Robotics: Science and Systems 2018
Pittsburgh, PA, USA, June 26-30, 2018

Fast Online Trajectory Optimization for the Bipedal Robot Cassie

Taylor Apgar, Patrick Clary, Kevin Green, Alan Fern, and Jonathan Hurst
Collaborative Robotics and Intelligent Systems Institute
Oregon State University
Corvallis, Oregon 97331
{*apgart, claryp, greenkev, alan.fern, jhurst*}*@oregonstate.edu*

***Abstract*—We apply fast online trajectory optimization for multi-step motion planning to Cassie, a bipedal robot designed to exploit natural spring-mass locomotion dynamics using lightweight, compliant legs. Our motion planning formulation simultaneously optimizes over center of mass motion, footholds, and center of pressure for a simplified model that combines transverse linear inverted pendulum and vertical spring dynamics. A vertex-based representation of the support area combined with this simplified dynamic model that allows closed form integration leads to a fast nonlinear programming problem formulation. This optimization problem is continuously solved online in a model predictive control approach. The output of the reduced-order planner is fed into a quadratic programming based operational space controller for execution on the full-order system. We present simulation results showing the performance and robustness to disturbances of the planning and control framework. Preliminary results on the physical robot show functionality of the operational space control system, with integration of the trajectory planner a work in progress.**

I. Introduction

In this paper, we present preliminary results from using fast online trajectory optimization to control Cassie, a bipedal robot with lightweight, compliant legs.

Cassie (Figure 1), built by Agility Robotics, is the latest in a line of bipedal robots designed to use *spring-mass* principles to walk and run. Its direct predecessors include Mabel [6] and ATRIAS [15] [7], which have respectively demonstrated planar and 3D walking and running. These robots were capable of moving over uneven terrain while blind to the upcoming ground surface using a combination of lightweight legs with physical spring elements and compliant actuation control. Cassie was designed with additional degrees of freedom in its legs that make it physically capable of moving through complex and structured terrain that cannot be handled by the blind-walking approach used by its predecessors.

To allow Cassie to navigate obstacles that require advanced planning, we have implemented a fast online trajectory optimization formulation for multi-step motion planning. This formulation plans over a reduced-order model of the robot with a fixed contact schedule. The details of this formulation are covered in Section IV, with discussion of prior formulations that it builds upon in Section II.

The online planner utilizes a reduced-order model of the robot, which must then be mapped to the full-order robot. This is accomplished using an implementation of operational space control (OSC) for controlling the acceleration of selected points on the physical robot. The OSC formulation is discussed is Section V.

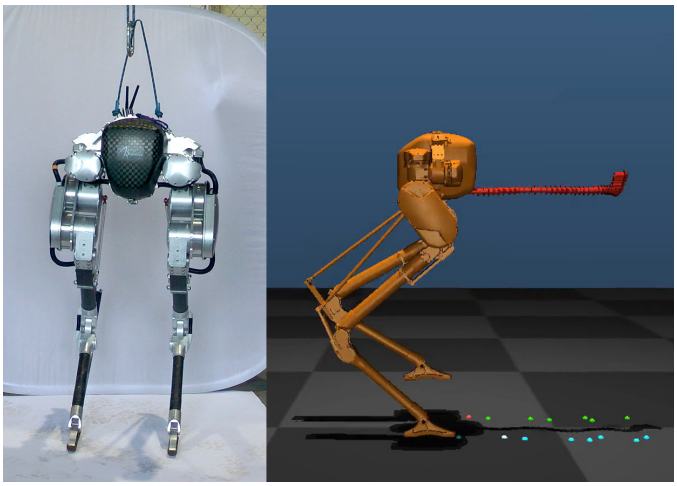

Fig. 1. Left: Cassie standing in place using operational space control. Right: Simulated Cassie showing center of mass trajectory and footsteps planned with reduced-order model.

Experiments on a simulated robot are discussed in Section VI-B. The simulated robot demonstrates walking on flat terrain with more precision than the robot's previous controller, inspired by the blind control used on its predecessors, provides. Robustness to unanticipated disturbances is also demonstrated.

Preliminary experiments with the physical robot are discussed in Section VI-C. Details of the real-time system implementation are presented, and the robot is shown standing and rejecting disturbances using the OSC implementation from Section V. We also discuss some of the challenges encountered during the physical experiments and how they have shaped the development of our control and planning formulation.

II. Background

The problem of controlling legged robots to walk and run is made difficult by nonlinearities, hybrid dynamics, strict constraints on ground reaction forces, instability, high dimensionality, and underactuation. Planning and executing state trajectories is one general approach that has been used to cope with these challenges.

The most general approach for trajectory planning is to calculate an optimal trajectory from the current state to the goal state for the full order robot. One approach for full body online trajectory optimization is differential dynamic programming, which is an unconstrained optimization approach. While DDP was demonstrated on a HRP-2 humanoid running at 20 Hz [11], it is generally not a very robust approach due to the fact that it has to discover contact instead of explicitly reasoning about it. Another method is to solve the optimal control problem using constrained optimization by formulating the problem as a quadratic or nonlinear program. Humanoid robots have shown success with this approach for behaviors such as climbing steps and gait transitions [14]. The central drawback of these methods comes from computational constraints because the planning is done using a full-order model of the robot.

The computational difficulty of the trajectory planning problem can be greatly reduced by planning on a reduced-order model. This does restrict the space of possible movements that can be found, but with careful co-design of the robot and reduced-order model, computation can be significantly reduced without a corresponding reduction in effective fidelity. One common reduced order model (ROM)for walking is the linear inverted pendulum (LIP) [9]. This model is computationally advantageous due to its linear dynamics. A more complex model is the spring-loaded inverted pendulum (SLIP). This model is able to produce both walking and running, and more closely reproduces the ground reaction force profiles observed in humans [16]. Unlike the LIP model, the SLIP model has nonlinear dynamics which increases the difficulty of trajectory planning.

Tracking a trajectory from a reduced-order model requires a sophisticated lower-level control layer that maps behavior to the full-order model. A widespread approach used to track the reduced order plans is OSC. OSC is an inverse dynamics method to find torques that accurately track desired accelerations in the task space of a robot. It was originally developed to unify force and motion control for manipulators [10]. There are several different ways to apply OSC to floating base legged robots. One method is to perform an orthogonal decomposition [12, 8]. Another method, which is used in this work, is a quadratic program formulation which treats the equations of motion as a constraint [20].

III. System Overview

The planning and control architecture presented here uses a fast nonlinear programing formulation for multi-step foot and body motion planning and OSC for controlling the robot about the reference trajectories. Since whole body motion planning is computationally expensive we use a simplified reduced order model for planning multiple footsteps. The full robot dynamics are then considered in our OSC formulation.

The planner uses a reduced-order model for the robot's center of mass dynamics as a function of foot placement. The contact schedule used by the planner is fixed based on step time and percentage of time spent in the double stance phase,

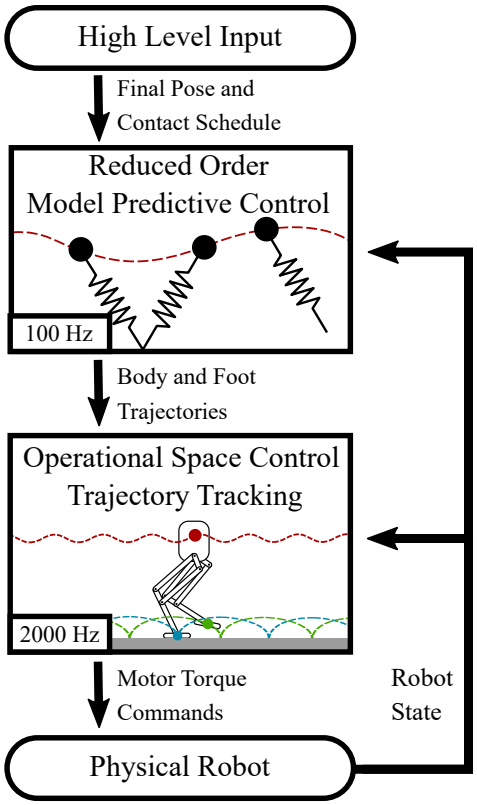

Fig. 2. The control hierarchy with execution frequency. Either a user or a higher level task planner inputs a goal state for the reduced order model as well as the contact schedule over time. The model predictive controller generates trajectories for the robot's body and feet which are tracked using operational space control through torque commands.

which are provided by the user. The phase sequence always cycles between double stance and single stance for a fixed number of steps. The planner optimizes over open parameters which are then converted to a center of mass trajectory and foot placements for OSC.

To get the target accelerations required by OSC, the center of mass trajectory found by the planner is analytically differentiated. The desired foot accelerations are determined by the contact schedule and the swing leg trajectory during single stance. OSC then finds the motor torques and contact forces required to reach these accelerations while obeying actuator and friction cone constraints.

As seen in Figure 2, the planner can be run at about 100 Hz, while the OSC runs at 2 kHz. Because the planner executes quickly, a Model Predictive Control (MPC) approach is used, where the planning parameters are continuously re-optimized as the robot moves. Once a new plan is found, it is sent to the controller, which accounts for the planning latency by jumping forward in the plan. This constant fast re-planning gives the robot robustness to disturbances and other environmental uncertainties.

IV. Trajectory Optimization

Whole body trajectory optimization for a complex walking robot with an undefined contact sequence is too computationally expensive to be done online at an acceptable rate. The first simplification that we use is to plan over a reduced-order dynamic model instead of a full-order model of the robot's dynamics. The second simplification that we use is to predefine the contact schedule. A predefined schedule of when which of the robot's feet will be on the ground is sufficient for the vast majority of desired walking behavior. It is not until the robot experiences extreme disturbances that it significantly benefits from allowing the optimizer to change the sequencing and duration of distinct contact phases.

A. Problem Formulation

This section describes how the nonlinear programming problem used for trajectory optimization is formulated. First, we describe the reduced-order model that allows closed form integration from [13], which is used for describing the robot's center of mass motion. Next, we discuss how foot placement and center of pressure are parametrized, as well as their constraints. Finally, we describe the swing leg trajectory and how planning is integrated with control.

The user provides the planner with the step time, double stance percentage, step height, number of steps, and desired end position. The planner then outputs a continuous center of mass trajectory and target foot positions. The trajectory optimization problem is formulated such that there are a fixed number of open parameters ϕ per phase. With two phases per step (single/double stance), there are $2N_{step}N_\phi$ total open variables, where N_{step} is the number of steps and N_ϕ is the number of open variables per phase. In our formulation there are 25 open parameters per phase. The states, control actions and parameters are

$$\boldsymbol{x}_i(t) = [\boldsymbol{c}_i(t) \quad \dot{\boldsymbol{c}}_i(t) \quad \theta_i(t) \quad \dot{\theta}_i(t) \quad \boldsymbol{p}_{L_i} \quad \boldsymbol{p}_{R_i}] \tag{1a}$$

$$\boldsymbol{u}_i = [\ddot{\theta} \quad \boldsymbol{\lambda}_0 \quad \boldsymbol{\lambda}_T \quad r_0 \quad r_T] \tag{1b}$$

$$\boldsymbol{\phi}_i = [\boldsymbol{x}_i(0) \quad \boldsymbol{u}_i]. \tag{1c}$$

The state $\boldsymbol{x}(t)$ of the reduced order model includes the positions and velocities of the main body and feet, while the control action $\boldsymbol{u}(t)$ includes the angular acceleration, vertex weightings, and spring reference.

The initial state of the center of mass for phase i is defined by the position $\boldsymbol{c}_i(0)$ and velocity $\dot{\boldsymbol{c}}_i(0)$, along with the heading $\theta_i(0)$ and rotational velocity $\dot{\theta}_i(0)$. The foot positions $\boldsymbol{p}_{L_i}$ and $\boldsymbol{p}_{R_i}$ contain the transverse position and angle of the foot at the end of the phase. The center of pressure is defined by vertex weightings $\boldsymbol{\lambda}$, which is a four dimensional vector in our application. The initial and final vertex weightings of the phase are optimized. Finally, we allow the reference position of the spring to change linearly through stance, where the initial and final positions are defined by r_0 and r_T, respectively.

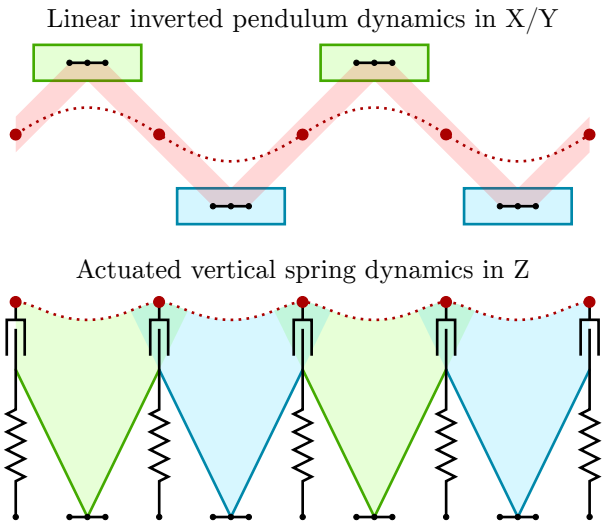

Fig. 3. A schematic of the different dynamics models used in the transverse (X/Y) and vertical (Z) dimensions. By splitting the model dynamics in this fashion, the majority of Cassie's physical dynamics are captured while keeping the model computationally efficient.

The trajectory optimization problem can be formulated as the nonlinear program

$$\begin{aligned}
&\text{find} && \phi_i, \quad for\ i \in [0, 2N_{step}] && && \text{(2a)}\\
&\text{subject to} && \boldsymbol{x}(0) - \boldsymbol{x}_0 = 0 && \text{(initial state)} && \text{(2b)}\\
&&& \boldsymbol{x}_{i+1}(0) - \boldsymbol{x}_i(T_i) = 0 && \text{(dynamics)} && \text{(2c)}\\
&&& \boldsymbol{h}(\phi_i) \leq 0 && \text{(kinematics)} && \text{(2d)}\\
&&& \boldsymbol{x}_{2N_{step}}(T) - \boldsymbol{x}_T = 0 && \text{(final state).} && \text{(2e)}
\end{aligned}$$

The constraints roughly fall into one of three categories: Center of mass motion, foot placement, and center of pressure motion. These constraints will be discussed in detail in the following section.

B. Equations of Motion

The spring reference r and vertex weightings $\boldsymbol{\lambda}$ are constrained to move linearly between the initial and final values over the phase, as shown by

$$r(t) = \frac{t}{T}(r_T - r_0) + r_0 \tag{3}$$

$$\boldsymbol{\lambda}(t) = \frac{t}{T}(\boldsymbol{\lambda}_T - \boldsymbol{\lambda}_0) + \boldsymbol{\lambda}_0, \tag{4}$$

where T is the time duration of the phase.

As previously mentioned, we are using the simplified robot model first discussed in [13]. The model uses a linearized inverted pendulum to describe the dynamics in the transverse directions, and a linear spring with equilibrium point actuation to describe the dynamics in the vertical direction. A schematic of the model is shown in Figure 3.

The transverse positions $x(t)$ and $y(t)$ are given by

$$x(t) = \beta_1 e^{\alpha t} + \beta_2 e^{-\alpha t} + u_x(t), \tag{5}$$

where

$$\beta_1 = (x_0 - u_{0,x})/2 + (\dot{x}_0 T - (u_{T,x} - u_{0,x}))/(2\alpha T), \quad (6a)$$

$$\beta_2 = (x_0 - u_{0,x})/2 - (\dot{x}_0 T - (u_{T,x} - u_{0,x}))/(2\alpha T), \quad (6b)$$

$$\alpha = \sqrt{g/z_0}, \quad (6c)$$

and $u(t)$ is the center of pressure position (see Section IV-D), and $x_0, z_0, \dot{x}_0$ are the initial center of mass positions and velocity. The equations of motion for $y(t)$ are given by the same equations but with the corresponding center of pressure dimension. As can be seen, the transverse position has a closed form solution. This is primarily due to a constraint on the center of pressure forcing it to only change linearly over the duration of the phase.

The vertical position $z(t)$ is given by

$$z(t) = d_1 \cos(\omega t) + d_2 \sin(\omega t) + r(t) - g/\omega^2, \quad (7)$$

where

$$d_1 = z_0 - r_0 + g/\omega^2 \quad (8a)$$

$$d_2 = \dot{z}_0/\omega - (r_t - r_0)/(T\omega), \quad (8b)$$

$$\omega = \sqrt{k/m}, \quad (8c)$$

and k is the constant spring stiffness and m is total mass. The vertical position also has a closed form solution due to the linear constraint on the reference spring motion during the phase. The motion of the main body heading is determined according to the linear equation

$$\theta(t) = \theta_0 + \dot{\theta}_0 t + \frac{1}{2}\ddot{\theta} t. \quad (9)$$

C. Foot Motion

The left and right foothold locations ($\boldsymbol{p}_{L_i} and \boldsymbol{p}_{R_i}$) are chosen during each phase of the trajectory. The foot locations are constrained such that the location of a foot in single stance phase must equal its location during the previous double stance phase. Likewise, the foothold location of a foot in swing phase should equal its location in the subsequent double stance phase.

In addition to the phase constraints, we also have to impose a reachability constraint to make sure the foot location is physically possible, given the center of mass motion over the phase. The reachability constraint is

$$\boldsymbol{p}_{nom} - \boldsymbol{r}^{x,y} < \boldsymbol{R}_{\theta_i(0)}(\boldsymbol{p}_{j_i} - \boldsymbol{c}_i(0)) < \boldsymbol{p}_{nom} + \boldsymbol{r}^{x,y}, \quad (10)$$

where $\boldsymbol{p}_{nom}$ is the nominal offset of the foot from the main body, $\boldsymbol{r}^{x,y}$ is the reachability constraint in body coordinates in the x and y directions, $\boldsymbol{p}_{j_i}$ is the left or right foot position, and $\boldsymbol{R}_{\theta_i(0)}$ is the rotation matrix from global to local coordinates according to the body yaw at the beginning of the phase as seen in Figure 4. Because of the stitching constraints, the foot position is also constrained according to the center of mass position at the end of the phase, and we therefore do not need to add that additional constraint here. The foot rotations are similarly constrained according to the rotation of the main body. An additional constraint is added to keep rotation of the swing leg the same as the rotation of the main body on touchdown.

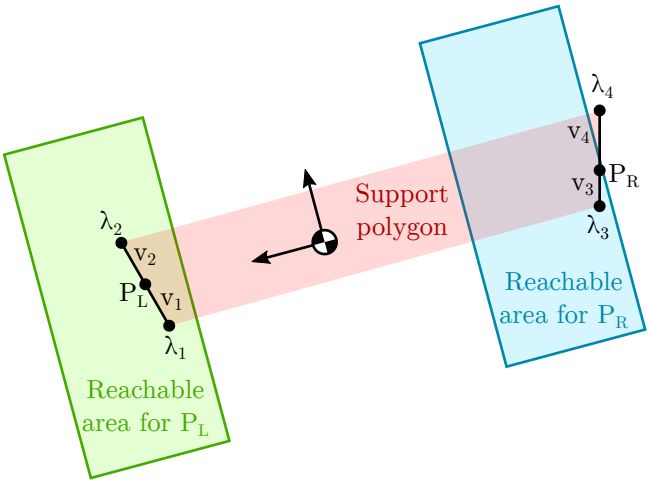

Fig. 4. Feet positions are constrained to be within the reachable areas shown by the rectangles on the left and right. The vertex weightings λ are applied at the front and back of each foot, which are centered at P_L and P_R.

D. Center of Pressure Motion

The benefits of using a vertex representation for the support polygon in double stance were first discussed in [21]. We use the same formulation here for determining the center of pressure (COP) motion over a phase. Unlike [21], we constrain the vertex weightings to change linearly over a phase to allow the closed form equations of motion discussed in Section IV-B. While this may seem overly restrictive, in practice it keeps the desired COP from rapidly jumping between points or getting too close to the edge of the support area. Large jumps in the desired COP are undesirable when operating on a real robot, as they will likely correspond to infeasible motions.

The center of pressure is calculated from the vertex weightings according to

$$\boldsymbol{u}(t) = \sum_{j \in L,R} \sum_{v=1}^{2} \lambda_j^v(t)(\boldsymbol{p}_j(t) + \boldsymbol{R}_{\theta_j(t)} \boldsymbol{v}_v), \quad (11)$$

where $\boldsymbol{v}$ is the vector from the center of the foot to the vertex point, $\boldsymbol{R}_{\theta_j(t)}$ is the rotation matrix of the foot, and $\boldsymbol{\lambda}(t)$ is calculated according to 4. This equation simply multiplies each vertex weighting by it's corresponding position to determine the center of pressure.

The vertex weightings are constrained such that they should sum to one and that values of the weightings on a foot that is not in contact should equal zero. These constraints are expressed as

$$\|\lambda\|_1 = 1 \quad (12a)$$

$$0 \leq \lambda_j^v(t) \leq c^j(t), \quad (12b)$$

where $c^j(t) \in 0, 1$ is the contact state for foot j which comes from the predefined mode schedule. If the foot is not in contact, then the vertex weightings on that foot must equal zero. This ensures that the center of pressure remains inside the support area.

E. Objectives

Our primary concern in designing the objective function is that the resulting behavior should be robust to disturbances and uncertainties in the environment. This means that the center of mass motion should be smooth. Additionally, as was previously mentioned, the COP should stay close to the center of the support polygon. The COP criteria is already enforced by constraining its motion to change linearly over a given phase. To produce smooth center of mass motion, we use the objective

$$\sum_{i=1}^{2*N_{step}} \sum_{t\in[0,\frac{T_i}{2}]} \ddot{c}_i(t)^2 \frac{T_i}{2} \tag{13}$$

which is the sum of body acceleration squared for discretized points along the trajectory. In [13], the authors integrated the acceleration squared over the plan. While integrating the continuous function is the correct measure, we found that it caused the optimization to run longer without a noticeable difference in the resulting solution. As we discuss in Section VI-B, the primary contributer to robustness was the planning frequency. The exact objective used was found to be much less important.

V. Trajectory Tracking

A controller is required to select torques to command to the physical robot such that it accurately tracks the body and foot trajectories generated by the MPC. The motion plan could include highly dynamic behaviors, so inertial and velocity product effects may be significant. OSC accounts for these effects by solving inverse dynamics on the full order rigid body model of the robot.

A. Constrained Forward Dynamics

The robot is modeled as a floating base rigid body system with generalized coordinates $q = [q_b; q_r] \in \mathbb{R}^n$. Here, $q_b \in \mathbb{R}^6$ is the position and orientation of the floating base (specified as three translational coordinates and three Euler angles) and $q_r \in \mathbb{R}^{n-6}$ is the joint configuration. The robot has p point contacts through which forces can be applied and m motors which are mapped to joint space through the transpose of the actuated joint selection matrix S_a^T.

The equation of motion is

$$M(q)\ddot{q} + h(q,\dot{q}) = S_a^T\tau + J_c(q)^T f + J_{eq}(q)^T\lambda, \tag{14}$$

where M is the mass matrix, h is the velocity product and gravitational terms, $\tau \in \mathbb{R}^m$ is the applied motor torques, J_c is the combined support Jacobian, $f \in \mathbb{R}^{3p}$ is the ground reaction forces, J_{eq} is the closed loop constraint Jacobian and λ is the vector of constraint forces [1]. The constraint forces are necessary because each Cassie leg contains a closed kinematic loop. The closed loop imposes an equality constraint between two points on the rigid body tree.

The velocity constraint takes the form of

$$\dot{x}_{eq} = J_{eq}\dot{q} = 0, \tag{15}$$

where $\dot{x}_{eq}$ is the relative velocity of the two points that close the loop. The acceleration constraint is

$$\ddot{x}_{eq} = J_{eq}\ddot{q} + \dot{J}_{eq}\dot{q} = 0. \tag{16}$$

The loop closure forces λ required to satisfy (16) can be substituted into (14) to produce a constrained equation of motion,

$$M\ddot{q} + N_s^T h + \gamma = N_s^T S_a^T \tau + N_s^T J_c^T f, \tag{17}$$

where

$$\gamma(q,\dot{q}) = -J_{eq}^T(J_{eq}M^{-1}J_{eq}^T)^\dagger \dot{J}_{eq}\dot{q} \tag{18}$$

is the velocity dependent terms that are produced by the constraint and

$$N_s = I - J_{eq}^T(J_{eq}M^{-1}J_{eq}^T)^\dagger J_{eq}M^{-1} \tag{19}$$

is the dynamically consistent null space projector for the constraint. In these expressions, $(\cdot)^\dagger$ represents the Moore-Penrose pseudo-inverse.

B. Ground Reaction Force Constraints

The ground reaction forces are constrained to prevent the foot from lifting off or slipping. The standard method to describe point contact friction constraints is the friction cone. For a friction coefficient μ and a surface normal in the $+z$ direction, the space of valid ground reaction forces is

$$\mathcal{C} = \left\{ (f_x, f_y, f_z) \in \mathbb{R}^3 \;\middle|\; f_z \geq 0; \sqrt{f_x^2 + f_y^2} \leq \mu f_z \right\}. \tag{20}$$

This friction model presents a problem for constraining fast optimizations because it is a quadratic inequality constraint.

An alternative which we use here is a pyramidal friction model,

$$\mathcal{P} = \left\{ (f_x, f_y, f_z) \in \mathbb{R}^3 \;\middle|\; f_z \geq 0; |f_x|, |f_y| \leq \frac{\mu}{\sqrt{2}} f_z \right\}. \tag{21}$$

This model is more conservative than the friction cone model, $\mathcal{P} \subset \mathcal{C}$, but has the advantage that it takes the form of a set of linear inequality constraint on the ground reaction forces.

C. Swing Leg Trajectory

The swing leg trajectory is determined independently from the ROM optimization in Section IV. Once the foothold locations are determined from the optimizer a cubic polynomial trajectory is generated from the foot start position to the target position. A single trajectory is used for the transverse directions, while two trajectories are generated for the vertical direction. The first trajectory goes to the target step height midway through swing and the second trajectory goes to the target position. The cubic trajectory equation is

$$x_s(t) = \sum_{i=0}^{3} a_i t^i, \tag{22}$$

where $x_s(t)$ is the position of the swing foot at time t. The initial and final position and velocity of the foot are known so coefficients a_2 and a_3 can be found using

$$\begin{bmatrix} T^3 & T^2 \\ 3T^2 & 2T \end{bmatrix} \begin{bmatrix} a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} x_s(0) + \dot{x}_s(0)T - x_s(T) \\ \dot{x}_s(0) - \dot{x}_s(T) \end{bmatrix}. \quad (23)$$

As the swing leg trajectory is twice differentiable, we use the desired acceleration as a feed forward term in our OSC formulation. We also include a soft PD in operational space about the desired swing leg trajectory.

D. Operational Space Control

The goal of the Operational Space Controller is to choose a dynamically consistent set of torques, ground reaction forces and generalized accelerations that minimize the error in the operational space accelerations. Most approaches to this problem can be classified as hierarchical [17] or weighted [13] (although a hybrid form is possible). The advantage of the hierarchical approach is that the priority of the tasks is strictly enforced, whereas in the weighted approach a low weight task is able to interfere with a high weighted task. The advantage of the weighted approach is that it only requires the inverse dynamics to be computed once per time step.

A weighted quadratic program formulation of OSC is used here for execution speed. If the task space accelerations are able to be closely satisfied, there will be little to no conflict between tasks. Our MPC trajectory generator should generally produce trajectories that are possible for the full-order robot to track.

Our desired trajectories are described as foot and center of mass positions, velocities, and accelerations. We choose a commanded task space acceleration using a proportional-derivative plus feed-forward controller in operational space,

$$\ddot{x}_{cmd} = K_p(x - x_d) + K_d(\dot{x} - \dot{x}_d) + \ddot{x}_d \quad (24)$$

where K_p and K_d are diagonal feedback matrices. The true task space positions are found using forward kinematics and velocities with the task space Jacobian, $\dot{x} = A\dot{q}$.

These operational space commanded accelerations are tracked using torques found by solving the following optimization problem:

$$\begin{aligned} \underset{\ddot{q},\tau,f}{\text{minimize}} \quad & \|A\ddot{q} + \dot{A}\dot{q} - \ddot{x}_{cmd}\|^2_W && (25a) \\ \text{subject to} \quad & M\ddot{q} + N_s^T h + \gamma = N_s^T S_a^T \tau + N_s^T J_c^T f && (25b) \\ & f \in \mathcal{P}^p. && (25c) \\ & S_f f = 0 && (25d) \\ & \underline{\tau} \leq \tau \leq \bar{\tau} && (25e) \end{aligned}$$

Minimization of (25a) is equivalent to minimizing the weighted two-norm of error in task accelerations. The weighting matrix W is not constant over all phases of the gait. For example, when the leg is in swing phase, the weighting on the foot acceleration is much less than when it is in contact with the ground. Constraint (25b) restricts the accelerations, toques, and forces to be dynamically consistent. When a foot is in contact with the ground and applying forces, one option is to constrain $\ddot{q}$ such that the contact point has zero acceleration [20]. However, in [3] it was reported that simply using a large weight on a desired zero acceleration gave better stability, which is what we do here. The friction constraints on the ground reaction forces are enforced through (25c). When the foot is not in contact with the ground, we use the ground reaction force selection matrix S_f in (25d) to constrain those ground reaction forces to be zero. Motor torque limits are enforced through (25e) where $\underline{\tau}$ and $\bar{\tau}$ are vectors of the minimum and maximum motor torques.

VI. Experiments

The planning and control algorithms described above were implemented in simulation and are in the process of being implemented on the physical robot. The simulation results for the complete system are promising, and the physical robot is currently able to stand and move its limbs accurately using the real-time OSC implementation. We will first discuss some implementation details, follow with simulation experiments, and finally discuss current progress on hardware.

A. System Setup

The hierarchy of subsystems in this system were divided into the MPC planner, OSC controller, and either the simulated or physical robot. Our simulation of Cassie is built using MuJoCo [18], a fast physics simulator for multibody jointed that uses soft contacts. As the planner and controller run at different frequencies, we created two separate C++ applications that communicate to each other asynchronously over UDP network sockets. The nonlinear programming problem used by the planner was solved using Ipopt [19]. While Ipopt provides the ability to hot start the optimization, our implementation presently only partially does so by initializing the open variables with the previous iteration's solution. Fully implementing the hot start capability will improve the average run time even further. The OSC application uses the Rigid Body Dynamics Library (RBDL) [2] for rigid body algorithms and qpOASES [5] as the QP solver. qpOASES is an active set QP solver that supports hot starting the solution [4], which our implementation takes advantage of to achieve a 2kHz QP OSC update rate.

B. Simulation Results

As mentioned previously, robustness to disturbances is a primary concern for this planning and control framework. To test robustness, we applied different magnitude disturbances in different directions to the main body of the simulated robot as it tried to walk forward a meter. We ran this procedure with two different planning objective functions. One objective was to minimize the sum of acceleration squared, formulated by (13). The other was a null objective, meaning the optimizer merely tried to satisfy the constraints. As shown in Figure 5, disturbance forces were applied in the four cardinal directions at 3 different locations along the robot's path. Three different impulse magnitudes were tested, equivalent to a 0.2, 0.25, and

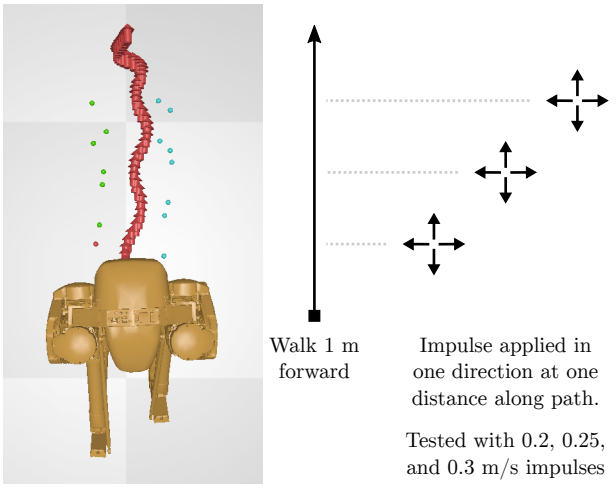

Fig. 5. Simulation experiment where the robot walks forward 1 m and varying impulses are applied at different segments in the trajectory.

	No Objective	$\sum \ddot{c}^2 T$
Run Time (ms) min|ave|max	5.4||6.6||71.0	18.9||60.3||120.4
Recover% 0.20 m/s impulse	100%	83%
Recover% 0.25 m/s impulse	91%	58%
Recover% 0.30 m/s impulse	75%	25%

TABLE I
SIMULATION RESULTS FOR DISTURBANCE FORCE REJECTION USING THE SUM OF SQUARE OBJECTIVE AND NO OBJECTIVE FORMULATIONS.

0.3 m/s change in the robot's velocity. A trial was considered a success if the robot reached the final position without falling. We collected data on run time statistics for the two objectives as well as the percentage of successful trials. The data can be found in table I.

We found that the null objective that simply satisfies constraints performed much better than the acceleration minimization objective. As can be seen in table I, the average replanning time is significantly faster for the null objective. This suggests that a small replanning time is a much larger contributor to robustness then the exact objective formulation used. Initially, it was surprising that the constraint satisfaction formulation produced such good results. However, after viewing many solutions, we realized that the problem formulation is actually fairly constrained. Therefore, faster solutions are typically better as the robot stays closer to feasible solutions. When the robot drifts too far between replanning iterations the robot state gets closer to the infeasible region which in turn increases the replanning times.

C. Hardware Results

We are currently able to test OSC on the physical Cassie robot. We are able to show stable crouching behavior, small disturbance rejection, and balancing on unstable and moving ground. The results of running OSC can be seen in a video accompanying this submission. We have begun to port the motion planner used for walking in simulation over to the physical robot. Current and future work, as well as some of the more interesting issues that have been encountered in the process, is discussed in section VII-A.

VII. CONCLUSION

In this paper we present an end to end planning and control architecture for a bipedal robot. In simulation, we demonstrate that the planning formulation presented above is robust to large disturbances and is very responsive to changes in the desired end goal. We show that this robustness is primarily due to the MPC formulation combined with the fast replanning. This allows us to reformulate the optimization problem into a constraint satisfaction problem with notable improvements in terms of robustness. Our OSC formulation is demonstrated on the physical Cassie hardware, showing that it can handle disturbances while standing.

A. Future Work

When implementing the OSC and planning on real hardware, we ran into several issues that we are working to address. The static friction in the actuated joints was not accounted for in the dynamic model. While Cassie has fairly transparent motors and transmissions, there is still a significant amount of friction in the gear boxes. These model inaccuracies affect the acceleration-to-torque calculation performed by OSC. This highlights the sensitivity to model inaccuracies of the OSC approach. Future work will attempt to better characterize the joint dynamics to allow for more precise acceleration control.

Another issue encountered is that the planner and controller operate in world coordinates, meaning the robot needs an estimate of its body position and velocity. These cannot be directly measured, so position and velocity are estimated by detecting foot contact, assuming no slip, and calculating them using kinematics. Performance of the planning and control algorithms will greatly improve as the body state estimator improves.

ACKNOWLEDGMENTS

This work was supported by DARPA contract W911NF-16-1-0002 and NSF Grant No. 1314109-DGE. We thank Agility Robotics for providing and supporting the Cassie robot.

REFERENCES

[1] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.

[2] Martin L Felis. RBDL: an efficient rigid-body dynamics library using recursive algorithms. *Autonomous Robots*, 41(2):495–511, 2017.

[3] Siyuan Feng, Eric Whitman, X Xinjilefu, and Christopher G Atkeson. Optimization-based Full Body Control for the DARPA Robotics Challenge. *Journal of Field Robotics*, 32(2):293–312, 2015.

[4] H.J. Ferreau, H.G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.

[5] H.J. Ferreau, C. Kirches, A. Potschka, H.G. Bock, and M. Diehl. qpOASES: A parametric active-set algorithm

for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[6] Jessy W Grizzle, Jonathan Hurst, Benjamin Morris, Hae-Won Park, and Koushil Sreenath. MABEL, a new robotic bipedal walker and runner. In *American Control Conference, 2009. ACC'09.*, pages 2030–2036. IEEE, 2009.

[7] Christian Hubicki, Jesse Grimes, Mikhail Jones, Daniel Renjewski, Alexander Spröwitz, Andy Abate, and Jonathan Hurst. ATRIAS: Design and validation of a tether-free 3D-capable spring-mass bipedal robot. *The International Journal of Robotics Research*, 35(12):1497–1521, 2016.

[8] M Hutter, M Hoepflinger, C David Remy, and R Siegwart. Hybrid Operational Space Control for Compliant Legged Systems. *Robotics Science and Systems (RSS)*, pages 129–136, 2012. ISSN 2330765X. doi: 10.3929/ethz-a-010184796. URL http://roboticsproceedings.org/rss08/p17.pdf.

[9] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 239–246. IEEE, 2001.

[10] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[11] Jonas Koenemann, Andrea Del Prete, Yuval Tassa, Emanuel Todorov, Olivier Stasse, Maren Bennewitz, and Nicolas Mansard. Whole-body model-predictive control applied to the HRP-2 humanoid. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3346–3351. IEEE, 2015.

[12] Michael Mistry, Jonas Buchli, and Stefan Schaal. Inverse Dynamics Control of Floating Base Systems Using Orthogonal Decomposition. *Control*, (3):3406–3412, 2010. doi: 10.1109/ROBOT.2010.5509646. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5509646.

[13] Igor Mordatch, Martin De Lasa, and Aaron Hertzmann. Robust physics-based locomotion using low-dimensional planning. In *ACM Transactions on Graphics (TOG)*, volume 29, page 71. ACM, 2010.

[14] Michael Posa, Scott Kuindersma, and Russ Tedrake. Optimization and stabilization of trajectories for constrained dynamical systems. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1366–1373. IEEE, 2016.

[15] Alireza Ramezani, Jonathan W Hurst, Kaveh Akbari Hamed, and Jessy W Grizzle. Performance analysis and feedback control of ATRIAS, a three-dimensional bipedal robot. *Journal of Dynamic Systems, Measurement, and Control*, 136(2):021012, 2014.

[16] William John Schwind. *Spring loaded inverted pendulum running: A plant model.* PhD thesis, 1998.

[17] L. Sentis and O. Khatib. Control of Free-Floating Humanoid Robots Through Task Prioritization. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 1718–1723, April 2005. doi: 10.1109/ROBOT.2005.1570361.

[18] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

[19] A Waechter and L T Biegler. *On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming*, volume 106. 2006. ISBN 1010700405.

[20] Patrick M Wensing and David E Orin. Generation of dynamic humanoid behaviors through task-space control with conic optimization. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3103–3109. IEEE, 2013.

[21] Alexander W Winkler, Farbod Farshidian, Diego Pardo, Michael Neunert, and Jonas Buchli. Fast trajectory optimization for legged robots using vertex-based zmp constraints. *IEEE Robotics and Automation Letters*, 2 (4):2201–2208, 2017.