Bipedal Walking on Constrained Footholds with MPC Footstep Control

Brian Acosta and Michael Posa
University of Pennsylvania GRASP Lab

***Abstract*— Bipedal robots promise the ability to traverse rough terrain quickly and efficiently, and indeed, humanoid robots can now use strong ankles and careful foot placement to traverse discontinuous terrain. However, more agile underactuated bipeds have small feet and weak ankles, and must constantly adjust their planned footstep position to maintain balance. We introduce a new model-predictive footstep controller which jointly optimizes over the robot's discrete choice of stepping surface, impending footstep position sequence, ankle torque in the sagittal plane, and center of mass trajectory, to track a velocity command. The controller is formulated as a single Mixed Integer Quadratic Program (MIQP) which is solved at 50-200 Hz, depending on terrain complexity. We implement a state of the art real-time elevation mapping and convex terrain decomposition framework to inform the controller of its surroundings in the form on convex polygons representing steppable terrain. We investigate the capabilities and challenges of our approach through hardware experiments on the underactuated biped Cassie.**

I. Introduction

While the ability to traverse unstructured terrain is a key motivation for bipedal robots, navigating these environments is an open problem. Humanoid robots can walk semi-autonomously on discontinuous terrains such as cinder-block piles, but rely on careful foot placement and decoupling footstep planning and balance control [1], [2]. In contrast, underactuated bipeds have limited ankle torque and small feet, allowing for efficient, agile motion but limiting horizontal center of mass (CoM) actuation. Therefore, footsteps must be continuously re-planned to maintain balance in light of disturbances and model error.

A simple yet powerful framework for online replanning of stabilizing footstep sequences is to use the step-to-step dynamics of the linear inverted pendulum model (LIP) [3] to synthesize MPC or LQR footstep controllers. This approach regulates walking speed without ankle torque by using foot placement to affect the initial conditions of each continuous single stance phase. Combined with output tracking via inverse-dynamics based whole body torque controllers, this approach has enabled dynamic and robust walking [4]. The Angular Momentum Linear Inverted Pendulum (ALIP) model, in particular, has been shown to accurately describe the bulk motion of walking even for robots with heavy legs [5], and has been used to stabilize walking on sloped terrain [6], synthesize specialized stair climbing controllers [7], and walk on pre-selected constrained footholds [8]. We extend this framework to rough terrain by modeling valid footholds as convex planar polygons, and enforcing via a mixed-integer formulation that each planned footstep lie in a valid foothold.

We transcribe our controller as a Mixed Integer Quadratic Program (MIQP). MIQPs are used extensively in robot motion planning, but they can be difficult to solve at high rates due combinatorial complexity in the planning horizon. For this reason, existing MIQP footstep planners run at 1-5 Hz [9], limiting their applicability to underactuated walking. In contrast, our controller achieves an average solve time of less than 20 ms even in challenging scenarios by using a low dimensional, linear dynamics model, planning over a short footstep horizon, and heuristically pruning foothold candidates.

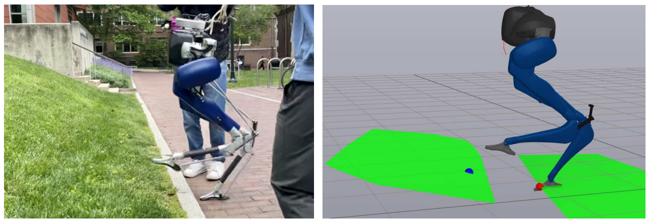
Fig. 1. We present a model predictive foot placement controller for underactuated bipedal walking with foothold constraints. Left: Cassie stepping over a curb onto a grassy hill using the proposed controller and a real-time terrain segmentation pipeline. Right: visualization of the terrain segmentation and footstep plan during the step-up.

We use a real-time elevation mapping and terrain decomposition pipeline [10] to represent the terrain as convex polygons online. As the components of this pipeline have been applied primarily to quadrupeds [11] [12], we discuss modifications needed for deployment on Cassie, and how certain design choices and properties of the perception stack affect the performance of the MIQP footstep controller. An overview of our perception and control stack can be seen in Fig. 2.

The key contributions of this paper are:

- An MPC style footstep planner which reasons over discrete foothold selection, footstep sequence, center of mass trajectory, and ankle torque, formulated as a single MIQP which can be solved at up to 200 Hz to stabilize underactuated walking on discontinuous terrain
- We extend an existing approach to vision-based real-time elevation mapping and terrain segmentation to be more robust to challenges inherent to underactuated bipedal walking. We introduce a simple algorithm which uses approximate convex decomposition [13] to find a convex polygon decomposition of the steppable terrain.
- Evaluation of the proposed controller on hardware as a full, vision integrated system. We demonstrate perceiving and stepping over curbs in real time, and discuss how perception accuracy limits the robustness of the controller.

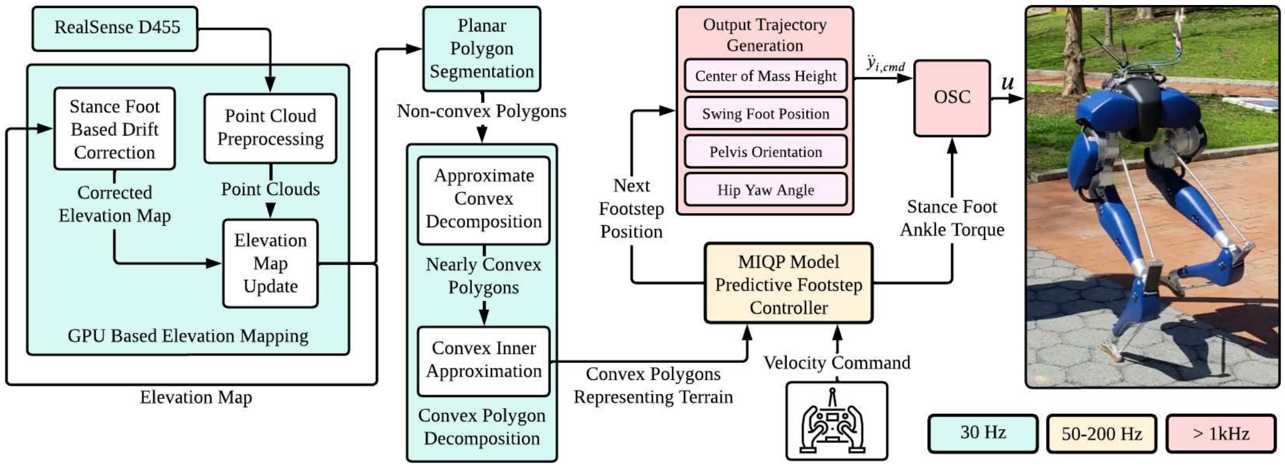
RealSense D455
Stance Foot Based Drift Correction
Point Cloud Preprocessing
Corrected Elevation Map
Point Clouds
Elevation Map Update
GPU Based Elevation Mapping
Elevation Map
Planar Polygon Segmentation
Non-convex Polygons
Approximate Convex Decomposition
Nearly Convex Polygons
Convex Inner Approximation
Convex Polygon Decomposition
Convex Polygons Representing Terrain
Output Trajectory Generation
Center of Mass Height
Swing Foot Position
Pelvis Orientation
Hip Yaw Angle
Next Footstep Position
$\ddot{y}_{i,cmd}$
OSC
u
Stance Foot Ankle Torque
MIQP Model Predictive Footstep Controller
Velocity Command
30 Hz
50-200 Hz
> 1kHz

Fig. 2. Block diagram of the control stack used to achieve perceptive locomotion on unstructured terrain. A real-time perception pipeline provides convex stepping stone constraints to our proposed MIQP footstep controller, which updates the commanded foot position at 50-200 Hz to maintain balance. This stack allows for foot placement control of underactuated bipeds in previously unseen environments by sensing the terrain on the fly and providing a terrain representation to the walking controller.

II. Related Work

A. MIQP Footstep Planning

Deits and Tedrake introduced the use of MIQPs for footstep planning in [14] by decomposing safe terrain into a collection of convex polygons, and using integer variables to assign every footstep to a polygon. Tonneau et al. [15] provide a convex approximation of this problem as a linear program, and Song et al. [16] show how both the mixed integer and linear programming formulations can be made more efficient by using a simplified trajectory planner to prune irrelevant footholds. In contrast to our work, these works only consider geometric and quasistatic stability criteria, and focus on long horizon footstep planning which may take seconds to solve.

MIQP footstep planning has also been used for quadruped robots. In [17], Risbourg et al. use the convex relaxation from [15] online to project the desired footstep sequence to the closest convex footholds, subject to kinematic constraints. In [9], Corberes et al. incorporate this footstep planning strategy as an online foothold scheduler at 1-5 Hz with vision in the loop. Due to the low planning rate, and the lack of dynamics constraints in the contact scheduler, they rely on a separate whole body MPC to find feasible robot trajectories. Aceituno-Cabezas et al. [18] formulate a full quadruped trajectory optimization problem using mixed integer constraints for assigning footsteps to footholds and to approximate the nonlinear manifold constraint for 3D rotations. Their trajectory optimization features both kinematic and dynamics constraints, but unlike our work, does not re-plan the footholds in real time.

B. Foot placement control for underactuated walking

A family of controllers has emerged which use LIP based linear control policies [19] [4] or MPC footstep planners [6], [20] to generate footstep plans which are realized by tracking outputs such as CoM height, swing foot position, and joint angles with some form of Quadratic Programming (QP) based whole-body torque control [21] . These controllers view the continuous dynamics of each stance phase as approximated by the autonomous LIP dynamics, and stabilize the walking motion by placing the next footstep in the appropriate position to arrest excess momentum accumulated during single stance. Our controller adopts the same philosophy as these works, but extends the applicability of this approach beyond flat or mildly sloped terrain to terrain which can reasonably be modeled as a collection of convex polygons.

III. Preliminaries

The controller is implemented in two coordinate frames. Y is the yaw frame, representing a rotation of the world frame about the z axis to match the yaw angle of the floating base. S is the stance frame. It is an identity rotation from the yaw frame, with its origin located at the bottom center of the current stance foot. We follow a x-forward, y-left, z-up convention.

A. Continuous ALIP model

The ALIP model is an approximation of the CoM dynamics of the robot during single stance based on the LIP [3], which uses angular momentum in place of CoM velocity to describe the speed of the robot. We direct the reader to [6] for a derivation of the 3D ALIP dynamics assuming piecewise planar terrain with a passive ankle. To take full advantage of Cassie's blade foot, we include ankle torque in the sagittal plane, u as an input to the continuous time ALIP model. The state of the ALIP model consists of the horizontal position of the center of mass and the horizontal components of the angular momentum of the robot about the contact point. The dynamics of the ALIP with ankle torque are given by

$$\underbrace{\begin{bmatrix} \dot{x}_{com} \\ \dot{y}_{com} \\ \dot{L}_x \\ \dot{L}_y \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \frac{1}{mH} \\ 0 & 0 & \frac{-1}{mH} & 0 \\ 0 & -mg & 0 & 0 \\ mg & 0 & 0 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_{com} \\ y_{com} \\ L_x \\ L_y \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}}_{B} u \quad (1)$$

where m is the robot's mass, and H is the height of the CoM above the terrain, and all quantities are in the stance frame.

B. *ALIP Reset Map*

The reset map enables control of the ALIP through foot placement by relating the positions of the robot's feet to a discrete jump in the ALIP state. On hardware, a double stance phase between footsteps helps avoid oscillations caused by rapidly unloading Cassie's leaf springs. Therefore we derive a reset map from x_-, the ALIP state just before footfall, to x_+, the ALIP state just after liftoff, including the double stance phase. We start by integrating the double stance dynamics, and then we apply a coordinate change to express the ALIP state with respect to the new stance foot.

During double stance, we treat the center of pressure (CoP) as a control input, and integrate the resulting dynamics with an assumed input trajectory,

$$p_{CoP} = p_- + \frac{t}{T_{ds}}(p_+ - p_-) \tag{2}$$

where p_- and p_+ are the pre- and post-touchdown stance foot positions in the yaw frame, T_{ds} is the duration of double stance, and t is the time since the beginning of double stance. We assume the CoM velocity v_{CoM} is approximately parallel to $p_+ - p_-$. This is reasonable, as the robot is generally stepping in the direction it is walking. Under this assumption, $(p_{CoP} - p_-) \times mv_{CoM} \approx 0$, so angular momentum about p_- and p_{CoP} are equal.

$$L_{CoP} = L_{p_-} + (p_{CoP} - p_-) \times mv_{CoM} \approx L_{p_-}. \tag{3}$$

By treating the CoP as a virtual contact point and applying (1), we arrive at the continuous dynamics

$$\dot{x} = Ax + \underbrace{\begin{bmatrix} 0_{2\times1} & 0_{2\times1} & 0_{2\times1} \\ 0 & mg & 0 \\ -mg & 0 & 0 \end{bmatrix}}_{B_{CoP}} (p_{CoP} - p_-). \tag{4}$$

The solution to (4) with the input (2) and $x(0) = x_-$ is a first order hold discretization of (4) over double stance,

$$x(T_{ds}) = A_r x_- + B_{ds}(p_+ - p_-). \tag{5}$$

where $A_r = \exp(AT_{ds})$ and

$$B_{ds} = A_r A^{-1}\left(\frac{1}{T_{ds}}A^{-1}\left(I - A_r^{-1}\right) - A_r^{-1}\right)B_{CoP}. \tag{6}$$

The remainder of the reset map is just a coordinate change,

$$x_+ = x(T_{ds}) + \underbrace{\begin{bmatrix} I_{2\times2} & 0_{2\times1} \\ 0_{2\times2} & 0_{2\times1} \end{bmatrix}}_{B_{fp}} \left({}^Yp_+ - {}^Yp_-\right). \tag{7}$$

with the fp subscript denoting "foot placement".

By sequentially applying (5) then (7), we arrive at a reset map from x_- to x_+ which is linear in x_-, x_+, p_- and p_+,

$$x_+ = \left[A_r \;\; (-B_{ds} - B_{fp}) \;\; \underbrace{(B_{ds} + B_{fp})}_{B_r}\right] \begin{bmatrix} x_- \\ p_- \\ p_+ \end{bmatrix}. \tag{8}$$

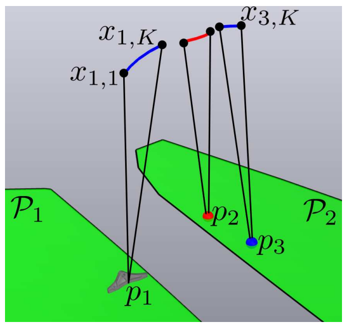

Fig. 3. Illustration of key MPFC problem data and decision variables for a horizon of 3 stance phases. The green polygons $\mathcal{P}_1$ and $\mathcal{P}_2$ are the foothold constraints. The black lines extend from the stance foot to the CoM position at the beginning and end of each single stance phase, and the CoM trajectories for each single stance phase are shown as alternating blue and red paths. The footstep positions are labeled $p_1 - p_3$, where p_1 is the current stance position and p_2 and p_3 are decision variables.

IV. MIQP Model Predictive Footstep Controller

The following section details the formulation of the MPFC as an MIQP. We simultaneously plan the footstep position, input, and ALIP trajectory over a horizon of N stance periods, satisfying the continuous dynamics (1) and discrete reset map (8). Each footstep is constrained to lie in a steppable region $\mathcal{P}_i$, represented as a 2D polygon embedded in 3D space. We discretize each single stance period of fixed duration T_{ss} into K knot points in order to apply intra-mode workspace constraints on the center of mass. For convenience, we index state knot points by their stance period and their order within the stance period, so the k^{th} knot point of the n^{th} stance period would be denoted $x_{n,k}$. An illustration of key problem parameters is shown in figure Fig. 3.

The MIQP formulation features continuous variables for the state and input trajectories, $\boldsymbol{x}$ and $\boldsymbol{u}$, as well as the stance foot position for each step, $\boldsymbol{p}$. We assign one binary variable, $\mu_{n,i}$ per foothold per step, indicating whether the foothold is used for that step. We can now introduce the problem statement of the MPFC (9), and dedicate the rest of this section to elaborating on the cost and constraints.

$$\underset{\boldsymbol{x},\boldsymbol{u},\boldsymbol{p},\boldsymbol{\mu}}{\text{minimize}} \sum_{n=1}^{N}\sum_{k=1}^{K-1}\left(\tilde{x}_{n,k}^T Q \tilde{x}_{n,k} + u_{n,k}^T R u_{n,k}\right) + \tilde{x}_{N,K}^T Q_f \tilde{x}_{N,K} \tag{9a}$$

$$\text{subject to } x_{n,k+1} = A_d x_{n,k} + B_d u_{n,k} \tag{9b}$$

$$x_{n+1,1} = A_r x_{n,K} + B_r(p_{n+1} - p_n) \tag{9c}$$

$$\mu_{n,i} = 1 \implies p_n \in \mathcal{P}_i \tag{9d}$$

$$\sum_{i\in\mathcal{I}} \mu_{n,i} = 1 \tag{9e}$$

$$\mu_{n,i} \in \{0,1\} \tag{9f}$$

CoM, Input, and Footstep limits

where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x_d}$ is the error from a desired reference trajectory, and $\mathcal{P}_i, i \in \mathcal{I}$ are the footholds.

A. Dynamics and Reset Map Constraints

The dynamics constraint (9b) is the discretization of (1) as a sampled system. Letting $\Delta t = T_{ss}/(K-1)$, then

$$A_d = \exp(A\Delta t) \tag{10}$$

$$B_d = A^{-1}(A_d - I)B. \tag{11}$$

The reset map (9c) is (8) expressed with MPFC indexing.

B. Foothold Constraints

Each convex polygonal foothold is defined by a plane $f_i^T p = b_i$ and a set of linear constraints $F_i p \leq c_i$. The logical constraint (9d) is enforced with the big-M formulation

$$F_i p_n \leq c_i + M(1 - \mu_{n,i}) \tag{12a}$$

$$f_i^T p_n \leq b_i + M(1 - \mu_{n,i}) \tag{12b}$$

$$-f_i^T p_n \leq -b_i + M(1 - \mu_{n,i}). \tag{12c}$$

With appropriately normalized F_i and f_i, (12) corresponds to relaxing each foothold constraint by M meters when $\mu_i = 0$. Since our problem scale is on the order of 2 m, we choose M = 10 for simplicity[1]. The binary constraint (9f) and the summation constraint (9e) imply that exactly one foothold must be chosen, and the remaining footholds relaxed.

C. State, Input, and Footstep Limits

We add a bounding box constraint on the CoM position with conservative bounds on y_{CoM} to avoid hip-roll joint limits. We limit the ankle torque to 5 Nm, which is what the OSC can reliably provide without foot slip. Finally, we add a constraint to prevent the feet from crossing the $x - z$ plane.

D. Reference Design

Given a desired average horizontal velocity, $v_d \in \mathbb{R}^2$, we generate a reference trajectory for the MPFC by finding a periodic ALIP trajectory which achieves this average velocity with a user-specified stance width l. In addition to letting us tune the stance width directly, this also ensures the desired position and angular momentum are consistent with the ALIP dynamics without using ankle torque. First, the gait parameters v_d and l are encoded into a footstep sequence

$$p_{n+1} = p_n + v_d T_{s2s} + \sigma_n l \hat{e}_y \tag{13}$$

where $T_{s2s} = T_{ss} + T_{ds}$ and $\sigma_n = -1, 1$ for left and right stance, respectively. To find the corresponding periodic ALIP trajectory, we define the step-to-step dynamics, which combine the single stance dynamics and reset map to arrive at a discrete dynamical system which has state $x_{n,1}$, the ALIP state at the beginning of single stance, and takes ${}^S p_{n+1}$ as an input. The dynamics are

$$x_{n+1,1} = \exp(AT_{s2s})x_{n,1} + B_r {}^S p_{n+1}. \tag{14}$$

We find the reference gait by substituting (13) into (14) and rolling out the dynamics for two stance modes, then solving for $x_{1,1} = x_{3,1}$ [2].

[1]M must be large enough for every relaxed foothold to contain every unrelaxed foothold, but should otherwise be small for numerical stability

[2]This is a specific choice of Period-2 orbit [4] which achieves symmetry between left and right stance.

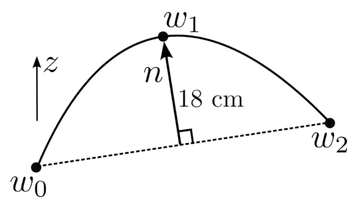

Fig. 4. We realize the planned footstep footstep positions by constructing a swing foot trajectory through the way points w_0, w_1, and w_2, where w_0 is the position of the swing foot at the beginning of the single stance phase, and w_2 is the most recent MPFC solution for the incoming stance foot.

V. Output Tracking via Operational Space Control

To realize the planned walking motion on the physical robot, MPC outputs are tracked with an inverse-dynamics based operational space controller (OSC). We use the same quadratic program as our previous Cassie examples [22] [23]. This section describes the construction of the task space trajectories tracked by the OSC.

A. Center of Mass Reference

Given a footstep plan from the MPC, we construct a CoM trajectory which enforces the local planarity assumption of the ALIP model by constructing the least-inclined plane passing through the current and imminent stance foot positions. Letting $p = {}^S p_{n+1}$, the plane parameters are the solution to

$$\begin{bmatrix} p_x & p_y \\ -p_y & p_x \end{bmatrix} \begin{bmatrix} k_x \\ k_y \end{bmatrix} = \begin{bmatrix} p_z \\ 0 \end{bmatrix}. \tag{15}$$

After solving for k_x and k_y, we define the reference trajectory for the CoM height in the stance frame as

$$z_c(t) = H + k_x x_c(t) + k_y y_c(t). \tag{16}$$

B. Swing Foot Reference

We generate swing foot trajectories by constructing a spline between the initial and final foot location during swing. First we generate an additional way point above the line connecting the initial and final foot location, as shown in Fig. 4. Then, as a heuristic for generating swing foot trajectories which will be tractable to track despite Cassie's leaf springs, we construct a minimum-snap spline through the way points. The motivation for this is that when considering the spring dynamics, the foot position is relative degree four to the motor torques.

C. Constant references

We track a constant pelvis roll and pitch of zero, and a constant hip yaw (abduction) angle of zero. We track a commanded pelvis yaw rate from the remote control, and a swing toe angle so that Cassie's foot makes an angle of $\arctan k_x$ with the ground.

VI. Perception Stack for Hardware Experiments

Here we describe the perception stack used to translate point clouds from the Intel RealSense D455 depth camera to convex foothold constraints in real time. The perception stack architecture is shown in Fig. 2.

A. State Estimator

We use the contact-aided invariant extended Kalman filter developed by Hartley et al. [24] to estimate the pose and velocity of the floating base. Due to the unobservability of Cassie's global position, we experience state estimator drift, especially vertically, which is accounted for in the elevation mapping node as described below.

B. RealSense D455 Depth Camera

The RealSense is mounted to Cassie's pelvis, looking down at the terrain in front of the robot (Fig. 6). We use the `realsense-ros`[3] ROS package to publish point-cloud data at 30 Hz, applying a decimation filter to reduce the number of points sent to the elevation mapping node.

C. GPU Based Elevation Mapping

We use the GPU based elevation mapping framework developed by Miki et al. in [10] to construct a robot-centric elevation map of the terrain. This framework represents the terrain as a grid around the robot, with the height of each cell updated by point cloud measurements through a Kalman filter. The quality of the convex planar decomposition, and ultimately the stability of the controller, depends on the accuracy of the elevation map, so we make several Cassie-specific modifications to [10], outlined below.

1) Point Cloud Preprocessing: To eliminate spurious measurements of the Cassie's front shell, we crop out a band of points along the near edge of the depth camera frame. Additionally, we crop out points outside user-specified minimum and maxium depths. We mask out Cassie's legs by removing all points from a bounding box extending up and back from the front of each foot.

2) Drift Correction using the Stance Foot: While [10] features a floating base drift correction feature which compares the height of the input point cloud to the height of the existing map, we found this to be insufficient for the severity of floating base drift we experience on Cassie. Because Cassie's state estimate experiences a consistent upward drift due to impacts during touchdown, and because we have only one depth camera on the front of the robot, terrain under and behind the robot is estimated to be lower than in reality. To correct for this, before each point cloud update, we adjust the height of the elevation map by adding the height difference between the elevation map and the current stance foot.

D. Planar Segmentation

We use the planar segmentation module provided by [10] (but described in [11]) to segment the elevation map into planar polygons. First, several filters are applied to the elevation map. In addition to the de-noising median filter described in [11], we apply an erosion filter and Gaussian blur to the height map to smooth out the terrain into its broad features. Next, each pixel in the elevation map is classified as steppable or not based on surface inclination and roughness in a neighborhood around the pixel. From this classification, connected components of steppable terrain are identified, and their outline is extracted as a 2D polygon embedded in 3D space, with a safety margin of 5 cm. It should be noted that the effective safety margin is higher, as the Gaussian blur rounds off sharp corners in the elevation map.

[3]https://github.com/IntelRealSense/realsense-ros

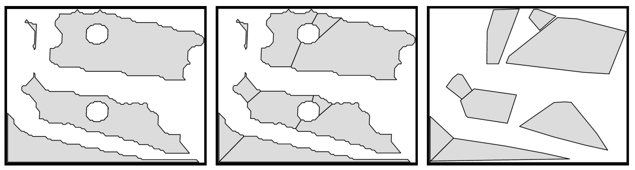
Fig. 5. Illustration of the convex polygon decomposition process. Left: original nonconvex polygons with holes. Middle: approximate convex decomposition of the original polygons. Right: convex inner approximation of the approximately convex components, with small components filtered out. When tuning the convex decomposition, there is a trade off between maximizing the total traversable area and maintaining a low number of convex polygons.

E. Convex Polygon Decomposition

In general, the planar segmentation yields non-convex polygons with holes (caused, for example, by small obstacles or other unsteppable areas), but we require convex foothold constraints for the MPFC. We use a two stage process (Fig. 5) to find a set of convex polygons whose union approximately matches the original non-convex polygon. This avoids creating many small triangles like an exact convex decomposition would, decreasing the number of integer variables in the MPFC.

First, we perform approximate convex decomposition (ACD)[13] on each polygon. ACD returns a decomposition of the original region into polygons which are τ-approximately convex, with τ representing the depth of the largest concave feature. After filtering out polygons with area less than 0.1 m^2, we find a convex inner-approximation of these nearly convex polygons with a greedy approach we name the whittling algorithm (Algorithm 1), after the way it makes incremental cuts to the polygon. We initialize the output polygon, $\mathcal{P}$ as the convex hull of the original polygon, then take the intersection of $\mathcal{P}$ with greedily chosen half-spaces until no vertices of the original polygon are contained in $\mathcal{P}$. While this does not guarantee containment of $\mathcal{P}$ in the original polygon, we do not see violations in practice.

Algorithm 1 Whittling Algorithm

```
Require: Input polygon vertices $V = \{v_0 \dots v_n\}$
  procedure WHITTLE($V$)
      $\mathcal{P} \leftarrow$ ConvexHull($V$)
      for all $v_i$ do
          if $v_i \in$ Interior($\mathcal{P}$) then
              $H$ = MakeCut($v_i$, $\mathcal{P}$)
              $\mathcal{P} \leftarrow \mathcal{P} \cap H$
      return $\mathcal{P}$
```

MakeCut($\mathcal{P}, v$) is a QP which finds a such that the half-space $H = \{x \mid a^T(x - v) \leq 0\}$ contains as much of $\mathcal{P}$ as possible, as measured by minimizing the squared hinge loss $\sum \max(a^T(p_i - v), 0)^2$, where p_i are the vertices of $\mathcal{P}$.

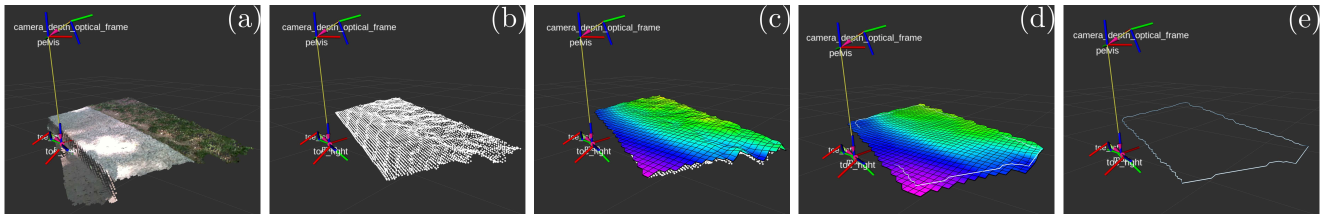

Fig. 6. Example of the elevation mapping and planar polygon extraction process showing the field of view of the Intel RealSense. Frame (a) shows the raw point-cloud data as seen in the world frame. Our point cloud pre-processing step (b) crops out potentially noisy measurements of Cassie's front shell and a bounding box around the robot's feet. Frame (c) shows the raw elevation map constructed from the cropped point cloud. Frame (d) Shows the map after applying a Gaussian blur and median filter, as well as the steppable planar polygon extracted from the map. (e) Shows the extracted polygon by itself.

VII. Results

Both layers of the control stack are implemented in C++ using the Drake [25] systems framework and mutlibody kinematics/dynamics. The MPFC and OSC use Drake's interfaces to the Gurobi and OSQP solvers respectively. The controllers are run in separate processes and communicate over LCM [26]. This abstracts the controller from source of the robot state information, allowing us to test identical code in simulation or on hardware. We use a horizon of 3 stance periods for the MPFC in order to plan multiple footsteps ahead.

A. Simulation Experiments

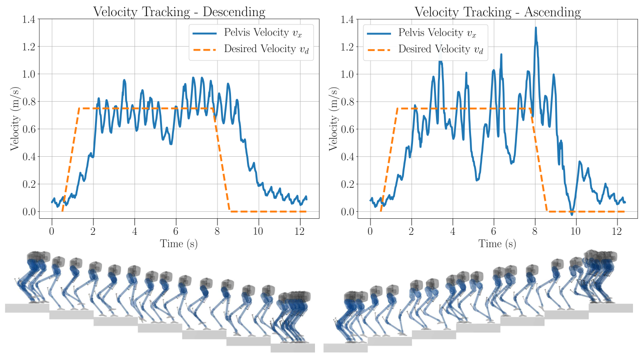

Fig. 7. Cassie descending and ascending a set of 1 m. long, 15 cm. tall steps with a pre-programmed velocity profile in simulation. Cassie tracks the commanded velocity with occasional deviations to satisfy foothold constraints.

To highlight the capabilities of the controller in an idealized environment, we demonstrate Cassie ascending and descending steps in simulation (Fig. 7). While we model Cassie as realistically as possible, including springs, reflected inertia, motor curves, and realistic joint limits, the MPFC and OSC have access to ideal state and terrain information. Cassie is able to track a commanded velocity of 0.75 m/s on meter long stairs, and automatically deviate from the velocity command to satisfy foothold constraints. Given Cassie's limited ankle torque, walking can only be stabilized on steps long enough to use footstep placement as the primary control input, 1 m for a commanded speed of 0.75 m/s, and 50 cm for a commanded speed of 0.5 m/s. Additionally, for steps taller than 15 cm, the ALIP model becomes a poor approximation of the dynamics, and the trailing swing foot starts to clip the stairs when stepping up. We witnessed this effect more severely on hardware, and discuss it further in Section VIII.

B. Hardware Experiments

For hardware deployment, we run the high speed state estimator and OSC loops on Cassie's onboard NUC computer and send joint torque commands to Cassie's target PC over UDP. The NUC is upgraded from stock to achieve better performance [4]. The MPFC and perception stack run on an offboard ThinkPad p15 Laptop with an 8-core, 2.3 GHz Intel 1180H processor and 24 GB of RAM. The laptop is carried in a backpack by one of the safety bar carriers and networked with the NUC over ethernet for LCM and ROS communication.

We demonstrate our walking controller ascending and descending a 9 cm. curb with vision in the loop. One trial can be seen in Fig. 8, and additional trials can be seen in the supplemental video. While we achieved multiple successful trials, several interactions between the controller and perception stack make the system brittle in practice, as detailed in Section VIII.

1) Controller Solve Times: Since the computational complexity of the MPFC scales with the number of foothold constraints, and fast re-planning is required for stable walking, we analyze the effect of the number of potential footholds on solve time (Fig. 9). While the minimum, mean, and 90th percentile solve times are similar, and all increase slowly up to 9 footholds, the maximum solve time is 2.5-5 times higher. These occasional long solve times can be up to 10 percent of a single stance phase, and introduce torque spikes when tracking commanded foot position with high feedback gains. We also examine the relationship between foothold constraint activation and solve time. When the MPFC solution contains a footstep on a foothold boundary, the convex relaxation of the MIQP is no longer optimal, leading to longer solve times. This scenario makes up 4.9 percent of the data in Fig. 9, and will increase with more challenging terrains.

VIII. Implementation Lessons

In this work, we synthesized a number of mature or maturing ideas. We combined LIP based footstep control and MIQP based kinematic footstep planning to design a novel

[4]https://github.com/DAIRLab/cassie_documentation/wiki/Upgrading-the-Intel-NUC

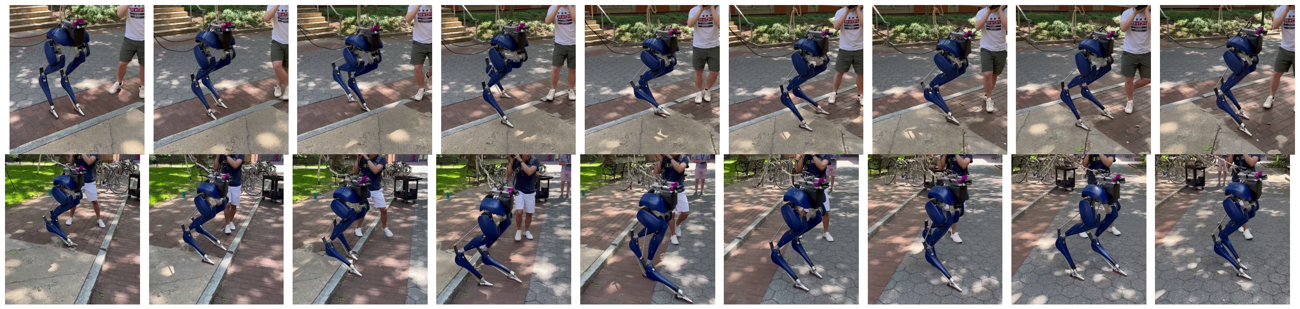

Fig. 8. Motion tiles showing Cassie stepping up (top row) and then down (bottom row) a curb using the perception and control stack outlined in this paper. Neither the controller nor the perception stack have any prior knowledge of the environment.

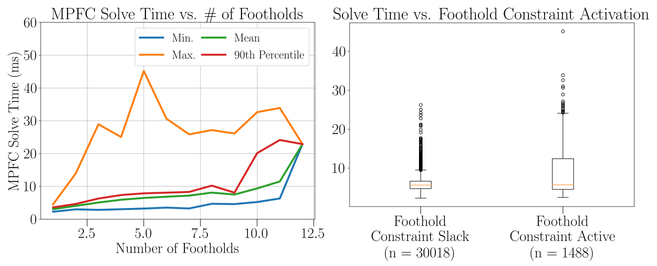

Fig. 9. Analysis of MPFC solve times over 6 trials totaling 3:35 minutes of walking and 8 surface transitions. Left: Plot of solve time vs. the number of foothold constraints. Solve time increases Right: Box and whisker plot showing the relationship between constraint activation and solve time. Solve time increases when the optimal footstep is on the boundary of a stepping stone, as the convex relaxation of the MIQP is no longer optimal

walking controller. We informed this controller with a state of the art elevation mapping and convex planar decomposition framework. Here, we discuss successes and shortcomings of these methods and how they integrate into the full perceptive locomotion system.

MPFC demonstrates yet again the effectiveness of (A)LIP based foot placement control for underactuated walking. When using MPFC on flat terrain, it is capable of achieving robust, dynamic walking similar to our baseline ALIP walking controller based on [5]. Our simulation results also show that with ideal terrain information, incorporating mixed integer footstep constraints expands the capabilities of ALIP based walking control to discontinuous terrain. Fast solve times point to the fact that more advanced solvers are making it possible to use MIQPs for high-rate control tasks.

Where we experienced challenges was our controller's sensitivity to perception error and noise. The GPU based elevation mapping framework has so far been applied primarily to quadruped robots, which are lighter, more stable, and less susceptible to the effects of impacts when walking. Additionally, existing controllers with real-time perceptive footstep planning generally plan footholds about once per stride [1], [9], [11], making them less susceptible to noise in the perception system. The MPFC foothold constraints, on the other hand, are updated in real time, and are the direct output of a 30 Hz perception system. Therefore the whole system's performance is dependent on accurate, consistent perception.

A. *Elevation Mapping Artifacts*

Errors and noise in the elevation map can propagate through the perception pipeline to the controller, leading to failure. When walking on grass, for example, the stance foot drift correction conflicts with the height of the point cloud, as Cassie's foot rests below the top of the grass. This leads to discontinuities in the height map when walking on flat grass. Additionally, the controller is sensitive to errors in the estimated ground height, which alter the swing foot touchdown time, ultimately causing the swing-foot to miss its commanded target.

B. *Safety Margins*

Because we model Cassie's feet as points in the MPFC, we rely on a safety margin in the planar segmentation algorithm to account for the length of the blade foot. Tuning this margin is quite difficult in practice, as there is a trade off between collision risk and traversable area. Insufficient margin leads to collisions when stepping up onto a higher surface, but unnecessary margin leads to terrain gaps too large to cross at reasonable walking speeds. We experienced both of these failure modes with the same margin, depending on the commanded velocity and initial conditions of the step. While we attempted shaping the swing foot trajectory to avoid collisions, these trajectories were difficult to track due to large accelerations needed to start the swing phase moving away from the target position. The walking speed dependence of this effect increased the skill and concentration required from the operator. Future work will consider reformulating the MPFC cost function to address these challenges, as well as allowing for asymmetric safety margins in the planar polygon contour extraction process depending on the relative height of each surface.

C. *Perception Noise*

When the planned footsteps are at or near the boundary of a foothold, noise in the robot state or foothold boundaries can cause sudden jumps in the impending footstep position by changing which foothold sequence is optimal. To avoid causing large jumps which would be impossible to track, we introduce an additional bounding box constraint on the next footstep position. The constraint is applied when 250ms remain in the swing phase, and constrains the upcoming footstep to a bounding box with half length 10 cm, centered at the footstep solution from the most recent MPFC solve.

IX. Conclusions and Future Work

We present a new model predictive footstep controller which allows underactuated bipeds to walk on constrained terrain without predefined foothold sequences. We formulate our controller as a single Mixed Integer Quadratic Program which can be solved online faster than the 30 Hz rate of the realtime perception system used to model the terrain as convex polygonal footholds. We demonstrate the controller on Cassie with a fully integrated vision system.

Future work will focus on improving the reliability of the perception pipeline to give a consistent and accurate terrain representation while maintaining its real-time performance, and improving the robustness of the MPFC and OSC to perception error.

X. Acknowledgements

We thank Yu-Ming Chen and William Yang for assistance with experiments and years of collaboration and mentorship working on Cassie. We also thank Chandravan Kunjeti, Wei-Cheng Huang, and Alp Aydinoglu for help with experiments and helpful discussions. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1845298. Toyota Research Institute also provided funds to support this work.

References

[1] D. Calvert, B. Mishra, S. McCrory, S. Bertrand, R. Griffin, and J. Pratt, "A Fast, Autonomous, Bipedal Walking Behavior over Rapid Regions," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, Nov. 2022, pp. 24–31.

[2] M. F. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, "Continuous humanoid locomotion over uneven terrain using stereo fusion," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov. 2015, pp. 881–888.

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

[4] X. Xiong and A. Ames, "3-D Underactuated Bipedal Walking via H-LIP Based Gait Synthesis and Stepping Stabilization," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2405–2425, Aug. 2022.

[5] Y. Gong and J. Grizzle, "One-Step Ahead Prediction of Angular Momentum about the Contact Point for Control of Bipedal Locomotion: Validation in a LIP-inspired Controller," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 2832–2838.

[6] G. Gibson, O. Dosunmu-Ogunbi, Y. Gong, and J. Grizzle, "Terrain-Adaptive, ALIP-Based Bipedal Locomotion Controller via Model Predictive Control and Virtual Constraints," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 6724–6731.

[7] O. Dosunmu-Ogunbi, A. Shrivastava, G. Gibson, and J. W. Grizzle, "Stair Climbing using the Angular Momentum Linear Inverted Pendulum Model and Model Predictive Control," *arXiv preprint arXiv:2307.02448*, Jul. 2023.

[8] M. Dai, X. Xiong, and A. Ames, "Bipedal Walking on Constrained Footholds: Momentum Regulation via Vertical COM Control," in *2022 International Conference on Robotics and Automation (ICRA)*, May 2022, pp. 10 435–10 441.

[9] T. Corbères, C. Mastalli, W. Merkt, I. Havoutis, M. Fallon, N. Mansard, T. Flayols, S. Vijayakumar, and S. Tonneau, "Perceptive Locomotion through Whole-Body MPC and Optimal Region Selection," *arXiv preprint arXiv:2305.08926*, May 2023.

[10] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using gpu," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 2273–2280.

[11] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model-predictive control," *IEEE Transactions on Robotics*, pp. 1–20, 2023.

[12] F. Jenelten, R. Grandia, F. Farshidian, and M. Hutter, "TAMOLS: Terrain-Aware Motion Optimization for Legged Systems," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3395–3413, Dec. 2022.

[13] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polygons," *Computational Geometry*, vol. 35, no. 1, pp. 100–123, Aug. 2006.

[14] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov. 2014, pp. 279–286.

[15] S. Tonneau, D. Song, P. Fernbach, N. Mansard, M. Taïx, and A. Del Prete, "Sl1m: Sparse l1-norm minimization for contact planning on uneven terrain," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6604–6610.

[16] D. Song, P. Fernbach, T. Flayols, A. D. Prete, N. Mansard, S. Tonneau, and Y. J. Kim, "Solving Footstep Planning as a Feasibility Problem Using L1-Norm Minimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5961–5968, Jul. 2021.

[17] F. Risbourg, T. Corbères, P.-A. Léziart, T. Flayols, N. Mansard, and S. Tonneau, "Real-time Footstep Planning and Control of the Solo Quadruped Robot in 3D Environments," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 12 950–12 956.

[18] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous Contact, Gait, and Motion Planning for Robust Multilegged Locomotion via Mixed-Integer Convex Optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, Jul. 2018.

[19] D. Kim, S. J. Jorgensen, J. Lee, J. Ahn, J. Luo, and L. Sentis, "Dynamic locomotion for passive-ankle biped robots and humanoids using whole-body locomotion control," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 936–956, Jul. 2020.

[20] X. Xiong, J. Reher, and A. D. Ames, "Global Position Control on Underactuated Bipedal Robots: Step-to-step Dynamics Approximation for Step Planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 2825–2831.

[21] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 3103–3109.

[22] W. Yang and M. Posa, "Impact Invariant Control with Applications to Bipedal Locomotion," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2021, pp. 5151–5158.

[23] Y.-M. Chen and M. Posa, "Optimal Reduced-order Modeling of Bipedal Locomotion," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 8753–8760.

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

[25] Russ Tedrake and the Drake Development Team, "Drake: Model-Based Design and Verification for Robotics," 2019.

[26] A. S. Huang, E. Olson, and D. C. Moore, "LCM: Lightweight Communications and Marshalling," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 4057–4062.