Temporal Logic Guided Locomotion Planning and Control in Cluttered Environments

Sutej Kulgod*[1] Wentao Chen*[2] Junda Huang*[2] Ye Zhao[2] Nikolay Atanasov[1]

***Abstract*— We present planning and control techniques for non-periodic locomotion tasks specified by temporal logic in rough cluttered terrains. Our planning approach is based on a discrete set of motion primitives for the center of mass (CoM) of a general bipedal robot model. A deterministic shortest path problem is solved over the Büchi automaton of the temporal logic task specification, composed with the graph of CoM keyframe states generated by the motion primitives. A low-level controller based on quadratic programming is proposed to track the resulting CoM and foot trajectories. We demonstrate dynamically stable, non-periodic locomotion of a kneed compass gait bipedal robot satisfying complex task specifications.**

I. Introduction

Legged bipedal robots are becoming widely adopted to accomplish complex tasks in human environments, due to their potential to navigate cluttered rooms and buildings and interact with objects of interest. While humans can ambulate intelligently and robustly, legged bipedal robots face challenges in doing the same due to the inherent instability of their under-actuated locomotion dynamics and the complexity of planning for complex task specifications.

Numerous works have investigated motion planning and control of bipedal walking robots [1]–[5]. Few, however, have explored a comprehensive strategy integrating motion planning for high-level tasks and low-level control simultaneously. Our work proposes an approach that integrates task design, motion planning and optimal control of non-periodic dynamic legged locomotion over complex terrain.

Synthesis of high-level reactive task and motion plans has been widely studied in the formal method community [6]–[8]. Kress-Gazit et al. [9] propose a discrete controller for linear temporal logic (LTL) tasks that specify complex behavior for a ground mobile robot in a 2-D environment. Generalizing ground mobility tasks to highly dynamic locomotion, Zhao et al. [10] propose a whole-body locomotion planning approach by solving a two-player game between a bipedal robot contact planner and its possibly adversarial environment. Fu et al. [11] propose a reduction of a subclass of LTL formulas to a deterministic shortest path problem (DSP), achieving joint task and motion planning in probabilistic semantic maps. Similar to [11], we consider a known environment and reduce an LTL task specification to a DSP. Meanwhile, we incorporate the switching strategies in [10] to determine a set of feasible motions that achieve stable locomotion behaviors while maneuvering confined space and rough terrain environments.

We gratefully acknowledge support from ONR SAI N00014-18-1-2828 and NSF Award No. 1924978.

*The first three authors contributed equally to this work.

[1]S. Kulgod and N. Atanasov are with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093, USA {skulgod, natanasov}@ucsd.edu.

[2]W. Chen, J. Huang, and Y. Zhao are with the George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA {wenche, jhuang496, yzhao301}@gatech.edu.

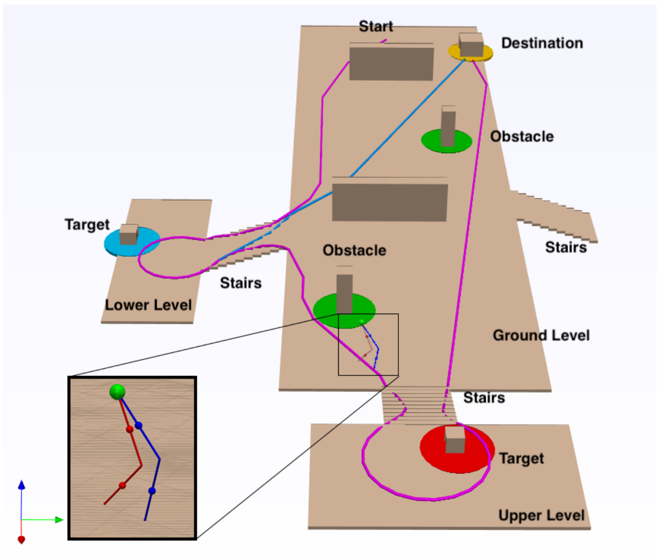


Fig. 1: Example executions of different tasks with a kneed bipedal robot in an environment with stairs and obstacles. The magenta curve is a plan for satisfying the task “visit a blue and a red region before going to a yellow region, while avoiding green regions.” The blue curve is a plan for a similar task requiring either a blue *or* a red region to be visited before the yellow region.

Our approach to efficient task planning hinges on the careful design of motion primitives that can be tracked by various bipedal robots. The primary aim of motion primitives is to describe a finite set of pre-computed feasible trajectories and use it to reduce the complexity of high-dimensional continuous-space planning problems. The use of motion primitives in planning for various bipedal robots has been explored in [12]–[15]. For example, a library of common high-dimensional trajectories for motions such as walking, climbing and the transitions between them is proposed in [13], [14]. In [12], a set of primitives for the compass-gait biped is derived with controlled reduction and a discrete-search algorithm is used to plan open-loop primitive sequences for walking. The importance of energy optimal trajectory generation is discussed in [16]. In this work, we propose a motion primitive design based on phase-space keyframe states [10], capturing characteristics about the robot's center of mass and stance foot, that allows generation of dynamically feasible optimal trajectories for a wide variety of bipedal robots.

Stabilizing locomotion trajectories over complex terrain is challenging, even if dynamically feasible CoM and foot trajectories are predefined to satisfy specified tasks. Quadratic programming (QP) is a powerful and well-established method for low-level control of bipedal robots during recent years. QP methods have been extensively investigated for controlling full-body humanoid robots [17], [18], and achieving control objectives subject to control barrier safety constraints [19]. To enable stable walking for bipedal robots, the work of [20], [21] introduces Zero Moment Point (ZMP) and Inverse Kinematics (IK) to the QP controller, while the work in [22] explicitly computes stability polygon and constructs task-based QP controllers based on quasi-static motion. Although their research has shown robustness on rough terrain to some degree, the walking process does not have the same success rate as dynamic walking. On the contrary, we implement a QP controller with trajectory re-planning to realize dynamic walking over rough terrain. As taking the keyframe based CoM trajectory and heuristic foot trajectory as inputs, we verify the feasibility of keyframe based motion primitives and achieve the locomotion trajectory tracking by using the model-based QP controller.

The contributions of this work are summarized as follows.

- We propose motion primitive design based on phase-space keyframe states that can be used with general bipedal robot models to achieve non-periodic motion.
- We demonstrate that a robot task specified via a linear temporal logic formula can be realised by solving a deterministic shortest path problem over motion primitive compositions.

II. Problem Formulation

Reduced-order models have been extensively employed for modeling dynamic legged locomotion. In this study, we develop a motion planner that relies on a prismatic inverted pendulum with center-of-mass (CoM) position constrained on a parametric surface as described in [23]. We assume instantaneous foot contact switching during walking. The start of a walking step is defined when the CoM is on top of the stance foot sagittal position, while its end is when the CoM is above the next stance foot location after one foot switch. Legged locomotion is described by hybrid swing and contact dynamics.

A. Discrete Dynamics

We first describe the discrete variables, which remain constant during the step duration. Let $t_k, t_{k+1}, \dots$ be the discrete times at which the steps begin. The *step state* at time t_k is defined as $\mathbf{s}_k := (\mathbf{p}_k^T, \psi_k)^T$, where $\mathbf{p}_k \in \mathbb{R}^3$ is the initial stance foot position and $\psi_k \in \mathbb{R}$ is the heading angle. Using $\mathbf{s}_k$ and a step length along the transverse plane $l_k \in \mathbb{R}_{>0}$, the change in height $\delta z_k \in \mathbb{R}$ for the step is determined based on the terrain. The evolution of the heading angle ψ_k is determined by a desired change in heading angle $\delta\psi_k \in \mathbb{R}$. Thus, the input determining the evolution of $\mathbf{s}_k$ is $\mathbf{u}_k = \{\delta\psi_k, l_k\} \in \mathbb{R}^2$, leading to discrete dynamics:

$$\begin{aligned} \mathbf{s}_{k+1} &:= \begin{bmatrix} \mathbf{p}_{k+1} \\ \psi_{k+1} \end{bmatrix} = f(\mathbf{s}_k, \mathbf{u}_k) \\ &:= \begin{bmatrix} \mathbf{p}_k + l_k R_z(\psi_k)\,\mathbf{e}_1 + \delta z_k \mathbf{e}_3 \\ \psi_k + \delta\psi_k \end{bmatrix} \end{aligned} \tag{1}$$

where $\mathbf{e}_i \in \mathbb{R}^3$ are standard basis vectors and $R_z(\psi) \in SO(3)$ specifies a rotation around the z-axis by angle ψ.

B. Continuous Dynamics

The remaining robot states evolve in continuous time during each walking step, specified by $\mathbf{s}_k$. Let the origin of the local coordinate frame of the k^{th} step be defined by the stance foot location $\mathbf{p}_k$. The x-coordinate is aligned with the walking direction, the y-coordinate is in the transverse plane perpendicular to x, and the z-coordinate is in the sagittal plane perpendicular to x. Let the CoM-state in local coordinates be $\boldsymbol{\xi}_k(t) := (\mathbf{x}_k(t)^T, \dot{\mathbf{x}}_k(t)^T)^T \in \mathbb{R}^6$, specified by its position $\mathbf{x}_k(t) := (x_k(t), y_k(t), z_k(t))^T \in \mathbb{R}^3$ and velocity $\dot{\mathbf{x}}_k(t) \in \mathbb{R}^3$ for $t \in [t_k, t_{k+1}]$. Let $\zeta_k \in [t_k, t_{k+1}]$ be the time at which the stance foot switches. The CoM height at the switching point $z_k(\zeta_k)$ as well as the apex velocity $\dot{x}_k(t_{k+1})$ and CoM height $z_k(t_{k+1})$ at the end of the step are constrained for each step in order to obtain desired gait characteristics. Let these constraints be specified by $\mathbf{v}_k = (\dot{x}_k(t_{k+1}), z_k(t_{k+1}), z_k(\zeta_k))^T \in \mathbb{R}^3$. Finally, let $\mathbf{c}(t) \in \mathbb{R}^6$ be the global coordinates of the CoM state $\boldsymbol{\xi}_k(t)$ for $t \in [t_k, t_{k+1}]$. The continuous-time CoM dynamics,

$$\dot{\mathbf{c}}(t) = g(\mathbf{c}(t), \mathbf{s}_k, \mathbf{s}_{k+1}, \mathbf{v}_k), \qquad t \in [t_k, t_{k+1}], \tag{2}$$

are defined precisely in Sec. IV.

C. Linear Temporal Logic Specifications

An LTL formula φ is used to specify the robot's task over a finite set of atomic propositions $\mathcal{AP}$, evaluated to be true when certain conditions are met by the robot's CoM state $\mathbf{c}(t)$. Temporal logic utilizes propositions, Boolean operators and temporal operators to represent system properties and requirements, especially the temporal ordering of events [24]. An LTL formula φ is composed of atomic propositions $\pi \in \mathcal{AP}$, based on the following grammar,

$$\varphi := \pi | \neg\varphi | \varphi \wedge \varphi | \varphi \vee \varphi | \bigcirc \varphi | \varphi \mathcal{U} \varphi$$

with Boolean constants True and False, logic operators negation ($\neg$), disjunction ($\vee$), conjunction ($\wedge$), implication ($\Rightarrow$), equivalence ($\Leftrightarrow$), and temporal operators next ($\bigcirc$), until ($\mathcal{U}$), eventually ($\Diamond\varphi :=$ True$\mathcal{U}\varphi$), and always ($\Box\varphi := \neg\Diamond\neg\varphi$).

A labeling function $\ell : \mathbb{R}^6 \to 2^{\mathcal{AP}}$ provides the set of propositions $\ell(\mathbf{c}(t))$ that evaluate true at the environment location specified by the CoM state $\mathbf{c}(t)$. We convert the task φ into a deterministic finite-state automaton (DFA) $\mathcal{A}_\varphi$ using the Spot library [25] (see Fig. 2). The DFA is defined as $\mathcal{A}_\varphi = \left(\mathcal{Q}, 2^{\mathcal{AP}}, h, q_0, \mathcal{Q}_S, \mathcal{Q}_F\right)$, where $\mathcal{Q}$ is a finite set of automaton states, $2^{\mathcal{AP}}$ is the alphabet, $h : \mathcal{Q} \times 2^{\mathcal{AP}} \to \mathcal{Q}$ is the transition function and $q_0 \in \mathcal{Q}$ is the initial state. The transition function h defines the continuous-time evolution of the discrete automaton state $q(t)$ as a function of the propositions satisfied by the CoM:

$$q^+(t) = h\left(q^-(t), \ell\left(\mathbf{c}(t)\right)\right), \quad t \in [t_k, t_{k+1}] \tag{3}$$

where $q^-(t)$ and $q^+(t)$ are the automation states before and after the instantaneous update at time t. The set $\mathcal{Q}_S \subset \mathcal{Q}$ denotes unaccepting sink states (e.g., collision) such that $q(t) = h(q(t), \ell)$ for all $\ell \in 2^{\mathcal{AP}}$ and the task φ can never be satisfied, while $\mathcal{Q}_F \subset \mathcal{Q}$ denotes accepting final states at which the task φ is satisfied. The planning problem considered in this work follows.

Problem 1. Let A_φ be a deterministic finite-state automaton with initial state q_0 specifying the robot task. Given an initial step state $\mathbf{s}_0$ and CoM state $\mathbf{c}(0)$, determine a control sequence $\mathbf{u}_k$ for $k = 0, \ldots, K-1$ that satisfies the task, i.e., an accepting automaton state $q(t_K) \in \mathcal{Q}_F$ is reached at beginning of the K^{th} step, while minimizing:

$$\begin{aligned}
\min_{K, \mathbf{u}_{0:K-1}, \mathbf{v}_{0:K-1}} & \sum_{k=0}^{K-1} \rho(\mathbf{c}(t_k), \mathbf{c}(t_{k+1})) + \mathbf{u}_k^T R \mathbf{u}_k + \mathbf{v}_k^T G \mathbf{v}_k \\
\text{s.t.} \quad & \mathbf{s}_{k+1} = f\left(\mathbf{s}_k, \mathbf{u}_k\right), \qquad k = 0, \ldots, K-1 \\
& \dot{\mathbf{c}}(t) = g(\mathbf{c}(t), \mathbf{s}_k, \mathbf{s}_{k+1}, \mathbf{v}_k), \quad t \in [t_k, t_{k+1}] \\
& q^+(t) = h\left(q^-(t), \ell\left(\mathbf{c}(t)\right)\right), \quad t \in [t_k, t_{k+1}] \\
& q(0) = q_0, \quad q(t) \notin \mathcal{Q}_S, \quad q(t_K) \in \mathcal{Q}_F,
\end{aligned} \tag{4}$$

where ρ is a positive definite motion cost function and R,G are positive definite matrices.

III. Task Specification and Keyframe State Selection

The key idea to solve Problem 1 is to define a finite set of *motion primitives* that serve to reduce the path planning problem to a finite-state deterministic shortest path problem. In our work, motion primitives are defined by a discrete set of possible inputs $\mathbf{u}_k$, constraints $\mathbf{v}_k$ and associated continuous-time CoM trajectories $\mathbf{c}(t)$ for $t \in [t_k, t_{k+1}]$ that can be selected at the beginning of each step. This section describes the task specification φ and the selection of constraints for motion primitive generation.

A. Task Specification

Let $\mathcal{E} \subset \mathbb{R}^3$ represent the environment in which the robot is operating. The environment $\mathcal{E} = \cup_i \mathcal{T}_i$ contains different terrains $\mathcal{T}_i$, such as lower ground, stairs, or ramps (see Fig. 1). In addition, there are regions $\mathcal{R}_i \subset \mathcal{E}$ of interest, such as obstacles or locations that robot should visit (see Fig. 1). To specify a meaningful task for the robot, the regions $\mathcal{R}_i$ are associated with atomic propositions $\alpha_i \in \mathcal{A} \subset \mathcal{AP}$ that evaluate to True according to the labeling function, i.e., $\alpha_i \in \ell(\mathbf{c}(t))$, when the robot's CoM $\mathbf{c}(t)$ is in the corresponding region $\mathcal{R}_i$. Note that the regions $\mathcal{R}_i$ may overlap, in which case the labeling function returns multiple valid propositions α_i. The robot's task is specified via a temporal logic formula over the propositions α_i.

Example 1. Consider a task requiring the robot to visit a Red region and then a finish its mission at a Yellow region

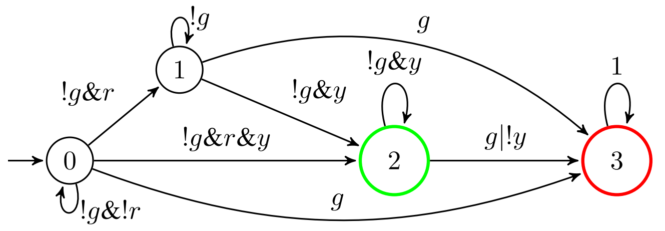


Fig. 2: Deterministic finite-state automation obtained from φ in Example 1. Propositions α_{Red}, α_{Yellow}, and α_{Green} are denoted by r, y, g, respectively, for readability. Automaton state 2 is a final state (green) and state 3 is a sink state (red).

while always avoiding Green regions (see Fig. 1). This task may formally be specified by the LTL formula φ in (5) and the corresponding deterministic finite-state automation is as given in Fig. 2:

$$\varphi = \Diamond \alpha_{\text{Red}} \wedge \Diamond \Box \alpha_{\text{Yellow}} \wedge \Box \neg \alpha_{\text{Green}} \tag{5}$$

B. Environment Action Set

Similar to associating atomic propositions to the interest regions $\mathcal{R}_i$, the terrains $\mathcal{T}_i$ are associated with atomic propositions $\nu_i \in \mathcal{B} \subset \mathcal{AP}$ and $\nu_i \in \ell(\mathbf{c}(t))$ when the robot's CoM $\mathbf{c}(t)$ is on the corresponding terrain $\mathcal{T}_i$. Thus, $\mathcal{B}$ and $\mathcal{A}$, together, form the set of atomic propositions $\mathcal{AP}$ used to define Problem 1, i.e., $\mathcal{A} \cup \mathcal{B} = \mathcal{AP}$. We call $\mathcal{B}$ the *environment action set* because the propositions $\nu_i \in \mathcal{B}$ are used to define rules for selecting the robot inputs $\mathbf{u}_k$ and constraints $\mathbf{v}_k$ depending on the terrain $\mathcal{T}_i$.

Example 2. Consider an environment containing `stairs` and three elevation levels: `lower`, `ground`, and `upper`. The corresponding environment action set $\mathcal{B}$ consists of the following atomic propositions:

$$\mathcal{B} := \{\beta_{\text{st}}, \beta_{\text{ll}}, \beta_{\text{gd}}, \beta_{\text{ul}}\} \tag{6}$$

where the first proposition β_{st} is true when the robot is one step away or stepping on the stairs, and the remaining ones indicate whether the robot is on the lower level/ground level/upper level of the environment. The environment actions other than β_{st} are mutually exclusive.

C. Determining Keyframe States using Environment Actions

A *keyframe state* $\boldsymbol{\theta}_k := (\mathbf{u}_k^T, \mathbf{v}_k^T) \subset \mathbb{R}^5$ at time t_k is defined as the input $\mathbf{u}_k$ for the k^{th} step discrete dynamics and the CoM state constraints $\mathbf{v}_k$. The environment actions that evaluate to True at time t_k are used to determine the keyframe state $\boldsymbol{\theta}_k$ depending on the terrain and desired gait characteristics. A propositional logic formula ϖ is specified over ν_i to define such rules as illustrated in the next example. The keyframe states that satisfy ϖ form a *finite* set $\Theta_k \subset \mathbb{R}^5$. The figurative presentation of keyframe states in the CoM phase-space plane and CoM sagittal plane can be visualized in Fig. 3 and 4.

Example 3. Suppose that each of the five dimensions (heading angle change $\delta\psi_k$, step length l_k, apex velocity $\dot{x}_k(t_{k+1})$, CoM height $z_k(t_{k+1})$, switching CoM height $z_k(\zeta_k)$) of a keyframe state $\boldsymbol{\theta}_k$ can take on values from a finite set $\{*, \mathsf{s}, \mathsf{m}, \mathsf{h}\}$. The symbols specify a small (s), medium (m), high (h), or unspecified ($*$) value for the keyframe state dimensions. When a dimension of $\boldsymbol{\theta}_k$ is unspecified ($*$), its value needs to be determined during the optimization in Problem 1. Given the environment actions in Example 2, we define the following keyframe state selection rules.

- When the robot is approaching stairs, it should switch to a large step length, small apex velocity, large CoM switching height and large apex height:
$$\varpi_1 = \Big(\beta_{\text{st}} \Rightarrow \big(\boldsymbol{\theta} = (*, \mathsf{h}, \mathsf{s}, \mathsf{h}, \mathsf{h})\big)\Big)$$
- When on the ground level, the robot should adopt a medium step length, large apex velocity, medium apex height, and medium switching height:
$$\varpi_2 = \Big(\big(\beta_{\text{gd}} \wedge \neg\beta_{\text{st}}\big) \Rightarrow \big(\boldsymbol{\theta} = (*, \mathsf{m}, \mathsf{h}, \mathsf{m}, \mathsf{m})\big)\Big)$$
- When on the lower level or upper levels, the robot should adopt a small step length, medium apex velocity, medium apex height, and medium switching height:
$$\varpi_3 = \Big(\big(\beta_{\text{ul}} \vee \beta_{\text{ll}}\big) \wedge \neg\beta_{\text{st}} \Rightarrow \big(\boldsymbol{\theta} = (*, \mathsf{s}, \mathsf{m}, \mathsf{m}, \mathsf{m})\big)\Big)$$

The combined formula $\varpi := \varpi_1 \wedge \varpi_2 \wedge \varpi_3$ specifies the rules for selecting a keyframe state $\boldsymbol{\theta}_k$ instantaneously depending on the terrain $\mathcal{T}_i$ where the CoM $\mathbf{c}(t_k)$ is located, i.e., ϖ does not involve temporal operators.

IV. Motion Primitive Generation

At the beginning of the k^{th} step, the robot's CoM $\mathbf{c}(t_k)$ determines the environment actions as detailed in the previous section, which in turn determines the possible inputs and constraints $\boldsymbol{\theta}_k \in \Theta_k$. Each element of Θ_k specifies a possible continuous-time CoM trajectory, which we refer to as a *motion primitive*.

The CoM motion is constrained to two planar surfaces perpendicular to the sagittal plane. The surfaces are defined by the equations:

$$\gamma_i x_k(t) + \nu_i - z_k(t) = 0, \quad i \in \{1, 2\} \tag{7}$$

where $i = 1$ in the first half of the walking cycle, i.e., before the stance foot switching time, $t_k \le t < \zeta_k$, and $i = 2$ in the second half of the walking cycle, i.e., for $\zeta_k \le t < t_{k+1}$. The constants γ_i, ν_i may be determined from $\mathbf{c}(t_k)$, $\mathbf{u}_k$, and $\mathbf{v}_k$ as described next. The CoM position $\mathbf{x}_k$ follows the dynamics:

$$\ddot{\mathbf{x}}_k(t) = \begin{cases} W_1 \mathbf{x}_k(t), & t_k \le t < \zeta_k \\ W_2 \mathbf{x}_k(t) - R_z^T(\psi_k)(\mathbf{p}_{k+1} - \mathbf{p}_k), & \zeta_k \le t \le t_{k+1} \end{cases} \tag{8}$$

where W_i is determined by an asymptotic slope ω_i with acceleration due to gravity g_c:

$$W_i = \begin{bmatrix} \omega_i^2 & 0 & 0 \\ 0 & \omega_i^2 & 0 \\ \gamma_i \omega_i^2 & 0 & 0 \end{bmatrix} \quad \forall i \in \{1, 2\} \tag{9}$$

$$\omega_1 = \sqrt{\frac{g_c}{z_k(t_k)}}, \quad \omega_2 = \sqrt{\frac{g_c}{z_{k+1}(t_{k+1})}} \tag{10}$$

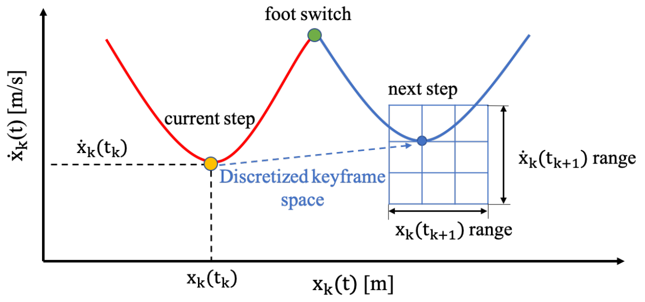


Fig. 3: Evolution of the CoM position $x_k(t)$ and velocity $\dot{x}_k(t)$ in the walking direction. The terminal value $x_k(t_{k+1})$ is determined by the step length l_k in the input $\mathbf{u}_k$, while $\dot{x}_k(t_{k+1})$ is available from the constraints $\mathbf{v}_k$.

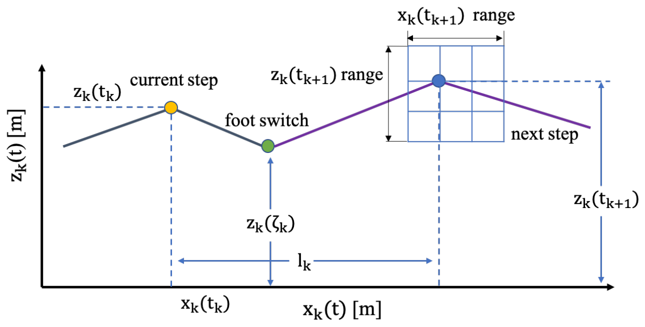


Fig. 4: Evolution of the CoM sagittal coordinates $x_k(t)$ and $z_k(t)$. The terminal value $x_k(t_{k+1})$ is determined by the step length l_k in the input $\mathbf{u}_k$, while $z_k(t_{k+1})$ is available from the constraints $\mathbf{v}_k$.

A. Stance Foot Switching Time

Separating the sagittal dynamics from (8) results in:

$$\ddot{x}_k(t) = \begin{cases} \omega_1^2 x_k(t), & t_k \le t < \zeta_k \\ \omega_2^2 x_k(t) - \mathbf{e}_1^T R_z^T(\psi_k)(\mathbf{p}_{k+1} - \mathbf{p}_k), & \zeta_k \le t \le t_{k+1} \end{cases} \tag{11}$$

The keyframe constraint $\mathbf{v}_k$ determines $\dot{x}_k(t_{k+1})$. From the step definition and since $\mathbf{p}_k$ is the origin of the local frame, we have $x_k(t_k) = 0$ and $x_k(t_{k+1}) = l_k$, where the step length l_k is available from the input $\mathbf{u}_k$. Hence, the time of stance foot switch ζ_k is determined by integrating forward from $x_k(t_k), \dot{x}_k(t_k)$ and backwards from $x_k(t_{k+1}), \dot{x}_k(t_{k+1})$ using ϵ perturbations [26], [27] until the x and $\dot{x}$ curves intersect. The evolution of $x_k(t)$, $\dot{x}_k(t)$ and the point of foot switching are illustrated in Fig. 3.

B. Surface Parameter Computation

Eq. (8) and (9) indicate that $x_k(t)$ and $y_k(t)$ are independent of the surface constants γ_i, ν_i. Hence, the surface constants determine the value of $z_k(t)$ for $t \in [t_k, t_{k+1}]$. The keyframe constraint $\mathbf{v}_k$ determines $z_k(\zeta_k)$ and $z_k(t_{k+1})$. See Fig. 4 for an illustration. The surface constants can then be obtained:

1: Since $x_k(t_k) = 0$ from (7) $\nu_1 = z_k(t_k)$. Using $x_k(\zeta_k)$, computed from (11), and $z_k(\zeta_k)$, available from $\mathbf{v}_k$, γ_1 is determined from (7).
2: Substituting the values $x_k(\zeta_k)$, $z_k(\zeta_k)$, $x_k(t_{k+1}) = l_k$ (from $\mathbf{u}_k$) and $z_k(t_{k+1})$ (from $\mathbf{v}_k$) in (7) results in two equations with two unknowns ν_2 and γ_2.

C. CoM-State Dynamics

Eq. (8) may be re-written in state-space form using the local CoM-state coordinates $\boldsymbol{\xi}_k(t) = (\mathbf{x}_k(t)^T, \dot{\mathbf{x}}_k(t)^T)^T$:

$$\dot{\boldsymbol{\xi}}_k(t) = \underbrace{\begin{bmatrix} 0 & I \\ W_i & 0 \end{bmatrix}}_{A_i} \boldsymbol{\xi}_k(t) - \underbrace{\begin{bmatrix} 0 \\ R_z^T(\psi_k) \end{bmatrix}}_{B} \mathbf{d}_i, \tag{12}$$

where $\mathbf{d}_1 = 0$ and $\mathbf{d}_2 = (\mathbf{p}_{k+1} - \mathbf{p}_k)$. The solution to this linear time-invariant system is:

$$\boldsymbol{\xi}_k(t) = \begin{cases} e^{A_1(t-t_k)}\boldsymbol{\xi}_k(t_k) & t_k \le t < \zeta_k \\ e^{A_2(t-\zeta_k)}\boldsymbol{\xi}_k(\zeta_k) & \\ \quad + \int_{\zeta_k}^{t} e^{A_2(t-s)} ds B(\mathbf{p}_{k+1} - \mathbf{p}_k) & \zeta_k \le t \le t_{k+1} \end{cases} \tag{13}$$

Then, the global coordinates $\mathbf{c}(t)$ of the CoM state may be obtained:

$$\mathbf{c}(t) = \begin{bmatrix} R_z(\psi_k) & 0 \\ 0 & R_z(\psi_k) \end{bmatrix} \boldsymbol{\xi}_k(t) + \begin{bmatrix} I \\ 0 \end{bmatrix} \mathbf{p}_k, \quad t \in [t_k, t_{k+1}]. \tag{14}$$

V. LTL Task Planning

Having defined the hybrid locomotion dynamics and their spatial discretization via keyframe states $\boldsymbol{\theta}_k$, we are now ready to address Problem 1. The keyframe states were designed to limit the evolution of the CoM dynamics to a finite set of possible trajectories. This reduces Problem 1 to a deterministic shortest path (DSP) problem over the graph of possible CoM states $\mathbf{c}(t_k)$ composed with the task automaton A_φ. The DSP problem may be solved by a standard motion planning algorithm, such as A* [28] or RRT [29]. In detail, the state of the DSP problem consists of the automaton state $q(t_k)$, the step state $\mathbf{s}_k$, and the CoM state $\mathbf{c}(t_k)$. The discrete-time state evolution is described by the automaton transition function h, the discrete locomotion dynamics f in eq. (1), and the CoM state trajectory in eq. (13) and (14). This may be used to define the EXTEND function in RRT or the SUCCESSORS function in A* (see Alg. 1).

Inspired by [11], we use the A* algorithm because an accurate heuristic function may be designed to guide the planning process based on the level sets of the automaton A_φ. The automation states $\mathcal{Q}$ are partitioned into level sets where $\mathcal{Q}_0 := \mathcal{Q}_F$ and for $i > 0$ $\mathcal{Q}_{i+1} := \{q \in \mathcal{Q} \setminus \bigcup_{k=0}^{i} \mathcal{Q}_k | \exists q' \in \mathcal{Q}_i, p \in 2^{\mathcal{AP}}, \text{ such that } h(q,p) = q'\}$. From the description of the levels, it can be seen that the automation state has to pass through the level sets sequentially in order to reach the accepting set $\mathcal{Q}_F$. Given $(q(t_k), \mathbf{s}_k, \mathbf{c}(t_k))$ with $q(t_k) \in \mathcal{Q}_i$, a heuristic function that underestimates the cost in eq. (4) may be obtained by computing the cost from $\mathbf{c}(t_k)$ to the set of CoM states $\mathbf{c}(t)$ that produce labels $\ell(\mathbf{c}(t))$ triggering a transition to $\mathcal{Q}_{i-1}$. If the cost function ρ is a distance metric, a more precise heuristic can be obtained by adding the set-to-set distances between the CoM-state sets that trigger transitions to $\mathcal{Q}_{i-1}$, followed by $\mathcal{Q}_{i-2}$, …, until $\mathcal{Q}_0$.

Algorithm 1 Determining successors for LTL-constrained A*

```
 1: input: automaton state $q(t_k)$, step state $\mathbf{s}_k$, CoM state $\mathbf{c}(t_k)$, keyframe
    set $\Theta_k$
 2: $Succ \leftarrow \emptyset$, $Cost \leftarrow \emptyset$
 3: for $(\mathbf{u}_k, \mathbf{v}_k)$ in $\Theta_k$ do
 4:     $\mathbf{s}_{k+1} \leftarrow f(\mathbf{s}_k, \mathbf{u}_k)$
 5:     for $t = t_k, t_k + \Delta t, \ldots, t_{k+1} - \Delta t$ do
 6:         Compute $\mathbf{c}(t+\Delta t)$ from $\mathbf{c}(t)$, $\mathbf{s}_k$, $\mathbf{s}_{k+1}$, $\mathbf{v}_k$ via eq. (14), (13)
 7:         $q(t+\Delta t) \leftarrow h(q(t), \ell(\mathbf{c}(t+\Delta t)))$
 8:         if $\mathbf{c}(t+\Delta t)$ hits obstacle or $q(t+\Delta t) \in \mathcal{Q}_s$ then
 9:             break
10:     $r \leftarrow \rho(\mathbf{c}(t_k), \mathbf{c}(t_{k+1})) + \mathbf{u}_k^T R \mathbf{u}_k + \mathbf{v}_k^T G \mathbf{v}_k$
11:     $Succ \leftarrow Succ \cup \{q(t_{k+1}), \mathbf{c}(t_{k+1}), \mathbf{s}_{k+1}\}$, $Cost \leftarrow Cost \cup \{r\}$
12: return $Succ$, $Cost$
```

VI. Quadratic Programming Control

Quadratic programming (QP) is one of the mainstream optimal control methods in robot manipulation and locomotion. In this study, we employ QP to achieve multiple-step dynamic locomotion stabilization on level ground and stairs. The features of our QP controller are highlighted as: (i) trajectory replanning for tracking error reduction and robust walking; (ii) slack variables encoded in the foot contact constraint for feasibility of numerical optimization; (iii) controlled point-foot walker maneuvering on rough terrain.

The task planning algorithm developed in Sec. V provides desired CoM trajectory $\mathbf{c}(t)$ and stance foot sequence $\mathbf{p}_k$. Desired swing foot position and velocity trajectory $(\mathbf{f}_\mathrm{k}(t), \dot{\mathbf{f}}_\mathrm{k}(t)) \in \mathbb{R}^6$ is obtained in the global frame of reference using two piece-wise constant acceleration trajectories such that the velocity and height are 0 at t_k and t_{k+1} and non-zero in between. Given the robot state $(\boldsymbol{q}_\mathrm{gc}(t), \dot{\boldsymbol{q}}_\mathrm{gc}(t)) \in \mathbb{R}^{22}$, where $\boldsymbol{q}_\mathrm{gc}(t)$ are generalized coordinates, consisting of the 6-D base pose, two knee joints, and three hip joints, and $\dot{\boldsymbol{q}}_\mathrm{gc}(t)$ are their velocities, our QP formulation aims to track the desired CoM, stance foot, and swing-foot trajectories. The QP controller takes these desired trajectories as inputs, and optimizes the joint torques $\boldsymbol{\tau} \in \mathbb{R}^5$ subject to the full-body dynamics and constraints as follows:

$$\begin{aligned}
\min_{\ddot{\boldsymbol{q}}_\mathrm{gc}, \boldsymbol{\lambda}, \boldsymbol{\tau}, \boldsymbol{\eta}_\mathrm{st}, \boldsymbol{\eta}_\mathrm{sw}} \quad & w_\mathrm{com} \left\| \boldsymbol{J}_\mathrm{com} \ddot{\boldsymbol{q}}_\mathrm{gc} + \dot{\boldsymbol{J}}_\mathrm{com} \dot{\boldsymbol{q}}_\mathrm{gc} - \ddot{\mathbf{x}}_k \right\|^2 \\
& + w_\tau \left\| \boldsymbol{\tau} - \boldsymbol{\tau}_\mathrm{pre} \right\|^2 + w_\mathrm{st} \left\| \boldsymbol{\eta}_\mathrm{st} \right\|^2 \\
& + w_\mathrm{sw} \left\| \boldsymbol{\eta}_\mathrm{sw} \right\|^2 && \text{(15a)} \\
\text{subject to} \quad & \boldsymbol{H}_f \ddot{\boldsymbol{q}}_\mathrm{gc} + \boldsymbol{C}_f = \boldsymbol{J}_f^\mathrm{T} \boldsymbol{\lambda} && \text{(15b)} \\
& \boldsymbol{H}_a \ddot{\boldsymbol{q}}_\mathrm{gc} + \boldsymbol{C}_a = \boldsymbol{J}_a^\mathrm{T} \boldsymbol{\lambda} + \boldsymbol{B}_a \boldsymbol{\tau} && \text{(15c)} \\
& \boldsymbol{J}_\mathrm{st} \ddot{\boldsymbol{q}}_\mathrm{gc} + \dot{\boldsymbol{J}}_\mathrm{st} \dot{\boldsymbol{q}}_\mathrm{gc} = -\alpha \boldsymbol{J}_\mathrm{st} \dot{\boldsymbol{q}}_\mathrm{gc} + \boldsymbol{\eta}_\mathrm{st} && \text{(15d)} \\
& \boldsymbol{J}_\mathrm{sw} \ddot{\boldsymbol{q}}_\mathrm{gc} + \dot{\boldsymbol{J}}_\mathrm{sw} \dot{\boldsymbol{q}}_\mathrm{gc} = \ddot{\mathbf{f}}_k + \boldsymbol{\eta}_\mathrm{sw} && \text{(15e)} \\
& \boldsymbol{\tau} \in [\boldsymbol{\tau}_\mathrm{min}, \boldsymbol{\tau}_\mathrm{max}] && \text{(15f)} \\
& q_{gc,i} \in [q_\mathrm{min}, q_\mathrm{max}], \ \forall i \in \mathcal{I}_\mathrm{actuated} && \text{(15g)} \\
& \boldsymbol{\lambda}_j \in \widehat{F}_j, \quad \forall j \in \mathcal{I}_\mathrm{contact} && \text{(15h)}
\end{aligned}$$

where $\boldsymbol{H_f}$ and $\boldsymbol{H_a}$ are the inertial matrices corresponding to the under-actuated CoM dynamics and actuated joint dynamics, $\boldsymbol{C_f}$ and $\boldsymbol{C_a}$ represent the gravitational and Coriolis terms, $\boldsymbol{B_a}$ is the actuation selection matrix, and $\boldsymbol{J_f}^\mathrm{T}$ and $\boldsymbol{J_a}^\mathrm{T}$ are external contact-force Jacobian matrices. For our kneed compass gait walker model, $\boldsymbol{\lambda} = \left[\boldsymbol{\lambda_1}^{\mathrm{T}} \quad \boldsymbol{\lambda_2}^{\mathrm{T}}\right]^{\mathrm{T}}$ is a contact-force vector acting at two contact points. The matrices $\boldsymbol{J}$ and $\dot{\boldsymbol{J}}$ are task-level Jacobians which can be defined for the CoM, swing foot and stance foot tasks in Cartesian space. The indices $\mathcal{I}_{\text{actuated}}$ and $\mathcal{I}_{\text{contact}}$ are the actuated joint set and contact point set, respectively, and $\widehat{F}_j = \left\{\boldsymbol{\lambda}_j | \boldsymbol{\lambda}_j \in \mathbb{R}^3, \langle \mathbf{n}_j, \boldsymbol{\lambda}_j \rangle < \arctan \mu_j \right\}$ is the friction cone of the j^{th} contact point, where $\mathbf{n}_j$ and μ_j are normal and Columbus coefficient at the j^{th} contact point respectively, angle bracket represents the angle between the two vector arguments. The constraints (15b) and (15c) enforce full body dynamics; eq. (15d) is a no-slip constraint to ensure the stance leg stays on the ground (we set α to 0); eq. (15h) represents contact-force constraints; eq. (15f) represents joint torque limits. Finally, eq. (15e) is a swing-leg constraint, where $\ddot{\mathbf{f}}_k$ is the desired swing-foot trajectory acceleration.

The weight parameter, w_{com}, is used to balance the importance of tracking the CoM trajectory, while w_{st} and w_{sw}, are used to weight the ground-slip prevention and swing-foot motion tracking. We set $w_{\text{st}} = 10^8$ to enforce the no-slip constraint, while w_{sw} and w_{com} were set to 10^5.

VII. Results

This section evaluates the performance of our planner in satisfying different tasks in different environments and our controller in achieving periodic and non-periodic locomotion. We perform simulations in Drake [30] using a kneed compass gait model [31], [32] to navigate environments containing obstacles, stairs, and ramps.

A. LTL Task Planning

We consider a task requiring the robot to obtain objects located in red and blue regions, and deliver them to a yellow region, while avoiding dangerous obstacle regions. Formally, we define regions $\mathcal{R}_i \in \{\texttt{Blue, Red, Yellow, Obs}\}$ and associate atomic propositions α_{Blue}, α_{Red}, α_{Yellow}, α_{Obs} with them. Two variations of this task are considered.

In the first task, the robot is assumed to have limited capacity to hold objects. In this case, the robot is tasked to reach either the blue or the red region before delivering the object to the yellow region. The specification φ_1 is:

$$\begin{aligned}\varphi_1 =& (\Diamond\alpha_{\text{Blue}} \wedge \Diamond\Box\alpha_{\text{Yellow}} \wedge \Box\neg\alpha_{\text{Red}}) \\ & \vee (\Diamond\alpha_{\text{Red}} \wedge \Diamond\Box\alpha_{\text{Yellow}} \wedge \Box\neg\alpha_{\text{Blue}}) \wedge \Box\neg\alpha_{\text{Obs}}\end{aligned} \quad (16)$$

The task φ_1 automaton is visualized in Fig. 5.

In the second task, the robot is assumed to have sufficient storage capacity to hold the objects from both red and blue regions. The robot is tasked to visit both the regions, without specific order, and finally reach the yellow region, while avoiding the dangerous regions. The specification φ_2 is:

$$\varphi_2 = \Diamond\alpha_{\text{Blue}} \wedge \Diamond\alpha_{\text{Red}} \wedge \Diamond\Box\alpha_{\text{Yellow}} \wedge \Box\neg\alpha_{\text{Obs}} \quad (17)$$

The switching behavior of the keyframe states $\boldsymbol{\theta}_k$ in response to the environment actions defined in eq. (6) was evaluated via the SLUGS toolbox [33]. The realizability of

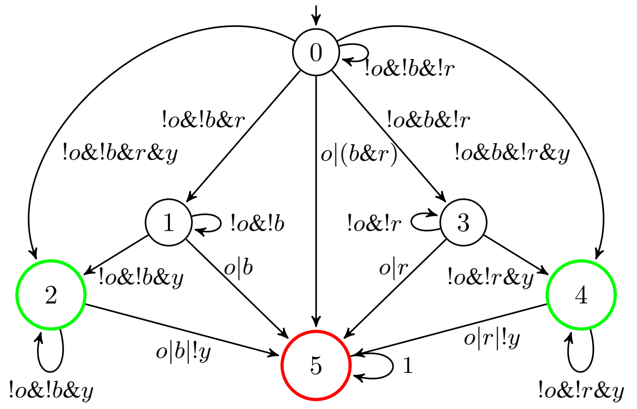
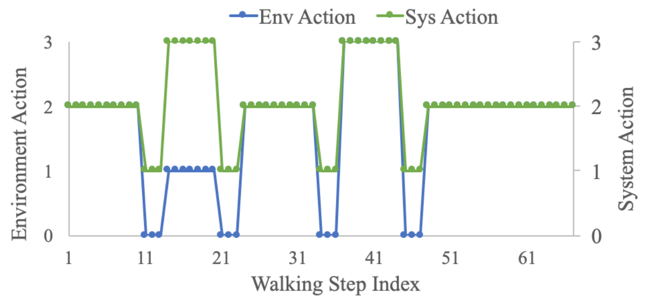


Fig. 5: Deterministic finite-state automation obtained from the task φ_1 in (16) using Spot [25]. The propositions α_{Red}, α_{Blue}, α_{Yellow}, and α_{Obstacle} are denoted by r, b, y, o, respectively. Automaton



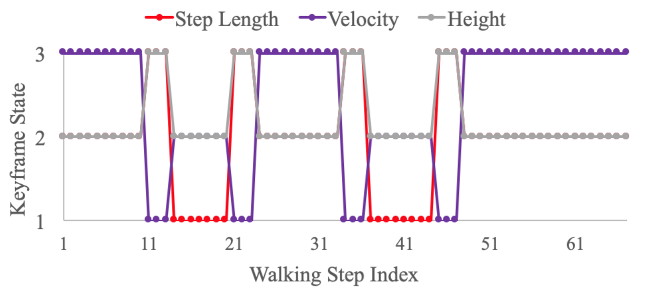


Fig. 6: Mode switching visualized by SLUGS [33]. The top plot shows the switching of system actions in response to the environment actions. The environment actions in eq. (6) are numbered 0 to 3 in order. The keyframe states $(l_k, \dot{x}_k(t_{k+1}), z_k(t_{k+1})) \in \{(\mathsf{h}, \mathsf{s}, \mathsf{h}), (\mathsf{m}, \mathsf{h}, \mathsf{m}), (\mathsf{s}, \mathsf{m}, \mathsf{m}), (\mathsf{m}, \mathsf{m}, \mathsf{m})\}$ are also numbered 0 to 3 in order. The bottom plot shows the corresponding keyframe state switching. The keyframe state values `1, 2, 3`, correspond to small (s), medium (m), and high (h), respectively, as in Example 3.

the switching behavior of the step length l_k, apex velocity $\dot{x}_k(t_{k+1})$ and CoM height $z_k(t_{k+1})$ is verified in Fig. 6.

Our task planning approach was used to generate a discrete-time stance-foot sequence and a continuous-time CoM trajectory for both tasks in three different environments: `env.1`, `env.2`, and `env.3`. The generated CoM trajectories in the three environments are visualized in Fig. 1, Fig. 7, and Fig. 8, respectively. The performance of the planning algorithm, in terms of motion cost, expanded nodes, and planning time, was evaluated in these environments for two different scenarios. In the first scenario (`scn.1`), the rules specified in Example 3 for selecting keyframe states $\boldsymbol{\theta}_k = (\mathbf{u}_k^T, \mathbf{v}_k^T)^T$ were used. In the second scenario (`scn.2`), the keyframe selection rules in Example 3 were

TABLE I: Planing results for two tasks across two scenarios in three environments.

Environment	Scenario	Task	Cost	Expanded Nodes	Planning Time(s)
env.1	scn.1	φ_1	53.86	32569	0.17
		φ_2	93.71	194790	1.2
	scn.2	φ_1	53.17	55488	0.99
		φ_2	89.69	246239	3.38
env.2	scn.1	φ_1	44.578	24184	0.12
		φ_2	∞	403601	-
	scn.2	φ_1	43.091	13592	0.36
		φ_2	∞	3849118	-
env.3	scn.1	φ_1	186.3	2185502	19.6
		φ_2	235.801	3354012	35.1
	scn.2	φ_1	182.3	2398659	42.1
		φ_2	232.699	3658481	68.2

modified to allow the step length $l_k \in \{\mathsf{s}, \mathsf{m}, \mathsf{h}\}$ to be optimized during planning. The allowable keyframe values were $\delta\psi_k \in \{-0.3, 0, 0.3\}$, $l_k \in \{0.4, 0.5, 0.6\}$, $x_k(t_{k+1}) \in \{0.4, 0.5, 0.6\}$, $z_k(t_{k+1}) \in \{0.6, 0.65, 0.7\}$ and $z_k(\zeta_k) \in \{0.63, 0.65, 0.68\}$. The following motion cost was used:

$$\rho(\mathbf{c}(t_k), \mathbf{c}(t_{k+1})) = \|\mathbf{c}(t_k) - \mathbf{c}(t_{k+1})\|_2$$
$$R := \begin{bmatrix} 3.33 & 0 \\ 0 & 0 \end{bmatrix} \qquad G := 0_{3\times 3} \tag{18}$$

The time complexity of the proposed planning algorithm is $O(|\Theta|^K)$ where $|\Theta|$ is the number of motion primitives and K is the number of steps in the optimal path. The number of steps K is proportional to the sum of the distance between the atomic propositions that trigger level set transitions. Hence, K is dependent both on the size of the map, number of level sets introduced by the LTL formula and the distance between them as seen in Table I.

The computation time and the number of expanded nodes are considerably higher for env.3 as it is 5 times larger than env.1 and planning is more challenging due to its maze-like structure. Since the task φ_2 has an extra level set when compared to φ_1, it can be seen that the planning time and the expanded nodes is higher for φ_2. For both tasks and in all environments, the path cost for scn.1 is slightly larger than for scn.2 because in the latter the step length l_k is optimized. However, the value of the keyframe constraints can be seen in that the planning time and number of expanded nodes are much smaller for scn.2 due to the reduction of the number of motion primitives considered for expansion, while the generated paths are of negligibly lower quality. Well-defined keyframe rules may significantly decrease the computational complexity of the task planning problem.

B. QP Implementation

A kneed bipedal robot model with point feet and five point masses on its hip and legs was used (see Fig. 1). Locomotion of the robot is studied on flat and rough terrain. In both scenarios, our QP controller employs a replanning strategy for the CoM and foot trajectories to reduce the tracking error after each step. The desired foot location is modified using a bisection method to ensure that the keyframe state of each step remains within a bounded region.

For flat terrain walking, we study a scenario with a constant heading angle and step length. Desired trajectories

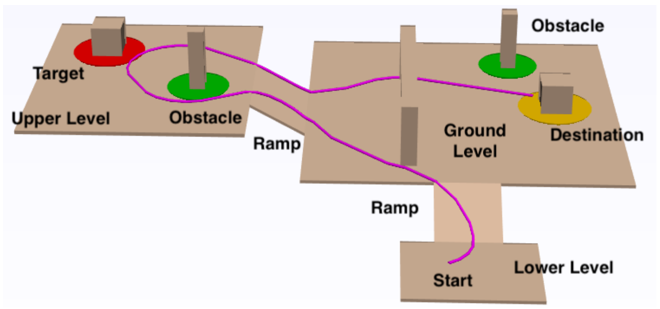


Fig. 7: Planned task execution in env.2 for task specification φ_1 in (16). Task φ_2 in (17) does not generate a feasible path as Blue region is not present.

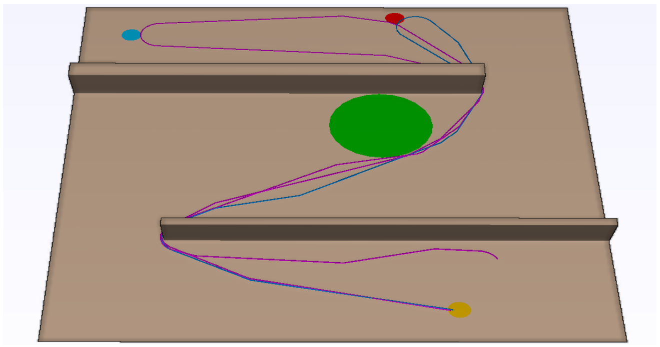


Fig. 8: Planned paths in env.3 for the LTL task specification φ_1 in (16) represented by the blue curve and the task specification φ_2 in (17) represented by the magenta curve.

are extracted from the task planner and tracked by the QP controller. Although the CoM and stance foot position tracking is challenging due to point-feet and under-actuated dynamics, the QP controller generates torque commands that accomplish the stabilization of a 10-step dynamic walking process, as illustrated in Fig. 9. We also evaluate our QP controller for walking upstairs, and Fig. 10 shows a four-step walking process of this scenario.

As an on-going work, we aim at applying this QP controller to an infinite number of steps while steering the heading angle. In this case, the QP controller would be capable of stabilizing the robot for long-distance dynamic locomotion. During our experiments, we realized that the crux of achieving this goal is to devise a robust contact switching and a lateral foot placement strategy. We are implementing a lateral foot placement search algorithm according to the work [27] and a robust hybrid trajectory tracking strategy based on sensitivity analysis [34].Fault Diagnosis [35] [36] will also be an important topic in the future control design.

VIII. Conclusion

This work proposed temporal-logic-guided planning and control techniques to achieve non-periodic locomotion tasks in cluttered rough terrains. The body of our work is the design of keyframe states and motion primitives to reduce LTL task planning with hybrid locomotion dynamics to a discrete-space planning problem. We also developed a novel QP control formulation that tracks the planned CoM and foot trajectories of a kneed bipedal robot. The proposed method offers a promising direction for achieving complex semantically meaningful behavior for humanoid robots in

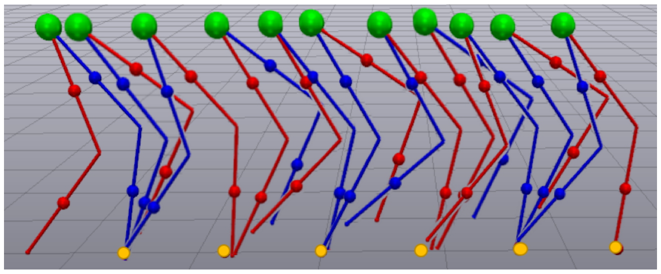


Fig. 9: Walking on level ground using the proposed QP controller. Yellow dots on the ground represent desired stance foot positions.

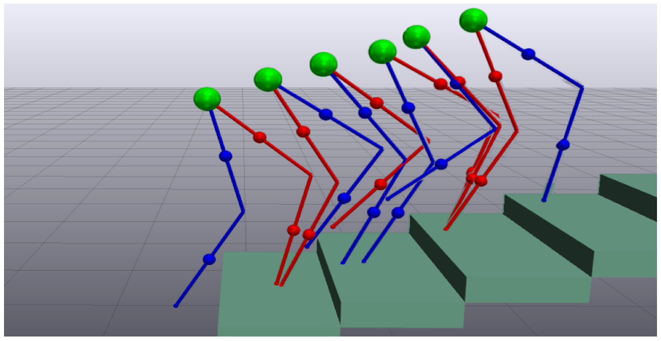


Fig. 10: Climbing stairs using the proposed QP controller.

unstructured human environments. Future work will focus on extensions to full body planning and control, a larger class task specifications using signal temporal logic.

REFERENCES

[1] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback control of dynamic bipedal robot locomotion*. CRC press, 2018.

[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, 2016.

[3] M. S. Motahar, S. Veer, and I. Poulakakis, "Composing limit cycles for motion planning of 3d bipedal walkers," in *IEEE Conference on Decision and Control (CDC)*, 2016, pp. 6368–6374.

[4] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[5] A. Hofmann, S. Massaquoi, M. Popovic, and H. Herr, "A sliding controller for bipedal balancing using integrated movement of contact and non-contact limbs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004, pp. 1952–1959.

[6] J. Liu, N. Ozay, U. Topcu, and R. M. Murray, "Synthesis of reactive switching protocols from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 58, no. 7, pp. 1771–1785, 2013.

[7] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia, "Reactive synthesis from signal temporal logic specifications," in *International Conference on Hybrid Systems: Computation and Control (HSCC)*. ACM, 2015, pp. 239–248.

[8] K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "Reactive synthesis for finite tasks under resource constraints," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[9] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[10] Y. Zhao, U. Topcu, and L. Sentis, "High-level planner synthesis for whole-body locomotion in unstructured environments," in *IEEE Conference on Decision and Control (CDC)*, 2016, pp. 6557–6564.

[11] J. Fu, N. Atanasov, U. Topcu, and G. J. Pappas, "Optimal temporal logic planning in probabilistic semantic maps," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016, pp. 3690–3697.

[12] R. D. Gregg, A. K. Tilton, S. Candido, T. Bretl, and M. W. Spong, "Control and planning of 3-d dynamic walking with asymptotically stable gait primitives," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1415–1423, 2012.

[13] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *The International Journal of Robotics Research (IJRR)*, vol. 27, no. 11-12, 2008.

[14] M. J. Powell, H. Zhao, and A. D. Ames, "Motion primitives for human-inspired bipedal robotic locomotion: walking and stair climbing," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[15] B. Lim, J. Lee, J. Kim, M. Lee, H. Kwak, S. Kwon, H. Lee, W. Kwon, and K. Roh, "Optimal gait primitives for dynamic bipedal locomotion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 4013–4018.

[16] N. Sun, Y. Wu, H. Chen, and Y. Fang, "An energy-optimal solution for transportation control of cranes with double pendulum dynamics: Design and experiments," *Mechanical Systems and Signal Processing*, vol. 102, pp. 87–101, 03 2018.

[17] S. Kuindersma, F. Permenter, and R. Tedrake, "An efficiently solvable quadratic program for stabilizing dynamic locomotion," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[18] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, A. Escande, K. Bouyarmane, K. Kaneko, M. Morisawa, P. Gergondet *et al.*, "Multi-contact vertical ladder climbing with an hrp-2 humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 561–580, 2016.

[19] S.-C. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," in *American Control Conference (ACC)*, 2015, pp. 4542–4548.

[20] S. Feng, X. Xinjilefu, W. Huang, and C. G. Atkeson, "3d walking based on online optimization," in *IEEE-RAS International Conference on Humanoid Robots*, 2013, pp. 21–27.

[21] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization based full body control for the atlas robot," in *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 120–127.

[22] H. Audren, A. Kheddar, and P. Gergondet, "Stability polygons reshaping and morphing for smooth multi-contact transitions and force control of humanoid robots," in *IEEE International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 1037–1044.

[23] Y. Zhao, B. R. Fernandez, and L. Sentis, "Robust phase-space planning for agile legged locomotion over various terrain topologies." in *Robotics: Science and Systems*, 2016.

[24] E. A. Emerson, "Temporal and modal logic," in *Formal Models and Semantics*. Elsevier, 1990, pp. 995–1072.

[25] A. Duret-Lutz, A. Lewkowicz, A. Fauchille, T. Michaud, E. Renault, and L. Xu, "Spot 2.0 — a framework for LTL and ω-automata manipulation," in *International Symposium on Automated Technology for Verification and Analysis*, ser. Lecture Notes in Computer Science, vol. 9938. Springer, 2016, pp. 122–129.

[26] L. Sentis and B. Fernandez, "Perturbation theory to plan dynamic locomotion in very rough terrains," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2267–2273.

[27] Y. Zhao and L. Sentis, "A three dimensional foot placement planner for locomotion in very rough terrains," in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 726–733.

[28] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with Provable Bounds on Sub-Optimality," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2003.

[29] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[30] R. Tedrake et al., "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

[31] H. Dai and R. Tedrake, "L2-gain optimization for robust bipedal walking on unknown terrain," in *IEEE Int. Conf. on Robotics and Automation*, 2013.

[32] K. Byl and R. Tedrake, "Approximate optimal control of the compass gait on rough terrain," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1258–1263.

[33] R. Ehlers and V. Raman, "Slugs: Extensible GR(1) synthesis," in *Int. Conf. on Computer Aided Verification*, 2016, pp. 333–339.

[34] A. Saccon, N. van de Wouw, and H. Nijmeijer, "Sensitivity analysis of hybrid systems with state jumps with application to trajectory tracking," in *IEEE Conf. on Decision and Control*, 2014.

[35] Y. Wu, B. Jiang, and N. Lu, "A descriptor system approach for estimation of incipient faults with application to high-speed railway traction devices," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.

[36] Y. Wu, B. Jiang, and Y. Wang, "Incipient winding fault detection and diagnosis for squirrel-cage induction motors equipped on crh trains," *ISA transactions*, 2019.