The International Journal of Robotics Research
©The Author(s) XXXX
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
http://mms.sagepub.com

Reactive Task and Motion Planning for Robust Whole-Body Dynamic Locomotion in Constrained Environments

Ye Zhao [*1], Yinan Li[2], Luis Sentis[3], Ufuk Topcu[3,4], Jun Liu[2].
[1]*George W. Woodruff School of Mechanical Engineering, Georgia Tech, USA.*
[2] *Department of Applied Mathematics, University of Waterloo, Canada.*
[3]*Department of Aerospace Engineering and Engineering Mechanics, UT Austin, USA.*
[4]*Institute for Computational Engineering and Sciences, UT Austin, USA.*

Abstract
Contact-based decision and planning methods are becoming increasingly important to endow higher levels of autonomy for legged robots. Formal synthesis methods derived from symbolic systems have great potential for reasoning about high-level locomotion decisions and achieving complex maneuvering behaviors with correctness guarantees. This study takes a first step toward formally devising an architecture composed of task planning and control of whole-body dynamic locomotion behaviors in constrained and dynamically changing environments. At the high level, we formulate a two-player temporal logic game between the multi-limb locomotion planner and its dynamic environment to synthesize a winning strategy that delivers symbolic locomotion actions. These locomotion actions satisfy the desired high-level task specifications expressed in a fragment of temporal logic. Those actions are sent to a robust finite transition system that synthesizes a locomotion controller that fulfills state reachability constraints. This controller is further executed via a low-level motion planner that generates feasible locomotion trajectories. We construct a set of dynamic locomotion models for legged robots to serve as a template library for handling diverse environmental events. We devise a replanning strategy that takes into consideration sudden environmental changes or large state disturbances to increase the robustness of the resulting locomotion behaviors. We formally prove the correctness of the layered locomotion framework guaranteeing a robust implementation by the motion planning layer. Simulations of reactive locomotion behaviors in diverse environments indicate that our framework has the potential to serve as a theoretical foundation for intelligent locomotion behaviors.

Keywords
Task and motion planning, Legged locomotion, Temporal logic, Robust reachability, Sequential composition.

1 Introduction

The goal of this paper is to devise a reactive task and motion planning framework for whole-body dynamic locomotion (WBDL) behaviors in constrained environments. We employ formal methods for synthesis of a symbolic task planner and design of reachability controllers to achieve legged locomotion behaviors that are reactive to the environment. Although

[*] Corresponding author; E-mail addresses: ye.zhao@me.gatech.edu, {yinan.li, j.liu}@uwaterloo.ca, {utopcu, lsentis}@utexas.edu.

widely used in mobile robot motion planning [Wongpiromsarn et al. (2012); Kloetzer and Belta (2010); Fu and Topcu (2016)] and autonomous driving [Campbell et al. (2010); Xu et al. (2018)], formal methods have not been previously used to reason about keyframe states of dynamic locomotion behaviors. To that end, we rely on dynamic locomotion abstractions that reduce the dimensionality of the reasoning process [Zhao et al. (2017)]. These abstractions allow to sequentially compose locomotion modes by reasoning about the previously mentioned keyframe dynamic locomotion states and achieve advanced reactive behaviors that can respond to dynamic events in the environment as well as to disturbances, a hallmark of intelligent locomotion behaviors. The complex locomotion behaviors studied in this paper could not be achieved by using motion planners alone without a high-level decision-making process. Reasoning about keyframe dynamic locomotion states has several advantages allowing to 1) take advantage of the passive dynamics of legged robots, 2) directly compose behaviors in the phase space of the locomotion process, 3) achieve goal state reachability considering robustness margins, and 4) adjust locomotion behaviors in response to disturbances.

Our technical approach relies on a suite of template-based locomotion modes that span a spectrum of desired whole-body dynamic locomotion behaviors. Sequentially composing these modes via the proposed reactive synthesis enables us to formally combine tasks such as multi-contact locomotion, swinging movements, and hopping motions, as shown in Fig. 1. Using simplified models to characterize locomotion dynamics has been widely pursued such as the use of the linear inverted pendulum model (LIPM) [Kajita et al. (2001)], the spring-loaded inverted pendulum model [Piovan and Byl (2015)], the brachiation-like pendulum model [Bertram et al. (1999)], the multi-contact model [Sentis et al. (2010b); Caron and Kheddar (2016)], and our recently proposed prismatic inverted pendulum model (PIPM) [Zhao and Sentis (2012)], to name a few. Usually, these models are separately considered in their own specific scenarios and lack a framework to seamlessly integrate them. Seminal locomotion results using template models [Raibert (1986); Full and Koditschek (1999); Alexander (1984); De and Koditschek (2015)] and sequential composition of these models Burridge et al. (1999) championed the advantages of using simplified models to uncover the fundamental locomotion principles related to the fine details of multi-body mechanism and dynamics. The work in [Arslan and Saranli (2012)] employs sequential composition to achieve reactive and robust planning against both model uncertainty and measurement noise without replanning. Nevertheless, no high-level decision-making algorithms with formal guarantees have been investigated, although the mentality of hierarchical planning and control had been proposed in [Full and Koditschek (1999)].

In this study, we aim at bridging this gap by proposing formal symbolic-level decision-making theories to sequentially compose more challenging – highly dynamic, versatile, non-periodic – locomotion behaviors reactive to dynamic environmental events. In the vein of work addressing rough terrain locomotion [Englsberger et al. (2015); Sreenath et al. (2013); Zhao et al. (2016)], we address the variability of the terrains by allowing the robot to respond to sudden environmental events. The behaviors we synthesize are required to satisfy formal task specifications in a provably correct manner, which we guarantee by using formal methods with discrete abstractions of hybrid systems [Alur et al. (2000)]. To the best of our knowledge, our study is the first attempt to use formal methods applied to phase-space keyframe state during for dynamic locomotion behaviors.

The inherent hybrid dynamics of the locomotion process and our use of keyframe dynamic locomotion states facilitate the discrete planning synthesis. Instead of discretizing the robot's state space, we rely on a discretization of the phase space keyframe states for synthesizing symbolic-level decisions which are further sent to the underlying motion planner. We focus on the integration between the symbolic-level discrete task planner and the continuous motion planner. This top-down planning approach significantly reduces the computational complexity compared to bottom-up approaches [Liu et al. (2013); Tabuada (2009); Belta et al. (2017); Liu and Ozay (2016)]. The correctness of our top-down hierarchy is guaranteed via a correct-by-construction synthesis at the task planner level and a reachability control synthesis at the motion planner level.

The contributions of this paper are as follows. The first one is on devising symbolic reasoning methods that make decisions on keyframe states of the dynamic locomotion process in response to the dynamically changing environment.

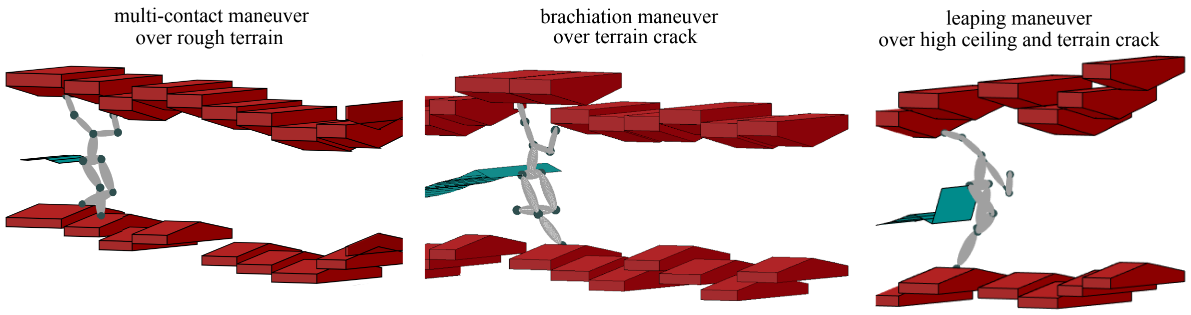

multi-contact maneuver over rough terrain
brachiation maneuver over terrain crack
leaping maneuver over high ceiling and terrain crack

Fig. 1. Maneuvering in a constrained environment via multi-contact whole-body dynamic locomotion. Three locomotion modes are illustrated. The contact decisions are made according to the high-level symbolic task planner.

Our second contribution is on ensuring robust locomotion under bounded disturbances by reasoning about keyframe state reachability. The third contribution is on using game theory to compose complex dynamic locomotion behaviors sequentially. The final contribution is on reasoning about the correctness of the overall planning framework.

This paper is organized as follows. Section 3 introduces various dynamic locomotion models and the problem formulation of switched systems, phase space planning, and temporal logic preliminaries. In Section 4, we present the task specifications for whole-body dynamic locomotion and a reactive planner winning strategy via defining a two-player logic game. Section 5 introduces a robust finite transition system for the hybrid locomotion process to reason about local robustness with respect to the bounded disturbances and proposes robustness margin sets using phase-space Riemmanian metrics. In Section 6, we reason about the one-walking-step robust reachability and the correctness of the overall planning strategy. Simulation results of whole-body dynamic locomotion behaviors over changing environments are shown in Section 7. In Sections 8 and 9, we discuss the results and the conclusions of this paper. The Appendix presents supplementary mathematical formulations, algorithms, and propositions. A preliminary version of this paper was published in a conference proceeding [Zhao et al. (2016)]. Compared to that proceeding, this paper presents a new study on robust reachability control synthesis, incorporates additional locomotion modes and more diverse task specifications, proposes a replanning strategy, and implements more sophisticated behaviors with a diversity of environmental events.

2 Related Work

Formal methods have been widely investigated for mobile navigation [Kress-Gazit et al. (2011); Raman et al. (2015); DeCastro et al. (2015); Sadigh and Kapoor (2016)]. The authors in [Kloetzer and Belta (2010)] proposed an automated computational framework for decentralized communications and control of a team of mobile robots from global task specifications. This work suffers from high computational complexity and does not address reactive response to environmental changes. To alleviate the computational burden, the work in [Wongpiromsarn et al. (2012)] proposed a receding-horizon based hierarchical framework that reduced the complex synthesis problem to a set of significantly smaller problems with a shorter horizon. An autonomous vehicle navigation process is simulated in the presence of exogenous disturbances. Provable correctness is an important property of temporal logic based control and planning approaches. The work of [Kress-Gazit et al. (2009)] allows mobile robots to react to the environment in real time and guarantees the provable correctness of controllers. The approach proposed in [Liu et al. (2013)] extended controller synthesis with guaranteed-correctness to nonlinear switched systems and designed a reactive mechanism in response to an adversarial environment at runtime. Given a high-level discrete controller encoding reactive task behaviors, the work in [DeCastro and Kress-Gazit (2015)] designed low-level controllers to guarantee the correctness of a high-level controller. More recently, the work of [Duperret and Koditschek (2020)] solves a formal discrete leaping navigation problem of legged robots to reach a goal set while in the interim reactively avoiding a set of obstacle states. However, all of the work above is applied to 2D-world mobile robots or a single-leg hopper, which have simple dynamics unlike our focus on underactuated and hybrid legged robots. Although the recent works in [Warnke et al. (2020); Kulgod et al. (2020); Gu et al. (2021)] explore the use of temporal-logic-based formal methods to solve bipedal robot navigation problems, the studied environments are well structured such as level ground or mild rough terrain with stairs. In addition, contact sequence planning for bipedal robots is straightforward due to the unique option of alternating two legs for contact. Formal-method-based planning for multi-limb robots, such as the one in this paper, requires to consider highly confined environment constraints and contact sequence planning which do not present in grounded mobile or bipedal walking robots.

2.1 Formal methods for manipulation and locomotion

Formal methods have also gained increasing attention in the mobile manipulation community via task and motion planning (TAMP) methods [Kaelbling and Lozano-Pérez (2011); Srivastava et al. (2014); Dantam, Kingston, Chaudhuri, and Kavraki (Dantam et al.); He et al. (2015); Zhao et al. (2021)] or reactive synthesis methods [Sharan (2014); Chinchali et al. (2012); He et al. (2017)]. However, many existing TAMP approaches rely on sampling-based motion planners which ignore the underlying physical dynamics. To fill this gap, the recent work of [Toussaint et al. (2018)] proposed a logic-geometric program to incorporate manipulation dynamics into the task and motion planning process, where discrete logic rules are used to specify the mode sequence for dynamic manipulation tasks. However, this work lacked a reactive mechanism in response to environment actions and manipulated objects. More importantly, formal methods are yet to be used to reason about dynamic legged locomotion, or for more complex dynamic tasks for humanoid robots like the ones described in this paper. The authors in [Antoniotti and Mishra (1995)] determined goals for legged robots by using computational tree logic and synthesized controllers for locomotion. However, their work is restricted to static locomotion tasks which do not allow robots to walk dynamically or jump similarly to humans. An abstraction-based controller was proposed in [Ames et al. (2015)] for bipedal robots using virtual constraints, but this work focused on controller generation without addressing symbolic task reasoning. Recently, the work of [Maniatopoulos et al. (2016)] proposed an end-to-end approach to automatically synthesize temporal-logic-based plans on an Atlas humanoid robot. Reaction to low-level failures was formally incorporated by simply terminating the execution. However, the robot behaviors focus on manipulation and grasping tasks, instead of locomotion behaviors. The work of [Sreenath et al. (2013)] proposed a two-layer hybrid controller for locomotion over varying-slope terrains with imprecise sensing. To account for terrain uncertainties, a high-level controller implements a partially observable Markov decision process to make sequential decisions for controller switching. Once again, this work does not address symbolic task reasoning for dynamic locomotion. In addition, this work is limited to walking on terrains with mild roughness while our focus is locomotion on highly rough terrain and constrained environments.

2.2 Robustness reasoning of formal methods

Robustness to disturbances and reactiveness to changing environments are major challenges in robotic systems. Related work includes [Fainekos and Pappas (2009)] which studies the robust satisfaction of temporal logic specifications associated with continuous-time signals. Signal temporal logic (STL) [Donzé and Maler (2010)] allows to reason about dense-time, real-valued signals, enabling for the evaluation of the extent to which the specifications are satisfied or violated. This property makes STL especially suitable to quantify robustness [Farahani et al. (2015); Deshmukh et al. (2015); Sadraddini and Belta (2015)]. The focus of all the work above is on the robust semantics of temporal logic, while our objective is to design robust locomotion planners where robustness margin sets are quantified as a goal in the reachability analysis under bounded disturbances. The work of [Majumdar et al. (2011)] studied robust controller synthesis on discrete transition systems against disturbances and proposed a robust metric to ensure that the state deviation from the nominal system is bounded by the magnitude of the disturbance. The work of [Topcu et al. (2012)], on the other hand, investigated the amount of uncertainty that can be tolerated while the controller still satisfies the given specifications. Both of the two

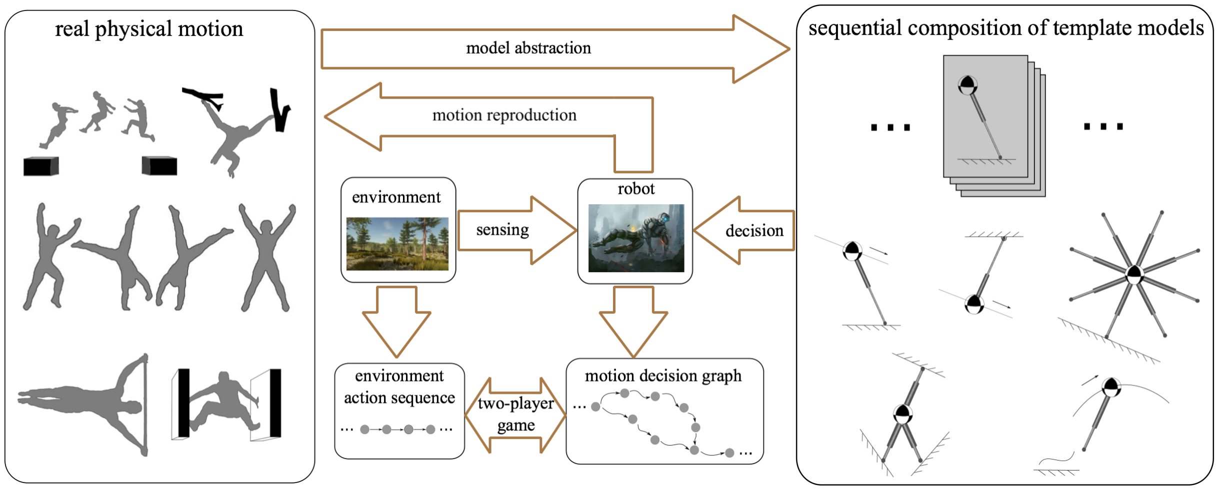


Fig. 2. Illustration of template-based locomotion behaviors dynamically interacting with complex environments. Inspired from real-world human and animal motions, our study focuses on how to make model abstractions and high-level decisions for complex environments. A fundamental problem is how to use template models to characterize essential locomotion modes and sequentially compose these modes to achieve agile and robust locomotion.

papers above, however, focused on robustness reasoning in a purely discrete model, whereas our proposed method reasons about robustness in a hybrid locomotion system and incorporates the underlying physical dynamics. Recently, the work in [Plaku et al. (2010); Bhatia et al. (2010); He et al. (2015)] proposed a multi-layered synergistic framework such that the low-level sampling-based planner communicates with the high-level discrete planner through a middle coordinating layer. This coordinating layer allows the motion planner to ask the task planner for a new high-level plan when a failure occurs at the low level. This synergy between multiple planning layers enhances the robustness of the planning framework. As an alternative, the work of [Dantam et al. (2016)] incrementally incorporated geometric information from the failure event of the motion planner into the task planner via the so-called incremental constraint updates. The robustness in the two lines of research above is reasoned from a replanning perspective. While our study employs a similar replanning strategy as theirs, our focus is on the formal synthesis of a task planner that can react to sudden event changes in the environment. In this paper, we address the robustness as follows: (1) at the task planning level, we devise a reactive mechanism that chooses appropriate system actions according to environmental actions, and (2) at the motion planning level, we achieve robustness against bounded state disturbances by designing robust keyframe transitions for dynamic locomotion.

2.3 *Multi-contact legged locomotion*

Multi-contact locomotion planning and control for humanoid robots have gained good traction as legged robots operate within complex environments more frequently in recent years [Sentis et al. (2010a); Chung and Khatib (2015); Bouyarmane and Kheddar (2011); Hauser (2014); Posa et al. (2016)]. The work in [Bretl (2006)] studied multi-contact locomotion as a hybrid control problem while the work in [Hauser et al. (2005)] posed the multi-contact planning problem as a hierarchy that first reasons about contacts, and then interpolated these contacts with trajectories computed from a probabilistic planner. The study in [Kudruss et al. (2015)] formulated multi-contact centroidal momentum dynamics as an optimal control problem. However, all of the work above focused on either static or quasi-static mobility behaviors. Instead, our planning framework tackles highly dynamic behaviors, i.e., non-periodic multi-contact dynamic locomotion over rough and constrained environments. The work in [Caron et al. (2015)] employed contact wrench cones to geometrically construct dynamic supports in arbitrary virtual planes for multi-contact behaviors. This work did not employ a rich set of locomotion templates due to restrictive assumptions on the center of mass behavior. Once again, all the work does not address symbolic reasoning of dynamic locomotion behaviors.

3 Preliminaries and Problem Formulation

Problem Statement: This study focuses on the reactive and robust synthesis of dynamic whole-body locomotion behaviors for robots equipped with arms and legs to maneuver in complex environments exposed to unexpected emergency events. We use a variety of reduced-order models characterizing the robot's center-of-mass dynamic behaviors. Robot actions are parameterized by discrete contact decisions (i.e., limb contact configurations) while environmental actions are composed of various features, such as stair height variations and emergency events, including the appearance of humans, terrain cracks, high ceilings, and narrow passages. A two-player game based on the linear temporal logic method is employed for the robot to be reactive to environmental events. We combine the reactive synthesis and reachability control to provide formal guarantees for locomotion in terms of correctness and robustness. While the synthesized actions and continuous control policies are designed off-line, we make them available as look-up tables for real-time online execution of reactive whole-body locomotion decisions and control commands. In this paper, we choose a specific set of environmental actions to demonstrate the versatility of our method in employing multiple limbs for locomotion and responding to a diversity of environmental changes and emergency events. Our method is flexible to incorporate more diverse environments, such as including the contact from lateral supporting walls or obstacles coming from different directions.

3.1 Dynamic locomotion modes

We design a phase-space motion planner that consists of a palette of locomotion modes. To begin with, we introduce centroidal momentum dynamics in a general form. Dynamics of mechanical systems can be represented by their rate of linear and angular momenta, which are affected by external wrenches (i.e., force/torque) exerted on the system. We characterize this class of dynamical systems via the balance of moments around the system's centroid.

$$\dot{\boldsymbol{l}} = m\ddot{\boldsymbol{p}}_{\rm com} = \sum_{i}^{N_c} \boldsymbol{f}_i + m\boldsymbol{g}, \tag{1}$$

$$\dot{\boldsymbol{k}} = \sum_{i}^{N_c} (\boldsymbol{p}_i - \boldsymbol{p}_{\rm com}) \times \boldsymbol{f}_i + \boldsymbol{\tau}_i, \tag{2}$$

where $\boldsymbol{l} \in \mathbb{R}^3$ and $\boldsymbol{k} \in \mathbb{R}^3$ represent the centroidal linear and angular momenta, respectively. $\boldsymbol{f}_i \in \mathbb{R}^3$ is the $i^{\rm th}$ ground reaction force, m is the total mass of the robot, $\boldsymbol{g} = (0, 0, -g)^T$ corresponds to the gravity field, $\boldsymbol{f}_{\rm com} = m\ddot{\boldsymbol{p}}_{\rm com} = m(\ddot{x}, \ddot{y}, \ddot{z})^T$ is the vector of center-of-mass inertial forces. Eq. (1) represents the rate of spatial linear momentum is equal to the total linear external forces. $\boldsymbol{p}_i = (p_{i,x}, p_{i,y}, p_{i,z})^T$ is the position of the $i^{\rm th}$ limb contact position. $\boldsymbol{\tau}_i \in \mathbb{R}^3$ is the $i^{\rm th}$ contact torque. Eq. (2) reveals that the rate of angular momentum is equal to the sum of the torques generated by contact wrenches at the CoM.

Given this general model, certain assumptions are commonly imposed to make the problem tractable [Audren et al. (2014)]. In our case, six locomotion modes are proposed to produce various WBDL behaviors.

Mode (a): prismatic inverted pendulum model. For single foot contact, Eq. (2) is simplified to $(\boldsymbol{p}_{\rm com} - \boldsymbol{p}_{\rm foot}) \times (\boldsymbol{f}_{\rm com} + m\,\boldsymbol{g}) = -\boldsymbol{\tau}_{\rm com}$. Given a piece-wise linear CoM path surface to follow, the system dynamics are expressed as

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = \omega^2_{\rm PIPM} \begin{pmatrix} x - x_{\rm foot} - \frac{\tau_y}{mg} \\ y - y_{\rm foot} - \frac{\tau_x}{mg} \end{pmatrix}, \tag{3}$$

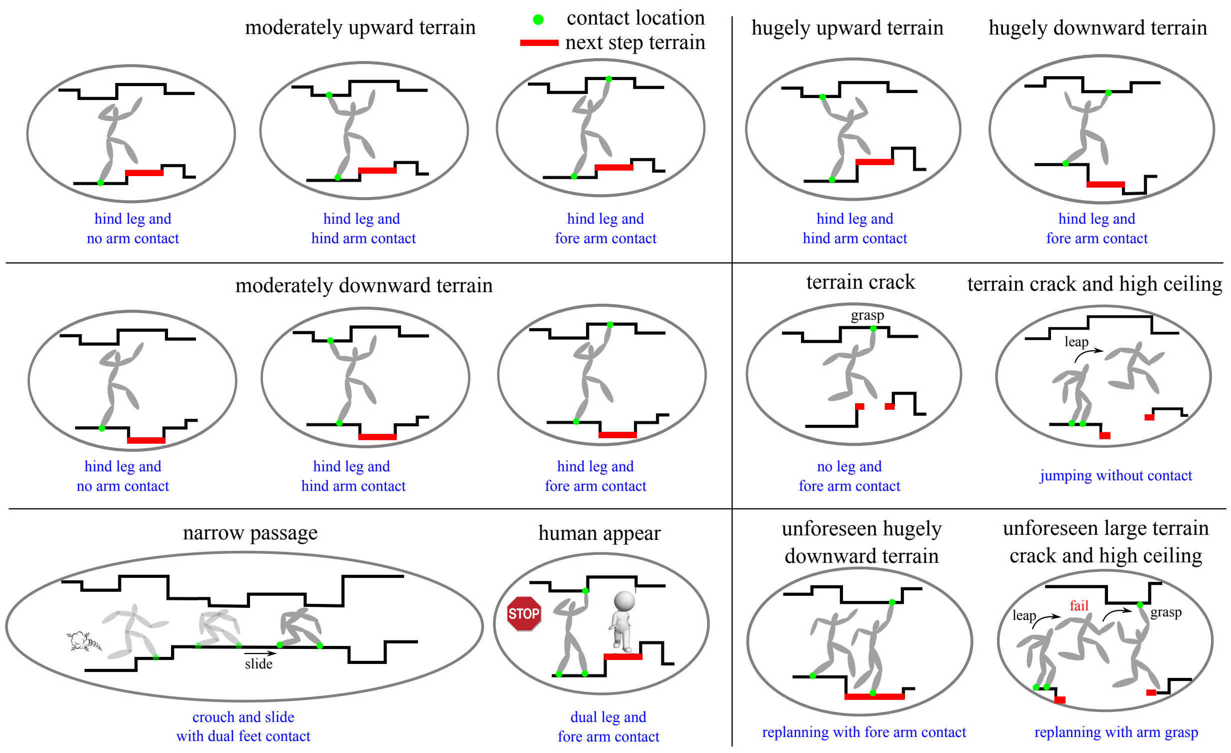

moderately upward terrain
● contact location
🟥 next step terrain
hugely upward terrain
hugely downward terrain
hind leg and no arm contact
hind leg and hind arm contact
hind leg and fore arm contact
hind leg and hind arm contact
hind leg and fore arm contact
moderately downward terrain
terrain crack
terrain crack and high ceiling
grasp
leap
hind leg and no arm contact
hind leg and hind arm contact
hind leg and fore arm contact
no leg and fore arm contact
jumping without contact
narrow passage
human appear
unforeseen hugely downward terrain
unforeseen large terrain crack and high ceiling
STOP
slide
leap
fail
grasp
crouch and slide with dual feet contact
dual leg and fore arm contact
replanning with fore arm contact
replanning with arm grasp

Fig. 3. Contact planning strategies for locomotion in rough terrains. We discretize the terrain height to decide what locomotion actions to take, and define them as environmental actions to set up a two player game decision problem. For instance, given a moderately upward or downward terrain, there can be multiple contact actions to deal with it. Events motivated by ordinary accidents in human daily lives, such as a crack on the terrain and the sudden appearance of a human, are treated as emergency events, and incorporated into the allowable environment. Detailed definitions of environment and system actions are provided in Section 4.1.

where $\ddot{x}$ and $\ddot{y}$ are linear CoM acclerations aligned with sagittal and lateral directions as defined in Eq. (1). The PIPM phase-space asymptotic slope [Zhao et al. (2017)] is defined as $\omega_{\text{PIPM}} = \sqrt{g/z^{\text{apex}}_{\text{PIPM}}}$, $z^{\text{apex}}_{\text{PIPM}} = (a \cdot x_{\text{foot}} + b \cdot x_{\text{foot}} + c - z_{\text{foot}})$, where a and b are the slopes for the piecewise linear CoM path surface $\psi_{\text{CoM}}(x, y, z) = z - ax - by - c = 0$. Thus, the dynamics in the vertical direction are represented by $\ddot{z} = a\ddot{x} + b\ddot{y}$ and not explicitly shown here. The control input is $\boldsymbol{u} = (x_{\text{foot}}, y_{\text{foot}}, \omega_{\text{PIPM}}, \tau_x, \tau_y)^T$. For more details, please refer to the result in [Zhao et al. (2017)].

Mode (b): prismatic pendulum model. When the terrain is cracked, the robot has to grasp the overhead support to swing over an unsafe region using brachiation. The system dynamics can be approximated as a prismatic pendulum model (PPM). For a single hand contact, we have

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = -\omega^2_{\text{PPM}} \begin{pmatrix} x - x_{\text{hand}} - \dfrac{\tau_y}{mg} \\ y - y_{\text{hand}} - \dfrac{\tau_x}{mg} \end{pmatrix}, \tag{4}$$

where similarly we can define $\omega_{\text{PPM}} = \sqrt{g/z^{\text{apex}}_{\text{PPM}}}$, $z^{\text{apex}}_{\text{PPM}} = (z_{\text{hand}} - a \cdot x_{\text{hand}} - b \cdot x_{\text{hand}} - c)$, given the same piece-wise linear CoM path surface $\psi_{\text{CoM}}(x, y, z) = z - ax - by - c = 0$ in Mode (a). Similarly, vertical direction dynamics are represented by $\ddot{z} = a\ddot{x} + b\ddot{y}$. A difference between modes (a) and (b) lies in that PPM dynamics are inherently stable since the CoM is always attracted to move towards the apex position while the PIPM dynamics are not. This study assumes the robot can firmly grasp the overhead support once receiving the upper limb contact command. Fine reasoning of the low-level grasping model and potential failure scenarios are out of the scope of this paper, though important, and will be studied in future work.

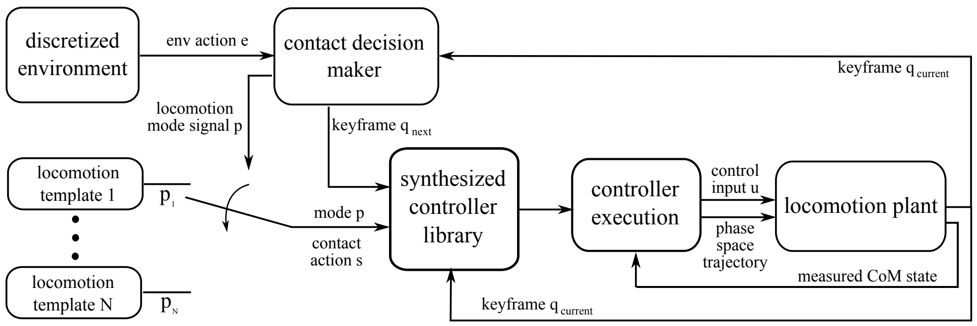


Fig. 4. Logic-based locomotion planner structure. A set of locomotion templates is devised for maneuvering in constrained dynamic environments. Each template is indexed by a locomotion mode signal p. The discrete environment actions are represented by the variable e while control actions are represented by the variable s describing limb contact actions. The discretized dynamic locomotion keyframes are represented by $q = (p_{\rm contact}, \dot{x}_{\rm apex})$. Based on an environmental action e and a keyframe state q at the current walking step, the contact decision maker decides the locomotion mode signal p and the next keyframe locomotion state. More details on the usage of this decision process are discussed in Section 5.

Mode (c): stop-launch model. When a human appears, the robot has to come to a stop, wait until human disappears, and start to move forward. The task in this mode consists on decelerating the CoM motion to zero and accelerating it from zero again. We name this model as a stop-launch model (SLM) with a constant CoM sagittal accelerations. The resulting phase-space trajectory is a parabolic manifold.

Mode (d): multi-contact model. In this mode, a multi-contact model (MCM) is proposed built upon the centroidal momentum dynamics. To make the dynamics tractable, we assume a known constant vertical acceleration a_z in each step and neglect of the angular momentum k_z around the z-axis [Audren et al. (2014)]. Therefore, we have a constant resultant vertical external force, i.e., $\sum_i^{N_c} f_{i,z} = m(\ddot{z} - g)$, where N_c is the number of limb contacts. Since our model has point contacts, $\boldsymbol{\tau}_i = 0, \forall i \leq N_c$, and the dynamics are described by

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\varphi} \\ \ddot{\theta} \end{pmatrix} = \begin{pmatrix} \sum_i^{N_c} f_{i,x}/m \\ \sum_i^{N_c} f_{i,y}/m \\ -(\ddot{z} - g)\cdot y + z \cdot \sum_i^{N_c} f_{i,y}/m - \sum_i^{N_c} p_{i,z}\cdot f_{i,x}/m + \sum_i^{N_c} p_{i,z}\cdot f_{i,z}/m \\ (\ddot{z} - g)\cdot x - z \cdot \sum_i^{N_c} f_{i,x}/m + \sum_i^{N_c} p_{i,z}\cdot f_{i,x}/m - \sum_i^{N_c} p_{i,y}\cdot f_{i,y}/m \end{pmatrix},$$

where φ and θ are torso roll and pitch angles aligned with the CoM sagittal and lateral directions as derived from Eq. (2). The external force vector $(f_{i,x}, f_{i,y}, f_{i,z})$ represents the $i^{\rm th}$ contact force. The vertical position z is a function of x and y defined *a priori*.

Mode (e): hopping model. This model applies when the locomotion model needs to jump over an unsafe region. In this case, the CoM dynamics follow a free-falling ballistic trajectory. We have $\ddot{x} = \ddot{y} = 0, \ddot{z} = -g$. The trajectory is fully controlled by the initial condition, where a discontinuous jump in the CoM state can occur and be used to generate a desired linear momentum. For instance, when the robot jumps over a cracked terrain, it needs to push the ground as the foot lifts to generate a sufficiently large sagittal linear acceleration.

Mode (f): sliding model. This model applies when the robot needs to slide through a constrained region. The CoM dynamics are subject to a constant friction force. Thus, $\ddot{x}$ is a constant negative value, and we assume $\ddot{y} = 0, \ddot{z} = 0$. The sagittal linear velocity decays at a constant rate.

Given the locomotion modes above, we define the set of locomotion modes as

$$\mathcal{P} := \{p_{\rm PIPM}, p_{\rm MCM}, p_{\rm PPM}, p_{\rm SLM}, p_{\rm HM}, p_{\rm SM}\}.$$

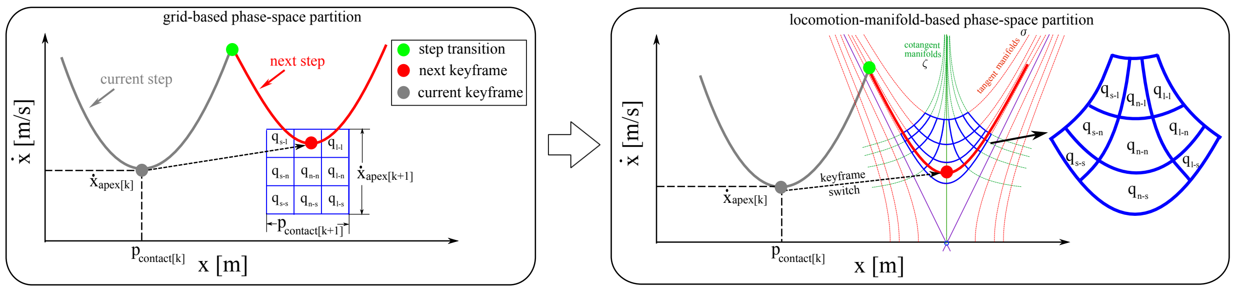


Fig. 5. Phase-space partition of locomotion manifolds for keyframe design. The left figure shows a grid-based partition while the right figure is a non-Euclidean partition that follows the phase-space locomotion manifolds. The latter partition is consistent with locomotion dynamics and we define it as the "locomotion-manifold-based partition". This partition will be used to achieve robust locomotion. We use different granularities for two orthogonal axes.

All the locomotion modes above are illustrated in Fig. 3. Each mode has closed-form solutions for their phase-space tangent and cotangent manifolds as will be derived in Section 5 and Appendix C. The timing synchronization between the sagittal and lateral dynamics is guaranteed by a Newton-Raphson foot placement searching algorithm [Zhao et al. (2017)]. Likewise, more complex tasks can be defined in the locomotion mode set $\mathcal{P}$. For instance, cartwheel, dense gaps, and spinkick behaviors as shown in [Peng et al. (2018)] are promising behaviors to be explored.

Our phase-space planning process produces three-dimensional locomotion. However, the planning framework of this study focuses on forward walking using sagittal keyframes. Given high-level sagittal keyframes, the robot's lateral dynamic behavior is automatically computed by our motion planner. Turning behaviors can be incorporated in our framework by using the method that we introduced in[Zhao et al. (2017)].

3.2 *Switched systems and phase-space planning*

Given the continuous locomotion modes above, we formulate the locomotion planning problem as a switched system [Liberzon (2012)]. The dynamics of the whole-body dynamic locomotion (WBDL) process are defined as

$$\dot{\boldsymbol{\xi}}(\zeta) = f_p\big(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta), d(\zeta)\big),\ p \in \mathcal{P}, \tag{5}$$

where $\boldsymbol{\xi}(\zeta) \in \Xi \subseteq \mathbb{R}^{12}$ denotes the full system state vector at $\zeta \in \mathbb{R}_{\geq 0}$, i.e., the twelve dimensional center-of-mass position and angular state vector of the robot during the locomotion process[1]. The phase progression variable ζ, analogous to time, represents the current phase progression on a locomotion trajectory. The control input is denoted by $\boldsymbol{u}(\zeta) = (\boldsymbol{p}_{\text{contact}}, \omega, \tau_x, \tau_y, \tau_z) \in \mathcal{U}$, where $\boldsymbol{p}_{\text{contact}}$ represents a set of contact position vectors, where each contact position vector is three-dimensional; ω represents the slope of the phase-space asymptote dependent on specific locomotion modes as defined in Section 3.1; and (τ_x, τ_y, τ_z) represents a three-dimensional torso torque vector. Each locomotion mode merely involves a subset of the full state and control vectors. In addition, $d \in \mathcal{D} \subseteq \mathbb{R}^d$ represents an external disturbance. The locomotion mode p (i.e., the locomotion mode) indexes a specific locomotion mode belonging to the set $\mathcal{P}$ and $f_p(\cdot)$ denotes a vector field associated with the locomotion mode p. A logic-based switched system modeling the locomotion process is shown in Fig. 4.

Our phase-space planning is a three-dimensional hybrid bipedal locomotion planning framework based on robustly tracking a set of non-periodic keyframe states. This framework focuses on non-periodic gait generation for robust and agile locomotion over various challenging terrains and under external disturbances. The keyframe state in the phase-space is defined as

[1] The state vector $\boldsymbol{\xi}$ is reused to represent the center-of-mass sagittal states $(x, \dot{x})$ when we model specific locomotion modes in later sections.

Definition 1 (Phase-space keyframe). *A keyframe state in the phase-space of a locomotion system is a critical point on the locomotion manifold normally located either at the point of minimal or maximal velocity, or at an approximately central position of the phase-space manifold of one continuous walking step (see the gray and red dots in Fig. 5).*

In general, this keyframe state refers to the apex state when the center-of-mass (CoM) velocity reaches the local minimal or maximum velocity in the CoM sagittal axis.[2] Given two consecutive keyframe states, the phase-space planner evolves continuously and computes the contact transitions of one walking step as defined below.

Definition 2 (Phase-space contact switch). *A phase-space contact switch, i.e., a contact transition, is defined by the intersection of two adjacent phase-space trajectories (see green dot in Fig. 5(a)).*

Our contact-triggered switching strategy is especially suitable for non-periodic locomotion, which is abstracted as a progression map Φ between keyframe states, that is, driving the robot's center-of-mass from one desired keyframe to the next one via the control input $\boldsymbol{u}$, i.e. $(p_{\text{contact}_{k+1}}, \dot{x}_{\text{apex}_{k+1}}) = \Phi(p_{\text{contact}_k}, \dot{x}_{\text{apex}_k}, \boldsymbol{u})$, where p_{contact_k} and $\dot{x}_{\text{apex}_k}$ denote the k^{th}-step CoM sagittal position and velocity at the contact apex, respectively. To accomplish whole-body dynamic locomotion behaviors, we will compose a sequence of locomotion modes with planned keyframes. This can be achieved by synthesizing a high-level task planner protocol which makes proper contact decisions like the ones shown in Fig. 3 and determines the switching strategy of the low-level motion planner.

Definition 3 (One walking step in the phase-space). *One walking step (OWS) of the locomotion process is defined as two consecutive semi-step phase-space trajectories (see Fig. 5(a)). The first semi-step trajectory starts at the first keyframe state (gray dot) and ends at the contact switch (green dot) while the second semi-step trajectory starts at the contact switch and ends at the second keyframe (red dot).*

Instead of using generalized coordinates associated with the robot joints, our planning framework chooses to use the robot's center-of-mass state as the output space. This simplified coordinate choice is often used in the locomotion communities. Alternatives for dimensionality reduction include, for instance, differential flatness [Liu et al. (2012)] and partial hybrid zero dynamics [Ames et al. (2015)].

The switched system dynamics in Eq. (5) can be represented by a tuple

$$\mathcal{SS} = (\Xi, \Xi_0, \mathcal{U}, \mathcal{P}, f, AP, \mathcal{L}) \tag{6}$$

where $\Xi_0 \subseteq \Xi$ is a set of initial conditions, AP is a set of atomic propositions and $\mathcal{L} : \Xi \rightarrow 2^{AP}$ is a labeling function. Then a control strategy for $\mathcal{SS}$ is a partial function defined as

$$\Omega_i(\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_i) = \boldsymbol{u}_i \in \mathcal{U}^{[0,\Delta\zeta_i]}, \forall i = 0, 1, 2, ... \tag{7}$$

where $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_i$ is a finite sequence of sampled states evaluated at discrete phase progression instants $\zeta_0, \zeta_1, \ldots, \zeta_i$ satisfying $\zeta_{j+1} - \zeta_j = \Delta\zeta_i, \forall 0 \le j \le i-1$, and $\mathcal{U}^{[0,\Delta\zeta_i]}$ denotes the set of control input signals from $[0, \Delta\zeta_i]$ to $\mathcal{U}$. It is assumed that $\boldsymbol{u}_i$ is the constant control input with a phase progression duration $\Delta\zeta_i$.

Contact switching planner synthesis problem: Given a switched system $\mathcal{SS}$ in Eq. (6) and a specification φ expressible in the linear temporal logic (LTL) form, synthesize a contact planning strategy for the system that (i) only generates correct phase-space trajectories $\kappa = (\boldsymbol{\xi}, \rho, \eta, \mu)$ in the sense that $\kappa \models \varphi$ for all initial conditions in Ξ_0, (ii) generates a locomotion mode μ in response to the environment actions at runtime. φ is realizable by $\mathcal{SS}$ if there exists such a switching strategy. ρ, η and μ are the continuous counterparts of the discrete environment action e, contact action s, and locomotion mode p. More detailed definitions will be introduced in Section 5.

[2] In the special case of the phase-space trajectory having a constant slope, we choose the CoM state locating at the central position of the phase-space trajectory as the keyframe state.

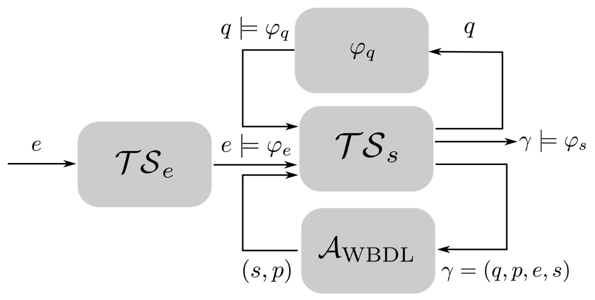


Fig. 6. The interconnected feedback diagram of system and environment finite transition systems $\mathcal{TS}_s$ and $\mathcal{TS}_e$, and a winning strategy $\mathcal{A}_{\text{WBDL}}$ synthesized in Section 4.4.

3.3 *Finite transition systems and LTL preliminaries*

We now define system, environment, and product finite transition systems and describe linear temporal logic (LTL) preliminaries.

Definition 4 (Finite transition system of the robot system). *A finite transition system of the robot system is a tuple,*

$$\mathcal{TS}_s := (\mathcal{Q}, \mathcal{P}, \mathcal{S}, \mathcal{T}_s, \mathcal{I}_s, AP_s, \tilde{\mathcal{L}}_s), \tag{8}$$

where $\mathcal{Q}$ is a finite set of states, $\mathcal{P}$ is a set of system modes as mentioned in Eq. (5), $\mathcal{S}$ is a finite set of controllable robot contact actions, $\mathcal{T}_s \subseteq \mathcal{Q} \xrightarrow{\mathcal{P}\times\mathcal{S}} \mathcal{Q}$ is a transition, $\mathcal{I}_s = \mathcal{Q}_0 \subseteq \mathcal{Q}$ is a set of initial states, AP_s is a set of atomic propositions, $\tilde{\mathcal{L}}_s : \mathcal{Q} \to 2^{AP_s}$ is a labeling function mapping the state to an atomic proposition. $\mathcal{TS}_s$ is finite if $\mathcal{Q}, \mathcal{P}, \mathcal{S}$ and AP_s are finite.

Definition 5 (Finite transition system of the environment). *A finite transition system of the environment is a tuple,*

$$\mathcal{TS}_e := (\mathcal{E}, \mathcal{T}_e, \mathcal{I}_e, AP_e, \tilde{\mathcal{L}}_e), \tag{9}$$

where $\mathcal{E}$ is a finite set of environmental states, $\mathcal{T}_e \subseteq \mathcal{E} \times \mathcal{E}$ is a transition, $\mathcal{I}_e = \mathcal{E}_0 \subseteq \mathcal{E}$ is a set of initial states, AP_e is a set of atomic propositions, $\tilde{\mathcal{L}}_e : \mathcal{E} \to 2^{AP_e}$ is a labeling function mapping the state to an atomic proposition. $\mathcal{TS}_e$ is finite if $\mathcal{E}$ and AP_e are finite.

Definition 6 (Open finite product transition system). *Given $\mathcal{TS}_s$ and $\mathcal{TS}_e$, we define an open finite product transition system (OFPTS) to describe the overall system behavior, including the robot and its environment as,*

$$\mathcal{TS}_{\text{prod}} := (\mathcal{Q}, \mathcal{P}, \mathcal{S}, \mathcal{E}, \mathcal{T}, \mathcal{I}, \widetilde{AP}, \tilde{\mathcal{L}}), \tag{10}$$

where $\mathcal{Q}, \mathcal{P}$ and $\mathcal{S}$ are defined as previously, $\mathcal{E}$ is a finite set of uncontrollable environmental actions, $\mathcal{F} = \mathcal{Q} \times \mathcal{P} \times \mathcal{S} \times \mathcal{E}$, $\mathcal{T} \subseteq \mathcal{F} \to \mathcal{F}$ is a transition, $\mathcal{I} = \mathcal{F}_0 \subseteq \mathcal{F}$ is a set of initial states, $\widetilde{AP}$ is a set of atomic propositions, $\tilde{\mathcal{L}} : \mathcal{Q} \to 2^{\widetilde{AP}}$ is a labeling function mapping the state to an atomic proposition. $\mathcal{TS}_{\text{prod}}$ is finite if $\mathcal{Q}, \mathcal{P}, \mathcal{E}, \mathcal{S}$ and $\widetilde{AP}$ are finite.

Note that, the environment states $\mathcal{E}$ in $\mathcal{TS}_e$ are treated as uncontrollable actions in $\mathcal{TS}_{\text{prod}}$. This is why $\mathcal{TS}_{\text{prod}}$ is called a "open" finite transition system [Topcu et al. (2012)]. Without loss of generality, it is assumed that for every pair $(q, e) \in \mathcal{Q} \times \mathcal{E}$, there exists at least one pair (p, s) such that $(q, e) \xrightarrow{p,s} (q', e')$. The OFPTS considered in this study has non-deterministic transitions.

Definition 7 (Execution and word of an OFPTS). *An execution γ of an OFPTS $\mathcal{TS}_{\text{prod}}$ is an infinite path sequence $\gamma = (q_0, p_0, e_0, s_0)(q_1, p_1, e_1, s_1)(q_2, p_2, e_2, s_2)\ldots$, with $\gamma_i = (q_i, p_i, e_i, s_i) \in \mathcal{Q} \times \mathcal{P} \times \mathcal{E} \times \mathcal{S}$ and $\gamma_i \xrightarrow{\mathcal{T}} \gamma_{i+1}$. The word generated from γ is $w_\gamma = w_\gamma(0)w_\gamma(1)w_\gamma(2)\ldots$, with $w_\gamma(i) = \tilde{\mathcal{L}}(\gamma_i),\ \forall i \geq 0$.*

The word w_γ is said to satisfy a LTL formula φ, if and only if the execution γ satisfies φ. If all executions of $\mathcal{TS}_{\text{prod}}$ satisfy φ, we say that $\mathcal{TS}_{\text{prod}}$ satisfies φ, i.e., $\mathcal{TS}_{\text{prod}} \models \varphi$. Please refer to Fig. 6 for an illustration of the finite transition systems. Linear temporal logic is an extension of propositional logic that incorporates temporal operators. Preliminaries of linear temporal logic are explained in Appendix B.

3.4 Discrete task planner synthesis formulation

Given the preliminaries above, we formulate a discrete task planner synthesis problem and introduce a specific fragment of the temporal logic for the task specifications.

Discrete task planner synthesis problem: Given a product transition system $\mathcal{TS}_{\text{prod}}$ and a LTL specification φ following the assume-guarantee form [Bloem et al. (2012)],

$$\varphi := \big(\varphi_e \Rightarrow (\varphi_q \wedge \varphi_s)\big), \tag{11}$$

where φ_e and φ_q, φ_s are propositions for the admissible environment actions, the keyframe states, and the correct overall system behavior, respectively; in particular, φ_s incorporates the behaviors of locomotion mode p and contact action s; we synthesize a contact planner switching strategy γ that generates only correct executions (q, p, e, s), i.e., $(q, p, e, s) \models \varphi$.

To make the computation tractable, we employ a fragment of LTL formulae with a favorable polynomial complexity, named the Generalized Reactivity (1) (GR (1)) formulae [Bloem et al. (2012)]. This class of formulae is expressed as, for $v \in \{e, q, s\}$,

$$\varphi_v = \varphi^v_{\text{init}} \bigwedge_{i \in I_{\text{safety}}} \Box \varphi^v_{\text{trans},i} \bigwedge_{i \in I_{\text{goal}}} \Box\Diamond \varphi^v_{\text{goal},i}, \tag{12}$$

where φ^v_{init} are the propositional formulae defining initial conditions. $\varphi^v_{\text{trans},i}$ refer to the transitional propositional formulae (i.e., safety conditions) incorporating the state at next step. $\varphi^v_{\text{goal},i}$ are the propositional formulae describing the goals to be reached infinitely often (i.e., liveness conditions).

Remark 1. *The GR(1) formula is an efficient fragment of LTL and reasons over a rich set of states and actions and makes the task planner synthesis process tractable. A motivation of using this automated synthesis is to lay the theoretical foundation of devising a correct-by-construction decision-maker for composing complex locomotion trajectories.*

4 Task Planning for Whole-body Dynamic Locomotion

In this section, we introduce the temporal logic specifications for locomotion in a possibly adversarial environment. We will specify a two-player game where the environment and keyframe state are the first player while the robot action is the second player. Our task specifications will capture two types of environmental events: (i) varying height terrains are treated as ordinary events; and (ii) sudden incidents, such as a person appearing on the robot's path or a crack on the terrain, are treated as adversarial actions, since if the robot does not respond properly they may cause an accident.

The design of linear temporal logic (LTL) specifications relies on human designers who specify locomotion tasks and models of the environment. Our task specifications below are designed according to locomotion heuristics. In general, there are no unique ways to evaluate the efficacy of the LTL design process. For instance, we could decide to add an additional environmental specification to forbid repeating the same environmental actions terrainCrack-normalCeiling $e_{\text{tc-nc}}$, expressed as $\square(e_{\text{tc-nc}} \Rightarrow \neg\bigcirc e_{\text{tc-nc}})$. Using locomotion heuristics is an effective way to generate natural and safe locomotion behaviors. The heuristics that we have chosen employ human intuition, which often lead to natural and recognizable behaviors. In addition, another set of heuristics is used to guarantee safety.

4.1 Environment specifications

As previously stated, we treat the environment as a player "acting" against the robot's locomotion process. We define an environmental action set, $\mathcal{E}$, as the composition of two subsets: a set for varying height terrain and a set for emergencies (i.e., the so-called sudden events), respectively.

$$\mathcal{E} := \mathcal{E}_{\text{terrain}} \cup \mathcal{E}_{\text{emergency}} = \{e_{\text{md}}, e_{\text{hd}}, e_{\text{mu}}, e_{\text{hu}}\} \cup \{e_{\text{tc-nc}}, e_{\text{tc-hc}}, e_{\text{ha}}, e_{\text{np}}\}, \tag{13}$$

where the elements in the set $\mathcal{E}_{\text{terrain}}$ denote different height terrain actions, as illustrated in Fig. 3. For instance, e_{md} denotes moderatelyDownward terrain. The actions in $\mathcal{E}_{\text{emergency}}$ represent sudden events, i.e. terrainCrack-normalCeiling, terrainCrack-highCeiling, humanAppear, and narrowPassage. The environmental action set specified above is generalizable to other environmental events while maintaining computational tractability. Given the environmental actions above, we design the following specifications. First, the following sudden environmental actions are assumed to not occur at the initial instant:

$$\varphi^{e}_{\text{init}} = \neg e_{\text{tc-nc}} \wedge \neg e_{\text{tc-hc}} \wedge \neg e_{\text{ha}} \wedge \neg e_{\text{np}} \tag{14}$$

Since only one environmental action can be True at any time, we enforce the following transitional proposition

$$\square\Big(\big(e_{\text{md}} \wedge \bigwedge_{e\in\mathcal{E}\backslash e_{\text{md}}} (\neg e)\big) \bigvee \big(e_{\text{hd}} \wedge \bigwedge_{e\in\mathcal{E}\backslash e_{\text{hd}}} (\neg e)\big) \bigvee \ldots \bigvee \big(e_{\text{np}} \wedge \bigwedge_{e\in\mathcal{E}\backslash e_{\text{np}}} (\neg e)\big)\Big). \tag{15}$$

where the operator $\bigwedge_{e\in\mathcal{E}\backslash e_{\text{md}}}$ is used to represent the conjunction of multiple environmental propositions $\neg e, \forall e \in \mathcal{E}\backslash e_{\text{md}}$. To enable the robot to maneuver through the dynamic environment, certain sudden environmental actions are forbidden to occur consecutively, as shown in the following transitional specifications:

- ($S_{e\text{-}1}$) If the current environmental action is terrainCrack-highCeiling, then the next environmental action can not be terrainCrack-highCeiling, humanAppear, nor narrowPassage.

$$\square\Big(e_{\text{sc-hc}} \Rightarrow \neg(e_{\text{sc-hc}} \wedge e_{\text{ha}} \wedge e_{\text{np}})\Big) \tag{16}$$

- ($S_{e\text{-}2}$) If the current environmental action is terrainCrack-normalCeiling, then the next environmental action can not be terrainCrack-highCeiling, humanAppear, nor narrowPassage.

$$\square\Big(e_{\text{sc-nc}} \Rightarrow \neg(e_{\text{sc-hc}} \wedge e_{\text{ha}} \wedge e_{\text{np}})\Big) \tag{17}$$

- ($S_{e\text{-}3}$) If the current environmental action is narrowPassage, then the next environmental action can not be terrainCrack-normalCeiling nor terrainCrack-highCeiling.

$$\square\Big(e_{\text{np}} \Rightarrow \neg(e_{\text{sc-hc}} \wedge e_{\text{sc-nc}})\Big) \tag{18}$$

To evaluate the effectiveness of our proposed approach handling all the allowable environmental actions, we enforce them to occur infinitely often via the goal proposition:

$$\varphi^e_{\text{goal}} = (\Box\Diamond e_{\text{md}}) \wedge (\Box\Diamond e_{\text{hd}}) \wedge \ldots \wedge (\Box\Diamond e_{\text{np}}) \tag{19}$$

To ensure the robot makes progress (i.e., continuously moves forward within the constrained environment), we define the following liveness condition:

$$\varphi^e_{\text{liveness}} \coloneqq \neg\Diamond\Box e_{\text{ha}} \wedge \neg\Diamond\Box e_{\text{np}} \tag{20}$$

which is consistent with the goal proposition of Eq. (12). This specification establishes that the robot cannot eventually always encounter the conditions humanAppear or narrowPassage. In fact, this liveness condition should also include the environmental action terrainCrack-highCeiling, i.e., $\neg\Diamond\Box e_{\text{sc-hc}}$, which is already guaranteed by $\Box(e_{\text{sc-hc}} \Rightarrow \neg e_{\text{sc-hc}})$ in specification ($S_{\text{e-1}}$).

4.2 *Robot specifications*

To maneuver in the environment using whole-body dynamic locomotion, we define the following robot actions

$$\mathcal{S} \coloneqq \{s_{\text{li-aj}},\ \forall(i,j) \in \mathcal{S}_{\text{index}}\}, \tag{21}$$

where the indices 'l' and 'a' are short for leg and arm, respectively. $(i,j) \in \mathcal{S}_{\text{index}}$ corresponds to the contact limb with $\mathcal{S}_{\text{index}} = \{(\mathrm{h},\mathrm{n}), (\mathrm{h},\mathrm{h}), (\mathrm{h},\mathrm{f}), (\mathrm{d},\mathrm{h}), (\mathrm{d},\mathrm{f}), (\mathrm{d},\mathrm{d}), (\mathrm{d},\mathrm{n}), (\mathrm{n},\mathrm{f}), (\mathrm{n},\mathrm{n})\}$, where the letters 'h', 'f', 'd' and 'n' represent hind, fore, dual and no contacts, respectively. For instance, $s_{\text{lh-af}}$ specifies the legHindArmFore contact action in the sense that the robot's hind leg and the fore arm are in contact for that action while the other two limbs are not in contact. Notice that we don't specify left and right limbs explicitly as the hind and fore adjectives lead to unique assignments during the locomotion process.

We enforce the robot not to take actions responding to emergency events of the environment $\mathcal{E}_{\text{emergency}}$, i.e., $\varphi^s_{\text{init}} = \neg s_{\text{ln-af}} \wedge \neg s_{\text{ln-an}} \wedge \neg(s_{\text{ld-ah}} \vee s_{\text{ld-af}})$, which are already guaranteed by the initial propositions defined for the environmental actions in Eq. (14). Given a specific set of locomotion modes $\mathcal{P}$ as defined in Section 3.1, the robot transitional specifications φ^s_{trans} are defined as follows:

- ($S_{\text{robot-1}}$) Robot actions in response to varying-height terrain $\mathcal{E}_{\text{terrain}}$ are specified as

$$\Box\Big((e_{\text{md}} \vee e_{\text{mu}}) \Rightarrow (p_{\text{PIPM}} \wedge s_{\text{lh-an}}) \vee \big(p_{\text{MCM}} \wedge (s_{\text{lh-ah}} \vee s_{\text{lh-af}})\big)\Big)$$
$$\bigwedge \Box(e_{\text{hu}} \Rightarrow p_{\text{MCM}} \wedge s_{\text{lh-ah}}) \bigwedge \Box(e_{\text{hd}} \Rightarrow p_{\text{MCM}} \wedge s_{\text{lh-af}}),$$

  where moderate terrain variations allow for the use of more robot contact actions than in the case of huge terrain variations. For instance, if $e = e_{\text{hu}}$, i.e. the terrain has an action hugelyUpward, the robot has only one action to choose from, consisting of using its hind arm for contact such that it can push forward its center of mass to overcome the huge terrain variation as shown in Fig. 3.
- ($S_{\text{robot-2}}$) If the environmental action terrainCrack-normalCeiling occurs, i.e., a crack on the terrain appears and the ceiling above the robot has a normal height (assumed to be accessible by the robot), the robot will grab a supposedly existing handle on the overhead support using its forearm (i.e., $s_{\text{ln-af}}$). On the other hand, when there is no crack on

the terrain, we don't allow the use of that action:

$$\square(e_{\text{tc-nc}} \Rightarrow p_{\text{PPM}} \wedge s_{\text{ln-af}}) \bigwedge \square(\neg e_{\text{tc-nc}} \Rightarrow \neg p_{\text{PPM}} \wedge \neg s_{\text{ln-af}}).$$

- ($S_{\text{robot-3}}$) If the environmental action humanAppear occurs, i.e., a person appears in front of the robot, the robot comes to a stop using the legDual contacts and the arm contacts. On the other hand, when the person disappears, the robot should continue walking from where it stopped before:

$$\square\big(e_{\text{ha}} \Rightarrow p_{\text{SLM}} \wedge (s_{\text{ld-ah}} \vee s_{\text{ld-af}} \vee s_{\text{ld-an}})\big) \bigwedge \square\big(\neg e_{\text{ha}} \Rightarrow \neg p_{\text{SLM}} \wedge \neg(s_{\text{ld-ah}} \vee s_{\text{ld-af}} \vee \neg s_{\text{ld-an}})\big).$$

- ($S_{\text{robot-4}}$) If a narrow passage narrowPassage appears, the robot will slide on the ground using two feet and no arm contacts. On the other hand, if there is no narrow passage, the robot will not use the sliding mode

$$\square(e_{\text{np}} \Rightarrow p_{\text{SM}} \wedge s_{\text{ld-an}}) \bigwedge \square(\neg e_{\text{np}} \Rightarrow \neg p_{\text{SM}}).$$

- ($S_{\text{robot-5}}$) If the environmental action terrainCrack-highCeiling appears, i.e., a crack appears on the terrain and there is a high ceiling, the robot will have to leap over the cracked region using a hopping motion (i.e., $s_{\text{ln-an}}$). On the other hand, when this environmental action does not occur, we do not allow to use that action:

$$\square(e_{\text{tc-hc}} \Rightarrow p_{\text{HM}} \wedge s_{\text{ln-an}}) \bigwedge \square(\neg e_{\text{tc-hc}} \Rightarrow \neg p_{\text{HM}} \wedge \neg s_{\text{ln-an}}).$$

As for the goal proposition of the robot, we require that all locomotion modes and contact actions will occur infinitely often to verify their correctness.

$$\varphi^s_{\text{goal}} = (\square\diamond p_{\text{PIPM}}) \wedge (\square\diamond s_{\text{ld-ah}}) \wedge (\square\diamond s_{\text{ld-af}}) \wedge (\square\diamond s_{\text{ld-an}}) \tag{22}$$

where we do not list all the goal propositions of locomotion modes and contact actions. The reason is that the other goal propositions regarding contact actions and locomotion modes are implied by the goal propositions of the environment defined in Eq. (19).

4.3 Keyframe specifications

Our phase-space motion planner relies on a keyframe state vector $q = \{p_{\text{contact}}, \dot{x}_{\text{apex}}\}$ as defined in Section 3.2. In the task planner, the keyframe state is designed to be non-deterministic. We define a discretized phase-space region to choose keyframe states for each walking step using a Riemannian geometry decomposition as shown in Fig. 5(b). The keyframe states consist of ordinary and special types (see further below)

$$\begin{aligned}\mathcal{Q} := \mathcal{Q}_{\text{ordinary}} \cup \mathcal{Q}_{\text{special}} =&\{q_{i\text{-}j\text{-}k},\ i \in \mathcal{I}_{\text{ordinary-behavior}},\ \forall(j,k) \in \mathcal{I}_{\text{level}} \times \mathcal{I}_{\text{level}}\}\\ &\cup \{q_{i\text{-}j},\ i \in \mathcal{I}_{\text{special-behavior}},\ \forall j \in \mathcal{I}_{\text{level}}\}\end{aligned} \tag{23}$$

where ordinary behaviors are $\mathcal{I}_{\text{ordinary-behavior}} = \{$walk, brachiation$\}$ while special behaviors are $\mathcal{I}_{\text{special-behavior}} = \{$stop, hop, slide$\}$. A apex velocity index j and a step length index k refer to the set $\mathcal{I}_{\text{level}} = \{s, m, l\}$ whose elements are three different keyframe "levels": s (Small), m (Medium) and l (Large).[3] For instance, $q_{\text{walk-s-l}}$ represents walkSmallVelocityLargeStep, a walking keyframe with a small apex velocity, and a large step length. In our case, the ordinary locomotion

[3] More levels can be introduced at the expense of a combinatorial increase on the total number of the keyframe states.

behaviors (i.e., walk and brachiation) comprise 9 keyframe states, respectively while the special locomotion behaviors (i.e., stop, hop and slide) comprise 3 keyframe states, respectively.

Given the environmental actions in Section 4.1, the specifications for keyframe states are designed as follows.

- ($S_{q\text{-}1}$) If the next environmental action is moderatelyDownward e_{md}, the level for the next keyframe state q remains constant or increases by one level either from step length or apex velocity:

$$\begin{aligned}
&\Box\big((q_{\text{walk-s-s}} \wedge \bigcirc e_{\text{md}}) \Rightarrow \bigcirc(q_{\text{walk-s-s}} \vee q_{\text{walk-s-m}} \vee q_{\text{walk-m-s}})\big)\\
&\bigwedge\Box\big((q_{\text{walk-s-m}} \wedge \bigcirc e_{\text{md}}) \Rightarrow \bigcirc(q_{\text{walk-s-m}} \vee q_{\text{walk-s-l}} \vee q_{\text{walk-m-m}})\big)\\
&\qquad \dots\\
&\bigwedge\Box\big((q_{\text{walk-l-m}} \wedge \bigcirc e_{\text{md}}) \Rightarrow \bigcirc(q_{\text{walk-l-m}} \vee q_{\text{walk-l-l}})\big) \bigwedge \Box\big((q_{\text{walk-m-l}} \wedge \bigcirc e_{\text{md}}) \Rightarrow \bigcirc(q_{\text{walk-m-l}} \vee q_{\text{walk-l-l}})\big)\\
&\bigwedge\Box\big((q_{\text{walk-l-l}} \wedge \bigcirc e_{\text{md}}) \Rightarrow \bigcirc q_{\text{walk-l-l}}\big) \bigwedge \Box\Big(\big((q_{\text{brachiation}} \vee q_{\text{stop}}) \wedge \bigcirc e_{\text{md}}\big) \Rightarrow \bigcirc(q_{\text{walk-s-m}} \vee q_{\text{walk-m-m}} \vee q_{\text{walk-l-m}})\Big),
\end{aligned}$$

  where, if $q = q_{\text{walk-s-s}}$, $\bigcirc q$ can be $q_{\text{walk-s-s}}$ (remaining constant), $q_{\text{walk-s-m}}$ (step length increases one level) or $q_{\text{walk-m-s}}$ (apex velocity increases one level). All the other keyframes in ordinary scenarios follow the same pattern. There are three special cases: (i) when $q = q_{\text{walk-l-m}}$, there are only two choices for $\bigcirc q$, i.e., $q_{\text{walk-l-m}}$ and $q_{\text{walk-l-l}}$; (ii) the same situation applies to $q_{\text{walk-m-l}}$; (iii) when $q = q_{\text{walk-l-l}}$, the only choice is $\bigcirc(q = q_{\text{walk-l-l}})$. In emergency cases, we assign $\bigcirc q$ by $q_{\text{walk-s-m}}$, $q_{\text{walk-m-m}}$ or $q_{\text{walk-l-m}}$.
- ($S_{q\text{-}2}$) If the next environmental action is hugelyDownward e_{hd}, the level for the next keyframe state increases by one or two units, either on the step length or on the apex velocity. The only exception is as follows: when the current keyframe is $q = q_{\text{l-l}}$, then the next step is only allowed to choose the keyframe $q_{\text{l-l}}$.

$$\begin{aligned}
&\Box\big((q_{\text{walk-s-s}} \wedge \bigcirc e_{\text{hd}}) \Rightarrow \bigcirc(q_{\text{walk-m-s}} \vee q_{\text{walk-s-m}} \vee q_{\text{walk-l-s}} \vee q_{\text{walk-s-l}} \vee q_{\text{walk-m-m}})\big)\\
&\bigwedge\Box\big((q_{\text{walk-s-m}} \wedge \bigcirc e_{\text{hd}}) \Rightarrow \bigcirc(q_{\text{walk-s-l}} \vee q_{\text{walk-m-m}} \vee q_{\text{walk-m-l}})\big)\\
&\qquad \dots\\
&\bigwedge\Box\Big(\big((q_{\text{walk-l-m}} \vee q_{\text{walk-m-l}} \vee q_{\text{walk-l-l}}) \wedge \bigcirc e_{\text{hd}}\big) \Rightarrow \bigcirc q_{\text{walk-l-l}}\Big)\\
&\bigwedge\Box\Big(\big((q_{\text{branchiation}} \vee q_{\text{stop}}) \wedge \bigcirc e_{\text{hd}}\big) \Rightarrow \bigcirc(q_{\text{walk-s-m}} \vee q_{\text{walk-m-m}} \vee q_{\text{walk-l-m}})\Big)
\end{aligned}$$

  where, if $q = q_{\text{walk-s-s}}$, $\bigcirc q$ increases by (i) one unit level, i.e., $q_{\text{walk-s-m}}$ and $q_{\text{walk-m-s}}$, or (ii) two unit levels, i.e., $q_{\text{walk-m-m}}$, $q_{\text{walk-l-s}}$ and $q_{\text{walk-s-l}}$. Special cases are $q_{\text{walk-l-m}}$, $q_{\text{walk-m-l}}$ and $q_{\text{walk-l-l}}$ where $q_{\text{walk-l-l}}$ is the only choice for the next walking step.
- ($S_{q\text{-}3}$) If there is a crack on the terrain with a normal-height ceiling, i.e., $e_{\text{sc-nc}}$, then the next keyframe state is $q_{\text{brachiation}}$ relying on a different set of apex velocities and step lengths than for walking behaviors:

$$\Box\big(\bigcirc e_{\text{sc-nc}} \Rightarrow \bigcirc(q_{\text{brachiation-s}} \vee q_{\text{brachiation-m}} \vee q_{\text{brachiation-l}})\big).$$

- ($S_{q\text{-}4}$) If there is a crack on the terrain and there is a high ceiling, i.e., $e_{\text{sc-hc}}$, then the keyframe state is q_{hop} relying on a specific apex velocity, regardless of the current q:

$$\Box\big(\bigcirc e_{\text{sc-hc}} \Rightarrow \bigcirc(q_{\text{hop-s}} \vee q_{\text{hop-m}} \vee q_{\text{hop-l}})\big).$$

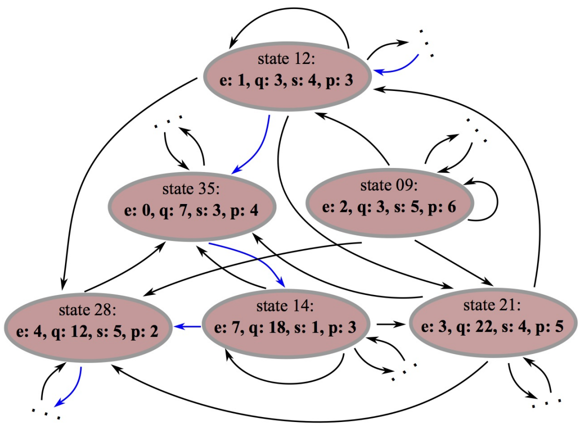


Fig. 7. A fragment of the synthesized automaton for the WBDL contact planner. Nondeterministic transitions are encoded in this automaton. The blue transitions represent a specific execution. For illustration, we index both the environmental action $\mathcal{E}$ in Eq. (13) and the system action $\mathcal{S}$ in Eq. (21) as $\{0,\ldots,7\}$ and $\{0,\ldots,8\}$ in order, respectively. The robot keyframe state $\mathcal{Q}$ is indexed as $\{0,\ldots,26\}$ in order. For instance, when the automaton state is at number 12, we encounter environmental action $e=1$. The winning strategy assigns keyframe state $q=3$, robot contact action $s=4$ and locomotion switching mode $p=3$. This state allows several nondeterministic transitions for the next walking step decision.

- ($S_{q\text{-}5}$) If a human appears in front of the robot, i.e., e_{ha}, then the next keyframe state is q_{stop} relying on a specific step length, regardless of the current q:

$$\square\big(\bigcirc e_{\text{ha}} \Rightarrow \bigcirc(q_{\text{stop-s}} \vee q_{\text{stop-m}} \vee q_{\text{stop-l}})\big).$$

- ($S_{q\text{-}6}$) If there is a narrow passage, i.e., e_{np}, then the next key frame state is q_{slide} relying on a specific apex velocity, regardless of the current q:

$$\square\big(\bigcirc e_{\text{np}} \Rightarrow \bigcirc(q_{\text{slide-s}} \vee q_{\text{slide-m}} \vee q_{\text{slide-l}})\big).$$

The remaining eight scenarios involving different environment and system action combinations are defined in a similar manner omitted here for brevity. The specifications in ($S_{q\text{-}1}$)-($S_{q\text{-}6}$) and all others belong to φ^q_{trans}.

From a high-level perspective, the goal of our task planner is to enable the robot to continuously maneuver through constrained environments by repeatedly selecting contact actions among $\mathcal{S}$. To be consistent with the environmental goal specification in Eq. (15), we enforce the following liveness specification for the keyframe states.

$$\varphi^q_{\text{goal}} = \bigwedge_{q\in\mathcal{Q}} (\square\diamond q) \tag{24}$$

All the task specifications have been proposed such that $\varphi = \big((\varphi_q \wedge \varphi_e) \Rightarrow \varphi_s\big)$ holds.

4.4 *Synthesis of a high-level reactive task planner*

Here we formulate the high-level locomotion planning problem as a game between the robot and its possibly adversarial environment. Given the task specifications defined above, a reactive control protocol is synthesized such that the controlled legged robot behaviors satisfy all the designed specifications whatever admissible uncontrollable environment behaviors are.

Definition 8 (Game played by the WBDL task planner). *A game for the whole-body dynamic locomotion task planner is a tuple*

$$\mathcal{G} := \langle \mathcal{V}, \mathcal{X}, \mathcal{Y}, \theta_i, \theta_o, \Psi_i, \Psi_o, \phi_{\text{win}} \rangle$$

with the following elements

•$\mathcal{X} := \mathcal{E}$ *is a set of input variables for player 1;*
•$\mathcal{Y} := \mathcal{Q} \times \mathcal{S} \times \mathcal{P}$ *is a set of output variables for player 2;*
•$\mathcal{V} = \mathcal{X} \times \mathcal{Y}$ *is a finite set of proposition state variables over finite domains in the game;*
•θ_i *and* θ_o *are atomic propositions characterizing initial states of the input and output variables, respectively;*
•$\Psi_i(\mathcal{V}, \mathcal{X}')$ *and* $\Psi_o(\mathcal{V}, \mathcal{X}', \mathcal{Y}')$ *are the transition relations for the input and output variables for next steps, respectively;*
•ϕ_{win} *is the winning condition given by an LTL formula.*

A winning strategy for the task planner represented by the pair $(\mathcal{TS}_{\text{prod}}, \varphi)$ is defined as a partial function $(\gamma_0\gamma_1 \cdots \gamma_{i-1}, e_i) \mapsto (q_i, s_i, p_i)$, where a keyframe state q_i, a contact action s_i, and a switching mode p_i are chosen according to the state sequence history and the current environmental action in order to satisfy the assume-guarantee form in Eq. (11). All the specifications are satisfied whatever admissible yet uncontrollable environmental actions are.

Proposition 1 (Existence of a winning WBDL strategy). *A winning WBDL strategy* $\mathcal{A}_{\text{WBDL}}$ *exists for the game* $\mathcal{G}$ *in Definition 8 if and only if* $(\mathcal{TS}_{\text{prod}}, \varphi)$ *is realizable.*

Fig. 7 shows an automaton fragment of the WBDL contact planner $\mathcal{A}_{\text{WBDL}}$. Self-transition exists in moderatelyUpward states (e.g., state 09) and moderatelyDownward states (e.g., states 12 and 14) while hugelyDownward states (e.g., state 35) do not have a self-transition according to proposition ($S_{es\text{-}1}$). There is no transition between states 09 and 14 due to infeasible keyframe state transition. States 21 and 28 in red nodes represent humanAppear and terrainCrack events, respectively.

Remark 2. *Non-deterministic transitions exist in the synthesized automaton as follows: (i) environmental actions are non-deterministic. (ii) given an environmental action, several non-deterministic keyframe states can be chosen. (iii) even when both an environmental action and a keyframe state are given, non-deterministic system contact actions exist for certain transitions. This non-deterministic transitions allow self-transitions. In this case, we can guarantee the robot to make progress (i.e., maneuvering forward) due to the properties of locomotion keyframe states.*

The keyframe specifications in this section purely reason about logic-level decisions and have no knowledge of underlying locomotion dynamics. However, the locomotion dynamics, especially those affected by external disturbances or model uncertainties, often result in the desired keyframe transitions being unrealizable. As such, we need to propose keyframe transitions with robustness margins and synthesize a reachability based controller to determine realizable keyframe transitions by the low-level locomotion dynamical system as proposed in the next section.

5 Robust Reachability Control of Hybrid Locomotion Systems

When we model robot dynamics and estimate physical environments, uncertainty is ubiquitous due to sensor noise, model inaccuracy, external disturbance, sudden environmental changes, contact surface geometry uncertainty, and so on. As a result, commands from the symbolic task planner are potentially not achievable by the low-level motion planner. Additionally, a mismatch between the high-level discrete and low-level continuous planners is usually caused by the abstraction techniques applied on the underlying continuous systems. To handle these difficulties, we define a robust finite transition system and compute its keyframe transitions via synthesizing reachability controllers for every single walking step. In order to use phase-space locomotion manifolds to define robustness margin sets, a phase-space mapping needs to be

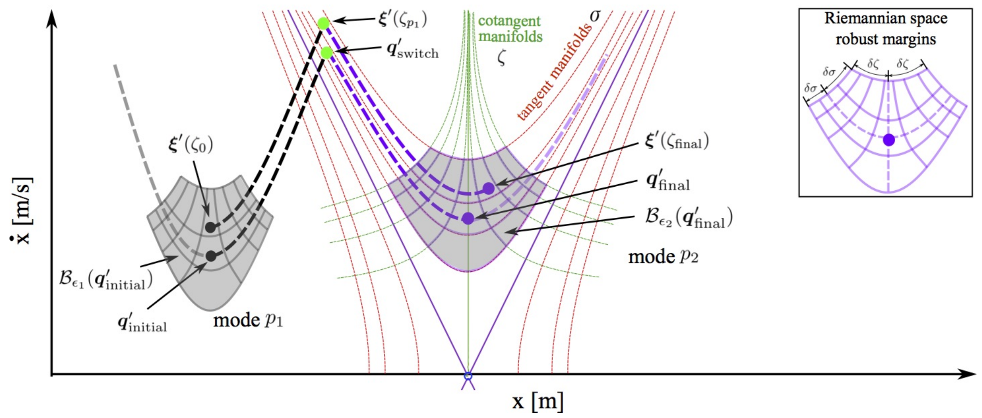

$\boldsymbol{\xi}'(\zeta_{p_1})$
$\boldsymbol{q}'_{\rm switch}$
cotangent manifolds
ζ
σ
tangent manifolds
Riemannian space robust margins
$\delta\sigma$
$\delta\zeta$
$\delta\zeta$
$\delta\sigma$
$\boldsymbol{\xi}'(\zeta_0)$
$\boldsymbol{\xi}'(\zeta_{\rm final})$
$\boldsymbol{q}'_{\rm final}$
$\mathcal{B}_{\epsilon_2}(\boldsymbol{q}'_{\rm final})$
mode p_2
$\dot{x}$ [m/s]
$\mathcal{B}_{\epsilon_1}(\boldsymbol{q}'_{\rm initial})$
$\boldsymbol{q}'_{\rm initial}$
mode p_1
x [m]

Fig. 8. Phase-space abstraction via locomotion-manifold-based partition. This figure shows a keyframe state transition process with robustness margins. Compared to the conventional grid-based partition in Euclidean phase space of Fig 5(a), this partition complies with locomotion dynamics, further enabling us to define robustness margins based on closed-form locomotion manifolds.

defined between the Euclidean and Riemmanian spaces to evaluate whether a phase-space state is in the robustness margin or not.

5.1 Phase-space Euclidean-to-Riemmanian mapping

We first consider a specific locomotion process, e.g., the prismatic inverted pendulum model (PIPM) (see Section 3.1 for more details) in order to establish a Euclidean-to-Riemmanian mapping in the phase space. Our previous study derives closed-form solutions of phase-space tangent and cotangent manifolds for this process [Zhao et al. (2017)] as follows.

Proposition 2 (PIPM phase-space tangent manifold). *Given the PIPM of Eq. (3) with initial conditions* $(x_0, \dot{x}_0) = (x_{\rm foot}, \dot{x}_{\rm apex})$ *and known foot placement* $x_{\rm foot}$*, the phase-space tangent manifold is characterized by the states* $(x, \dot{x}, x_{\rm foot}, \dot{x}_{\rm apex})$ *such that*

$$\sigma(x, \dot{x}, x_{\rm foot}, \dot{x}_{\rm apex}) = \frac{\dot{x}^2_{\rm apex}}{\omega^2_{\rm PIPM}}\big(\dot{x}^2 - \dot{x}^2_{\rm apex} - \omega^2_{\rm PIPM}(x - x_{\rm foot})^2\big), \tag{25}$$

where $\sigma = 0$ *represents the nominal phase-space manifold. When* $\sigma \neq 0$*, it represents the Riemannian distance to the nominal phase-space manifold.*

The tangent manifold can be used to measure deviations from the nominal locomotion trajectory in the phase-space. We use this manifold to quantify the width of a phase-space robustness margin.

Proposition 3 (PIPM phase-space cotangent manifold). *Let* ζ_0 *be a nonnegative scaling value representing the initial phase of a cotangent manifold. Given the PIPM of Eq. (3) and a specific initial state* $(x_0, \dot{x}_0)$ *different from the keyframe* $(x_{\rm foot}, \dot{x}_{\rm apex})$*, the cotangent manifold is characterized by the states* $(x, \dot{x}, x_0, \dot{x}_0)$ *such that*

$$\zeta(x, \dot{x}, x_0, \dot{x}_0) = \zeta_0\Big(\frac{\dot{x}}{\dot{x}_0}\Big)^{\omega^2_{\rm PIPM}}\frac{x - x_{\rm foot}}{x_0 - x_{\rm foot}}, \tag{26}$$

where ζ_0 *is chosen as the phase progression value at the keyframe state in this study.*

This cotangent manifold represents the arc length along the tangent manifold σ in Eq. (25). We use this cotangent manifold to quantify the length of a phase-space robustness margin. Detailed derivations of these two closed-form solutions above, i.e., $\sigma(x, \dot{x}, x_{\rm foot}, \dot{x}_{\rm apex}) = 0$ and $\zeta(x, \dot{x}, x_0, \dot{x}_0) = 0$, are provided in [Zhao et al. (2017)]. A similar analysis can be performed for other locomotion models as described in Section 3.1 (see the propositions in Appendix C for other locomotion modes). Given these analytical solutions, we define a mapping between the Euclidean and Riemmanian spaces as

$$\begin{pmatrix} \zeta \\ \sigma \end{pmatrix} = \mathcal{Z}_p(\boldsymbol{\xi}) = \begin{pmatrix} \mathcal{Z}_{p,\zeta}(x, \dot{x}) \\ \mathcal{Z}_{p,\sigma}(x, \dot{x}) \end{pmatrix} \tag{27}$$

where $\mathcal{Z}_p(\boldsymbol{\xi})$ is a nonlinear mapping of the CoM state $(x, \dot{x})$ to the Riemannian space states and obtained by using the phase space manifold of the p^{th} locomotion mode. This mapping will be used for the robust finite transition system definition in order to quantify the location of the phase-space state in the Riemmanian space.

5.2 *Robust finite transition system for one walking step*

We now focus on a case of the one-walking-step locomotion process as defined in Def. 3. As illustrated in Fig. 5(b), the discrete task planner uses a Riemannian discretization of the local state space, which is defined by an abstraction map $\mathcal{M}_{\text{Riem}} : \Xi \to \mathcal{Q}$ such that for all $(\boldsymbol{\xi}, \boldsymbol{q}) \in \Xi \times \mathcal{Q}$,

$$|\boldsymbol{\xi} - \boldsymbol{q}| \preceq \nu \implies \boldsymbol{q} = \mathcal{M}_{\text{Riem}}(\boldsymbol{\xi}), \tag{28}$$

where ν is the granularity of the discretization[4]. The operators $|\cdot|$ and $\preceq$ above represent vectorized absolute values and element-wise inequality, respectively.[5]

To guarantee that the motion planner yields feasible phase-space plans robust to disturbances, such as state measurement errors and disturbances in the dynamics, we introduce ϵ_1 and ϵ_2 as the bounds of initial and final robustness margins in the one-walking-step transition system. Namely, we not only consider the nominal initial and final keyframe states $\boldsymbol{q}_{\text{initial}}$ and $\boldsymbol{q}_{\text{final}}$ assigned by the task planner, but also neighbourhood keyframe cells overlapping the ϵ-neighbourhood of nominal keyframe states.

Definition 9 (**Robustness margin sets**). *The initial and final robustness margin sets around the nominal keyframe states $\boldsymbol{q}_{\text{initial}}, \boldsymbol{q}_{\text{final}} \in \mathcal{Q}$ are defined as*

$$\tilde{\mathcal{B}}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}}) := \big\{\boldsymbol{\xi}(\zeta_0) \mid |\mathcal{Z}_{p_1}(\boldsymbol{\xi}(\zeta_0)) - \mathcal{Z}_{p_1}(\boldsymbol{q}_{\text{initial}})| \preceq \epsilon_1\big\}.$$
$$\tilde{\mathcal{B}}_{\epsilon_2}(\boldsymbol{q}_{\text{final}}) := \big\{\boldsymbol{\xi}(\zeta_{\text{final}}) \mid |\mathcal{Z}_{p_2}(\boldsymbol{\xi}(\zeta_{\text{final}})) - \mathcal{Z}_{p_2}(\boldsymbol{q}_{\text{final}})| \preceq \epsilon_2\big\}.$$

where $\epsilon_1, \epsilon_2 \in \mathbb{R}^2$ represent the bounds of $\tilde{\mathcal{B}}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and $\tilde{\mathcal{B}}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$, respectively. p_1 and p_2 denote the locomotion modes before and after a contact switch, respectively.

The robustness margins ϵ_1 and ϵ_2 in Def. 9 are defined in the Riemannian space. A mapping $\mathcal{Z}$ is applied on the Euclidean states $\boldsymbol{\xi}$ and $\boldsymbol{q}$ to convert them to the Riemannian space. We design $\epsilon_1 \succ \nu$ and $\epsilon_2 \succ \nu$ such that the robustness margins are larger than the discretized cell. To provide different robust margins, we allow for non-uniform sets, i.e., non-identical values for (ϵ_1, ϵ_2). This non-uniform set design makes the size of the total number of allowable keyframe transitions more manageable.

Now we describe how to simplify the robustness margin sets based on the closed-form phase-space manifolds defined in Section 5.1.

[4] In this section, we use a bold symbol $\boldsymbol{q}$ to represent a keyframe state since it is a multi-dimensional state vector. In the task planner, the keyframe state is represented by a non-bold symbol q due to its pure discrete property.
[5] For two given n-vectors x and y, we have $|x| = (|x_1|, |x_2|, \dots, |x_n|)$ and $|x| \preceq |y| \Leftrightarrow |x_i| \leq |y_i|, \forall i \in \{1, \dots, n\}$.

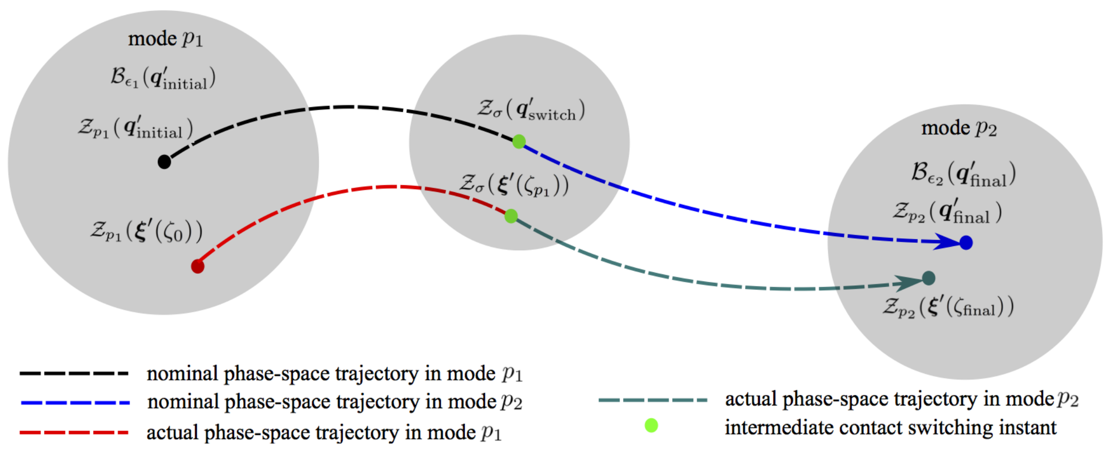

mode p_1
$\mathcal{B}_{\epsilon_1}(\boldsymbol{q}'_{\text{initial}})$
$\mathcal{Z}_{p_1}(\boldsymbol{q}'_{\text{initial}})$
$\mathcal{Z}_{p_1}(\boldsymbol{\xi}'(\zeta_0))$
$\mathcal{Z}_{\sigma}(\boldsymbol{q}'_{\text{switch}})$
$\mathcal{Z}_{\sigma}(\boldsymbol{\xi}'(\zeta_{p_1}))$
mode p_2
$\mathcal{B}_{\epsilon_2}(\boldsymbol{q}'_{\text{final}})$
$\mathcal{Z}_{p_2}(\boldsymbol{q}'_{\text{final}})$
$\mathcal{Z}_{p_2}(\boldsymbol{\xi}'(\zeta_{\text{final}}))$
nominal phase-space trajectory in mode p_1
nominal phase-space trajectory in mode p_2
actual phase-space trajectory in mode p_1
actual phase-space trajectory in mode p_2
intermediate contact switching instant

Fig. 9. Keyframe state reachability with robustness margins for one walking step. Due to measurement error or external disturbance, the initial state $\boldsymbol{\xi}'(\zeta_0)$ may deviate from the desired keyframe state $\boldsymbol{q}'_{\text{initial}}$. A robustness region $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}'_{\text{initial}})$ is defined to bound the allowable state deviations. The actual and desired states evolve according to their respective system dynamics in locomotion modes p_1 and p_2, respectively. The state switches from mode p_1 to mode p_2 at the phase instant ζ_{p_1}. The bound of distance between the nominal intermediate keyframe $\boldsymbol{q}'_{\text{switch}}$ and $\boldsymbol{\xi}'(\zeta_{p_1})$ is shown in Eq. (38). Finally, these two states reach $\boldsymbol{\xi}'(\zeta_{\text{final}})$ and $\boldsymbol{q}'_{\text{final}}$, respectively. To compute the keyframe transition, we require that $\mathcal{Z}_{p_2}(\boldsymbol{\xi}'(\zeta_{\text{final}}))$ should be ϵ_2-close to $\mathcal{Z}_{p_2}(\boldsymbol{q}'_{\text{final}})$, i.e., being in the margin $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}'_{\text{final}})$. More details of the definitions of robustness margin set are in Def. 10. Note that, since the robustness margin set $\mathcal{B}$ is defined in the Riemannian space, the mapping $\mathcal{Z}_p$ is applied on the states $\boldsymbol{q}$ and $\boldsymbol{\xi}'$ in the figure symbols for consistency.

Definition 10 (Phase-space robustness margin sets). *Given closed-form locomotion phase-space manifolds from Propositions 2 and 3, the initial and final robustness margin sets are simplified to*

$$\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}}) := \{\boldsymbol{\xi}(\zeta,\sigma) \mid \zeta \in [\zeta_0 - \delta\zeta_{\epsilon_1}, \zeta_0 + \delta\zeta_{\epsilon_1}], \sigma \in [-\delta\sigma_{\epsilon_1}, \delta\sigma_{\epsilon_1}]\}, \tag{29}$$

$$\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}}) := \{\boldsymbol{\xi}(\zeta,\sigma) \mid \zeta \in [\zeta_{\text{final}} - \delta\zeta_{\epsilon_2}, \zeta_{\text{final}} + \delta\zeta_{\epsilon_2}], \sigma \in [-\delta\sigma_{\epsilon_2}, \delta\sigma_{\epsilon_2}]\}, \tag{30}$$

where $\boldsymbol{\xi} = (x, \dot{x})$, the initial state $\boldsymbol{q}_{\text{initial}} = \boldsymbol{\xi}(\zeta_0, 0)$ and the final state $\boldsymbol{q}_{\text{final}} = \boldsymbol{\xi}(\zeta_{\text{final}}, 0)$, $\epsilon_1 = [\delta\zeta_{\epsilon_1}, \delta\sigma_{\epsilon_1}]$ and $\epsilon_2 = [\delta\zeta_{\epsilon_2}, \delta\sigma_{\epsilon_2}]$ quantify the uncertainty bounds of $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$, respectively.

These two pairs of bounds represent Riemannian distances in phase-space, as shown in the upper right miniature subfigure in Fig. 8. A locomotion-manifold-based partition is illustrated in Fig. 5(b). The proposed robust finite transition system will use this partition to design the robustness margin around keyframe states as shown in Fig. 8. A merit of our analysis is that this partition is consistent with the vector field of the locomotion dynamics. Additionally, this partition simplifies mathematical descriptions of robustness margin sets.

Definition 11 (Robust finite transition system for one walking step). *Given two triples composed of nominal keyframe states, locomotion modes, and system contact actions $(\boldsymbol{q}_{\text{initial}}, p_1, s_1)$ and $(\boldsymbol{q}_{\text{final}}, p_2, s_2)$*[6]*, a finite transition subsystem $\mathcal{TS}_{\text{OWS}}$ with robustness margins ϵ_1 and ϵ_2 for one walking step (OWS) is defined as a tuple*

$$\mathcal{TS}_{\text{OWS}} := (\mathcal{Q}_{\text{OWS}}, \mathcal{I}_{\text{OWS}}, \mathcal{P}_{\text{OWS}}, \mathcal{S}_{\text{OWS}}, \mathcal{A}_{\text{OWS}}, \mathcal{T}_{\text{OWS}}) \tag{31}$$

with the following elements

•*$\mathcal{Q}_{\text{OWS}}$ is a set of keyframe states determined by the nominal keyframe pair $(\boldsymbol{q}_{\text{initial}}, \boldsymbol{q}_{\text{final}})$ and robustness margins (ϵ_1, ϵ_2). $\mathcal{Q}_{\text{OWS}} \subseteq \mathcal{Q}$, where $\mathcal{Q}$ is the set of all the allowable keyframe states defined in Eq. (23). $\mathcal{Q}_{\text{OWS}} = \mathcal{Q}_{p_1,\text{OWS}} \cup \mathcal{Q}_{p_2,\text{OWS}}$,*

[6] In the robust finite transition system layer, we use a bold symbol $\boldsymbol{q}$ to represent a keyframe state since it is a multi-dimensional state vector in phase-space. In the task planner, the keyframe state is represented by a non-bold q due to the discrete domain reasoning.

where $\mathcal{Q}_{p_1,\mathrm{OWS}}$ and $\mathcal{Q}_{p_2,\mathrm{OWS}}$ are defined as

$$\mathcal{Q}_{p_1,\mathrm{OWS}} = \{\boldsymbol{q}'_{\mathrm{initial}} \in \mathcal{Q} \mid \mathcal{M}^{-1}_{\mathrm{Riem}}(\boldsymbol{q}'_{\mathrm{initial}}) \cap \mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\mathrm{initial}}) \neq \emptyset\}, \tag{32}$$

$$\mathcal{Q}_{p_2,\mathrm{OWS}} = \{\boldsymbol{q}'_{\mathrm{final}} \in \mathcal{Q} \mid \mathcal{M}^{-1}_{\mathrm{Riem}}(\boldsymbol{q}'_{\mathrm{final}}) \cap \mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\mathrm{final}}) \neq \emptyset\}. \tag{33}$$

•$\mathcal{I}_{\mathrm{OWS}} = \mathcal{Q}_{p_1,\mathrm{OWS}}$ *is a set of initial states.*
•$\mathcal{P}_{\mathrm{OWS}} = \{p_1, p_2\}$ *is a pair of locomotion modes for one walking step.*
•$\mathcal{S}_{\mathrm{OWS}} = \{s_1, s_2\}$ *is a pair of contact actions for one walking step.*
•$\mathcal{A}_{\mathrm{OWS}} = \{\mathcal{A}_{p_1}, \mathcal{A}_{p_2}\}$ *with* $\mathcal{A}_{p_1} = \bigcup_{\zeta_0 \leq \zeta \leq \zeta_{p_1}} \mathcal{U}_{p_1}^{[\zeta_0,\zeta]}$ *and* $\mathcal{A}_{p_2} = \bigcup_{\zeta_{p_1} \leq \zeta \leq \zeta_{\mathrm{final}}} \mathcal{U}_{p_2}^{[\zeta_{p_1},\zeta]}$*,*[7] *where* ζ_{p_1} *represents the phase instant when the contact switch occurs.*
•$((\boldsymbol{q}'_{\mathrm{initial}}, p_1, s_1), \boldsymbol{a}, (\boldsymbol{q}'_{\mathrm{final}}, p_2, s_2)) \in \mathcal{T}_{\mathrm{OWS}}$ *(i.e.,* $(\boldsymbol{q}'_{\mathrm{initial}}, p_1, s_1) \xrightarrow{\boldsymbol{a}} (\boldsymbol{q}'_{\mathrm{final}}, p_2, s_2)$*) for* $\boldsymbol{q}'_{\mathrm{initial}} \in \mathcal{Q}_{p_1,\mathrm{OWS}}, \boldsymbol{q}'_{\mathrm{final}} \in \mathcal{Q}_{p_2,\mathrm{OWS}}$ *if there exists a control sequence* $\boldsymbol{a} = \{\boldsymbol{u}_{p_1}(\zeta_0), \ldots, \boldsymbol{u}_{p_1}(\zeta_{p_1}), \ldots, \boldsymbol{u}_{p_2}(\zeta_{p_2}), \ldots, \boldsymbol{u}_{p_2}(\zeta_{\mathrm{final}})\} \in \mathcal{A}_{p_1} \cup \mathcal{A}_{p_2}$ *for all bounded external disturbances* $d : [\zeta_0, \zeta_{\mathrm{final}}] \to D_{\mathrm{OWS}} \subseteq \mathbb{R}^d$ *such that the resulting solution* $\boldsymbol{\xi}' : [\zeta_0, \zeta_{\mathrm{final}}] \to \mathbb{R}^n$ *follows,*

$$\dot{\boldsymbol{\xi}}'(\zeta) = f_{p_1}\big(\boldsymbol{\xi}'(\zeta), \boldsymbol{u}_{p_1}(\zeta), d(\zeta)\big), \forall \zeta \in [\zeta_0, \zeta_{p_1}], \tag{34}$$

$$\dot{\boldsymbol{\xi}}'(\zeta) = f_{p_2}\big(\boldsymbol{\xi}'(\zeta), \boldsymbol{u}_{p_2}(\zeta), d(\zeta)\big), \forall \zeta \in [\zeta_{p_1}, \zeta_{\mathrm{final}}], \tag{35}$$

satisfying

$$|\mathcal{Z}_{p_1}(\boldsymbol{\xi}(\zeta_0)) - \mathcal{Z}_{p_1}(\boldsymbol{q}'_{\mathrm{initial}})| \preceq \epsilon_1, \boldsymbol{q}'_{\mathrm{initial}} \in \mathcal{M}_{\mathrm{Riem}}(\boldsymbol{\xi}(\zeta_0)), \tag{36}$$

$$|\mathcal{Z}_{p_2}(\boldsymbol{\xi}(\zeta_{\mathrm{final}})) - \mathcal{Z}_{p_2}(\boldsymbol{q}'_{\mathrm{final}})| \preceq \epsilon_2, \boldsymbol{q}'_{\mathrm{final}} \in \mathcal{M}_{\mathrm{Riem}}(\boldsymbol{\xi}(\zeta_{\mathrm{final}})), \tag{37}$$

$$|\mathcal{Z}_{p_1,\sigma}(\boldsymbol{\xi}'(\zeta_{p_1})) - \mathcal{Z}_{p_1,\sigma}(\boldsymbol{q}'_{\mathrm{switch}})| \leq \delta\sigma_{\epsilon_1}, |\mathcal{Z}_{p_2,\sigma}(\boldsymbol{\xi}'(\zeta_{p_1})) - \mathcal{Z}_{p_2,\sigma}(\boldsymbol{q}'_{\mathrm{switch}})| \leq \delta\sigma_{\epsilon_2}, \tag{38}$$

where $\mathcal{Z}(\cdot)$ is the two-dimensional Euclidean-to-Riemannian mapping introduced in Section 5.1. The system vector fields f_{p_1} and f_{p_2} are jointly determined by the locomotion mode set $\mathcal{P}_{\mathrm{OWS}}$ and the contact action set $\mathcal{S}_{\mathrm{OWS}}$, respectively.

The mapping function $\mathcal{Z}$ has two dimensions in the phase-space: tangent σ and cotangent ζ manifolds as defined in Eq. (27). Initial and final bound conditions are represented by Eqs. (36) and (37), respectively. Eq. (38) essentially defines an intermediate set where the mode switch takes place, and determines the bound for the switching instant ζ_{p_1}. The inequalities in Eqs. (36) and (37) are element-wise.

A conceptual illustration of this transition computation is shown in Fig. 9. Using the robustness margins, we construct the transition set $\mathcal{T}_{\mathrm{OWS}}$ of the robust finite transition subsystem $\mathcal{TS}_{\mathrm{OWS}}$, as defined in Def. 11, by adding all the feasible transitions $\{(\boldsymbol{q}'_{\mathrm{initial}}, p_1, s_1) \xrightarrow{\boldsymbol{a}} (\boldsymbol{q}'_{\mathrm{final}}, p_2, s_2)\}$ where $\mathcal{Z}_{p_2}(\boldsymbol{\xi}'(\zeta_{\mathrm{final}}))$ is within the ϵ_2-distance to the targeted state $\mathcal{Z}_{p_2}(\boldsymbol{q}'_{\mathrm{final}})$.

The construction of $\mathcal{Q}_{\mathrm{OWS}}$ is shown in Fig. 10. The keyframe transitions in the robust finite transition subsystem $\mathcal{TS}_{\mathrm{OWS}}$ can be computed as follows: for all $\boldsymbol{q}'_{\mathrm{initial}} \in \mathcal{Q}_{p_1,\mathrm{OWS}}$ and $\boldsymbol{a} \in \mathcal{A}_{\mathrm{OWS}}$, if there exists $\boldsymbol{q}'_{\mathrm{final}} \in \mathcal{Q}_{p_2,\mathrm{OWS}} \cap \mathcal{B}_{\epsilon_2}(\boldsymbol{\xi})$, then we add $\{\boldsymbol{q}'_{\mathrm{initial}} \xrightarrow{\boldsymbol{a}} \boldsymbol{q}'_{\mathrm{final}}\}$ to the transition set $\mathcal{T}_{\mathrm{OWS}}$.

Algorithm 1 details the construction of the robust finite transition subsystem above. The high-level task planner specifies the inputs of the algorithm, i.e., two pairs of keyframe states, locomotion modes, and contact actions $(\boldsymbol{q}_{\mathrm{initial}}, p_1, s_1)$ and $(\boldsymbol{q}_{\mathrm{final}}, p_2, s_2)$ with robustness margins ϵ_1 and ϵ_2, respectively, and by Def. 11, determines the set of finite states $\mathcal{Q}_{\mathrm{OWS}}$.

[7] It is worth noting that $\mathcal{A}_{p_1}$ and $\mathcal{A}_{p_2}$ represent a sequence of discrete control inputs, respectively, since this abstraction is designed for two consecutive walking steps involving a sequence of inter-sampling time steps.

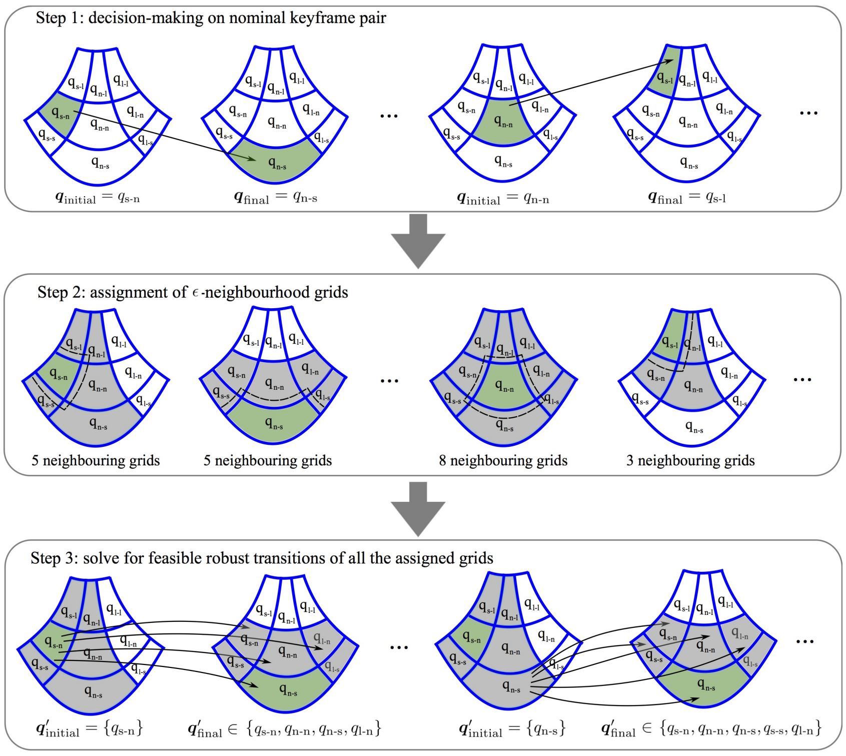

Step 1: decision-making on nominal keyframe pair
$q_{\text{initial}} = q_{\text{s-n}}$
$q_{\text{final}} = q_{\text{n-s}}$
$q_{\text{initial}} = q_{\text{n-n}}$
$q_{\text{final}} = q_{\text{s-l}}$
Step 2: assignment of ϵ-neighbourhood grids
5 neighbouring grids
5 neighbouring grids
8 neighbouring grids
3 neighbouring grids
Step 3: solve for feasible robust transitions of all the assigned grids
$q'_{\text{initial}} = \{q_{\text{s-n}}\}$
$q'_{\text{final}} \in \{q_{\text{s-n}}, q_{\text{n-n}}, q_{\text{n-s}}, q_{\text{l-n}}\}$
$q'_{\text{initial}} = \{q_{\text{n-s}}\}$
$q'_{\text{final}} \in \{q_{\text{s-n}}, q_{\text{n-n}}, q_{\text{n-s}}, q_{\text{s-s}}, q_{\text{l-n}}\}$

Fig. 10. Sequential procedure of designing feasible robust keyframe transitions. Step 1 determines the nominal keyframe state pair from the symbolic task planner. In Step 2, we design discrete state set $\mathcal{Q}_{\text{OWS}}$ in the robust finite transition system. Four cases are shown for illustration. The green cell represents the nominal keyframe state determined from the task planner while its surrounding gray cells represent other allowable discrete states in $\mathcal{Q}_{\text{OWS}}$. Finally, all feasible robust transitions are determined in Step 3.

This is the top-down component of our approach. The bottom-up component is the reachability control synthesis introduced in the next subsection. Algorithm 1 integrates the top-down and bottom-up components.

The proposed robust finite transition system (RFTS) differs from the abstraction approaches in [Liu and Ozay (2014, 2016); Tabuada (2009); Belta et al. (2017)] with respect to the following points: (i) The most salient difference is that our planning approach is a hierarchy consisting of both top-down and bottom-up components. The RFTS is an interface taking the desired command from the high-level symbolic task planner (i.e., the top-down component) and use this command to synthesize a reachability controller in the low-level motion planner (i.e., the bottom-up component). The approach in [Liu and Ozay (2014, 2016)] is an abstraction of the underlying continuous dynamical system and represents a bottom-up approach. (ii) By using the proposed hierarhical structure, we are able to solve a more challenging problem with whole-body dynamic locomotion in a constrained environment, instead of simple examples such as 2D mobile robot or vehicle. (iii) Our RFTS reasons about the robustness to bounded state disturbances at not only the inter-sampling level, but also the locomotion keyframe level capturing the essential locomotion dynamics. (iv) We incorporate hybrid dynamics into our RFTS design, which is constructed for the one walking step. Overall, our planning framework sequentially composes multiple locomotion modes. (v) Instead of a grid-based partition, we use a locomotion-manifold-based partition to characterize the robustness margins of the keyframe states in the phase-space.

5.3 *One-walking-step reachability control synthesis*

To determine the transitions satisfying the conditions in Eqs. (36)-(38) of Def. 11, we employ abstraction-based control synthesis developed for general dynamical systems. The idea of this approach is to automatically and rigorously compute the set of states that can be controlled to realize a given specification and generate feedback controllers for those states.

Algorithm 1 One-walking-step robust finite transition subsystem $\mathcal{TS}_{\rm OWS}$ with robustness margins (ϵ_1, ϵ_2)

```
1: Input: $\epsilon_1, \epsilon_2, \boldsymbol{q}_{\rm initial}, \boldsymbol{q}_{\rm final}, p_1, p_2, s_1, s_2$
2: Define $\mathcal{A}_{p_1}, \mathcal{A}_{p_2}$ as finite subsets of $\bigcup_{\zeta_0 \leq \zeta \leq \zeta_{p_1}} \mathcal{U}_{p_1}^{[\zeta_0,\zeta]}$ and $\bigcup_{\zeta_{p_1} \leq \zeta \leq \zeta_{\rm final}} \mathcal{U}_{p_2}^{[\zeta_{p_1},\zeta]}$.
3: Define the discrete state of keyframe $\mathcal{Q}_{\rm OWS}$ according to Eqs. (32) and (33) of Def. 11 and Fig. 10.
4: Initialize transition set $\mathcal{T}_{\rm OWS} \leftarrow (\mathcal{Q}_{p_1,{\rm OWS}} \times \{p_1\}) \times \mathcal{A}_{\rm OWS} \times (\mathcal{Q}_{p_2,{\rm OWS}} \times \{p_2\})$ {initialize all possible transitions}
5: for $\boldsymbol{q}'_{\rm initial} \in \mathcal{Q}_{p_1,{\rm OWS}}$ do
6:   for $\boldsymbol{q}'_{\rm final} \in \mathcal{Q}_{p_2,{\rm OWS}}$ do
7:     Construct an inter-sampling finite abstraction $\mathcal{TS}_{\rm OWS,INT}$
8:     isReachable ← ReachabilityControl($\mathcal{B}_{\epsilon_1}(\boldsymbol{q}'_{\rm initial}), \mathcal{B}_{\epsilon_2}(\boldsymbol{q}'_{\rm final}), \mathcal{TS}_{\rm OWS,INT}$)
9:     if isReachable == false then
10:      $\mathcal{T}_{\rm OWS} = \mathcal{T}_{\rm OWS} \backslash \{(\boldsymbol{q}'_{\rm initial}, p_1) \xrightarrow{\boldsymbol{a}} (\boldsymbol{q}'_{\rm final}, p_2)\}$     {delete unqualified transitions}
11:    end if
12:  end for
13: end for
14: return $\mathcal{TS}_{\rm OWS} = (\mathcal{Q}_{\rm OWS}, \mathcal{I}_{\rm OWS}, \mathcal{P}_{\rm OWS}, \mathcal{A}_{\rm OWS}, \mathcal{T}_{\rm OWS})$
```

Generally, abstraction-based control synthesis consists of three steps: 1) Construct a finite transition system (also called a finite abstraction) that over-approximates the dynamics of the original continuous system. 2) Design control algorithms based on the finite transition system with respect to the given specification. This step not only verifies whether the given specification is realizable by the low level robot dynamics, but also synthesizes a controller for the abstraction if realizable. 3) Interpolate the synthesized controller to be executed in the original continuous system.

We consider a one-walking-step locomotion subsystem defined on a local state space determined by two keyframe states.

Definition 12 (One-walking-step locomotion subsystem). *Given the switched system tuple in Eq. (6), a one-walking-step (OWS) locomotion subsystem from a given keyframe state $\boldsymbol{q}'_{\rm initial}$ with a robustness margin ϵ_1 in the $(p_1^{\rm th}, s_1^{\rm th})$ mode to another keyframe state $\boldsymbol{q}'_{\rm final}$ with a robustness margin ϵ_2 in the $(p_2^{\rm th}, s_2^{\rm th})$ mode is formulated as:*

$$\mathcal{SS}_{\rm OWS} = (\Xi_{\rm OWS}, \Xi_{\rm OWS,0}, \mathcal{U}_{\rm OWS}, \mathcal{P}_{\rm OWS}, f_{\rm OWS}) \tag{39}$$

where the state space of the subsystem $\Xi_{\rm OWS} \subseteq \Xi$ is a local area determined by the two keyframe states $\boldsymbol{q}'_{\rm initial}$ and $\boldsymbol{q}'_{\rm final}$; $\Xi_{\rm OWS,0} = \mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\rm initial})$ is the set of initial continuous states, and $\Xi_{\rm OWS} = \Xi_{p_1,{\rm OWS}} \cup \Xi_{p_2,{\rm OWS}}$ with $\Xi_{p_1,{\rm OWS}}$ and $\Xi_{p_2,{\rm OWS}}$ representing the local state space of the locomotion modes p_1 and p_2, respectively; $\mathcal{U}_{\rm OWS}$ is the allowable control input set for one walking step; $\mathcal{P}_{\rm OWS} = \{p_1, p_2\}$ represents a locomotion mode set composed of two consecutive walking steps; $f_{\rm OWS} = \{f_{p_1}, f_{p_2}\}$ is the set of vector fields determined by $(p_1^{\rm th}, s_1^{\rm th})$ and $(p_2^{\rm th}, s_2^{\rm th})$ command pairs, respectively. The mode transition instant ζ_{p_1} is determined by Eq. (38).

Remark 3. *The state spaces $\Xi_{p_1,{\rm OWS}}$ and $\Xi_{p_2,{\rm OWS}}$ overlap so that a contact switch can happen during one walking step. This overlap should fully cover the intersection of two robust tubes defined in Eqs. (38). A straightforward option is to make both $\Xi_{p_1,{\rm OWS}}$ and $\Xi_{p_2,{\rm OWS}}$ identical to the state space fully covering one walking step.*

The control actions $\boldsymbol{a}$ defined in Def. 11 are a sequence of control signals for one walking step. We discrete the control space and maintain a constant control signal for each time step. In the following, we propose a finite abstraction of the one-walking-step locomotion subsystem $\mathcal{SS}_{\rm OWS}$. This abstraction is based on a predefined time step $\delta\zeta$ for the construction of control signals, a bounded disturbance d, and a finer Euclidean space discretization (i.e., an abstraction map $\mathcal{M}_{\rm OWS,Euc}$) rather than the one used in the task planner.

$$\begin{tikzcd}
\mathcal{TS}_{\text{prod}} \arrow[r] & \mathcal{TS}_{\text{OWS}} \arrow[r] & \mathcal{TS}_{\text{OWS,INT}} \arrow[r] & \mathcal{SS}_{\text{OWS}}
\end{tikzcd}$$

Fig. 11. Top-down hierarchy of layered abstractions: $\mathcal{TS}_{\text{prod}}$ represents the finite product transition system of the high-level task planner in Section 3; $\mathcal{TS}_{\text{OWS}}$ denotes the robust finite transition system for one walking step in Def. 11; $\mathcal{TS}_{\text{OWS,INT}}$ denotes the inter-sampling finite abstraction of one walking step in Def. 13; $\mathcal{SS}_{\text{OWS}}$ represents the continuous one-walking-step locomotion subsystem in Def. 12.

Definition 13 (Inter-sampling finite abstraction of one walking step). *Given a one-walking-step locomotion subsystem $\mathcal{SS}_{\text{OWS}}$, an abstraction map $\mathcal{M}_{\text{OWS,Euc}} : \Xi_{\text{OWS}} \to \mathcal{Q}_{\text{OWS,INT}}$, and a time step $\delta\zeta$, a finite transition system*

$$\mathcal{TS}_{\text{OWS,INT}} = (\mathcal{Q}_{\text{OWS,INT}}, \mathcal{Q}_{\text{OWS,INT},0}, \mathcal{A}_{\text{OWS,INT}}, \mathcal{T}_{\text{OWS,INT}})$$

is defined as an inter-sampling finite abstraction of $\mathcal{SS}_{\text{OWS}}$, denoted by $\mathcal{SS}_{\text{OWS}} \preceq_{(\delta\zeta,d,\mathcal{M}_{\text{OWS,Euc}})} \mathcal{TS}_{\text{OWS,INT}}$ if the following conditions hold

•$\mathcal{Q}_{\text{OWS,INT}} = \mathcal{Q}_{p_1,\text{OWS,INT}} \cup \mathcal{Q}_{p_2,\text{OWS,INT}} = \bigcup_{\boldsymbol{\xi}\in\Xi_{p_1,\text{OWS}}} \mathcal{M}_{\text{OWS,Euc}}(\boldsymbol{\xi}) \cup \bigcup_{\boldsymbol{\xi}\in\Xi_{p_2,\text{OWS}}} \mathcal{M}_{\text{OWS,Euc}}(\boldsymbol{\xi})$ *is a finite set of discrete states; an initial set of discrete states is defined as* $\mathcal{Q}_{\text{OWS,INT},0} = \bigcup_{\boldsymbol{\xi}\in\Xi_{\text{OWS},0}} \mathcal{M}_{\text{OWS,Euc}}(\boldsymbol{\xi})$.
•$\mathcal{A}_{\text{OWS,INT}} = \{\mathcal{U}_{p_1}, \mathcal{U}_{p_2}\}$ *is the set of control values, where $\mathcal{U}_{p_1}$ and $\mathcal{U}_{p_2}$ are the allowable control input ranges in the p_1^{th} and p_2^{th} locomotion modes, respectively.*
•$(\boldsymbol{q}, \boldsymbol{a}_{\text{INT}}, \boldsymbol{q}') \in \mathcal{T}_{\text{OWS,INT}}$ *for $\boldsymbol{q}, \boldsymbol{q}' \in \mathcal{Q}_{p_1,\text{OWS,INT}}$ (or $\boldsymbol{q}, \boldsymbol{q}' \in \mathcal{Q}_{p_2,\text{OWS,INT}}$, respectively), if there exists $\boldsymbol{a}_{\text{INT}} \in \mathcal{A}_{\text{OWS,INT}}$ being constant with one time-step duration $\delta\zeta$ and some external disturbance $d : [0, \delta\zeta] \to D_{\text{OWS}}$ such that the resulting solution $\boldsymbol{\xi} : [0, \delta\zeta] \to \Xi_{p_1,\text{OWS,INT}}$ (or $\boldsymbol{\xi} : [0, \delta\zeta] \to \Xi_{p_2,\text{OWS,INT}}$, respectively) satisfies $\boldsymbol{\xi}(0) = \boldsymbol{\xi}_0$, $\boldsymbol{\xi}_0 \in \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q})$, $\boldsymbol{\xi}(\delta\zeta) \in \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q}')$ and the system dynamics in Eq. (34) (or Eq. (35), respectively).*

The abstraction map $\mathcal{M}_{\text{OWS,Euc}} : \Xi_{\text{OWS,INT}} \to \mathcal{Q}_{\text{OWS,INT}}$ maps a continuous state in $\Xi_{\text{OWS,INT}}$ into a discrete state in the set $\mathcal{Q}_{\text{OWS,INT}}$. Equivalently, $\Xi_{\text{OWS,INT}} := \bigcup_{\boldsymbol{q}\in\mathcal{Q}_{\text{OWS,INT}}} \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q})$. A typical implementation of such a map $\mathcal{M}_{\text{OWS,Euc}}$ is a uniform partition with a specific granularity. The condition in the last item of Def. 13 indicates that $\mathcal{TS}_{\text{OWS,INT}}$ is an over-approximation of $\mathcal{SS}_{\text{OWS}}$. That is, all the transitions will be included as long as a transition is possible by using the locomotion dynamics under bounded disturbances. For instance, let us examine two consecutive inter-sampling discrete states $\boldsymbol{q}$ and $\boldsymbol{q}'$. We add a transition $(\boldsymbol{q}, \boldsymbol{a}_{\text{INT}}, \boldsymbol{q}')$ if

$$\mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q}') \cap \mathcal{R}_{\delta\zeta}(\mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q}), \boldsymbol{a}_{\text{INT}}) \neq \emptyset, \tag{40}$$

where $\mathcal{R}_{\delta\zeta}(\cdot,\cdot)$ is defined as $\mathcal{R}_{\delta\zeta}(\Xi_0, \boldsymbol{u}) := \{\boldsymbol{\xi}(\delta\zeta) \mid \dot{\boldsymbol{\xi}}(\tau_r) = f_{p_i}\big(\boldsymbol{\xi}(\tau_r), \boldsymbol{u}(\tau_r), d(\tau_r)\big), \tau_r \in [0, \delta\zeta], \boldsymbol{\xi}(0) \in \Xi_0, i = 1, 2\}$, representing the reachable set of $\Xi_0 \subseteq \Xi_{\text{OWS,INT}}$ after a time step $\delta\zeta$ under the constant control input $\boldsymbol{u}$. For nonlinear dynamics, it is difficult to compute the exact reachable set $\mathcal{R}_{\delta\zeta}(\Xi_0, \boldsymbol{u})$. To circumvent this hurdle, we compute an over-approximation of the exact reachable set $\mathcal{R}_{\delta\zeta}(\Xi_0, \boldsymbol{u})$, denoted as $\widehat{\mathcal{R}}_{\delta\zeta}(\Xi_0, \boldsymbol{u})$. This over-approximation is obtained via employing interval-valued functions (refer to [Jaulin (2001)] for the details) of the discretized low-level dynamics. As a counterpart of real-valued functions, such an interval-valued function is evaluated over intervals and obeys interval arithmetic. As such, all the reachable states after a time step $\delta\zeta$ from any state in Ξ_0 are captured in the output of an interval-valued function. By refining the set Ξ_0 into smaller intervals, we can approximate the reachable set $\mathcal{R}_{\delta\zeta}(\Xi_0, \boldsymbol{u})$ with an arbitrary precision [Liu (2017)].

Next, we will discuss in detail how to compute the over-approximation $\widehat{\mathcal{R}}_{\delta\zeta}(\cdot,\cdot)$.

Algorithm 2 Reachability control synthesis

```
1: procedure ReachabilityControl(initial set $I$, target set $G$, inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$)
2: assign a queue Que $\leftarrow G$, $G \subseteq \mathcal{Q}$                {define a FIFO queue}
3: initialize a winning set $\mathcal{WIN} \leftarrow G$
4: $\mathcal{K} \leftarrow \mathbf{0}^{N\times M}$                {$N = |\mathcal{Q}_{\text{OWS,INT}}|_{\text{numel}}$ and $M = |\mathcal{A}_{\text{OWS,INT}}|_{\text{numel}}$}
5: while Que $\neq \emptyset$ do
6:   $\boldsymbol{q}' \leftarrow$ Que.pop()
7:   for all $\boldsymbol{q} \in \mathcal{Q}_{\text{OWS,INT}}$ and $a \in \mathcal{A}_{\text{OWS,INT}}$ such that $\boldsymbol{q} \xrightarrow{a} \boldsymbol{q}'$ do
8:     if $\boldsymbol{q}'' \in \mathcal{WIN}$ for all $\boldsymbol{q}''$ such that $\boldsymbol{q} \xrightarrow{a} \boldsymbol{q}''$ then
9:       if $\boldsymbol{q} \notin \mathcal{WIN}$ then
10:        Que $\leftarrow \boldsymbol{q}$
11:        $\mathcal{WIN} \leftarrow \boldsymbol{q}$
12:        $\mathcal{K}(\boldsymbol{q}, a) \leftarrow 1$
13:      end if
14:    end if
15:  end for
16: end while
17: if $\mathcal{WIN} \cap I \neq \emptyset$ then
18:   isReachable $\leftarrow$ true
19: else
20:   isReachable $\leftarrow$ false
21: end if
22: return isReachable, $\mathcal{WIN}$, $\mathcal{K}$
```

Assumption 1 (Disturbance additivity and boundedness). *We assume that the right-hand side of the disturbed switched system in Eq. (5) can be divided into a nominal part and a disturbance part:*

$$f_p\big(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta), d(\zeta)\big) = f_p\big(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta)\big) + g_p\big(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta)\big),\ p \in \mathcal{P}, \zeta \geq 0, \tag{41}$$

and the disturbance part is element-wise upper bounded by

$$\big|g_p\big(\boldsymbol{\xi}, \boldsymbol{u}\big)\big| \preceq r,\ \forall p \in \mathcal{P},\ \boldsymbol{\xi} \in \Xi,\ \boldsymbol{u} \in \mathcal{U}, \tag{42}$$

with the bound vector $r \in \mathbb{R}^n$.

Given a locomotion mode, we denote by $\delta\boldsymbol{\xi}(\zeta)$ the difference of two trajectories $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ at the same instant ζ. These two trajectories start from their initial states $\boldsymbol{\xi}_{1,0}$ and $\boldsymbol{\xi}_{2,0}$, respectively. With the Lipschitz condition and Assumption 1, we have

$$|\delta\dot{\boldsymbol{\xi}}(\zeta)| \preceq L|\delta\boldsymbol{\xi}(\zeta)| + r \quad \Longrightarrow \quad |\delta\boldsymbol{\xi}(\zeta)| \preceq |\boldsymbol{\xi}_{1,0} - \boldsymbol{\xi}_{2,0}|e^{L\zeta} + L^{-1}(e^{L\zeta} - \boldsymbol{I}_{n\times n})r,\ \forall\zeta \in [0, \delta\zeta],$$

which implies that under a disturbance bounded by the vector r, all the possible states after a time step $\delta\zeta$ stay within a ball centered at the nominal trajectory state with a radius vector $r_{\delta\zeta} = |\boldsymbol{\xi}_{1,0} - \boldsymbol{\xi}_{2,0}|e^{L\zeta} + L^{-1}(e^{L\delta\zeta} - \boldsymbol{I}_{n\times n})r$. Hence, the reachable set $\mathcal{R}_{\delta\zeta}(\mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q}), \boldsymbol{a}_{\text{INT}})$ in Eq. (40) can be over-approximated by the estimated reachable set of the nominal system $\dot{\boldsymbol{\xi}}(\zeta) = f_{p_i}\big(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta)\big)$ enlarged by $r_{\delta\zeta}$.

Given the abstraction defined in Def. 13, we synthesize a reachability controller for the inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$ of a one-walking-step subsystem $\mathcal{SS}_{\text{OWS}}$ as shown in Algorithm 2. This algorithm takes as inputs an initial set I, a target set G, and a finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$. Backward dynamics propagation is used to determine the realizability of the reachability controller. This algorithm returns a boolean value isReachable indicating the realizability of the target set G. If this target set is realizable, it outputs two additional sets: (i) a winning set $\mathcal{WIN}$ defined as all the states from which the reachability goal is satisfied under bounded state disturbances; and (ii) a boolean matrix $\mathcal{K}$ indexing the control strategy $\Omega : \mathcal{Q} \to 2^{\mathcal{A}}$. Otherwise, $\mathcal{WIN}$ and Ω are returned as empty sets. Note that, the operator $|A|_{\mathrm{numel}}$ on Line 4 of Algorithm 2 represents the total number of elements in the set A. Given a library of synthesized controllers in Algorithm 2, an execution of the complete reachability controller based on the robust finite transition system is shown in Algorithm 4 in the Appendix.

A merit of the proposed hierarchical structure is to decompose the overall high-dimensional contact-rich planning problem into tractable sub-problems with smaller state dimensions, circumventing prohibitive computational complexity. In particular, the symbolic task planner takes charge of the high-level decisions being reactive to the environment actions. The middle-level robust finite transition system reasons about the robustness of a local phase space region around the nominal keyframe state w.r.t bounded state disturbances. The low-level phase-space planner executes the continuous locomotion dynamics. This hierarchy is analogous to the receding horizon control approach in [Wongpiromsarn et al. (2012)], where the complex high-dimensional planning problem is decomposed into a set of solvable sub-problems. This strategy facilitates efficient decision making during dynamic interactions with uncertain environments.

Remark 4. *$\mathcal{TS}_{\mathrm{prod}}$ and $\mathcal{TS}_{\mathrm{OWS}}$ establish a hierarchical relationship for task decomposition. $\mathcal{TS}_{\mathrm{prod}}$ is a high-level decision maker of a nominal keyframe state while $\mathcal{TS}_{\mathrm{OWS}}$ reasons about the robustness of the local phase-space region around the nominal keyframe state determined from $\mathcal{TS}_{\mathrm{prod}}$. Overall, $\mathcal{TS}_{\mathrm{prod}}$ and $\mathcal{TS}_{\mathrm{OWS}}$ form a top-down hierarchy (see Fig. 11) to simultaneously achieve "global" phase-space decision making and "local" robustness reasoning.*

In next section, we will prove the robust reachability of the synthesized controller, i.e., the reachability goal is realizable for $\mathcal{SS}_{\mathrm{OWS}}$ if it is realizable for $\mathcal{TS}_{\mathrm{OWS}}$. With such a guarantee, the robust finite transition system $\mathcal{TS}_{\mathrm{OWS}}$ interfaces the high-level task planner commands with the low-level hybrid motion planner.

6 Correctness of The Reactive Task and Motion Planner

Correctness guarantees of the whole-body dynamic locomotion (WBDL) planner play a key role in the successful execution of robust legged locomotion interacting with dynamic environments. The objective of this section is to prove such a correctness. In particular, the correctness of our planning framework is interpreted as successful implementations of the high-level task planner on the low-level motion planner under bounded disturbances. Our locomotion planner is a hierarchy composed of a task planning layer with reactive synthesis and a robust motion planner layer synthesized by a robust finite transition system. A high-level task planner, i.e., a WBDL winning strategy $\mathcal{A}_{\mathrm{WBDL}}$, is synthesized via a two player game $\mathcal{G}$. The two-player game $\mathcal{G}$ is solved between the robot and its environment to make a decision $(p, s, \boldsymbol{q})$ representing the locomotion mode p, the contact action s, and the keyframe state $\boldsymbol{q}$, respectively. This locomotion decision determines a nominal phase-space motion plan and is sent to the robust finite transition system $\mathcal{TS}_{\mathrm{OWS}}$ such that the high-level decision is achieved in a robust manner by the low-level continuous motion planner.

6.1 One-walking-step robust reachability

To guarantee the robust finite transition system to be realizable by its underlying continuous system, we need to prove that the conditions in Eqs. (36)-(38) of Def. 11 also hold for continuous states. Since the robust finite transition system is based on the keyframes of one walking step, we name the keyframe reachability problem as "one-walking-step robust reachability". We model the bounded disturbance causing initial state deviations, model uncertainties, and external perturbations during the evolution of the locomotion trajectory. The term "robust reachability" refers to the reachability of the goal robustness margin set centered around the final keyframe from the initial robustness margin set.

Theorem 1 (**One-walking-step robust reachability**). *Consider a one-walking-step locomotion subsystem $\mathcal{SS}_{\text{OWS}}$ with two pairs of decisions $(\boldsymbol{q}_{\text{initial}}, p_1, s_1)$ and $(\boldsymbol{q}_{\text{final}}, p_2, s_2)$ and its inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$. Assume that $\mathcal{SS}_{\text{OWS}} \preceq_{(\delta\zeta, d, \mathcal{M}_{\text{OWS,Euc}})} \mathcal{TS}_{\text{OWS,INT}}$ as defined in Def. 13. If it is realizable for $\mathcal{TS}_{\text{OWS,INT}}$, this walking step is realizable for $\mathcal{SS}_{\text{OWS}}$, i.e., the robustness margin set $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$ of the final keyframe $\boldsymbol{q}_{\text{final}}$ is reachable from $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$.*

Proof. Suppose that the walking step from $(\boldsymbol{q}_{\text{initial}}, p_1, s_1)$ to $(\boldsymbol{q}_{\text{final}}, p_2, s_2)$ is realizable for $\mathcal{TS}_{\text{OWS,INT}}$, i.e., the winning set $\mathcal{WIN}_{\text{TS}} \neq \emptyset$ and there exists a control strategy $\Omega_{\text{TS}} : \mathcal{Q} \to 2^{\mathcal{A}}$ for $\mathcal{TS}_{\text{OWS,INT}}$. Let $\boldsymbol{q} \in \mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and $\boldsymbol{q} \in \mathcal{WIN}_{\text{TS}}$. Under bounded disturbances, all the possible state sequences starting from $\boldsymbol{q}$, generated by the locomotion dynamics and the control strategy synthesized by Algorithm 2, will finally reach the target set $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$. Let $\{\boldsymbol{q}_i\}_0^l$ ($l \in \mathbb{Z}$) be one of such sequences generated under a control sequence $\{\boldsymbol{a}_{i,\text{INT}}\}_0^{l-1}$ and a sequence of disturbances $\{\boldsymbol{d}_i\}_0^{l-1}$ such that $\boldsymbol{q}_0 = \boldsymbol{q}$ and $\boldsymbol{q}_l \in \mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$.

We construct a control strategy $\Omega_{\text{OWS}} : \Xi_{\text{OWS}} \to 2^{\mathcal{A}}$ for the one-walking-step locomotion system $\mathcal{SS}_{\text{OWS}}$ by

$$\boldsymbol{a}_{\text{INT}} \in \Omega_{\text{OWS}}(\boldsymbol{\xi}), \text{ if } \boldsymbol{a}_{\text{INT}} \in \Omega_{\text{TS}}(\mathcal{M}_{\text{OWS,Euc}}(\boldsymbol{\xi})).$$

Given the transitions of $\mathcal{TS}_{\text{OWS,INT}}$ assigned by Eq. (40), for $\forall \boldsymbol{\xi}(0) \in \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q})$, there always exist a state $\boldsymbol{q}'$ and a control input $\boldsymbol{a}_{\text{INT}}$ such that $\boldsymbol{\xi}(\delta\zeta) \in \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q}')$. Thus, for any $\boldsymbol{\xi}(0) \in \mathcal{M}^{-1}_{\text{OWS,Euc}}(\boldsymbol{q})$, the resulting solution with the same control sequence $\{\boldsymbol{a}_{i,\text{INT}}\}_0^{l-1}$ and disturbance $\{\boldsymbol{d}_i\}_0^{l-1}$ generated by Ω_{OWS} will be guaranteed to reach the target set $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$. This implies that at time $t_k = k \cdot \delta\zeta, \forall k \leq l$, $\boldsymbol{\xi}(t_k) \in \mathcal{WIN}_{\text{SS}}$ (i.e., the winning set of $\mathcal{SS}_{\text{OWS}}$). Therefore, $\mathcal{WIN}_{\text{SS}} \neq \emptyset$ and Ω_{OWS} is such a controller that can realize the one walking step. This completes the proof. □

6.2 *Correctness of the hierarchical WBDL planner*

Given the one-walking-step robust reachability of Theorem 1, we now prove the correctness of the top-down planning hierarchy, i.e., $\mathcal{TS}_{\text{prod}} \to \mathcal{TS}_{\text{OWS}} \to \mathcal{TS}_{\text{OWS,INT}} \to \mathcal{SS}_{\text{OWS}}$ as shown in Fig. 11. The correctness is defined in a robust sense, i.e., the actual keyframe states of the phase-space trajectory always stays within the robustness margins of the nominal keyframe states determined by the high-level task planner.

Definition 14. *Assume a low-level locomotion trajectory $\kappa = (\boldsymbol{\xi}, \rho, \eta, \mu)$ and a high-level decision sequence $\gamma = (q, e, s, p)$ as defined in Definition 7. The low-level trajectory κ is a continuous implementation of the high-level execution γ, if there exists a sequence of non-overlapping phase intervals $\mathcal{H} = H_1 \cup H_2 \cup H_3 \ldots$ and $\cup_{i=1}^{\infty} H_i = \mathbb{R}^+$ such that $\forall \zeta \in H_k, \forall k \geq 1$, the following mappings hold*

$$\boldsymbol{\xi}(\zeta_{\text{left-bnd}}) \in \mathcal{M}^{-1}_{\text{Riem}}(q_k), \boldsymbol{\xi}(\zeta_{\text{right-bnd}}) \in \mathcal{M}^{-1}_{\text{Riem}}(q_{k+1}), \rho(\zeta) = e_k, \eta(\zeta) = s_k, \mu(\zeta) = p_k,$$

where $\mathcal{M}_{\text{Riem}}$ is the Riemannian space abstraction defined in Eq. (28), which maps the continuous state $\boldsymbol{\xi}$ region centered at the keyframe state into the discrete keyframe q [Liu et al. (2013)]. $\zeta_{\text{left-bnd}}$ and $\zeta_{\text{right-bnd}}$ are the left and right boundary value of the interval H_k.

By this definition and stutter-equivalence [Baier and Katoen (2008)], we can conclude $\kappa \models \varphi$ if and only if $\gamma \models \varphi$, where φ is the task specifications in the symbolic task planner. For our phase-space planning, the interval H_k represents the phase duration of the k^{th} walking step. This guarantees that the left boundary point of H_k approaches to infinity as $k \to \infty$, and thus the continuous implementation guarantees the Zeno behavior to be ruled out. For detailed explanations, reader can refer to [Liu et al. (2013)] and the reference therein. Given these preliminaries, we have the following correctness theorem:

Theorem 2 (**Correctness of the WBDL task and motion planner**). *Given a robust finite transition system $\mathcal{TS}_{\text{OWS}}$, a winning WBDL strategy synthesized from the two-player game is guaranteed to be implementable by the underlying low-level phase-space motion planner in a provable correct manner.*

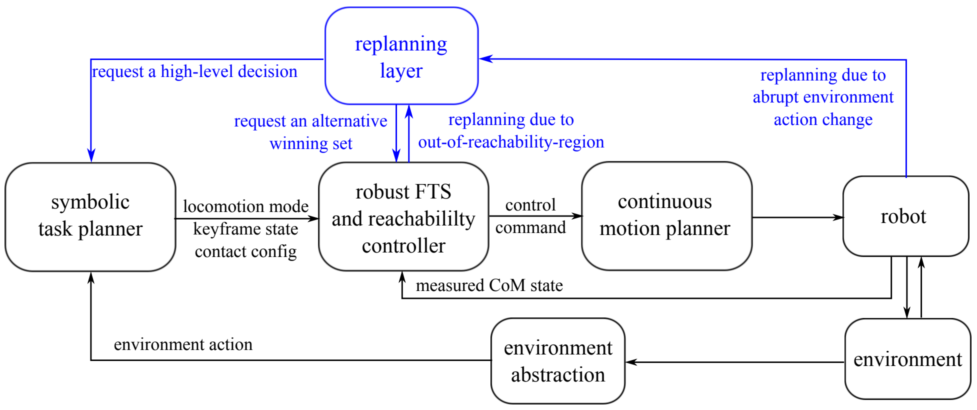

replanning layer
request a high-level decision
replanning due to abrupt environment action change
request an alternative winning set
replanning due to out-of-reachability-region
symbolic task planner
locomotion mode
keyframe state
contact config
robust FTS and reachabililty controller
control command
continuous motion planner
robot
measured CoM state
environment action
environment abstraction
environment

Fig. 12. Hierarchical structure of robust WBDL task and motion planner. This planner has three cascaded planning layers: high-level task planner, middle-level keyframe-based robust finite transition system, and low-level continuous motion planner. A replanning process (see blue lines) is triggered when the state is out of the reachability region (i.e., the winning set) or a sudden environmental change occurs.

Proof. By Proposition 1, a winning WBDL strategy $\mathcal{A}_{\rm WBDL}$ synthesized from the WBDL task planner game solves the discrete locomotion planning problem on $\mathcal{TS}_{\rm prod}$. This synthesis is correct-by-construction thanks to the properties of GR(1) formulae. According to $\mathcal{A}_{\rm WBDL}$, the system action s_{k+1}, switching mode p_{k+1}, and keyframe q_{k+1} at the next $(k+1)^{\rm th}$ walking step are derived from next environment actions e_{k+1} and current keyframe state q_k. To verify the correct implementation of a high-level decision sequence γ, we use the switching strategy semantics: given an initial state $\boldsymbol{\xi}(\zeta_0)$ and an initial environment action $\rho(\zeta_0) = e_0$, we assign $\eta(\zeta_0) = s_0$ and $\mu(\zeta_0) = p_0$ according to $\mathcal{A}_{\rm WBDL}$, where the step index $k = 0$. By using the control library synthesized from the robust finite transition system $\mathcal{TS}_{\rm OWS}$ with decision tuples of two consecutive walking steps (i.e., (p_0, s_0, q_0) and (p_1, s_1, q_1)) , we select a specific reachability controller synthesized by $\mathcal{TS}_{\rm OWS,INT}$ to achieve a robust keyframe transition at the next walking step. This is guaranteed by the one-walking-step robust reachability in Theorem 1, where the realizability of $\mathcal{TS}_{\rm OWS,INT}$ implies the realizability of the underlying continuous system $\mathcal{SS}_{\rm OWS}$. By executing this reachability controller, the continuous dynamics $\boldsymbol{\xi}(\zeta)$ evolve by following the dynamics of a specific locomotion mode $\dot{\boldsymbol{\xi}}(\zeta) = f_{p_0}(\boldsymbol{\xi}(\zeta), \boldsymbol{u}(\zeta), d(\zeta))$ under the bounded disturbance d. Once we detect a new environment action e_2 before the locomotion contact switch, a new decision tuple (p_2, s_2, q_2) is generated immediately based on $\mathcal{A}_{\rm WBDL}$. Given this new decision tuple, the same procedure is repeated as above for the future $k^{\rm th}$ walking step where $k \in \mathbb{Z}_{\geq 2}$. Therefore, it is proved that the low-level trajectory correctly implements the high-level decision sequence. □

6.3 *Replanning strategy and robustness*

It is worth noting that the proposed correctness holds under a set of assumptions on allowable environmental and system actions and disturbance boundedness. However, sometimes the real-world disturbance can violate the bounded disturbance assumption and perturb the state to be out of the local reachability region (i.e., the winning set). To handle this situation, we establish a replanning strategy to request a new high-level task planner command. Ideally, the union set of all local winning sets is expected to cover the entire state space of interest. However, the existence of a such a winning set union often can not be guaranteed. Thus, it is difficult to generalize formal correctness of one winning set to that of the union set of all winning sets. What we strive to is to maximize the phase-space coverage by the union set of all winning sets. From a practical implementation viewpoint, synthesizing a large number of winning sets enables our planner to cover a sufficiently large phase space such that it is always likely to find a feasible winning set when large disturbances occur.

Algorithm 3 Execution of reactive task and motion planner

```
1: procedure ExecuteReactivePlanner(nextEnvironmentAction $e_{\text{next}}$, currentKeyframe $q_{\text{current}}$, currentLocomotionMode
       $p_{\text{current}}$, automaton $\mathcal{A}_{\text{WBDL}}$)
2: $e_{\text{next}} \in \mathcal{E}$
3: $q_{\text{next}} \leftarrow$ getNextKeyframe($e_{\text{next}}, q_{\text{current}}$)        {look up $\mathcal{A}_{\text{WBDL}}$}
4: $s_{\text{next}} \leftarrow$ getNextContactAction($e_{\text{next}}, q_{\text{next}}$)
5: $p_{\text{next}} \leftarrow$ getNextLocomotionMode($e_{\text{next}}, q_{\text{next}}$)
6: $(\boldsymbol{\xi}_{\text{trans}}, t_{\text{trans}}) \leftarrow$ searchContactTransition($q_{\text{current}}, q_{\text{next}}$)
7: $(\boldsymbol{y}_{\text{limb,next}}) \leftarrow$ searchLateralLimbLocation($q_{\text{current}}, q_{\text{next}}$)
8: $(\boldsymbol{\xi}, \boldsymbol{\chi}) \leftarrow$ ExecuteOWSReachabilityControl(keyframeState $q_{\text{current}}, q_{\text{next}}$, locomotionMode $p_{\text{current}}, p_{\text{next}}$, contactAc-
       tion $s_{\text{current}}, s_{\text{next}}$, initialCoMState $\boldsymbol{\xi}_{\text{init}}$, transitionTime $t_{\text{trans}}$, nextEnvironmentAction $e_{\text{next}}$)
9: $(e, q, p, s)_{\text{current}} \leftarrow (e, q, p, s)_{\text{next}}$        {update task planner states}
10: repeat from Line 2
11: end procedure
```

In other words, there is no ground truth of "formal correctness" for real robotic systems. Even though we have a provably correct planner and implement it on a real robot in a correct way, the actual planner may not be formally "correct" due to many potential hardware issues. For instance, unmodelled actuator dynamics can easily break the correctness guarantee of task specifications at the high level. Thus, it makes more practical sense to target a formally correct approach that generates a palette of robust controllers, the winning sets of which jointly cover a sufficiently large phase-space of interest (if not a global state space). Our results indicate that a properly designed controller switching mechanism among these locomotion winning sets enables an effective replanning strategy such that a set of contiguous phase-space initial robustness margin sets can be controlled to reach a set of contiguous goal robustness margin sets under bounded disturbances.

Overall, the proposed planning framework reasons about robustness at the following three levels.

- The robust finite transition system $\mathcal{TS}_{\text{OWS}}$ explicitly incorporates neighbouring keyframe states via the robustness margin around the nominal keyframe state to handle initial state uncertainty in each walking step.
- If the disturbance is larger than the boundary value modeled in the controller synthesis, the state may be disturbed out of the winning set. In this case, an alternative winning set will be searched in the control library of allowable keyframe transitions determined by $\mathcal{TS}_{\text{OWS}}$. If no alternative winning set is feasible, a replanning signal will be sent to the high-level task planner. The task planner will use the synthesized automaton $\mathcal{A}_{\text{WBDL}}$ to assign a new locomotion decision $(p, s, \boldsymbol{q})$ and send it to the motion planner layer for replanning the next walking step.
- When an environment event changes suddenly, a replanning signal will be sent to the high-level task planner. Note that, this replanning process can only be executed before the next step transition. Otherwise, the contact of the next walking step already occurs. Fig. 23 in the Appendix shows a timing sequence of the replanning process. More details of this replanning process are shown in Algorithm 4 in the Appendix.

7 Results

We demonstrate whole-body dynamic locomotion (WBDL) results by sequentially composing the low-level locomotion modes via the symbolic task planner. In particular, we analyze in detail the robustness performance of the reachability control with respect to several key parameters. The Temporal Logic Planning (TuLiP) toolbox, a python-based embedded control software [Wongpiromsarn et al. (2011)], is used to synthesize the symbolic task planner. The gr1c[8] tool, involving the CU Decision Diagram Package, is used by TuLiP as the underlying synthesis solver. The synthesized planner is correct by construction, i.e., satisfying all the proposed specifications. The LTL synthesis procedure is offline and will take around

[8] `http://scottman.net/2012/gr1c`

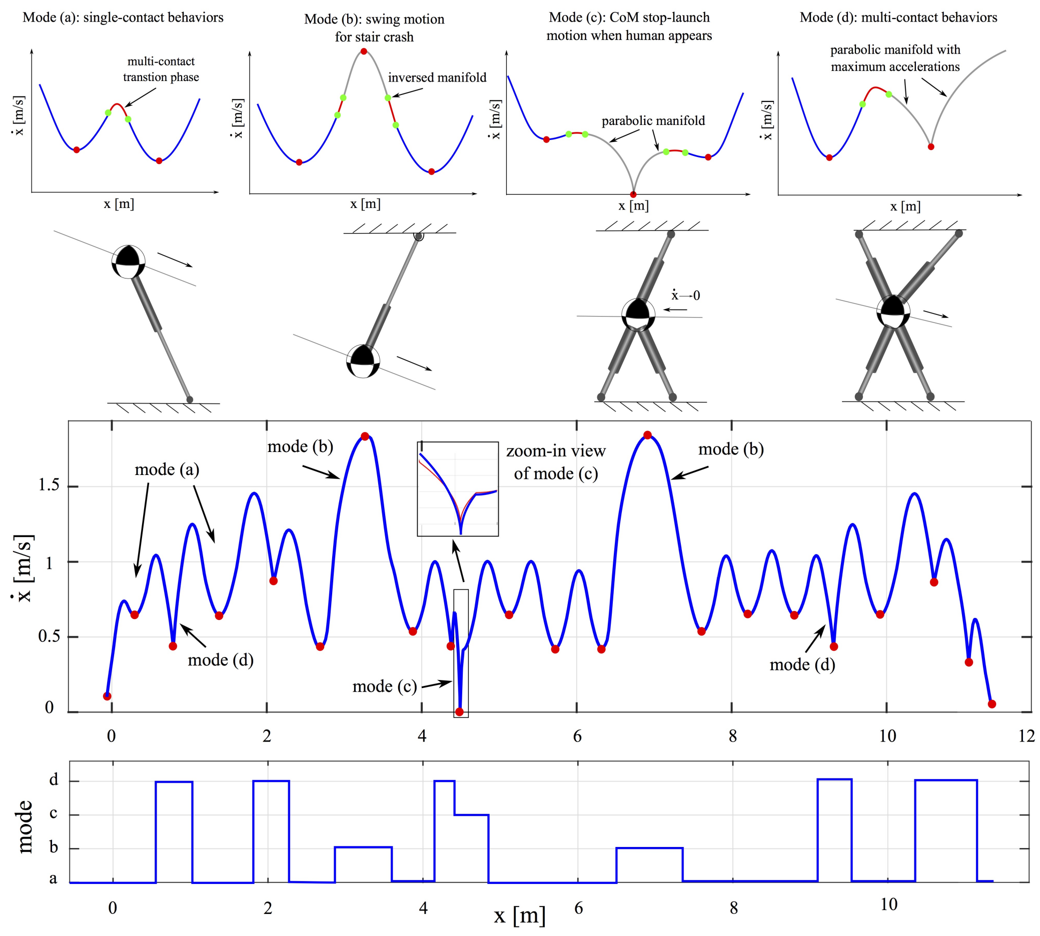


Fig. 13. Sequential composition of the sagittal CoM phase-space trajectories and mode switchings for a 20-step WBDL maneuver. The top four figures illustrate phase-space manifolds of the four locomotion modes. The mode switching is governed by the synthesized high-level contact planner. Among these steps, two terrain crack and one human appearance events are taken into account. A short multi-contact phase is designed between every two consecutive modes for a smooth transition (see the short red trajectory between two green dots).

20 minutes to generate an automaton on a MacBook with a 2.9 GHz Intel Core i9 processor and 32 GB of memory. Once the automaton is generated, the task planning process will be executed at run-time. To guarantee the successful implementations of the low-level motion plans by the high-level task planner, we perform a robust reachability analysis of the keyframe states by using the so-called robustly complete control synthesis (ROCS)[9] tool [Li and Liu (2018)], which currently supports abstraction-based control synthesis using both uniform and non-uniform discretizations. ROCS also generates feedback control strategies, which are used to design the control library of our locomotion tasks.

7.1 *Case I: locomotion with stopping and brachiation behaviors*

We first demonstrate a locomotion scenario involving environmental actions such as the appearance of a human and the terrain being crack in the scene. The synthesized discrete task planner is represented by a finite state automaton with 27 states and 148 transitions. The two-dimensional keyframe state q is composed of the apex velocity and step length. For either dimension, Small, Medium and Large labels are assigned to $\{1, 2, 3\}$ in order while Stop and Swing labels are assigned to $\{0, 4\}$. Fig. 13 illustrates the sequentially composed center-of-mass (CoM) sagittal phase-space trajectory of

[9] `https://git.uwaterloo.ca/hybrid-systems-lab/rocs`

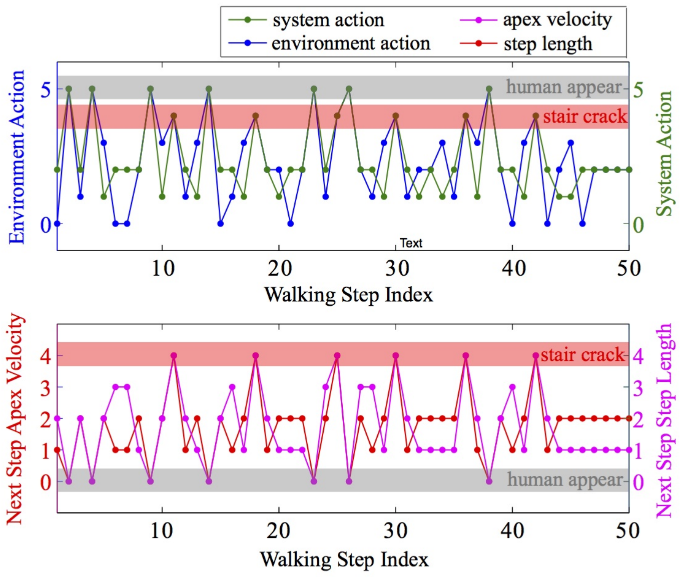


Fig. 14. Environment events, system actions and keyframe states of 50 walking steps according to the synthesized automaton. Actions and states are indexed by numbers. Emergency events, i.e, human appearance and terrain crack, are highlighted in the shaded regions. In the bottom subfigure, the numbers 0 to 4 on the vertical axis correspond to $\{0, 0.4, 0.6, 0.8, 1.7\}$ m/s for next step apex velocity and $\{0.15, 0.5, 0.6, 0.7, 0.6\}$ m for next step step length.

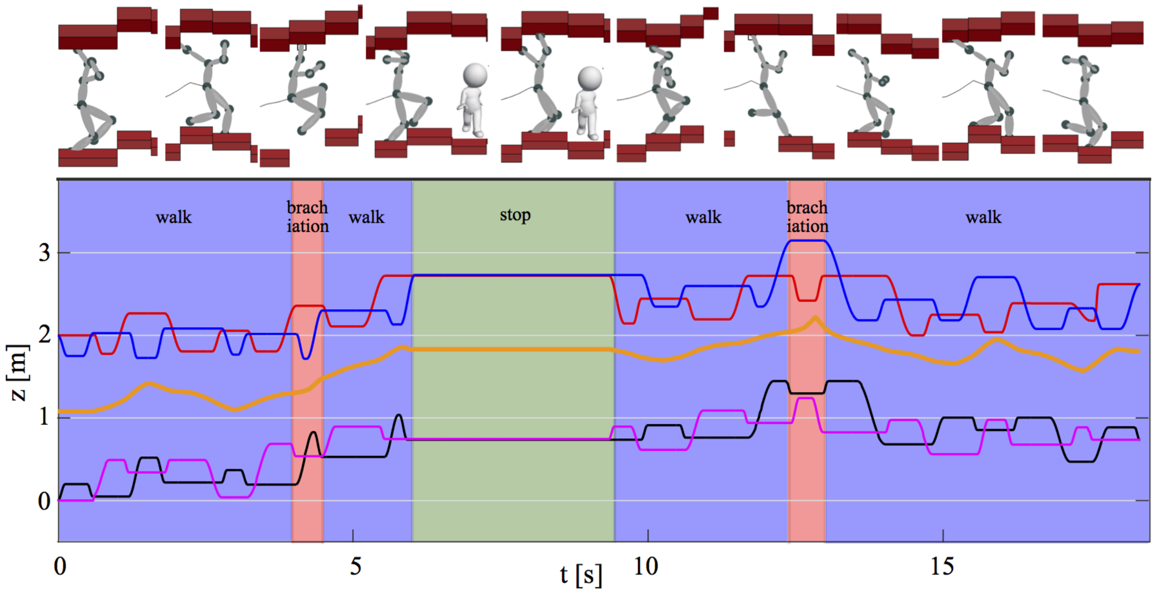


Fig. 15. Snapshots of the WBDL motions in respond to two environmental emergencies. The snapshots show a sequence of locomotion behaviors including a brachiation motion over the cracked terrain and a stopping motion when a human appears. The figure at the bottom shows the CoM vertical position trajectory (orange thick line), hand and feet trajectories (thin interlaced lines).

a 20-step walking process. Fig. 14 illustrates the discrete environment and system contact actions, and the corresponding keyframe states. At the low level, four locomotion modes (i.e., PIPM, PPM, MCM, SLM) are alternated according to the high-level decisions[10]. These high-level decisions are sent to the low-level motion planner one walking step ahead,

[10]For simplification, our model implements a simlified version of the above model with assumptions of a constant CoM acceleration and zero values of the states φ and θ.

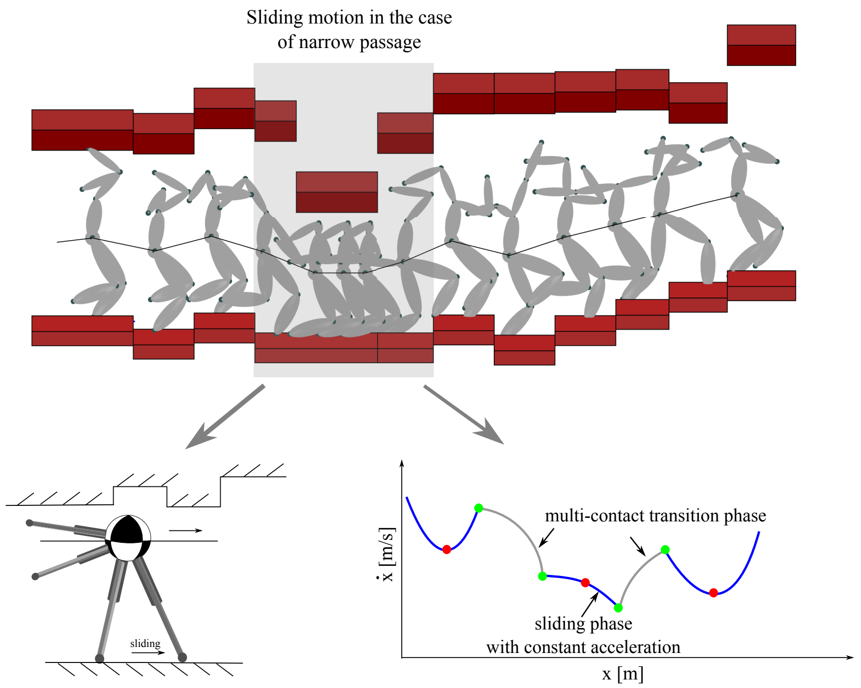

Sliding motion in the case of narrow passage
sliding
multi-contact transition phase
sliding phase with constant acceleration
ẋ [m/s]
x [m]

Fig. 16. Ground sliding motion when a narrow passage appears in the scene. The robot crouches and slides on the ground through the low-ceiling area with a constant negative acceleration. This ground sliding motion is preceded and succeeded by a multi-contact transition phase as shown in the phase-space subfigure at the lower right corner.

i.e., a one-walking-step horizon. By inspecting the discrete sequences, we can verify that all the system contact actions and keyframe states respond to the environmental actions correctly, i.e., all the task specifications are satisfied. Fig. 15 illustrates dynamic motion snapshots and continuous kinematic trajectories of the vertical CoM, foot, and hand positions. An accompanying video about the WBDL behaviors is available at `https://youtu.be/BdxYCmhRIMg`.

7.2 *Case II: locomotion with ground sliding and hopping behaviors*

When the robot maneuvers through a narrow passage, an ordinary locomotion mode (e.g., walk and brachiation) will not work anymore due to the confined height. As such, a natural solution is to use a ground sliding mode: the robot crouches and slides with two feet through this constrained space as shown in Fig. 16. The two arms are placed at a low position to avoid the contact with the ceiling. As shown in the bottom right subfigure of Fig. 16, there are two multi-contact transition phases before and after the sliding phase (see the gray trajectory segments). We assume a constant negative CoM acceleration during the sliding phase, and thus the phase space trajectory of the sliding phase is a parabola. The low-level locomotion model corresponds to the mode (f) in Section 3.1.

When a crack in the terrain and a high ceiling occur simultaneously, the robot can not grasp the overhead support any longer. To maneuver forward successfully, the robot has to leap over the unsafe region as shown in Fig. 17. Thus, a hopping phase will be executed with no contact with the environment. A constant CoM sagittal velocity shows up in the sagittal phase portrait while a parabola appears in the vertical position trajectory of Fig. 17(b). The keyframe state of this hopping motion is chosen to be the center of the horizontal line segment in the sagittal phase-space. The lateral velocity is set to zero to avoid a lateral drift. The state will stay at the red dot in Fig. 17(d) during the hopping motion. Since the robot locomotes forward, the lateral phase portrait in Fig. 17(d) behaves like a limit cycle but non-periodically due to the rough terrain. The low-level locomotion model corresponds to the mode (e) in Section 3.1. Via introducing specific locomotion modes, our planner is capable of handling emergency-motivated scenarios.

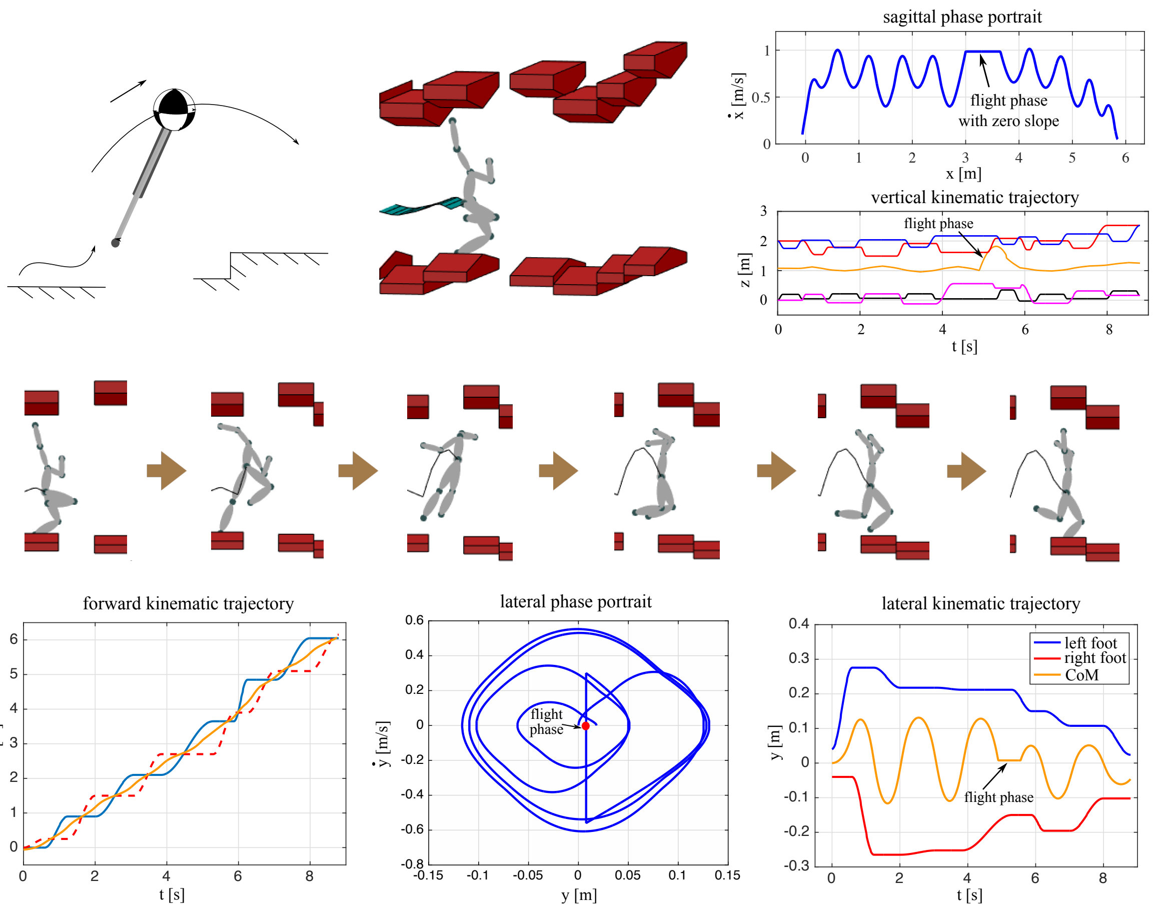


Fig. 17. Hopping motion when a crack in the terrain and a high ceiling occur simultaneously. In this case, no overhead support is available for grasping so the robot has to jump over the cracked terrain.

7.3 Case III: locomotion replanning strategy

When the robot is already leaping in the air and detects a sudden change from an ordinary terrain to a cracked terrain, it has to replan its contact action and locomotion mode to accommodate this sudden change on the fly. The robot will execute a replanning process to ask the high-level planner for a new decision, i.e., grasp the ceiling support and swing the robot's body over the cracked region as shown in Fig. 18. Otherwise, the robot will fail to locomote. The top subfigure of Fig. 18 shows a decision sequence including the replanning process. The three rows represent environment actions, locomotion modes, and keyframe states, respectively. The second column with three dotted blocks is the decision before the replanning process and not executed yet until the next walking step. The third column represents the replanned decision which is executed in response to the sudden environmental action change. This replanning process in the phase-space is illustrated in Fig. 23. Let us consider a more challenging terrain scenario (i.e., terrainCrack-highCeiling as described in Section 4), where there is a cracked terrain with a high ceiling. Then the robot can not replan by grasping the overhead support anymore, and the locomotion process will fail inevitably. To avoid this situation, the terrainCrack-highCeiling environment action is not allowed to occur consecutively in our environment specification $S_{\text{e-1}}$.

Besides the above replanning strategy in response to environment changes, there is another replanning strategy embedded in the reachability control library. When the robot state is perturbed to be out of the reachability region, i.e., the winning set currently being executed, the robot can not reach the robustness margin of the targeted keyframe. Then the robot calls for a new reachability controller in the library that covers the current perturbed state and uses this replanned controller to reach a new keyframe goal. Overall, the replanning process determines which new controller to call in the control library.

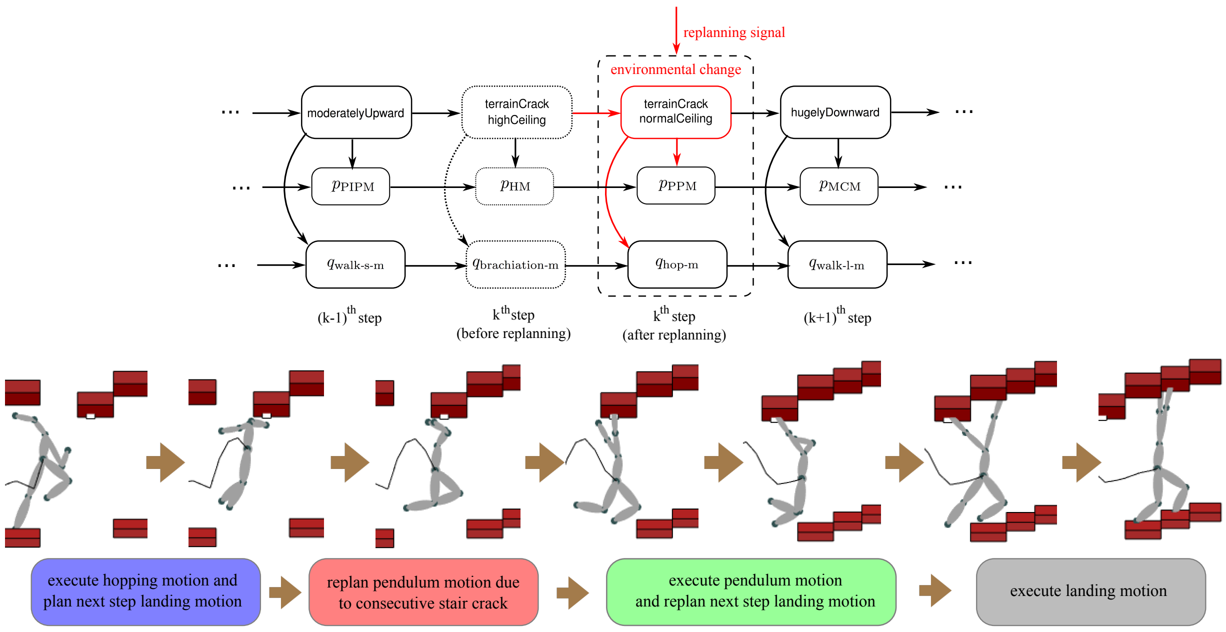

replanning signal
environmental change
moderatelyUpward
terrainCrack highCeiling
terrainCrack normalCeiling
hugelyDownward
p_{PIPM}
p_{HM}
p_{PPM}
p_{MCM}
$q_{\text{walk-s-m}}$
$q_{\text{brachiation-m}}$
$q_{\text{hop-m}}$
$q_{\text{walk-l-m}}$
(k-1)th step
kth step (before replanning)
kth step (after replanning)
(k+1)th step
execute hopping motion and plan next step landing motion
replan pendulum motion due to consecutive stair crack
execute pendulum motion and replan next step landing motion
execute landing motion

Fig. 18. Replanning in response to a sudden environment event change. Suppose that during the flight phase, the robot finds out that the next terrain is cracked. Accordingly, the robot triggers its replanning process by changing the locomotion mode to the prismatic pendulum model, i.e., grasping the overhead support, swinging the body over the second terrain crack region, and then landing on the non-cracked terrain. The top subfigure shows the decision sequence in the symbolic task planner. The bottom subfigure shows the snapshot sequence of the locomotion process.

7.4 *Case IV: Validation of the robust reachability controller*

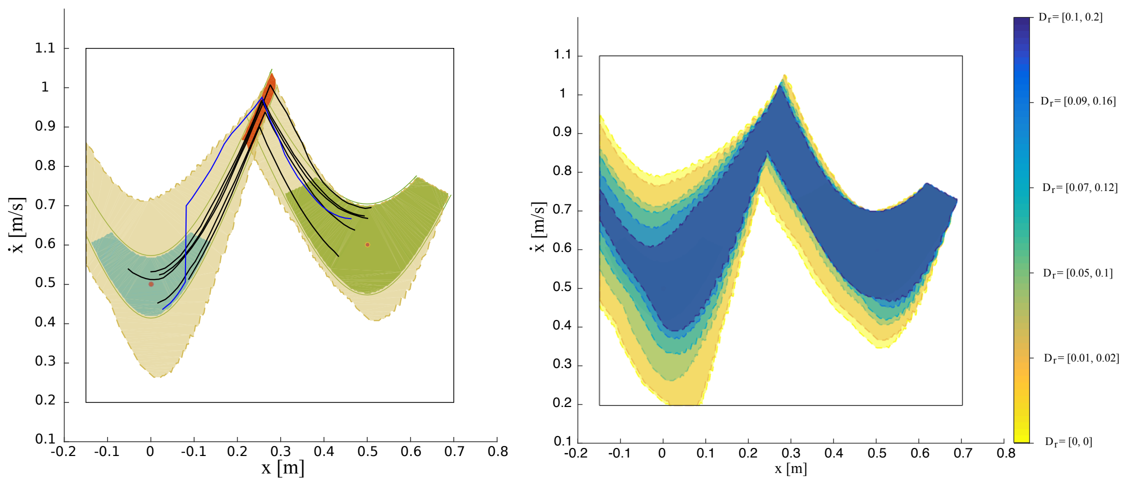
This case study evaluates the performance of the synthesized controller given a keyframe robust reachability goal. Let us first consider the prismatic inverted pendulum model (PIPM). Assume that we have a sagittal CoM state vector $\boldsymbol{\xi} = (x, v_x)^T$ and two consecutive locomotion modes denoted by PIPM_1 and PIPM_2, respectively. The PIPM dynamics in Eq. (3) are reformulated as the CoM dynamics below

$$\begin{pmatrix} \dot{x}(\zeta) \\ \dot{v}_x(\zeta) \end{pmatrix} = \begin{pmatrix} v_x(\zeta) \\ \omega^2_{\text{PIPM}}(x(\zeta) - x_{\text{foot}}) \end{pmatrix}, \tag{43}$$

where we assume zero torso angular momentum $(\tau_x, \tau_y) = \mathbf{0}$ and a predefined foot placement position x_{foot} for simplicity. The continuous control input $\omega_{\text{PIPM}} \in [\omega_{\text{nominal}} - \delta\omega, \omega_{\text{nominal}} + \delta\omega]$, where $\delta\omega$ is a predefined bound. Note that the parameters $x_{\text{foot},1}, x_{\text{foot},2}$, and ω_{nominal} are determined by the high-level symbolic task planner.

Let us define two nominal keyframe states $\boldsymbol{q}_{\text{initial}} = (\dot{x}(\zeta_0), v_x(\zeta_0)) = (0\text{m}, 0.5\text{m/s})$ and $\boldsymbol{q}_{\text{final}} = (\dot{x}(\zeta_{\text{final}}), v_x(\zeta_{\text{final}})) = (0.5\text{m}, 0.6\text{m/s})$[11] determined from the high-level planner. The goal is to solve the PIPM closed-loop phase-space trajectories starting from the initial robustness margin set $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and reaching the final robustness margin set $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$ as defined in Def. 10. To this end, we synthesize a controller to determine the realizability of a keyframe transition in one walking step for $\mathcal{TS}_{\text{OWS}}$ and generate a control strategy $\Omega : \Xi_{\text{OWS}} \to 2^{\mathcal{A}_{\text{OWS}}}$ if it is realizable. The intermediate robustness margin set $\mathcal{B}_{\text{inter}}$ between the locomotion modes PIPM_1 and PIPM_2 is defined by Eq. (38). Locomotion mode switching is only allowed when the state is within $\mathcal{B}_{\text{inter}}$. Overall, the controller synthesis of one walking step is composed of three steps: first, the CoM trajectory starts from $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and moves towards $\mathcal{B}_{\text{inter}}$; second, the state reaches $\mathcal{B}_{\text{inter}}$ and switches the locomotion mode; third, the CoM state reaches $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$. To reach $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$,

[11] In this paper, (a, b) represents a vector of two values a and b while $[a, b]$ represents an interval bounded by a and b.



(a) Controlled trajectories of PIPM one-walking step. (b) Winning sets under different levels of disturbances.

Fig. 19. The additive disturbances to the dynamics are bounded by $D_r = (0.05\text{m}; 0.1\text{m/s})$ in the subfigure (a). The shaded yellow region represents the winning set. The black trajectories are the five closed-loop trajectories simulated in five trials. The blue trajectory represents a trial suffering a large disturbance, i.e., a velocity jump in the phase-space. Since the disturbed state is still in the winning set, the CoM trajectory is guaranteed to reach the final robustness margin set. In subfigure (b), different levels of bounded disturbances are modeled in the computation of the winning sets. Naturally, a larger magnitude of the disturbance results in a smaller winning set.

the conditions in Eqs. (36)-(38) need to be satisfied by propagating the PIPM dynamics forward under bounded state disturbances and a bounded control input ω_{PIPM}. For this example, we assign the initial and final robustness margins of $\mathcal{B}_{\epsilon_1}(\boldsymbol{q}_{\text{initial}})$ and $\mathcal{B}_{\epsilon_2}(\boldsymbol{q}_{\text{final}})$ as $\delta\zeta_{\epsilon_1} = 0.05, \delta\sigma_{\epsilon_1} = 0.002$ and $\delta\zeta_{\epsilon_2} = 0.05, \delta\sigma_{\epsilon_2} = 0.006$, respectively.

As to the underlying continuous locomotion subsystem $\mathcal{SS}_{\text{OWS}}$, we assign the one-walking-step state space $\Xi_{\text{OWS}} = \bigcup_{p\in\mathcal{P}_{\text{OWS}}} \Xi_p$ where $\mathcal{P}_{\text{OWS}} = \{\text{PIPM}_1, \text{PIPM}_2\}$, $\Xi_p = [-0.1\text{m}, 0.7\text{m}] \times [0.1\text{m/s}, 1.2\text{m/s}]$ for all $p \in \mathcal{P}_{\text{OWS}}$, the initial state set $\Xi_{\text{OWS},0} = \{\boldsymbol{\xi} : |\mathcal{Z}_{p_1,\sigma}(\boldsymbol{\xi})| \leq \delta\sigma_{\text{bound,init}}\}$, the control space $\mathcal{U}_{\text{OWS}} = [2, 4]$. To construct an inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$, we uniformly discretize the state space Ξ_{OWS} with a granularity $[0.005\text{m}, 0.005\text{m/s}]$ and sample the control space $\mathcal{U}_{\text{OWS}}$ with a 0.02rad/s granularity, resulting in a sampled finite control set $\widehat{\mathcal{U}}_{\text{OWS}} = 0.02\mathbb{Z} \cap [2, 4]$. Let $\mathcal{A}_{\text{OWS}} = \widehat{\mathcal{U}}_{\text{OWS}}^{[0,\delta\zeta]}$ be a piece-wise trajectory with zero-order-hold values in $[0, \delta\zeta]$, where $\delta\zeta = 2$ ms is the time duration for each state in $\mathcal{T}_{\text{OWS}}$. The system dynamics in (43) are subject to additive disturbances bounded by $D_r = (0.05\text{m}; 0.1\text{m/s})$, i.e., position and velocity disturbances, respectively. Given the PIPM parameters above, we synthesize a reachability controller of $\mathcal{TS}_{\text{OWS,INT}}$ and the computed winning sets are shown in Fig. 19(a). As the result shows, the one-walking step reachability is realizable as long as the winning set overlaps (at least partially) the initial and final robustness margin sets. Five simulated trajectories under randomly-sampled bounded disturbances are shown as the black lines. Fig. 19(b) evaluates the changing size of the winning set under different levels of the disturbance. The winning set shrinks as the disturbance set increases because the synthesized controller needs to reach the goal robust set against a larger set of disturbances.

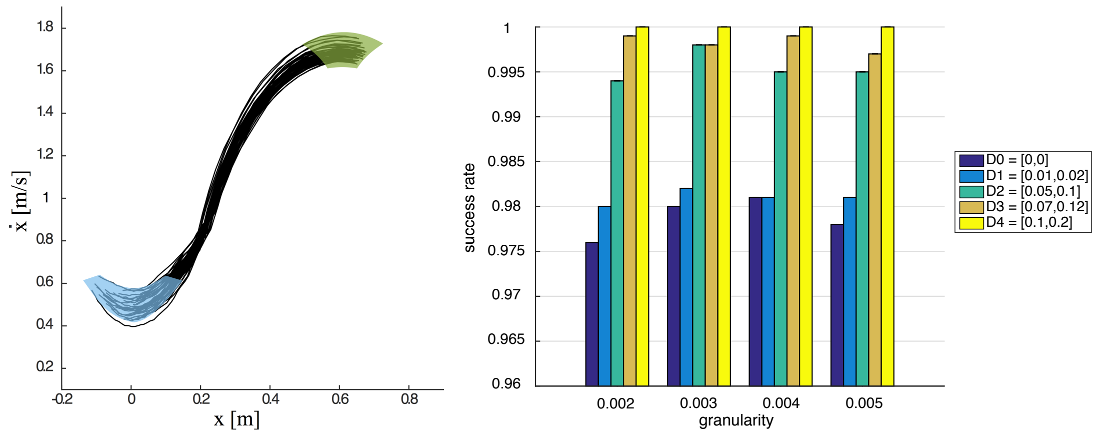
We design reachability controllers for all the combinations of the locomotion mode set $\mathcal{P}$. Consider another locomotion mode transition from the PIPM to the prismatic pendulum model (PPM). The PPM dynamics in Eq. (4) are reformulated as follows

$$\begin{pmatrix} \dot{x}(\zeta) \\ \dot{v}_x(\zeta) \end{pmatrix} = \begin{pmatrix} v_x(\zeta) \\ -\omega_{\text{PPM}}^2(x(\zeta) - x_{\text{hand}}) \end{pmatrix}, \tag{44}$$

with the assumption of $\tau_x = \tau_y = 0$ and a predefined hand contact position x_{hand}. Other parameters are defined in Table 1. To evaluate the robustness performance of the synthesized controller, we examine the success rate of reaching the goal

Table 1: Parameters of the PIPM-PPM mode transition.

Parameters	**Values**	**Parameters**	**Values**
initial keyframe $\boldsymbol{q}_{\text{initial}}$	$(0\text{m}, 0.5\text{m/s})$	final keyframe $\boldsymbol{q}_{\text{final}}$	$(0.6\text{m}, 1.7\text{m/s})$
initial tangent bound $\delta\sigma_{\text{bound,init}}$	0.002	initial cotangent bound $\delta\zeta_{\text{bound,init}}$	0.05
final tangent bound $\delta\sigma_{\text{bound,final}}$	0.06	final cotangent bound $\delta\zeta_{\text{bound,init}}$	0.005
mode set $\mathcal{P}_{\text{OWS}}$	$\{\text{PIPM}, \text{PPM}\}$	disturbance range D_r	$(0.15\text{m}, 0.3\text{m/s})$
OWS state space Ξ_p	$[-0.1\text{m}, 0.7\text{m}] \times [0.1\text{m/s}, 1.8\text{m/s}]$	control space $\mathcal{U}_{\text{OWS}}$	[2rad/s, 4rad/s]



(a) 50 simulated trajectories using the synthesized controller.
(b) Simulation success rate under various granularities and disturbances.

Fig. 20. Success rate of the simulations under varying granularities and disturbances. In subfigure (a), the system is subjected to disturbances bounded by $D_r = (0.15\text{m}; 0.3\text{m/s})$. All the 50 simulation trails can reach the goal robustness margin set successfully. In subfigure (b), we run 1000 trials for each case with a specific granularity and a bounded disturbance. The disturbance exerted in the simulation remains the same, i.e., $D_r = (0.1\text{m}, 0.2\text{m/s})$.

robust set through 50 simulation tests under different granularities and bounded disturbances. In Fig. 20(a), each trial is run for the one walking step with the PIPM-PPM mode pair. The exerted disturbance in the simulation is the same as the one used in the controller synthesis process, i.e., $D_r = (0.15\text{m}, 0.3\text{m/s})$. As shown in Fig. 20(a), all the trials reach the final robustness margin successfully. This agrees with the correctness guarantee by the one-walking-step robust reachability property of Theorem 1.

We evaluate the effect of the discretization granularity and the magnitude of disturbances used in the controller synthesis process as shown in Fig. 20(b). Given each controller synthesized using a specific granularity and for a specific disturbance bound, we simulate 1000 trials with the bounded disturbance $D_r = (0.1\text{m}, 0.2\text{m/s})$. Fig. 20(b) shows four sets of simulation results for different granularities ranging from 0.002 to 0.005. For each set of simulations, the success rate increases as the modeled disturbance in the controller synthesis increases, and it reaches 100% when the modeled disturbance matches the actual disturbance D_r used in the simulation. This is consistent with our expectation. Let us inspect the figure from another perspective. If we compare the results for different granularities with a specific disturbance set D_i $(i = 0, 1, 2, 3, 4)$, the success rate almost remains the same. This is because when constructing the abstraction for the robust reachability analysis, we have taken into consideration the effects of approximation errors caused by different discretization granularities, by using non-deterministic transitions that over-approximate the dynamics of the system. In addition, we observe that the success rates for all the synthesized controllers are greater than 97%, even in the case no disturbance is considered in the controller synthesis. This can again be interpreted by the over-approximation used in the abstraction. Nonetheless, as shown in the simulations, to achieve 100% correctness guarantee, the modeled disturbance has to be larger than (or at least match) the actual disturbance in the simulation. Moreover, under the same disturbance D_r, the nominal phase-space planner with a fixed open-loop control input only achieves a success rate of 29%. This huge discrepancy in success rate clearly shows the advantage of using an abstraction-based feedback controller over an open-loop phase-space planner.

7.5 Case V: Integrated multi-step locomotion via the reachability control library

This case evaluates an integrated multi-step locomotion example with the robust finite transition system $\mathcal{TS}_{\text{OWS}}$, the inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$, and the replanning strategy. Assume that the decision of the task planner renders a locomotion mode sequence involving the PIPM, PPM, and MCM modes below,

$$\text{PIPM} \to \text{PIPM} \to \text{PPM} \to \text{PIPM} \to \text{MCM} \to \text{PIPM} \to \text{PIPM}$$

To enable the initial and final keyframe robustness margin sets to cover a sufficiently larger phase space, we extend the default 3×3 keyframe grid to a 5×5 keyframe grid for each mode. This allows the reachability controllers to be applicable to a larger set of keyframe states. For each locomotion mode pair, we synthesize all the feasible controllers that reach the final keyframe robustness margin set under a bounded disturbance. We enumerate all the combinations of the allowable locomotion mode pairs and generate all the reachability control policies offline. These controllers are saved as a control library and are executed at runtime according to the high-level decision and measured states under bounded disturbances.

Parameters of constructing the inter-sampling finite abstraction $\mathcal{TS}_{\text{OWS,INT}}$ are defined as follows. The controller synthesis and execution process use the same disturbance bound $D_r = (0.05\text{m}; 0.1\text{m/s})$. The full discretized state space is $\Xi_{\text{full}} = [-0.2\text{m}, 3.8\text{m}] \times [0.2\text{m/s}, 1.9\text{m/s}]$ with a granularity $(0.003\text{m}, 0.003\text{m/s})$. The local state space of each walking step is chosen so that it is sufficiently large to cover the space around the two keyframe states. A time step $\delta\zeta = 0.02\text{s}$ is used for the abstraction construction of each walking step. The control inputs for PIPM, PPM and MCM satisfy $\omega_{\text{PIPM}} \in [2, 4]$, $\omega_{\text{PPM}} \in [2, 4]$ and $\omega_{\text{MCM}} \in [1, 3]$. We obtain the sets of sampled control values by a granularity of 0.02. The robustness margins of the phase space manifolds are $\delta\sigma_{\text{PIPM}} = 0.002$, $\delta\zeta_{\text{PIPM}} = 0.002$; $\delta\sigma_{\text{PPM}} = 0.04$, $\delta\zeta_{\text{PPM}} = 0.003$; $\delta\sigma_{\text{MCM}} = 0.15$, $\delta\zeta_{\text{MCM}} = 0.9 \times 10^{-5}$.

The computational time for constructing abstractions is around 30s on average, and 5s to 15s for synthesizing a reachability controller corresponding to each keyframe pair, depending on the number of states and transitions of the abstraction. Since we synthesize 625 (i.e., 25×25) reachability controllers for each walking step, the time to generate them is approximately 90 mins. In our simulation of six consecutive walking steps, all the local reachability control strategies are patched together to cover the overall state space. The time for simulating a single closed-loop walking trajectory is around 2s. As the results show in Fig. 21, we simulate six different trials with different initial conditions, i.e., starting from different initial robustness margin sets. Each locomotion trajectory is guaranteed to reach one of the robustness margin sets at the next walking step via using the reachability controller from the control library. In particular, a trial is tested to evaluate the replanning strategy when the state is perturbed out of the winning set.

8 Discussions and Future Work

8.1 Low-level uncertainties

This paper proposes a hierarchical approach to the task and motion planning of dynamic locomotion in complex environments. We achieve robustness against a general, bounded disturbance by synthesizing a middle-layer robust reachability controller with robustness margins to accommodate low-level uncertainties. Undoubtedly, a variety of low-level uncertainties can come from time delays, actuation limits, unmodeled dynamics, state estimation, and measurement error from the environment. These uncertainties severely deteriorate the execution success rate of the high-level planner, in particular when the robot performs highly agile motions in complex and unstructured environments. In addition, the abstraction methods can induce approximation errors between the high-level and low-level planners. Although not directly dealing with these

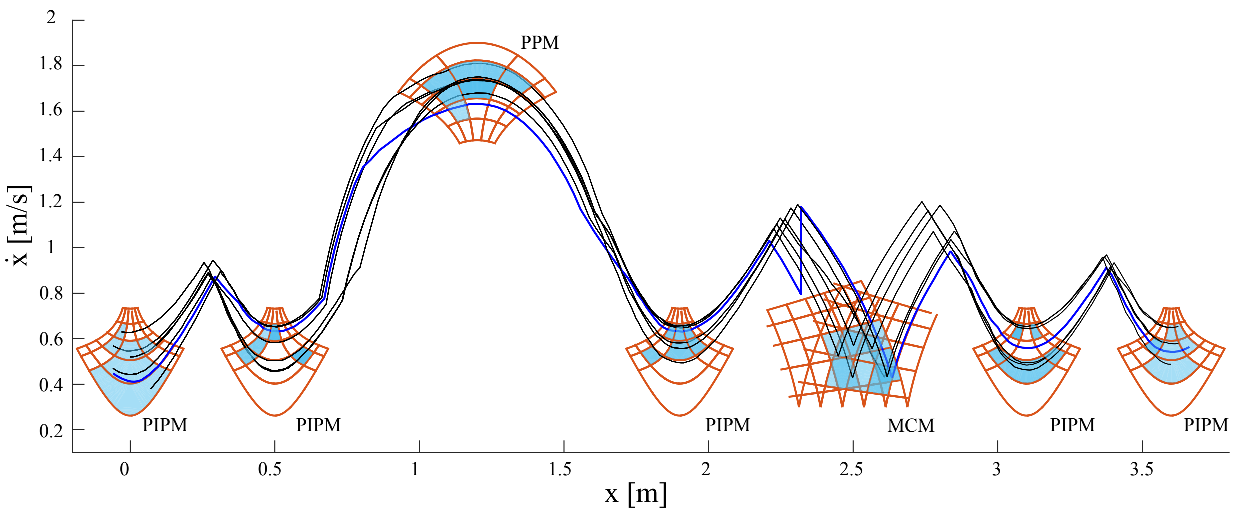


Fig. 21. Integrated phase-space trajectories of multi-walking step simulations under bounded disturbances. The replanning strategy is evaluated with a trial (see the blue trajectory) exerted with a velocity disturbance larger than the modeled disturbance in the MCM mode (around the position $x = 2.3$ m). In this case, the state is perturbed out of the winning set of the currently used reachability controller. A replanning signal is triggered, and the planner searches within the control library for a new winning set (together with a new reachability controller) that covers the perturbed state. Then the perturbed state will use that new controller to reach a new robustness margin set for continuous locomotion maneuvering.

low-level uncertainties and abstraction approximation errors, the keyframe-based robustness margin proposed in this paper can be viewed as an abstract representation of these uncertainties in the center-of-mass (CoM) state space. As long as a mapping can be established between these low-level uncertainties and the CoM phase-space deviations from the nominal trajectory, these uncertainties can be handled indirectly by the proposed reachability controller at run-time. Additionally, a replanning strategy is designed to handle large uncertainties that are not explicitly modeled in the reachability controller. In the future, abstraction refinement [Nilsson and Ozay (2014)] will be inspirational for designing a model abstraction with a proper granularity. More importantly, implementing the proposed high-level decision-making algorithms in the dynamic simulation and real hardware [Kim et al. (2016); Luo et al. (2017)], and evaluating the robustness performance against low-level uncertainties will be our main upcoming task.

8.2 *System and environment assumption relaxation*

To make the proposed hierarchical planning approach applicable to locomotion tasks in more complex and cluttered environments, it is important to relax the assumptions and approximations of the environment and model more realistic scenes. For instance, how to formally design recovery strategies for slippery terrains (i.e., with friction coefficient inaccuracies), large tilting angles, and swing foot obstacle collision is a practically meaningful topic.

Our current planner assumes that all limb contacts switch synchronously. To relax this conservative assumption, we will explore the asynchronous contact switching strategy in the future. This relaxation opens up the opportunity for designing more natural and diverse locomotion contact behaviors. From a more general perspective, contact actions and keyframe states may exhibit probabilistic features. Incorporating probabilistic models, such as Markov decision process (MDP) [Platt et al. (2004); Fu and Topcu (2014); Feng et al. (2015)], into the high-level decisions will be a promising direction. Accordingly, studying probabilistic correctness and completeness will be of our interest.

8.3 Generalization to complex tasks

Generalizing the proposed planning framework to more realistic locomotion tasks is of practical importance, in particular when robots are unleashed into the real world. Some more practical locomotion tasks include walking while carrying a payload, walking alongside human teammates, dynamically interacting with a human during motion [Alonso-Mora et al. (2018)], and multi-robot coordination [da Silva et al. (2016)]. To this end, how to design an automatic method for generating locomotion primitives of diverse tasks becomes important. Also, allocating computing resources efficiently among different planning layers is an essential topic. A mission planner will be needed to operate at a more abstract level to make decisions on task allocation, coordination, and synchronization. A key problem is how to properly design integrated, scalable, and reactive mission and motion planners [da Silva et al. (2016)] for legged robots to accomplish collaborative tasks in dynamic and unstructured environments.

At the individual robot level, our motion planner is designed for the three-dimensional case, although the demonstrated locomotion tasks are primarily straight walking. In the future, we will incorporate steering models [Zhao et al. (2017)] such that the locomotion behaviors are extendable to complex 3D motions with steering capabilities. An advantage of our planning framework is to use simplified models which allow us to efficiently compose multiple locomotion modes and achieve dynamic and complex locomotion behaviors in constrained environments. The high-level symbolic planner automates this sequential composition process and guarantees the formal correctness of the overall planning framework.

An application of the proposed whole-body dynamic locomotion methodology in the constrained environment is the following: The US Defense Advanced Research Projects Agency (DARPA) created a Subterranean Challenge [DARPA (2018)] aiming at augmenting underground operation capabilities. *"The Challenge aims to explore new approaches to rapidly map, navigate, and search underground environments ... and propose novel methods for tackling time-critical scenarios through unknown courses in mapping subsurface networks and unpredictable conditions, which are too hazardous for human first responders"*. Our proposed hierarchical decision-making approach for whole-body dynamic locomotion in constrained environments raise the importance of decision-making algorithms with formal guarantees for robots as complex as humanoid robots, a research topic of increasing importance as these robots begin to move out of the laboratories and work outdoors.

8.4 Planning horizon

Making planner decisions with a sufficiently long predictive horizon has great potential to enable intelligent and robust behaviors in complex and dynamically changing environments [Egerstedt et al. (2018)]. Our task planner has a one-walking-step horizon and may sometimes result in myopic locomotion decisions. For instance, if the disturbance is so large that the robot can not recover within one walking step, our planner will execute a replanning process. However, a natural alternative is to design a recovery strategy over the next multiple walking steps, which is commonly used in the recovery process of human locomotion. The downside of this strategy is the increase in computational complexity. Our planning process substantially relieves this computational burden by using the simplified locomotion models. In addition to this computational consideration, the design of the planning horizon should take into account the versatility of the locomotion behaviors. For instance, if the locomotion process is of high speed, being able to make predictions over a longer horizon will be advantageous. Overall, we should take into account the computational power and behavior versatility when designing the planning horizon. Algorithm 2 is designed in a general form and should be extendable to the multiple-walking-step scenario.

9 Conclusions

This paper employs temporal-logic-based formal methods to synthesize a high-level reactive task planner and designs a middle-level discrete control to achieve the one-walking-step robust reachability process for complex whole-body dynamic locomotion (WBDL) behaviors in constrained environments. A particular focus has been given to (i) the robustness of the keyframe state reachability with respect to bounded disturbances; (ii) the correctness of the top-down hierarchy from the high-level task planner to the low-level motion planner processes.

A diverse set of locomotion models are devised at the low-level to form a template library in response to various environmental events, including those adversarial ones such as cracked terrain, human appearance, and narrow passage. These adversarial events require specifically-designed locomotion modes to enable desired locomotion behaviors. Deviating from numerous existing studies primarily using a single simplified model for a specific locomotion task, our symbolic task planner focuses on integrating and unifying a variety of simplified models and achieves complex locomotion behaviors via sequential composition of trajectories. A key novelty of this task planner lies in solving the traditional contact planning problem via a two-player game. Contact decisions are determined according to the synthesized switching protocol in response to possibly-adversarial environment actions.

As for the reachability control under disturbances, we propose a robust metric of the keyframe state and use it to design a robust finite transition system realized by the underlying reachability synthesis. The proposed task and motion planner is validated through simulations of WBDL maneuvers in constrained environments. The performance of the reachability control is benchmarked via a series of synthesis and execution tests. We expect that this line of work acts as an entry point for the locomotion community to employ formal methods to verify and synthesize planners and controllers in legged and humanoid robots [Kuindersma et al. (2016); Hereid et al. (2016); Ramezani et al. (2014)]. Evaluating the proposed framework on dynamic bipedal robots is one of our high-priority future works.

Funding

This work was partially supported by the NSF Grant [#1924978, #1724360, #1652113], Office of Naval Research (ONR) Grant [grant #N000141512507], and partially supported by NSERC of Canada, the Canada Research Chairs program, and an Ontario MRIS Early Researcher Award.

Appendix

A Index to Multimedia Extensions

Extension	Type	Description
1	Video	Reactive task and motion planning for whole-body dynamic locomotion

B Linear temporal logic

Linear temporal logic is an extension of propositional logic that incorporate temporal operators. An LTL formula φ is composed of atomic propositions $\pi \in AP$. The generic form of a LTL formula has the following grammar,

$$\varphi ::= \pi \;\Big|\; \neg\varphi \;\Big|\; \varphi_1 \wedge \varphi_2 \;\Big|\; \varphi_1 \vee \varphi_2 \;\Big|\; \bigcirc\,\varphi \;\Big|\; \varphi_1 \mathcal{U} \varphi_2,$$

where the Boolean constants true and false are expressed by $\mathrm{false} = \neg\mathrm{true}$ and $\mathrm{true} = \varphi \vee \neg\varphi$, and we have the temporal operators $\bigcirc$ (“next"), $\mathcal{U}$ (“until"), $\neg$ (“negation") and $\wedge$ (“conjunction"). We can also define $\vee$ (“disjunction"), $\Rightarrow$ (“implication"), $\Leftrightarrow$ (“equivalence"). Another two key operators in LTL are $\diamond$ (“Eventually") and $\square$ (“Always"). We can interpret them $\diamond\varphi := \mathrm{true}\,\mathcal{U}\varphi$ for “Eventually" and $\square\varphi := \neg\diamond\neg\varphi$ for “Always".

LTL formulae are interpreted over an infinite sequence of states $\boldsymbol{q}$. We define $\boldsymbol{q}_i = q_i q_{i+1} q_{i+2} \ldots$ as a run from i^{th} position. It is said that a LTL formula φ holds at i^{th} position of $\boldsymbol{q}$, represented as $q_i \models \varphi$, if and only if φ holds for the

Algorithm 4 Execution of the synthesized controller based on robust finite transition system $\mathcal{TS}_{\text{OWS}}$

```
1: procedure ExecuteOWSReachabilityControl(keyframeState $\boldsymbol{q}_{\text{initial}}$, $\boldsymbol{q}_{\text{final}}$, locomotionMode $p_1$, $p_2$, contactConfiguration $s_1, s_2$, initialState $\boldsymbol{\xi}_{\text{init}}$, transitionTime $\zeta_{\text{trans}}$, environmentActionOfNextOWS $e_{\text{next-OWS}}$)
2: choose winning sets of the first and second semi-steps $\mathcal{WIN}_{p_1}$ and $\mathcal{WIN}_{p_2}$ via $(p_1, p_2)$, $(s_1, s_2)$, and $(\boldsymbol{q}_{\text{initial}}, \boldsymbol{q}_{\text{final}})$.
3: set initial and final robustness margins $(\delta\zeta_{\text{bound,init}}, \delta\sigma_{\text{bound,init}})$ and $(\delta\zeta_{\text{bound,final}}, \delta\sigma_{\text{bound,final}})$
4: set the intermediate robustness margin set $\mathcal{R}_{\text{inter}} = \{\boldsymbol{\xi}' : |\mathcal{Z}_{p_1,\sigma}(\boldsymbol{\xi}')| \leq \delta\sigma_{\text{bound,init}} \wedge |\mathcal{Z}_{p_2,\sigma}(\boldsymbol{\xi}')| \leq \delta\sigma_{\text{bound,final}}\}$.
5: initialize CoM state $\boldsymbol{\xi}_{\text{current}} \leftarrow \boldsymbol{\xi}_{\text{init}}$.
6: isEnvironmentAbruptChange ← **false**
7: **while** $\boldsymbol{\xi}_{\text{current}} \notin \mathcal{R}_{\text{inter}}$ **do**
8:   isEnvironmentAbruptChange ← detectEnvironmentAbruptChange($e_{\text{next-OWS}}$)
9:   **if** $\boldsymbol{\xi}_{\text{current}} \in \mathcal{WIN}_{p_1}$ **and** isEnvironmentAbruptChange == **false then**
10:     assign control input $u_{p_1}$ from $\mathcal{WIN}_{p_1}$ based on current state $\boldsymbol{\xi}_{\text{current}}$
11:   **else if** $\boldsymbol{\xi}_{\text{current}} \notin \mathcal{WIN}_{p_1}$ **and** isEnvironmentAbruptChange == **false then**
12:     (isAlternativeWinningSetFeasible, $\mathcal{WIN}_{\text{alternative}}$) ← detectAlternativeWinningSet($p_1$, $\mathcal{TS}_{\text{OWS}}$)
13:     **if** isAlternativeWinningSetFeasible == **true then**
14:       $\mathcal{WIN}_{p_1} \leftarrow \mathcal{WIN}_{\text{alternative}}$
15:     **else**
16:       **failure** occurs, **exit**, **replanning**, and **move** to Line 2          {no alternative winning set}
17:     **end if**
18:   **else if** isEnvironmentAbruptChange == **true then**
19:     **failure** occurs, **exit**, **replanning**, and **move** to Line 2          {environment abrupt change}
20:   **end if**
21:   send control command $u_{p_1}$ to robot for execution.
22:   measure the actual state $\boldsymbol{\xi}_{\text{next}}$ at next time step
23:   $\boldsymbol{\xi}_{\text{current}} \leftarrow \boldsymbol{\xi}_{\text{next}}, \zeta \leftarrow \zeta + \delta\zeta_{p_1}$ $\boldsymbol{\chi}_{\text{current}} \leftarrow$ generateLimbTraj($s_{p_1}, \boldsymbol{\xi}_{\text{current}}$)
24: **end while**
25: **while** $\boldsymbol{\xi}_{\text{current}} \in \mathcal{R}_{\text{inter}}$ and $\zeta < \zeta_{\text{trans}}$ **do**
26:   send control command $u_{p_1}$ to robot/simulator for execution.
27:   measured actual state at next time step $\boldsymbol{\xi}_{\text{next}}$
28:   $\boldsymbol{\xi}_{\text{current}} \leftarrow \boldsymbol{\xi}_{\text{next}}, \zeta \leftarrow \zeta + \delta\zeta_{p_1}, \boldsymbol{\chi}_{\text{current}} \leftarrow$ generateLimbTraj($s_{p_1}, \boldsymbol{\xi}_{\text{current}}$)
29: **end while**
30: $(\sigma, \zeta) \leftarrow \mathcal{Z}_{p_2}(\boldsymbol{\zeta}_{\text{current}})$
31: **while** $|\zeta - \zeta_{p_2}| > \delta\zeta_{\text{bound,final}}$ **or** $|\sigma - \sigma_{p_2}| > \delta\sigma_{\text{bound,final}}$ **do**
32:   **if** $\boldsymbol{\xi}_{\text{current}} \in \mathcal{WIN}_{p_2}$ **then**
33:     assign control input $u_{p_2}$ from $\mathcal{WIN}_{p_2}$ based on current state $\boldsymbol{\xi}_{\text{current}}$
34:   **else if** $\boldsymbol{\xi}_{\text{current}} \notin \mathcal{WIN}_{p_2}$ **then**
35:     (isAlternativeWinningSetFeasible, $\mathcal{WIN}_{\text{alternative}}$) ← detectAlternativeWinningSet($p_2$, $\mathcal{TS}_{\text{OWS}}$)
36:     **if** isAlternativeWinningSetFeasible == **true then**
37:       $\mathcal{WIN}_{p_2} \leftarrow \mathcal{WIN}_{\text{alternative}}$
38:     **else**
39:       **failure** occurs, **exit**, **replanning**, **select** a new feasible goal set from the task planner, and **move** to Line 30
40:                                                                        {no alternative winning set}
41:     **end if**
42:   **end if**
43:   send control command $u_{p_2}$ to the robot/simulator for execution.
44:   measure actual state at next time step $\boldsymbol{\xi}_{\text{next}}$
45:   $\boldsymbol{\xi}_{\text{current}} \leftarrow \boldsymbol{\xi}_{\text{next}}, \zeta \leftarrow \zeta + \delta\zeta_{p_2}, \boldsymbol{\chi}_{\text{current}} \leftarrow$ generateLimbTraj($s_{p_2}, \boldsymbol{\xi}_{\text{current}}$)
46:   $(\sigma, \zeta) \leftarrow \mathcal{Z}_{p_2}(\boldsymbol{\zeta}_{\text{current}})$.
47: **end while**
48: **return** a sequence of $(\boldsymbol{\xi}_{\text{current}}, \boldsymbol{\chi}_{\text{current}})$.
```

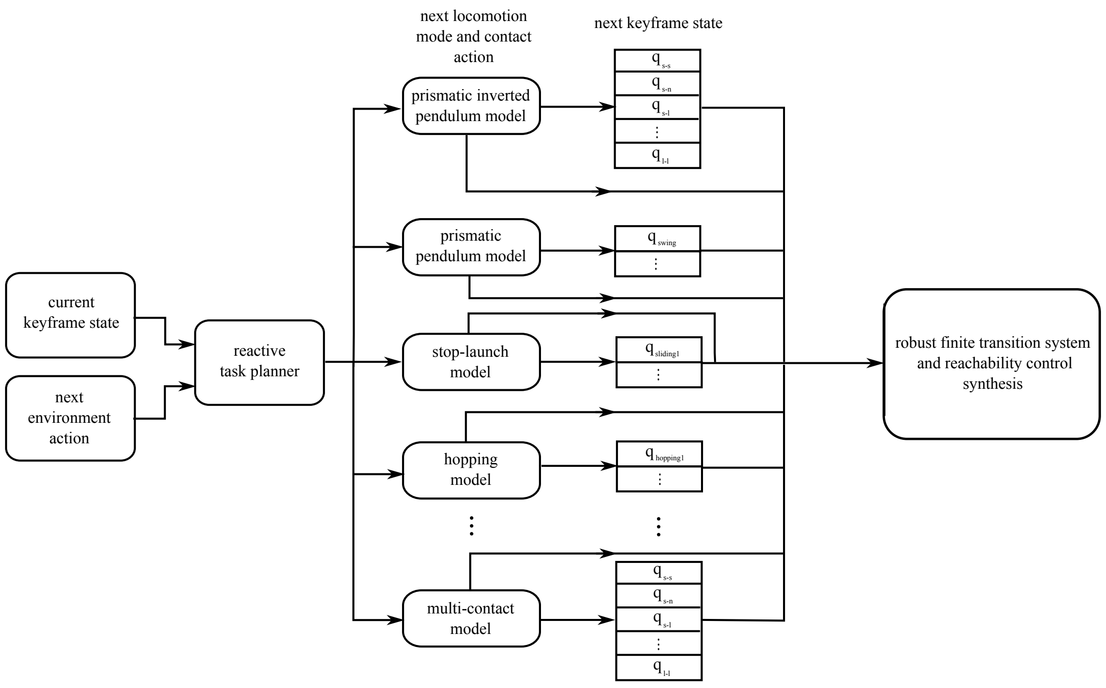


Fig. 22. An illustration of the top-down decision sequence of the high-level reactive task planner and middle-level reachability controller synthesis. It illustrates the relationship between the keyframe state, environment action, and system mode.

remaining sequence of $\boldsymbol{q}$ starting at i^{th} position. The semantics of LTL are defined inductively as

$$\begin{aligned}
q_i &\models \neg\varphi \text{ iff } q_i \not\models \varphi \\
q_i &\models \varphi_1 \wedge \varphi_2 \text{ iff } q_i \models \varphi_1 \wedge q_i \models \varphi_2 \\
q_i &\models \varphi_1 \vee \varphi_2 \text{ iff } q_i \models \varphi_1 \vee q_i \models \varphi_2 \\
q_i &\models \bigcirc\varphi \text{ iff } q_{i+1} \models \varphi \\
q_i &\models \varphi_1 \mathcal{U} \varphi_2 \text{ iff } \exists j \geq i, \text{s.t.} q_j \models \varphi_2 \\
&\qquad \text{and } q_k \models \varphi_1, \forall i \leq k \leq j
\end{aligned}$$

In these definitions, the notation $\bigcirc\varphi$ represents that φ is true at the next “step" (i.e., next position in the sequence), $\square\varphi$ represents φ is always true (i.e., true at every position of the sequence), $\diamond\varphi$ represents that φ is eventually true at some position of the sequence, $\square\diamond\varphi$ represents that φ is true infinitely often (i.e., eventually become true starting from any position), and $\diamond\square\varphi$ represents that φ is eventually always true (i.e., always becomes true after some point in time in the sequence) [Baier and Katoen (2008)].

C Phase-Space Manifold

Closed-form solutions of the phase-space manifolds are required to define the robustness margin sets in Def. 10. Besides the ones proposed for the prismatic inverted pendulum model (PIPM) in Propositions 2 and 3, this subsection proposes the closed-form solutions of additional locomotion modes, including the prismatic pendulum model (PPM) and the multi-contact model (MCM) as defined in Section 3.1.

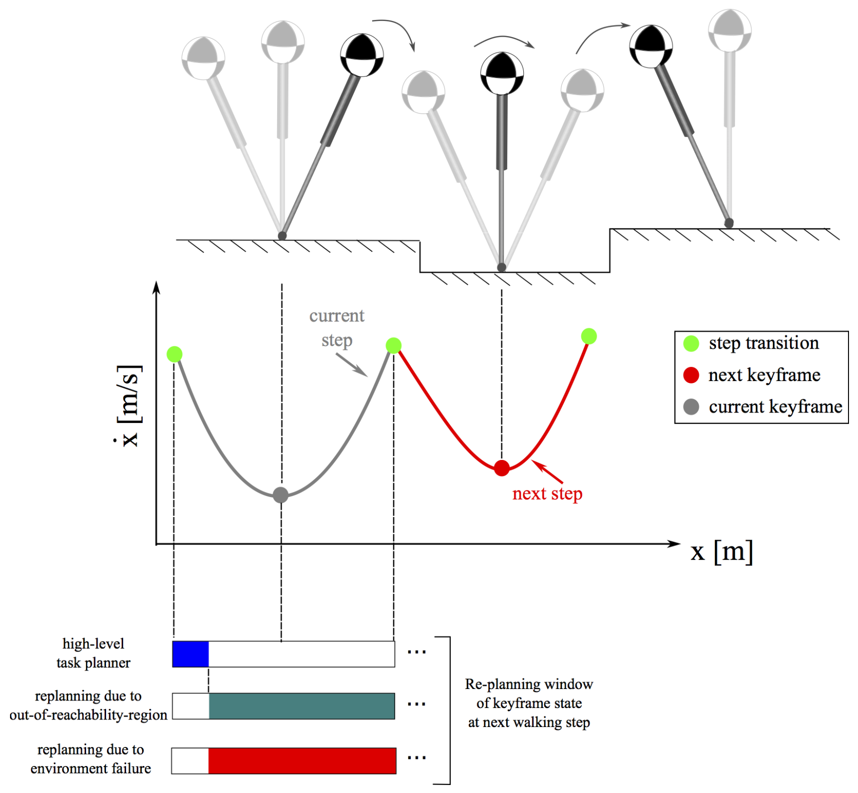


Fig. 23. Replanning timing for the next walking step. The high-level task planning for the next walking step is determined at the beginning of one walking step. Then during the remaining time of the current walking step (before switching to the next walking step), a replanning process can be triggered anytime if the state is out of the reachability region (i.e., the winning set) or the environment action change suddenly. This figure uses single-contact prismatic inverted pendulum model for illustration.

Proposition 4 (PPM phase-space tangent manifold). *Given the PPM of Eq. (44) with initial conditions* $(x_0, \dot{x}_0) = (x_{\rm foot}, \dot{x}_{\rm apex})$ *and known arm placement* $x_{\rm foot}$*, the PPM phase-space tangent manifold is defined as*

$$\sigma(x, \dot{x}, \dot{x}_{\rm apex}, x_{\rm foot}) = \frac{\dot{x}^2_{\rm apex}}{-\omega^2_{\rm PPM}}\big(\dot{x}^2 - \dot{x}^2_{\rm apex} + \omega^2_{\rm PPM}(x - x_{\rm foot})^2\big), \tag{45}$$

Compared to the PIPM tangent manifold in Proposition 2, the PPM tangent manifold has a negative asymptote slope square, i.e., $-\omega^2_{\rm PPM}$. Thus, the tangent manifold with $\sigma > 0$ locates beneath the nominal $\sigma = 0$ tangent manifold. This property is in contrast to that of the PIPM tangent manifold.

Proposition 5 (PPM phase-space cotangent manifold). *Given the PPM of Eq. (44), the PPM cotangent manifold is*

$$\zeta = \zeta_0\big(\frac{\dot{x}}{\dot{x}_0}\big)^{-\omega^2_{\rm PPM}}\frac{x - x_{\rm foot}}{x_0 - x_{\rm foot}}, \tag{46}$$

Proposition 6 (MCM phase-space tangent manifold). *Given the multi-contact model with a constant acceleration* $\omega_{\rm MCM}$ *(i.e., the control input), an initial condition* $(x_0, \dot{x}_0) = (x_{\rm foot}, \dot{x}_{\rm apex})$*, and a known foot placement* $x_{\rm foot}$*, the MCM phase-space tangent manifold is*

$$\sigma(x, \dot{x}, x_{\rm foot}, \dot{x}_{\rm apex}) = 2\omega_{\rm MCM}(x - x_{\rm apex}) - (\dot{x}^2 - \dot{x}^2_{\rm apex}), \tag{47}$$

where $\sigma = 0$ *represents the nominal phase-space tangent manifold.*

Proposition 7 (MCM phase-space cotangent manifold). *Given the multi-contact model with a constant acceleration and initial conditions* $(x_0, \dot{x}_0) = (x_{\text{foot}}, \dot{x}_{\text{apex}})$ *and known foot placement* x_{foot}*, the phase-space cotangent manifold is*

$$\zeta(x, \dot{x}, x_{\text{foot}}, \dot{x}_{\text{apex}}) = \omega_{\text{MCM}} \cdot \ln(\frac{\dot{x}}{\dot{x}_{\text{apex}}}) - (x - x_{\text{foot}}), \tag{48}$$

Again, for all the manifolds above, $\sigma = 0$ represents the nominal phase-space tangent manifold. The phase-space manifolds of the hopping model are trivial since its tangent phase-space manifold is a horizontal line. The stop-launch model and sliding model have similar phase-space manifolds (i.e., parabolic trajectories) as those of the multi-contact model since all of them has a constant sagittal acceleration. Their derivations are omitted for brevity.

References

Alexander, R. M. (1984). The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research 3*(2), 49–59.

Alonso-Mora, J., J. A. DeCastro, V. Raman, D. Rus, and H. Kress-Gazit (2018). Reactive mission and motion planning with deadlock resolution avoiding dynamic obstacles. *Autonomous Robots 42*(4), 801–824.

Alur, R., T. A. Henzinger, G. Lafferriere, and G. J. Pappas (2000). Discrete abstractions of hybrid systems. *Proceedings of the IEEE 88*(7), 971–984.

Ames, A. D., P. Tabuada, B. Schürmann, W.-L. Ma, S. Kolathaya, M. Rungger, and J. W. Grizzle (2015). First steps toward formal controller synthesis for bipedal robots. In *International Conference on Hybrid Systems: Computation and Control*, pp. 209–218.

Antoniotti, M. and B. Mishra (1995). Discrete event models + temporal logic = supervisory controller: Automatic synthesis of locomotion controllers. In *IEEE-RAS International Conference on Robotics and Automation*, Volume 2, pp. 1441–1446.

Arslan, O. and U. Saranli (2012). Reactive planning and control of planar spring–mass running on rough terrain. *Robotics, IEEE Transactions on 28*(3), 567–579.

Audren, H., J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida (2014). Model preview control in multi-contact motion-application to a humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4030–4035.

Baier, C. and J.-P. Katoen (2008). *Principles of model checking*. MIT press Cambridge.

Belta, C., B. Yordanov, and E. A. Gol (2017). *Formal methods for discrete-time dynamical systems*, Volume 89. Springer.

Bertram, J., A. Ruina, C. Cannon, Y. H. Chang, and M. J. Coleman (1999). A point-mass model of gibbon locomotion. *Journal of Experimental Biology 202*(19), 2609–2617.

Bhatia, A., L. E. Kavraki, and M. Y. Vardi (2010). Motion planning with hybrid dynamics and temporal goals. In *IEEE Conference on Decision and Control*, pp. 1108–1115.

Bloem, R., B. Jobstmann, N. Piterman, A. Pnueli, and Y. Saar (2012). Synthesis of reactive (1) designs. *Journal of Computer and System Sciences 78*(3), 911–938.

Bouyarmane, K. and A. Kheddar (2011). Multi-contact stances planning for multiple agents. In *IEEE International Conference on Robotics and Automation*, pp. 5246–5253.

Bretl, T. (2006). Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem. *The International Journal of Robotics Research 25*(4), 317–342.

Burridge, R. R., A. A. Rizzi, and D. E. Koditschek (1999). Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research 18*(6), 534–555.

Campbell, M., M. Egerstedt, J. P. How, and R. M. Murray (2010). Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 368*(1928), 4649–4672.

Caron, S. and A. Kheddar (2016). Multi-contact walking pattern generation based on model preview control of 3d com accelerations. In *IEEE-RAS International Conference on Humanoid Robots*, pp. 550–557.

Caron, S., Q.-C. Pham, and Y. Nakamura (2015). Zmp support areas for multi-contact mobility under frictional constraints. *arXiv preprint arXiv:1510.03232*.

Chinchali, S., S. C. Livingston, U. Topcu, J. W. Burdick, and R. M. Murray (2012). Towards formal synthesis of reactive controllers for dexterous robotic manipulation. In *IEEE-RAS International Conference on Robotics and Automation*, pp. 5183–5189.

Chung, S.-Y. and O. Khatib (2015). Contact-consistent elastic strips for multi-contact locomotion planning of humanoid robots. In *IEEE-RAS International Conference on Robotics and Automation*, pp. 6289–6294.

da Silva, R. R., B. Wu, J. Dai, and H. Lin (2016). Combined top-down and bottom-up approaches to performance-guaranteed integrated task and motion planning of cooperative multi-agent systems. *arXiv preprint arXiv:1607.07797*.

Dantam, N. T., Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki. An incremental constraint-based framework for task and motion planning. *The International Journal of Robotics Research, doi: 10.1177/0278364918761570*.

Dantam, N. T., Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki (2016). Incremental task and motion planning: A constraint-based approach. In *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan.

DARPA (2018). Darpa subterranean challenge, https://www.darpa.mil/program/darpa-subterranean-challenge.

De, A. and D. E. Koditschek (2015). The penn jerboa: A platform for exploring parallel composition of templates. *arXiv preprint arXiv:1502.05347*.

DeCastro, J. A., J. Alonso-Mora, V. Raman, D. Rus, and H. Kress-Gazit (2015). Collision-free reactive mission and motion planning for multi-robot systems. In *International Symposium on Robotics Research*.

DeCastro, J. A. and H. Kress-Gazit (2015). Synthesis of nonlinear continuous controllers for verifiably correct high-level, reactive behaviors. *The International Journal of Robotics Research 34*(3), 378–394.

Deshmukh, J. V., A. Donzé, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia (2015). Robust online monitoring of signal temporal logic. In *Runtime Verification*, pp. 55–70. Springer.

Donzé, A. and O. Maler (2010). Robust satisfaction of temporal logic over real-valued signals. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pp. 92–106. Springer.

Duperret, J. and D. E. Koditschek (2020). Towards reactive control of transitional legged robot maneuvers. In *Robotics Research*, pp. 145–162. Springer.

Egerstedt, M., J. N. Pauli, G. Notomista, and S. Hutchinson (2018). Robot ecology: Constraint-based control design for long duration autonomy. *Annual Reviews in Control*.

Englsberger, J., C. Ott, and A. Albu-Schaffer (2015). Three-dimensional bipedal walking control based on divergent component of motion. *IEEE Transactions on Robotics 31*(2), 355–368.

Fainekos, G. E. and G. J. Pappas (2009). Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science 410*(42), 4262–4291.

Farahani, S. S., V. Raman, and R. M. Murray (2015). Robust model predictive control for signal temporal logic synthesis. *IFAC-PapersOnLine 48*(27), 323–328.

Feng, L., C. Wiltsche, L. Humphrey, and U. Topcu (2015). Controller synthesis for autonomous systems interacting with human operators. In *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems*, pp. 70–79. ACM.

Fu, J. and U. Topcu (2014). Probably approximately correct mdp learning and control with temporal logic constraints. In *Proceedings of Robotics: Science and Systems*, Berkeley, CA.

Fu, J. and U. Topcu (2016). Synthesis of joint control and active sensing strategies under temporal logic constraints. *IEEE Transactions on Automatic Control 61*(11), 3464–3476.

Full, R. J. and D. E. Koditschek (1999). Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of experimental biology 202*(23), 3325–3332.

Gu, Z., N. Boyd, and Y. Zhao (2021). Reactive locomotion decision-making and robust motion planning for real-time perturbation recovery. *arXiv preprint arXiv:2110.03037*.

Hauser, K. (2014). Fast interpolation and time-optimization with contact. *The International Journal of Robotics Research 33*(9), 1231–1250.

Hauser, K., T. Bretl, and J.-C. Latombe (2005). Non-gaited humanoid locomotion planning. In *IEEE-RAS International Conference on Humanoid Robots*, pp. 7–12.

He, K., M. Lahijanian, L. E. Kavraki, and M. Y. Vardi (2015). Towards manipulation planning with temporal logic specifications. In *IEEE-RAS International Conference on Robotics and Automation*, pp. 346–352.

He, K., M. Lahijanian, L. E. Kavraki, and M. Y. Vardi (2017). Reactive synthesis for finite tasks under resource constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5326–5332.

Hereid, A., E. A. Cousineau, C. M. Hubicki, and A. D. Ames (2016). 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In *IEEE International Conference on Robotics and Automation*, pp. 1447–1454.

Jaulin, L. (2001). *Applied Interval Analysis: with Examples in Parameter and State Estimation, Robust Control and Robotics*, Volume 1. Springer Science & Business Media.

Kaelbling, L. P. and T. Lozano-Pérez (2011). Hierarchical task and motion planning in the now. In *IEEE International Conference on Robotics and Automation*, pp. 1470–1477.

Kajita, S., F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa (2001). The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Volume 1, pp. 239–246. Ieee.

Kim, D., Y. Zhao, G. Thomas, B. R. Fernandez, and L. Sentis (2016). Stabilizing series-elastic point-foot bipeds using whole-body operational space control. *IEEE Transactions on Robotics 32*(6), 1362–1379.

Kloetzer, M. and C. Belta (2010). Automatic deployment of distributed teams of robots from temporal logic motion specifications. *IEEE Transactions on Robotics 26*(1), 48–61.

Kress-Gazit, H., G. E. Fainekos, and G. J. Pappas (2009). Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics 25*(6), 1370–1381.

Kress-Gazit, H., T. Wongpiromsarn, and U. Topcu (2011). Correct, reactive, high-level robot control. *IEEE Robotics & Automation Magazine 18*(3), 65–74.

Kudruss, M., M. Naveau, O. Stasse, N. Mansard, C. Kirches, P. Soueres, and K. Mombaur (2015). Optimal control for whole-body motion generation using center-of-mass dynamics for predefined multi-contact configurations.

Kuindersma, S., R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake (2016). Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots 40*(3), 429–455.

Kulgod, S., W. Chen, J. Huang, Y. Zhao, and N. Atanasov (2020). Temporal logic guided locomotion planning and control in cluttered environments. In *2020 American Control Conference (ACC)*, pp. 5425–5432. IEEE.

Li, Y. and J. Liu (2018). ROCS: A robustly complete control synthesis tool for nonlinear dynamical systems. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 130–135.

Liberzon, D. (2012). *Switching in systems and control*. Springer Science & Business Media.

Liu, J. (2017). Robust abstractions for control synthesis: completeness via robustness for linear-time properties. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 101–110. ACM.

Liu, J. and N. Ozay (2014). Abstraction, discretization, and robustness in temporal logic control of dynamical systems. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 293–302.

Liu, J. and N. Ozay (2016). Finite abstractions with robustness margins for temporal logic-based control synthesis. *Nonlinear Analysis: Hybrid Systems 22*, 1–15.

Liu, J., N. Ozay, U. Topcu, and R. M. Murray (2013). Synthesis of reactive switching protocols from temporal logic specifications. *IEEE Transactions Automatic Control 58*(7), 1771–1785.

Liu, J., U. Topcu, N. Ozay, and R. M. Murray (2012). Synthesis of reactive control protocols for differentially flat systems. In *IEEE Conference on Decision and Control*.

Luo, J., Y. Zhao, D. Kim, O. Khatib, and L. Sentis (2017). Locomotion control of three dimensional passive-foot biped robot based on whole body operational space framework. *IEEE International Conference on Robotics and Biomimetics 26*, 28.

Majumdar, R., E. Render, and P. Tabuada (2011). Robust discrete synthesis against unspecified disturbances. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 211–220. ACM.

Maniatopoulos, S., P. Schillinger, V. Pong, D. C. Conner, and H. Kress-Gazit (2016). Reactive high-level behavior synthesis for an atlas humanoid robot. In *IEEE-RAS International Conference on Robotics and Automation*, pp. 4192–4199.

Nilsson, P. and N. Ozay (2014). Incremental synthesis of switching protocols via abstraction refinement. In *IEEE Conference on Decision and Control*, pp. 6246–6253.

Peng, X. B., P. Abbeel, S. Levine, and M. van de Panne (2018). Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *arXiv preprint arXiv:1804.02717*.

Piovan, G. and K. Byl (2015). Reachability-based control for the active slip model. *The International Journal of Robotics Research 34*(3), 270–287.

Plaku, E., L. E. Kavraki, and M. Y. Vardi (2010). Motion planning with dynamics by a synergistic combination of layers of planning. *IEEE Transactions on Robotics 26*(3), 469–482.

Platt, R., A. H. Fagg, and R. A. Grupen (2004). Manipulation gaits: Sequences of grasp control tasks. In *IEEE International Conference on Robotics and Automation*, Volume 1, pp. 801–806.

Posa, M., S. Kuindersma, and R. Tedrake (2016). Optimization and stabilization of trajectories for constrained dynamical systems. In *IEEE International Conference on Robotics and Automation*, pp. 1366–1373.

Raibert, M. H. (1986). Legged robots that balance. *MIT press*.

Raman, V., A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia (2015). Reactive synthesis from signal temporal logic specifications. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 239–248. ACM.

Ramezani, A., J. W. Hurst, K. A. Hamed, and J. Grizzle (2014). Performance analysis and feedback control of atrias, a three-dimensional bipedal robot. *Journal of Dynamic Systems, Measurement, and Control 136*(2), 021012.

Sadigh, D. and A. Kapoor (2016). Safe control under uncertainty with probabilistic signal temporal logic. In *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan.

Sadraddini, S. and C. Belta (2015). Robust temporal logic model predictive control. In *Annual Allerton Conference on Communication, Control, and Computing*, pp. 772–779.

Sentis, L., J. Park, and O. Khatib (2010a). Compliant control of multi-contact and center of mass behaviors in humanoid robots. *IEEE Transactions on Robotics 26*(3), 483–501.

Sentis, L., J. Park, and O. Khatib (2010b). Compliant control of multicontact and center-of-mass behaviors in humanoid robots. *IEEE Transactions on Robotics 26*(3), 483–501.

Sharan, R. (2014). *Formal methods for control synthesis in partially observed environments: application to autonomous robotic manipulation*. Ph. D. thesis, California Institute of Technology.

Sreenath, K., C. R. Hill Jr, and V. Kumar (2013). A partially observable hybrid system model for bipedal locomotion for adapting to terrain variations. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 137–142.

Srivastava, S., E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel (2014). Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE International Conference on Robotics and Automation*, pp. 639–646.

Tabuada, P. (2009). *Verification and control of hybrid systems: a symbolic approach*. Springer Science & Business Media.

Topcu, U., N. Ozay, J. Liu, and R. M. Murray (2012). On synthesizing robust discrete controllers under modeling uncertainty. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 85–94.

Toussaint, M., K. Allen, K. Smith, and J. Tenenbaum (2018). Differentiable physics and stable modes for tool-use and manipulation planning. *Proceedings of Robotics: Science and Systems, Pittsburgh, PA*.

Warnke, J., A. Shamsah, Y. Li, and Y. Zhao (2020). Towards safe locomotion navigation in partially observable environments with uneven terrain. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 958–965. IEEE.

Wongpiromsarn, T., U. Topcu, and R. M. Murray (2012). Receding horizon temporal logic planning. *IEEE Transactions Automatic Control 57*(11), 2817–2830.

Wongpiromsarn, T., U. Topcu, N. Ozay, H. Xu, and R. M. Murray (2011). Tulip: a software toolbox for receding horizon temporal logic planning. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pp. 313–314.

Xu, X., J. W. Grizzle, P. Tabuada, and A. D. Ames (2018). Correctness guarantees for the composition of lane keeping and adaptive cruise control. *IEEE Transactions on Automation Science and Engineering 15*(3), 1216–1229.

Zhao, Y., B. R. Fernandez, and L. Sentis (2016). Robust phase-space planning for agile legged locomotion over various terrain topologies. In *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan.

Zhao, Y., B. R. Fernandez, and L. Sentis (2017). Robust optimal planning and control of non-periodic bipedal locomotion with a centroidal momentum model. *The International Journal of Robotics Research 36*(11), 1211–1242.

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

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

Zhao, Z., Z. Zhou, M. Park, and Y. Zhao (2021). Sydebo: Symbolic-decision-embedded bilevel optimization for long-horizon manipulation in dynamic environments. *IEEE Access 9*, 128817–128826.