik, and D. Pathak, “Legged Locomotion in Challenging Terrains using Egocentric Vision,” Sep. 2022.
29. M. Nieuwenhuisen, J. Stueckler, A. Berner, R. Klein, and S. Behnke, “Shape-Primitive Based Object Recognition and Grasping,” in *ROBOTIK 2012; 7th German Conference on Robotics*, May 2012, pp. 1–5.
30. Y. Lin, C. Tang, F.-J. Chu, and P. A. Vela, “Using synthetic data and deep networks to recognize primitive shapes for object grasping,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 494–10 501.
31. L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation,” Aug. 2018.
32. Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick, “Detectron2,” 2019.
33. M. Asselmeier, J. Ivanova, Z. Zhou, P. A. Vela, and Y. Zhao, “Hierarchical Experience-informed Navigation for Multi-modal Quadrupedal Rebar Grid Traversal,” *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.
34. Z. Kingston and L. E. Kavraki, “Scaling Multimodal Planning: Using Experience and Informing Discrete Search,” *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 128–146, Feb. 2023.

35. D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous Robots*, vol. 35, no. 2, pp. 161–176, Oct. 2013.
36. J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A Unified MPC Framework for Whole-Body Dynamic Locomotion and Manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4688–4695, Jul. 2021.