2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Robust Autonomous Navigation of a Small-Scale Quadruped Robot in Real-World Environments

Thomas Dudzik[1], Matthew Chignoli[2], Gerardo Bledt[2], Bryan Lim[2],
Adam Miller[1], Donghyun Kim[2], and Sangbae Kim[2]

Abstract— **Animal-level agility and robustness in robots cannot be accomplished by solely relying on blind locomotion controllers. A significant portion of a robot's ability to traverse terrain comes from reacting to the external world through visual sensing. However, embedding the sensors and compute that provide sufficient accuracy at high speeds is challenging, especially if the robot has significant space limitations. In this paper, we propose a system integration of a small-scale quadruped robot, the MIT Mini-Cheetah Vision, that exteroceptively senses the terrain and dynamically explores the world around it at high velocities. Through extensive hardware and software development, we demonstrate a fully untethered robot with all hardware onboard running a locomotion controller that combines state-of-the-art Regularized Predictive Control (RPC) with Whole-Body Impulse Control (WBIC). We devise a hierarchical state estimator that integrates kinematic, IMU, and localization sensor data to provide state estimates specific to path planning and locomotion tasks. Our integrated system has demonstrated robust autonomous waypoint tracking in dynamic real-world environments at speeds of over $1\ \mathrm{m/s}$ with high rates of success.**

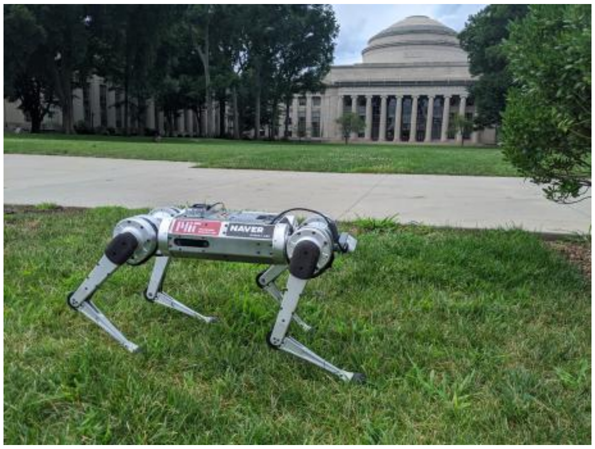

Fig. 1. **Mini-Cheetah Vision navigating a real-world environment.** The robot is able to autonomously navigate the rough garden terrain while avoiding obstacles such as shrubbery and light posts.

I. Introduction

The unique ability of legged systems to exploit discrete ground contacts provides a massive mobility advantage over wheeled and tracked systems when it comes to unstructured terrain. However, this advantage simultaneously invites a host of new and rich challenges related to planning, control, and hardware design. The immense upside of legged robots with animal-level locomotion capabilities has motivated considerable research in recent years [1]–[3]. Impressive developments in locomotion control algorithms [4]–[6] have opened the door for new research avenues related to increased autonomy and practical, task-oriented behaviors. In the area of quadruped locomotion, the majority of these control algorithms and their practical applications are designed specifically for medium to large-scale robots [7], [8]. Despite the many applications of small-scale quadrupeds including exploration of narrow regions such as tunnels, damaged buildings, ships, and submarines, they have, in comparison, received very little research attention [9]. This holds true to an even greater extent for vision-integrated small-scale legged robots.

The ability for a robot to blindly traverse terrain is critical for primitive locomotion balancing, but does not render it very useful in terms of task execution. As robots mature to the point where they can capably maneuver challenging terrain, the ability to actively plan intelligent paths and footholds within its environment becomes increasingly important. The robot should possess higher-level reasoning to compute an optimal path to take between locations as well as determine whether or not it is feasible to take a certain path. While these tasks have been studied in depth for various autonomous systems ranging from ground vehicles [10] to quadrotors [11], the interactions of controlled-limb systems with the environment has received arguably less attention. Of the prior published research, most of them focused on utilizing A* search for biped humanoid robots as in [12]–[14].

There has been plenty of success showing various techniques and frameworks for embedding perception into robot control systems [15]–[17]. Many demonstrate that tightly embedding external perception data into trajectory planning and balance, while maintaining the independence of the locomotion control, is crucial for a reliable and robust system. This work presents our methods for extending blind locomotion capabilities by incorporating perception and higher-level intelligent path planning.

A. Contributions

Much of the preceding work on the Mini-Cheetah robot platform focused on improving operator-guided blind locomotion performance. In this work, we present the considerable improvements done to the original vision-processing

[1] Department of Electrical Engineering & Computer Science, MIT, Cambridge, MA 02139, USA: {tdudz, ajm4}@mit.edu

[2] Department of Mechanical Engineering, MIT, Cambridge, MA 02139, USA: {chignoli, gbledt, bryanlwt, humanoid, sangbae}@mit.edu

978-1-7281-6211-9/20/$31.00 ©2020 IEEE

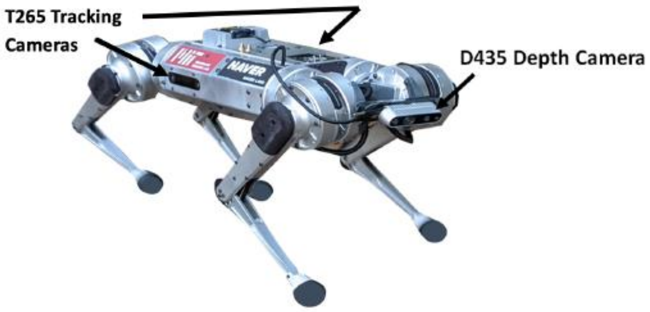

Fig. 2. **Onboard Sensing Suite.** Three Intel RealSense cameras are mounted on the Mini-Cheetah Vision. Two T265 tracking cameras are mounted internally on either side, and a D435 depth camera is mounted at a fixed angle on the front of the robot.

implementation as described in [18], and describe our addition of higher-level decision-making for autonomous path planning in dynamic environments. Our system integration of various components results in an onboard visual-perception system that is implemented on the small-scale, low-cost quadruped, Mini-Cheetah Vision. The locomotion framework previously presented is modified to directly make use of the visual data by informing the locomotion controllers of the surrounding terrain, while maintaining the inherent control robustness demonstrated in prior work. This framework combines intelligent, perceptive path planning and robust control to allow the robot to autonomously navigate real-world environments involving unsteady, unstructured terrain while deciding whether obstacles are traversable or should be planned around.

II. System Overview

The platform used in this study is the upgraded MIT Mini-Cheetah [19], dubbed the Mini-Cheetah Vision. The robot stands $0.3\ \mathrm{m}$ tall with a mass of approximately $9.6\ \mathrm{kg}$. Its upper and lower leg links measure $0.21\ \mathrm{m}$ and $0.20\ \mathrm{m}$, respectively. The robot is outfitted with a set of exteroceptive sensors that enable environmental perception and global state estimation which we describe below.

A. Onboard Exteroceptive Sensing

Perception of the Mini-Cheetah Vision's environment is achieved via three Intel RealSense cameras: a D435 depth camera and two T265 tracking cameras. The layout of the cameras on the robot is shown in Fig. 2. These particular cameras were selected because their small size, low power consumption, and high frame rate fulfill the strict requirements necessary for rapid onboard sensing.

The D435 depth camera, mounted on the front of the robot, is used to detect obstacles and build local maps of the robot's terrain. It measures $90\ \mathrm{mm} \times 25\ \mathrm{mm} \times 25\ \mathrm{mm}$, has a mass of $71.8\ \mathrm{g}$, and a $86° \times 57° \times 94°$ field of view. The sensor publishes 640×480 depth images at a rate of $90\ \mathrm{Hz}$ and has an accuracy of less than 1% error per meter from the sensor.

The T265 tracking cameras, mounted on either side of the robot, are used for localizing the global position and orientation of the robot. Each camera measures $108\ \mathrm{mm} \times 24.5\ \mathrm{mm} \times 12.5\ \mathrm{mm}$ with a mass of $55\ \mathrm{g}$. The two fisheye lenses in each camera have a combined field of view of $163 \pm 5°$. The T265 publishes a pose estimate produced by its visual-inertial SLAM algorithm at a rate of $200\ \mathrm{Hz}$.

B. Framework Overview

Our framework consists of four primary components: vision processing and high-level planning, locomotion control, state estimation, and a motion library. Vision processing computes the robot's absolute pose and builds a heightmap surrounding the robot. The outputs are then sent to the high-level planner which computes an optimal path to the desired goal location as a series of discrete waypoints. These points are then translated into joint torque and velocity commands for the locomotion controller. The Lightweight Communication and Marshalling (LCM) [20] library is used for all data serialization and transfering between modules. Lastly, the motion library saves pre-optimized motions such as jumping, backflipping, and recovery stand-up protocols that can be utilized when normal locomotion is insufficient.

Simultaneous, real-time operation of the vision and control systems poses a difficult challenge. The dense visual data processing done by the vision system and the nonlinear optimization of the locomotion controller both require significant computational bandwidth that must be carried out in realtime on the robot. A design modification involving extending the length of the robot's body frame allowed an NVIDIA Jetson TX2 to be fit inside the robot's body along with the standard UP board. This additional computing power allows for the Mini-Cheetah Vision to be a fully untethered platform.

C. Global State Estimation

State estimation for the robot is handled via a two-tiered hierarchical structure, similar to the framework described in [21] that integrates kinematic, IMU, and localization sensor data. The hierarchical framework leverages the benefits of the various sensor data streams to create a state estimation algorithm that is optimized for our dual planning and locomotion control architecture. High-level planning and low-level locomotion control are executed at different frequencies and depend on separate states of the robot. Specifically, high-level planning utilizes the estimated absolute position and yaw of the robot, while low-level planning requires the robot's linear and angular velocity as well as its estimated height relative to the ground beneath it. Furthermore, the level of accuracy required for each task is different. The hierarchical framework is able to accommodate the requirements of each task in a way that simultaneously allows for high-speed locomotion over challenging terrain and efficient, obstacle-robust planning.

The hierarchical approach differs significantly from approaches that tightly fuse and smooth all leg odometry, inertial measurements, and visual odometry into a single state estimate used for both planning and locomotion. Implementations following this approach have included fusing

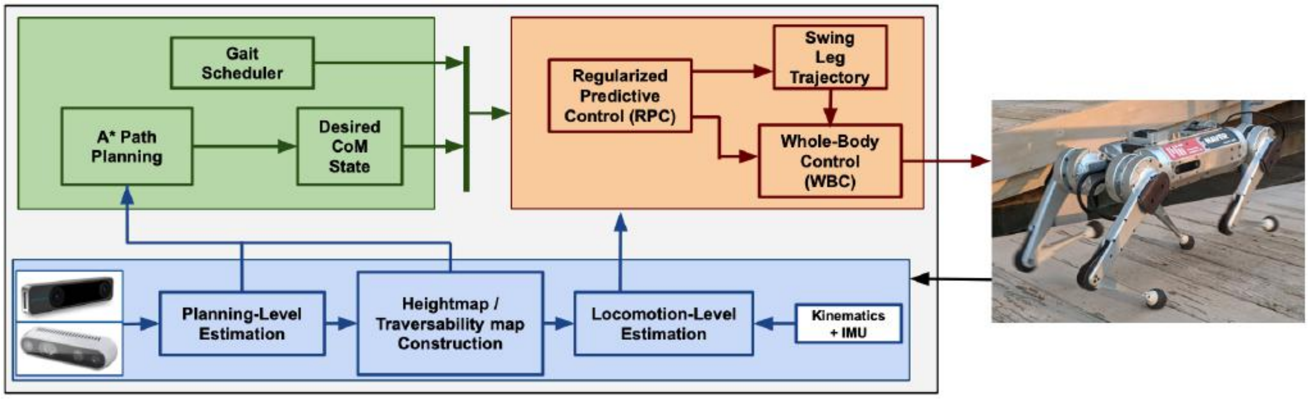
Gait Scheduler
A* Path Planning
Desired CoM State
Regularized Predictive Control (RPC)
Swing Leg Trajectory
Whole-Body Control (WBC)
Planning-Level Estimation
Heightmap / Traversability map Construction
Locomotion-Level Estimation
Kinematics + IMU

Fig. 3. **High-Level System Architecture.** Block diagram visualizing the software architecture and the interaction between modules. Green represents the perception and planning module, blue represents the state estimation module, and red represents the locomotion control module. The motion library (not shown) is a separate component. The main novelty of this paper is signified by improvements made to the the green section of this schematic.

and smoothing techniques such as extended Kalman Filters [22] and factor graph optimizations [23], [24]. Previous work involving high speed locomotion of the MIT Mini-Cheetah [25] employed a kinematic-inertial state estimation paradigm [26] that assumed flat ground. In this work, we remove the flat ground assumption by incorporating heightmap data from the D435 depth camera. Unlike the “robot-centric” approach [27], [28] that constructs a probabilistic local heightmap based purely on relative localization from kinematic and inertial measurements, our approach uses vision-based global localization to construct the heightmap that then informs the locomotion-level estimator. Estimates of the height of the terrain at each of the stance foot locations combined with the kinematic data from each stance leg enables fast and accurate estimation of the robot’s height relative to the ground. This technique for synthesizing the relative body height is preferred over a method that directly uses the localization data because the localization data updates at a slower frequency and is more sensitive to the highly dynamic motion of the body as it moves. Using our technique, the robot’s global position and yaw angle still suffer from drift, an unavoidable symptom of the kinematic-inertial estimation paradigm, but this drift has no effect on the locomotion controller since it does not depend explicitly on these states.

Autonomous path planning, however, does depend on the robot’s global position and yaw angle. To this end, the T265 cameras mounted on either side of the robot provide the localization measurements upon which the higher-level, visual-inertial estimation is based. The dual cameras allow for accurate, drift-robust localization even in cases where the view from one side of the robot is obstructed. In the case where both sides of the robot are obstructed or if the robot is operating in exceptionally low-light conditions, the state estimator will resort to using only kinematic-inertial data streams.

D. Heightmap Construction

Our heightmap construction algorithms are similar to work done in [18]. The Intel Realsense D435 camera mounted at the front of the robot publishes pointclouds that are transformed from the camera frame to the world frame using transformation matrices within the Special Euclidean group $SE(3)$. This world-frame pointcloud is then used to update a 2.5D heightmap where the height is encoded as an entry in a two-dimensional matrix. In particular, we discretize the world into square cells with side lengths of $10\ \mathrm{cm}$ and assign height values based on the z-coordinate of the most recent pointcloud point that falls within the corresponding grid cell.

We employ a heightmap filtering and traversability evaluation scheme, as the traversability map is a crucial component in the path-planning pipeline. The same erosion and dilation matrices are applied to the heightmap as in [18], as well as the gradient filters to create compute the traversability of the terrain. Since in reality the robot body is not a point mass and has non-negligible length and width, we further apply a Gaussian blur filter with kernel size 5×5 to expand the non-traversable regions around obstacles in the traversability map such that the robot can actually pass by the objects without any collisions. The result is a two-dimensional matrix where entries containing zeros mean that the cells are traversable by the robot’s center of mass (CoM) while entries containing non-zero values are not.

III. Locomotion Control

The robot is controlled by a set of hierarchical locomotion controllers that work together to take the desired CoM path and calculate joint torques that are sent to the robot. A Regularized Predictive Controller (RPC) uses a nonlinear optimization formulation on a simplified robot model to calculate ground reaction forces, footstep locations, and robot CoM states over a future prediction horizon. The results of this controller are passed into a Whole-Body Impulse Controller (WBIC), which finds the instantaneous joint torques that will minimize a quadratic program to track the desired CoM trajectory, as well as the ground reaction forces coming from the RPC on a full robot model. Together, the two controllers work to take the best parts of each, specifically the prediction capabilities of RPC and the use of the full system model in WBIC.

A. Regularized Predictive Control (RPC)

The RPC controller has been demonstrated on the MIT Cheetah 3 and Mini-Cheetah quadruped robots [5], [29], [30], which we briefly describe here. Using a simplified lumped-mass robot model with a massless leg assumption, the number of states describing the robot is largely reduced. The CoM states, $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{p}^T & \boldsymbol{\Theta}^T & \dot{\boldsymbol{p}}^T & \dot{\boldsymbol{\Theta}}^T\end{bmatrix}^T$, are composed of the position, $\boldsymbol{p}$, Euler angles, $\boldsymbol{\Theta}$, and their derivatives while the inputs to the system, $\boldsymbol{u} = \begin{bmatrix}\boldsymbol{r}_1^T & \boldsymbol{f}_1^T & \dots & \boldsymbol{r}_4^T & \boldsymbol{f}_4^T\end{bmatrix}^T$, are the footstep vector and corresponding ground reaction forces. With this definition of the system, we can describe the simplified discrete dynamics to be

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k h(\boldsymbol{\chi}_k, \boldsymbol{\Phi}_k) + \boldsymbol{d}_k \tag{1}$$

where $\boldsymbol{\Phi}$ describes the scheduled gait pattern and the matrices are linear and time varying as

$$\boldsymbol{A}_k = \begin{bmatrix}\mathbf{I}_6 & dt_k\mathbf{I}_6\\ \mathbf{0}_6 & \mathbf{I}_6\end{bmatrix} \boldsymbol{B}_k = \begin{bmatrix}\frac{dt_k^2}{2}\boldsymbol{I}^{-1}\\ dt_k\boldsymbol{I}^{-1}\end{bmatrix} \boldsymbol{d}_k = \begin{bmatrix}\frac{dt_k^2}{2}\boldsymbol{a}_g\\ dt_k\boldsymbol{a}_g\end{bmatrix}. \tag{2}$$

The net wrench on the CoM is calculated from the cumulative contributions of each stance foot's ground reaction force at the step location as

$$\begin{aligned} h(\boldsymbol{\chi}_k, \boldsymbol{\Phi}_k) &= \begin{bmatrix}\boldsymbol{f}\\ {}^{\mathcal{B}}\boldsymbol{\tau}\end{bmatrix}\\ &= \sum_{i=1}^{4}\begin{bmatrix}\mathbf{I}_3\\ \boldsymbol{R}(\boldsymbol{\Theta}_k)^T\left[\boldsymbol{r}_{i,k}\right]_\times\end{bmatrix}\boldsymbol{\Phi}_{i,k}\boldsymbol{f}_{i,k}. \end{aligned} \tag{3}$$

Here, the full CoM orientation rotation matrix $\boldsymbol{R}(\boldsymbol{\Theta}_k)$ is used to convert the torques into the body frame in order to maintain the linear simplicity of the rest of the discrete dynamics.

Decision variables in the optimization, $\boldsymbol{\chi} = \begin{bmatrix}\boldsymbol{x}^T & \boldsymbol{u}^T\end{bmatrix}^T$, are composed of a combination of robot CoM states and inputs for all timesteps within the decided prediction horizon, $k \in 0, ..., N-1$. The number of timesteps can be selected to be anything, although in practice it is common to use $N = 5$. Decision variables are regularized to simple heuristics that are expertly designed or extracted from simulation data and embedded into the cost function through error terms

$$\tilde{\boldsymbol{\chi}} = \boldsymbol{\mathcal{H}}_{\boldsymbol{\chi}}(\boldsymbol{\chi}, \boldsymbol{\Phi}, \boldsymbol{x}_d) - \boldsymbol{\chi}. \tag{4}$$

The process for extracting data-driven heuristics is detailed in [30]. The controller can be framed as a general nonlinear optimization formulation written to be

$$\begin{aligned} \min_{\boldsymbol{\chi}} \quad & J(\boldsymbol{\chi}) = \sum_{k=0}^{N-1}\tilde{\boldsymbol{\chi}}_k^T\boldsymbol{W}_k\tilde{\boldsymbol{\chi}}_k\\ \text{s.t.} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k h(\boldsymbol{\chi}_k, \boldsymbol{\Phi}_k) + \boldsymbol{d}_k\\ & \boldsymbol{\zeta}_k(\boldsymbol{\chi}_k, \boldsymbol{\Phi}_k) \le 0\\ & \boldsymbol{\zeta}'_k(\boldsymbol{\chi}_k, \boldsymbol{\chi}_{k+1}, \boldsymbol{\Phi}_k, \boldsymbol{\Phi}_{k+1}) \le 0 \end{aligned} \tag{5}$$

where $\boldsymbol{\zeta}_k$ and $\boldsymbol{\zeta}'_k$ are the instantaneous and linking constraints, respectively.

While the controller has shown exceptional blind locomotion robustness to various unexpected terrain changes, the constraint corresponding to placing the feet on the estimated ground height,

$$\hat{z}_g(p_i^x, p_i^y) - p_i^z = 0, \tag{6}$$

can be informed using the height and traversability maps output by the vision system. Rather than setting the nominal ground height to be 0 as was done with the blind controller, the vision system provides an actual estimated ground height, $\hat{z}_g(p_i^x, p_i^y)$, at each potential footstep location. This change is simple, but smooths the results to preemptively expect contact with the ground at a certain height rather than have to react after the fact. Previously, the controller required off-board computing to send control results to the Mini-Cheetah Vision robot, but with the upgrades outlined in Section II, it is now run entirely on the robot without the need for tethering.

B. Whole-Body Impulse Control (WBIC)

Using the reaction forces found by the RPC, the WBIC computes joint position, velocity, and torque commands. Quadratic programming then finds the reaction forces that reduce both errors in acceleration command tracking and reaction force command tracking while satisfying inequality constraints on the resultant reaction forces. The resultant reaction forces are used to find torque commands from the multi-body dynamics which can be written as

$$\boldsymbol{A}\begin{pmatrix}\ddot{\boldsymbol{q}}_f\\ \ddot{\boldsymbol{q}}_j\end{pmatrix} + \boldsymbol{b} + \boldsymbol{g} = \begin{pmatrix}\mathbf{0}_6\\ \boldsymbol{\tau}\end{pmatrix} + \boldsymbol{J}_c^\top\boldsymbol{f}_r, \tag{7}$$

where $\boldsymbol{A}$, $\boldsymbol{b}$, $\boldsymbol{g}$, $\boldsymbol{\tau}$, $\boldsymbol{f}_r$, and $\boldsymbol{J}_c$ are the generalized mass matrix, Coriolis force, gravitation force, joint torque, augmented reaction force and contact Jacobian, respectively. The term $\ddot{\boldsymbol{q}}_f \in \mathbb{R}^6$ is the acceleration of the floating base and $\ddot{\boldsymbol{q}}_j \in \mathbb{R}^{n_j}$ is the vector of joint accelerations, where n_j is the number of joints. We use $\mathbf{0}_6$ to represent a six-dimensional zero vector and $\mathbf{0}_n$ to represent a n-dimensional zero vector.

We compute the final reaction force with the acceleration command found in the previous step and the reaction force obtained from RPC. For the optimization, we use an open-source QP solver [31] that is efficient for small problems. The formulation of our QP problem is

$$\min_{\boldsymbol{\delta}_{\boldsymbol{f}_r}, \boldsymbol{\delta}_f} \quad \boldsymbol{\delta}_{\boldsymbol{f}_r}^\top\boldsymbol{Q}_1\boldsymbol{\delta}_{\boldsymbol{f}_r} + \boldsymbol{\delta}_f^\top\boldsymbol{Q}_2\boldsymbol{\delta}_f \tag{8}$$

s.t.

$$\begin{aligned} &\boldsymbol{S}_f\left(\boldsymbol{A}\ddot{\boldsymbol{q}} + \boldsymbol{b} + \boldsymbol{g}\right) = \boldsymbol{S}_f\boldsymbol{J}_c^\top\boldsymbol{f}_r && \text{(floating base dyn.)}\\ &\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}^{\text{cmd}} + \begin{bmatrix}\boldsymbol{\delta}_f\\ \mathbf{0}_{n_j}\end{bmatrix} && \text{(acceleration)}\\ &\boldsymbol{f}_r = \boldsymbol{f}_r^{\text{RPC}} + \boldsymbol{\delta}_{\boldsymbol{f}_r} && \text{(reaction forces)}\\ &\boldsymbol{W}\boldsymbol{f}_r \ge \mathbf{0}, && \text{(contact force constraints)} \end{aligned}$$

where $\boldsymbol{f}_r^{\text{RPC}}$ and $\boldsymbol{S}_f$ are reaction forces computed by the RPC and the floating base selection matrix, respectively. $\boldsymbol{J}_c$ and $\boldsymbol{W}$ are the augmented contact Jacobian and contact constraint matrix, respectively. $\boldsymbol{\delta}_f$ and $\boldsymbol{\delta}_{\boldsymbol{f}_r}$ are relaxation variables for the floating base acceleration and reaction forces. The method to find the joint position, velocity, and acceleration commands reflecting prioritized task hierarchy is explained in more detail in [25].

C. Path Planning

To efficiently reach the desired goal location without colliding into obstacles, we implemented an A* search variant introduced in [32]. The operator selects a 3D coordinate within the world frame as the destination goal which gets compared against the robot's current center-of-mass position computed by the state estimator. If the distance between the two points is larger than some small tolerance, the planner begins computing a path consisting of a discrete set of 3D points in the world frame. We chose a grid resolution of $10\ \mathrm{cm}$ and found that this value was appropriate in terms of computation time and optimality.

The planner runs at 100 Hz on the NVIDIA Jetson TX2 and continuously publishes paths over LCM to the robot's UP board computer where the locomotion controller executes the path. This continuous replanning allows the robot to be able to autonomously traverse dynamic environments as well as be robust to disturbances such as getting pushed or knocked over. Furthermore, since the selected goal point may initially be placed outside of the range of the perception sensors, continuous replanning allows the robot to modify its trajectory if it encounters a new obstacle anywhere along the path. If the goal point is placed so far away that the computation time would exceed the planner's alotted time to run, we time-bound the algorithm and return a path to an explored node in the graph with the smallest Euclidean distance to the goal. Planning is currently done only in 2D along the x and y axes for computational efficiency, but we note that the robot is still able to climb objects and change its z-coordinate so long as the calculated gradient along the path is below the threshold value during traversability map generation.

Once the planner finishes computing, the path is passed along to the lower-level controller to be converted into a meaningful command. To follow the path, the controller computes desired linear and angular velocities in a PD fashion based on where the next waypoint is located. We constrain the commands to only forward linear velocities and angular yaw rates in order to keep the robot in line with the path and facing the next waypoint, and take in as an input the maximum desired forward velocity. We note that in order to prevent jerkiness during locomotion, we smooth out the path by only requiring the robot's CoM to come within $5\ \mathrm{cm}$ of the currently tracked waypoint before moving on to the next one. Path tracking stops once the robot CoM reaches the final node in the planned path.

D. Posture Adaptation

We adjust the height and pitch of the robot based on the shape of the terrain where the robot is standing. In the computation of the height, the current stance feet heights and upcoming step location heights are used to find the average ground height. The desired height is then set as a constant offset from the average ground height. However, simply updating the desired height based on the upcoming location can cause sudden jumps in the command, and therefore, we use a low-pass filter to avoid a jerky height command.

Mini-Cheetah Vision adjusts to sloped terrain by changing the pitch of its body relative to the positions of its four feet. We represent individual footstep locations as $\boldsymbol{p}_i = (p_i^x, p_i^y, p_i^z)$ for $i = 4$ feet, and have $\boldsymbol{p}^x$, $\boldsymbol{p}^y$, $\boldsymbol{p}^z$ represent the locations of each foot in x, y, z respectively, e.g. $\boldsymbol{p}^x = (p_1^x, p_2^x, p_3^x, p_4^x)$. We calculate the least-squares plane for the feet contacting the ground to derive the local slope of the walking surface, where the surface is modeled as a plane:

$$z(x,y) = a_0 + a_1 x + a_2 y. \tag{9}$$

Using the plane, we determine the angle of pitch and adjust the desired posture accordingly. The coefficients $\boldsymbol{a} = (a_0, a_1, a_2)^T$ of (9) are calculated from the least squares problem:

$$\boldsymbol{a} = (\boldsymbol{W}^T\boldsymbol{W})^\dagger \boldsymbol{W}^T \boldsymbol{p}^z \tag{10}$$

$$\boldsymbol{W} = [\mathbf{1} \quad \boldsymbol{p}^x \quad \boldsymbol{p}^y]_{4\times 3}. \tag{11}$$

This finds the best-fitting plane for the collection of points for the most recent foot contact of all four legs. Note that $(\boldsymbol{W}^T\boldsymbol{W})^\dagger$ is the pseudoinverse of the non-square matrix $\boldsymbol{W}^T\boldsymbol{W}$. Pitch posture adaptation was constructed following the work of [33].

IV. Results

This section presents our results from experiments in a variety of real-world terrains using the fully-integrated Mini-Cheetah Vision robotic platform. Simulated environments were created to test the framework and visualize the path replanning as obstacles enter the field of view. Real-world testing in both indoor and outdoor environments with terrain surface changes as well as naturally-occurring obstacles verified the upgraded robot's online autonomy.

A. Path Planning in Simulation

Experimentation directly on the robotic hardware can be tedious and risky, so we first tested our entire implementation in simulation using the MIT Biomimetic Robotics Lab's open-source simulator[1]. In particular, we tested our fully-integrated A* path planner in numerous simulated environments with varying difficulties of terrain including flat ground, uneven ground, narrow hallways (Fig. 4), mazes (Fig. 5), and stairs (Fig. 6). In all cases, the robot was able to reach the desired goal point at velocities well surpassing $1\ \mathrm{m/s}$ without colliding into any walls or obstacles. Furthermore, the robot was able to successfully replan its path when the user changed the location of the waypoint during locomotion.

[1] `https://github.com/mit-biomimetics/Cheetah-Software`

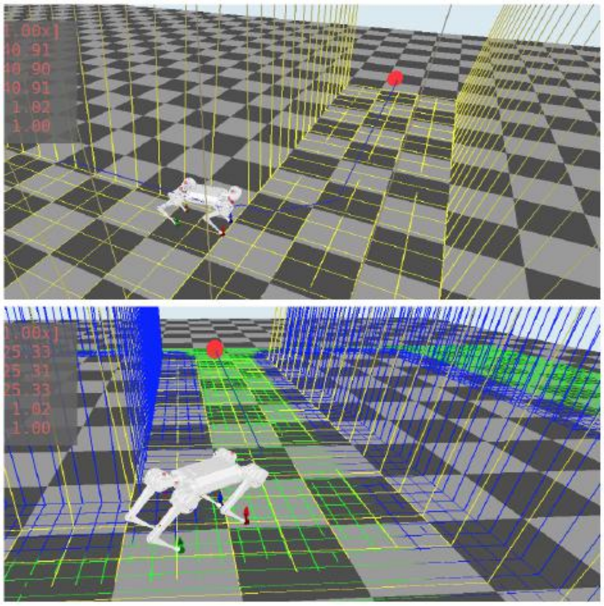

Fig. 4. **Path planning through a simulated hallway.** (a) The robot autonomously navigates through a simulated hallway to the desired waypoint. The planned path is displayed as a blue line. (b) The local traversability map is overlaid in simulation. The blue grid represents non-traversable locations for the robot's center of mass.

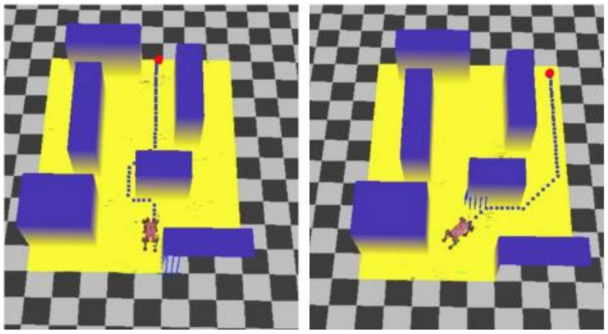

Fig. 5. **Path planning through a simulated maze.** The robot continuously replans as the waypoint (red) is moved throughout the maze. The planned path is displayed as a blue points.

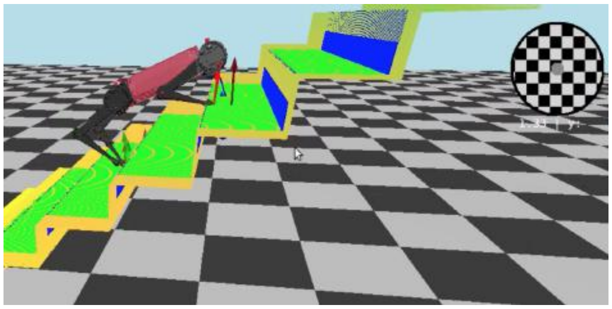

Fig. 6. **Path planning over stairs.** The path planner recognizes stairs as traversable and the heightmap allows the robot to accurately estimate its state on non-flat terrain, enabling the robot to climb stairs with steps that are 25 cm deep and 15 cm tall. state estimation and

B. Path Planning on Hardware

After verifying our code in simulation, we ran various experiments on the actual hardware platform in real-world environments. In particular, we chose environments, both indoor and outdoor, with assortments of obstacles for the robot to avoid including trees, rocks, shrubbery, uneven ground, buckets, and trash cans. During each of the tests, we manually set a waypoint approximately 40 m away from the robot and then let it walk on its own. We emphasize that the selection of the waypoint was the only human interaction required and that the rest of the experiments were completed in an entirely autonomous manner with the robot achieving an average speed of just above 1 m/s. Specific demonstrations of the real-time path planning implementation on hardware are shown in Fig. 7 and Fig. 8.

In addition to the general real-world tests, we also specifically evaluated the robot's ability to continuously replan trajectories in the presence of randomly-applied disturbances. We found that when a human pushed the robot while moving, it was consistently able to maintain balance and replan its path to reach the desired waypoint despite being thrown off of the original trajectory. Fig. 9 displays the logged results of the x and y positions of the robot, where it is clear that the robot recovered after each shove, showcasing its robustness to unforeseen disturbances in the real world.

C. Posture Adaption to Terrain

Without prior knowledge of the terrain, the Mini-Cheetah Vision robot is capable of adjusting its posture to changes in slope by deriving the plane of the feet location and adjusting the body pitch accordingly. This enables the robot to more robustly handle the traversal of unstructured environments where uneven footing makes a level body posture less advantageous, as is often the case when walking up ramps or climbing staircases. Fig. 10 demonstrates the difference between no posture adjustment in (a) and dynamic pitch control due to the sloped platform in (b). The robot is capable of deriving the best fitting plane of its feet contacts in realtime and using the forward aligned slope angle to adjust its own body posture pitch.

V. Conclusion and Future Work

The work presented in this paper represents a culmination of research efforts towards robot autonomy. The philosophy of the work by the MIT Biomimetic Robotics lab focuses on developing blind locomotion controllers [5], [34] to be able to traverse unstructured terrain without the reliance on external perception sensors, but to use knowledge of the environment when available to augment the locomotion controllers and autonomously plan paths. The blind controllers are able to tackle challenging environments including cluttered debris and stairs, as demonstrated in [33], [35]. However, since this work was done within controlled environments with user-guided velocity commands and simple rigid trajectories that did not incorporate intelligent planning, the robot would fail to recognize non-traversable obstacles in its path. Recently, a visual perception system was added to the Mini-Cheetah Vision robot in order to understand its environment and make modifications to simple CoM trajectories [18]. With the work

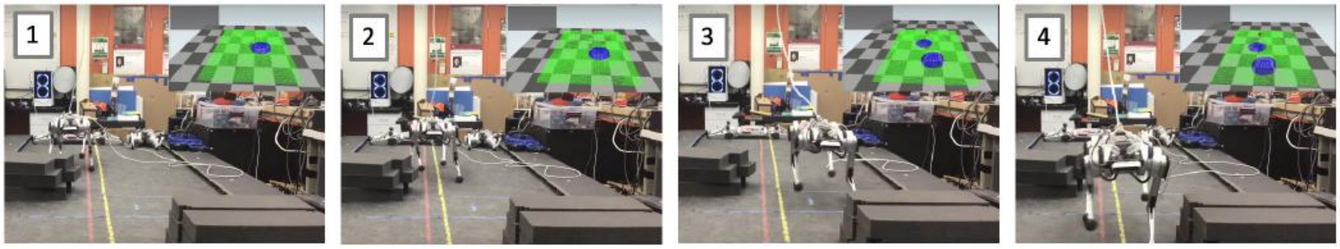

Fig. 7. **Real-time path planning and obstacle avoidance on hardware: Demo 1.** The robot recognizes and avoids two obstacles in its path as it walks across the platform. The visualization in the upper-right corner illustrates what the robot sees, with green cells representing traversable terrain, blue cells representing non-traversable terrain, and the blue line representing its planned path.

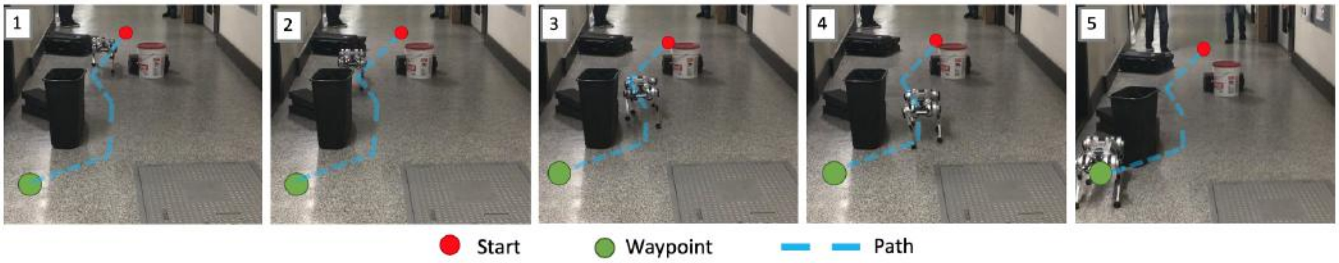

Fig. 8. **Real-time path planning and obstacle avoidance on hardware: Demo 2.** The robot walks through a cluttered hallway as it tracks the waypoint.

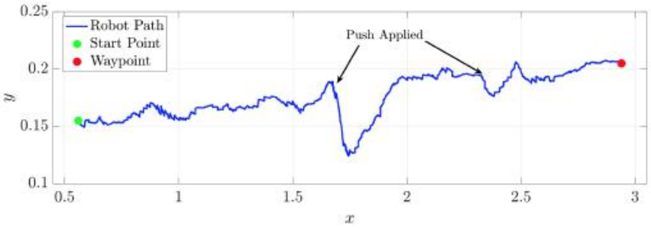

Fig. 9. **Perturbation-robust online path planning.** A graph showing the x and y position of the robot's CoM as it follows a planned trajectory. We see that the robot successfully reaches the desired waypoint even in the presence of human-applied pushes at randomly-selected times.

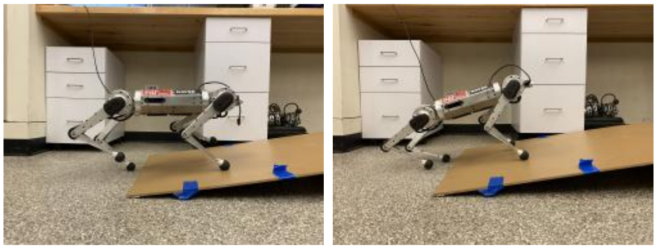

Fig. 10. **Posture pitch adjustment due to sloped terrain.** Mini-Cheetah Vision walking onto a sloped surface: (a) without pitch posture adjustment, and (b) with pitch posture adjustment added to the desired pitch of the body.

completed in this paper, we build on the aforementioned results by adding the capability for the robot to autonomously plan paths in arbitrary environments through higher-level decision making.

The added autonomy marks a crucial step towards deployment of robots in real-world environments. Despite the numerous successes in developing robots that can traverse any terrain, without a higher-level intelligent planner or constant operator guidance, the robot is not particularly useful in many of the intended applications of legged robots. By making use of the vision-modified planning framework presented, future work can now focus on goal-oriented tasks. For example, the robot will be able to receive a certain waypoint location to deliver a payload to and can continuously replan its trajectory based on the situation to autonomously avoid unexpected obstacles or other moving robots and humans.

In addition to adding more general autonomy, work is being done to reduce the computational load of the locomotion controllers for small-scale robots with limited compute power. With the NVIDIA Jetson TX2 computer, the RPC optimization is solved online at a frequency of only 50 Hz, which is within the requirements for steady-state locomotion, but as found in [29], the performance under disturbances diminishes with lower solve frequencies. At higher frequencies the controller has greater robustness against disturbances and overall better locomotion performance. An imitation learning approach in the form of behavior cloning similar to [36] is being developed to replace the difficult optimization problem with a fast neural network. The ground-truth labels for this supervised-learning problem will be collected from an expert demonstrator through simulation and used to train a policy network. Initial experiments performed using a neural network policy on the robot have shown promising results for balancing. This should allow robots with less computational power to approximate the robust results of RPC and greatly improve the accessibility of our presented work.

Acknowledgments

This work was supported by the Toyota Research Institute, Centers for ME Research and Education at MIT and SUSTech, Naver Labs, and the Air Force Office of Scientific Research.

References

[1] B. J. Stephens and C. G. Atkeson, "Push recovery by stepping for humanoid robots with force controlled joints," in *2010 10th IEEE-RAS International conference on humanoid robots*. IEEE, 2010, pp. 52–59.

[2] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.

[3] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based full body control for the darpa robotics challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[4] V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. R. De Pieri, and D. G. Caldwell, "A reactive controller framework for quadrupedal locomotion on challenging terrain," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 2554–2561.

[5] G. Bledt and S. Kim, "Implementing regularized predictive control for simultaneous real-time footstep and ground reaction force optimization," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Macau, China, Nov. 2019.

[6] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[7] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of hyq–a hydraulically and electrically actuated quadruped robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831–849, 2011.

[8] M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart, "Quadrupedal locomotion using hierarchical operational space control," *The International Journal of Robotics Research*, vol. 33, no. 8, pp. 1047–1062, 2014.

[9] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.

[10] B. Paden, M. Cáp, S. Z. Yong, D. S. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *CoRR*, vol. abs/1604.07446, 2016. [Online]. Available: http://arxiv.org/abs/1604.07446

[11] D. Zheng, H. Wang, Z. Xie, W. Chen, and X. Kong, "Autonomous navigation of a quadrotor in unknown environments," in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2017, pp. 1930–1935.

[12] A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz, "Anytime search-based footstep planning with suboptimality bounds," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, 2012, pp. 674–679.

[13] A. Hildebrandt, M. Klischat, D. Wahrmann, R. Wittmann, F. Sygulla, P. Seiwald, D. Rixen, and T. Buschmann, "Real-time path planning in unknown environments for bipedal robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1856–1863, 2017.

[14] N. Perrin, C. Ott, J. Englsberger, O. Stasse, F. Lamiraux, and D. G. Caldwell, "Continuous legged locomotion planning," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 234–239, 2017.

[15] J. Chestnutt, "Navigation planning for legged robots," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, December 2007.

[16] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1184–1189.

[17] ——, "Navigation planning for legged robots in challenging terrain," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1184–1189.

[18] D. Kim, D. Carballo, J. D. Carlo, B. Katz, G. Bledt, B. W. T. 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)*, Paris, France, June 2020.

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

[20] 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.

[21] M. F. Fallon, M. Antone, N. Roy, and S. Teller, "Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 112–119.

[22] S. Nobili, M. Camurri, V. Barasuol, M. Focchi, D. Caldwell, C. Semini, and M. Fallon, "Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots," 2017.

[23] R. Hartley, M. G. Jadidi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3783–3790.

[24] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4507–4514, 2019.

[25] D. Kim, J. D. Carlo, B. Katz, G. Bledt, and S. Kim, "Highly Dynamic Quadruped Locomotion via Whole-Body Impulse Control and Model Predictive Control," *arXiv.org*, Sep. 2019.

[26] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots-consistent fusion of leg kinematics and imu," *Robotics*, vol. 17, pp. 17–24, 2013.

[27] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *Mobile Service Robotics*. World Scientific, 2014, pp. 433–440.

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

[29] G. Bledt, "Regularized predictive control framework for robust dynamic legged locomotion," Ph.D. dissertation, Massachusetts Institute of Technology (MIT), 2020.

[30] G. Bledt and S. Kim, "Extracting legged locomotion heuristics with regularized predictive control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, June. 2020.

[31] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical Programming*, vol. 27, no. 1, pp. 1–33, Sep. 1983.

[32] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. SSC-4(2), pp. 100–107, 1968.

[33] G. Bledt, M. J. Powell, B. Katz, J. D. Carlo, P. M. Wensing, and S. Kim, "MIT cheetah 3: Design and control of a robust, dynamic quadruped robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Madrid, Spain, Oct. 2018.

[34] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "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, 2018, pp. 1–9.

[35] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, May 2018.

[36] J. Carius, F. Farshidian, and M. Hutter, "Mpc-net: A first principles guided policy search," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, p. 2897–2904, Apr 2020. [Online]. Available: http://dx.doi.org/10.1109/LRA.2020.2974653