Motion Planning for Legged Robots on Varied Terrain

Kris Hauser

Department of Computer Science
Stanford University
Stanford, CA 94305-5447, USA
khauser@cs.stanford.edu

Timothy Bretl

University of Illinois at Urbana-Champaign,
Urbana, IL 61801-2935, USA
tbretl@illinois.edu

Jean-Claude Latombe

Department of Computer Science
Stanford University
Stanford, CA 94305-5447, USA
latombe@cs.stanford.edu

Kensuke Harada

Humanoid Research Group
Intelligent Systems Research Institute
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki 305-8568, Japan
kensuke.harada@aist.go.jp

Brian Wilcox

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
Brian.H.Wilcox@jpl.nasa.gov

Abstract

In this paper we study the quasi-static motion of large legged robots that have many degrees of freedom. While gaited walking may suffice on easy ground, rough and steep terrain requires unique sequences of footsteps and postural adjustments specifically adapted to the terrain's local geometric and physical properties. In this paper we present a planner that computes these motions by combining graph searching to generate a sequence of candidate footfalls with probabilistic sample-based planning to generate continuous motions that reach these footfalls. To improve motion quality, the probabilistic planner derives its sampling strategy from a small set of motion primitives that have been generated offline. The viability of this approach is demonstrated in simulation for the six-legged Lunar vehicle ATHLETE and the humanoid HRP-2 on several example terrains, including one that requires both hand and foot contacts and another that requires rappelling.

KEY WORDS—Motion planning, legged robots, humanoids, probabilistic sample-based planning, motion primitives

1. Introduction

One of the main potential advantages of legged robots over other types of mobile robots (such as wheeled and track robots) is their mechanical ability to navigate on varied terrain. However, so far this ability has not been exploited fully. One

The International Journal of Robotics Research
Vol. 27, No. 11–12, November/December 2008, pp. 1325–1349
DOI: 10.1177/0278364908098447

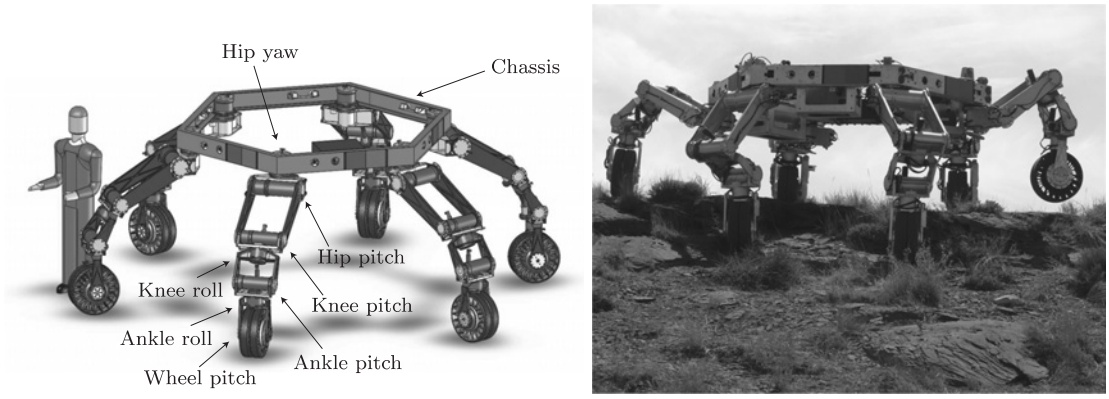


Fig. 1. The ATHLETE Lunar vehicle (Wilcox et al. 2007).

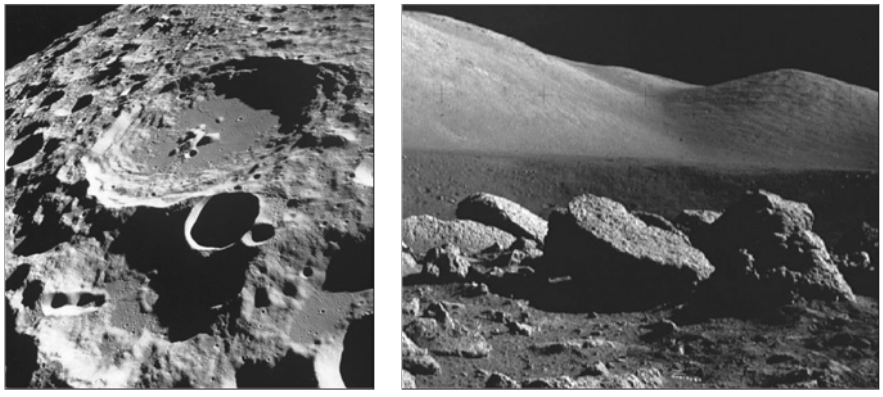


Fig. 2. Pictures of Lunar terrain from the Apollo missions (Heiken et al. 1991).

reason is the lack of an adequate motion planner capable of computing sequences of footsteps and postural adjustments specifically adapted to the local geometric and physical properties of the terrain. In this paper we describe the design and implementation of a motion planner that enables legged robots with many degrees of freedom (DOFs) to navigate safely across varied terrain. Although most of this planner is general, our presentation focuses on its application to two robots: the six-legged Lunar vehicle ATHLETE (Wilcox et al. 2007) and the humanoid HRP-2 (Kaneko et al. 2004).

1.1. ATHLETE and HRP-2

ATHLETE (shown in Figure 1) is large and highly mobile. A half-scale Earth test model has a diameter of 2.75 m and mass of 850 kg. It can roll at up to 10 km h^{-1} on rotating wheels over flat smooth terrain and walk carefully on fixed wheels over irregular and steep terrain. With its six articulated legs, ATHLETE is designed to scramble across terrain so rough that a fixed gait (for example, an alternating tripod gait) may prove insufficient. Such terrain is abundant on the Moon, most of which is rough, mountainous and heavily cratered—particularly in the polar regions, a likely target for future surface operations. These craters can be of enormous size and are filled with scattered rocks and boulders of a few centimeters to several meters in diameter (Figure 2). Crater walls are sloped at angles of between 10° and 45°, and sometimes have sharp rims (Heiken et al. 1991).

In comparison, HRP-2 (Figure 3) is relatively light and compact, with height 1.54m and mass 58kg, and is capable of moving at up to 2.5km/h. Using a fixed gait, it can walk on flat surfaces, along narrow paths, and up stairs of constant height. It has also been demonstrated crawling through tunnels, climbing stairs while holding onto handrails, and getting up after falling down (Harada et al. 2004; Kaneko et al. 2004) by performing specific motions that are carefully handcrafted through a long trial-and-error process. Like ATHLETE, HRP-2 is potentially capable of scrambling across rougher terrain, such as irregular outdoor terrain or urban rubble, where a fixed gait may be insufficient. An adequate planner is needed to make this capability a reality.

Fig. 3. The humanoid HRP-2 (Kaneko et al. 2004) and an example of varied terrain.

1.2. Other Robots and Applications

A variety of previous works have also focused on enabling robots to traverse irregular terrain. Locomotion of humanoids across somewhat uneven terrain has been studied by Kuffner et al. (2003) and Zheng and Shen (1990). Other legged robots, including quadrupeds (Hirose and Kunieda 1991), hexapods (Song and Waldron 1989; Krotkov and Simmons 1996), parallel walkers (Yoneda et al. 1999) and spherically symmetric robots (Pai et al. 1995), are capable of walking across rougher terrain. Wheeled robots with active or rocker-bogie suspension can also traverse rough terrain by changing wheel angles and the position of the center of mass (CM) (Iagnemma et al. 1999; Estier et al. 2000; Lauria et al. 2002). Careful descent is possible by rappelling as well, using either legs (Wettergreen et al. 1993; Hirose et al. 1997; Bares and Wettergreen 1999) or wheels (Mumm et al. 2004). In most cases, however, these robots perform tediously hand-crafted or teleoperated motions. The terrain we consider for ATHLETE and HRP-2 is also more irregular and steep than in most previous applications, although not as steep as for free-climbing robots (Bretl 2006).

Careful walking also resembles dexterous manipulation. Both ATHLETE and HRP-2 grasp the terrain in the same way a hand grasps an object, placing and removing footfalls rather than finger contacts. Legged robots have to remain in equilibrium as they move (only the object must remain in equilibrium during manipulation), and use fewer contact modes while walking (no sliding or rolling), but still face similar challenges (Bicchi and Kumar 2000; Okamura et al. 2000). Manipulation planning, involving the rearrangement of many objects with a simple manipulator, is another related application. A manipulator takes a sequence of motions with and without a grasped object (different states of contact) just like a legged robot takes a sequence of steps (Alami et al. 1995). In fact, for a legged robot to navigate among movable obstacles, it may be necessary to consider both walking and manipulation together (Stilman and Kuffner 2006).

1.3. Motion Planning for Legged Robots

On rough terrain, the walking motion of legged robots such as ATHLETE and HRP-2 is governed largely by two interdependent constraints: *contact*, that is, keep feet, fixed wheels or other body parts (such as hands or knees) at a carefully chosen set of footfalls (contacts); and *equilibrium*, that is, apply forces at these footfalls that exactly compensate for gravity without causing slippage. The range of forces that may be applied at the footfalls without causing slippage depends on their geometry (e.g. average slope) and their physical properties (e.g. coefficient of friction), both of which vary across the terrain. So every time the robot plans a step, it faces a dilemma: it cannot know the constraints on its subsequent motion until it chooses a footfall, a choice that itself depends on the constraints.

To handle this dilemma, we make a key design choice similar to that introduced by Bretl (2006) and Hauser et al. (2005) (Section 2): *to choose footfalls before computing motions*. We begin by identifying a number of potentially useful footfalls across the terrain. Each mapping of a robot's feet (or other allowed body parts) to a set of footfalls is a *stance*. Associated with this stance is a (possibly empty) set of feasible configurations that satisfy all motion constraints (including contact and equilibrium). A robot can take a step from one stance to another if they differ by a single footfall and if they share some feasible configuration, which we call a *transition*. Our planner proceeds in two stages: first, we generate a candidate sequence of footfalls by finding transitions between stances; then, we expand this sequence into a feasible, continuous trajectory by finding paths between subsequent transitions. This two-stage planning approach is motivated by the fact that a legged robot's motion on irregular and steep terrain is most constrained just as it places a foot at or removes a foot from a footfall (more generally, when it makes a new contact or breaks one). At this instant, the robot must be able to reach the footfall (the contact constraint) but cannot use it to avoid falling (the equilibrium constraint), since the applied force is zero at the instant of time when the contact is made or broken. So transitions are the "bottlenecks" of any motion: if we can find two subsequent transitions, it is likely we can find a path between them. This statement has been verified in our experiments.

Like the planners of Bretl (2006) and Hauser et al. (2005), the planner described in this paper combines graph searching to generate a sequence of candidate footfalls with a probabilistic roadmap (PRM) approach (see Chapter 7 of Choset et al. (2005)) to generate continuous motions that reach these footfalls. However, we add two key algorithmic tools in this framework (Section 3) to deal specifically with difficult computational issues raised by legged robots such as ATHLETE and HRP-2. One is a method of sampling feasible configurations

and of connecting pairs of configurations with local paths. This method addresses the challenge that these robots have many DOFs with terrain contacts that close many kinematic chains. While the closure constraint reduces the robot's feasible space at a given stance to a submanifold of the robot's configuration space, other constraints (such as equilibrium) restrict the feasible space further to a subset of small volume inside that manifold. Hence, sampling feasible configurations and connecting them with feasible paths is particularly difficult and potentially time consuming. The other tool is a heuristic to generate footfalls and guide our search through the highly combinatorial collection of stances. This heuristic addresses the challenge that moving across varied, but not extreme, terrain requires footfalls to be properly selected, even though the number of candidate stances is enormous.

1.3.1. Previous Planners for Legged Robots

Motion planning for legged robots requires computing both a sequence of footfalls and continuous motions that reach these footfalls. Previous planners differ primarily in which part of the problem they consider first:

- *Motion before footfalls.* When it does not matter where a robot contacts its environment, it makes sense to compute the robot's (or object's) overall motion first. For example, a manipulation planner might generate a trajectory for the grasped object ignoring manipulators, then compute manipulator trajectories that achieve necessary re-grasps (Koga and Latombe 1994). Similarly, a humanoid planner might generate a two-dimensional collision-free path of a bounding cylinder, then follow this path with a fixed, pre-defined gait (Kuffner 1999; Pettré et al. 2003). A related strategy is to plan a path for the robot's CM, then to compute footfalls and limb motions that keep the CM stable along this path (Eldershaw and Yim 2001). Some planners even avoid reasoning about footfalls entirely (Lee et al. 2006). These techniques are fast, but do not extend well to irregular and steep terrain.
- *Footfalls before motion.* When the choice of contact location is critical, it makes sense to compute a sequence of footfalls first. This approach is related to the work on manipulation planning presented by Alami et al. (1995), which expresses connectivity between different states of contact as a graph. For "spider-robots" with zero-mass legs walking on horizontal terrain, the exact structure of this graph can be computed quickly using analytical techniques (Boissonnat et al. 2000). For more general systems, the graph can sometimes be simplified by assuming partial gaits, for example restricting the order in which limbs are moved (Shapiro and Rimon 2003) or restricting footsteps to a discrete set (Kuffner et al. 2003). However, when motion is distinctly non-gaited (as in manipulation planning (Nielsen and Kavraki 2000; Sahbani et al. 2002), free-climbing (Bretl 2006), or for ATHLETE and HRP-2 on varied terrain), each step requires the exploration of a distinct configuration space. This fact motivates the two-stage search strategy we adopt in Section 2.

1.4. Improving Motion Quality

Without additional considerations for motion quality, the approach outlined above often generates motion that looks unnatural and inefficient. The reason is that, while robots such as ATHLETE and HRP-2 have many DOFs, we do not know in advance which of these DOFs are actually useful, or how many contacts may be needed. In some cases, there might exist too many feasible motions. On easy terrain such as flat ground or stairs of constant height, the motion of a legged robot is lightly constrained, so that most of its DOFs are unnecessary, and only feet need contact the ground. For example, although crawling would also be feasible for HRP-2 on flat ground, we would rather see the humanoid walk upright. Alternatively, on hard terrain such as steep rock or urban rubble, the robot's motion is highly constrained. In this case, most of its DOFs are essential and additional contacts (hands, knees, shoulders) might be required for balance. On varied terrain between these two extremes, the number of relevant DOFs and the types of required contacts may change from step to step. Since our basic planner always considers all DOFs (in order to find a feasible motion whenever one exists) it may then generate needless motions of arms or other DOFs that are not required for balance, or that may achieve balance in clearly sub-optimal ways. Eliminating such motions in post-processing is particularly hard. A better approach is to take motion quality into consideration during planning.

To solve this problem, we provide our planner with a small library of high-quality *motion primitives*, in a similar manner to Kuffner et al. (2003) and Yamane et al. (2004). These primitives might include a step on flat ground, a step up a staircase, or a step on sloped terrain with a hand contact on a rock for balance. Such primitives may be designed by hand, produced by off-line pre-computation (for instance, using optimization techniques), or extracted from captured motions of humans or animals. We record each motion primitive as a nominal path through the robot's configuration space. Then, instead of sampling across all of configuration space to find paths between stances, our planner samples a growing distribution of configurations around the nominal path associated with a chosen motion primitive. Our simulation results demonstrate not only a marked increase in motion quality[1] for legged robots

1. Exactly how motion quality should be measured is an open question, and is beyond the scope of this paper. Here, we define quality as inversely proportional to a linear combination of path length and sum-squared distance from an upright posture.

walking on varied terrain, but also a reduction in planning time. In the absence of a relevant primitive, the planner falls back on its general sampling method.

1.4.1. Other Ways to Improve Motion Quality

The most common way to improve motion quality is to post-process feasible motions using methods such as "short-cut" heuristics (Kavraki et al. 1996; Song et al. 2001) and gradient descent algorithms (Geraerts and Overmars 2004; Vougioukas 2005).. We also use similar methods in our planner, but, as mentioned above, for legged robots it is difficult to eliminate all needless motions in post-processing. For this reason, motion primitives and other types of maneuvers have been applied widely to legged robots and other vehicles with complex dynamics, as well as to digital animation of virtual actors. Four general strategies have been used.

- *Record and playback.* This strategy restricts motion to a library of maneuvers. Natural-looking humanoid locomotion on mostly flat ground can be planned as a sequence of pre-computed feasible steps (Kuffner et al. 2003). Robust helicopter flight can be planned as a sequence of feedfoward control strategies (learned by observing skilled human operators) to move between trim states (Gavrilets et al. 2001; Frazzoli et al. 2002b,a; Ng et al. 2004). Robotic juggling can be planned as a sequence of feedback control strategies (Burridge et al. 1999). The motion of peg-climbing robots can be planned as a sequence of actions such as "grab the nearest peg" (Bevly et al. 2000). In these applications, a reasonably small library of maneuvers is sufficient to achieve most desired motions. For legged robots on varied terrain, such a library may grow to impractical size.

- *Warp, blend or transform.* Widely used for digital animation, this strategy also restricts motion to a library of maneuvers, but allows these maneuvers to be superimposed or transformed to better fit the task at hand. For example, captured motions of human actors can be "warped" to allow characters to reach different footfalls (Witkin and Popović 1995) or "retargetted" to control characters of different morphologies (Gleicher 1998). The resulting motion is not guaranteed to satisfy all physical constraints, although techniques have been proposed that maintain some of these constraints (Popović and Witkin 1999; Shin et al. 2001).

- *Model reduction.* This motion-before-footfalls strategy first plans an overall motion in a configuration space of reduced dimension; then, it follows this motion with a concatenation of primitives. One way to generate natural-looking humanoid locomotion on flat ground is to approximate the robot as a cylinder, plan a two-dimensional collision-free path of this cylinder, and track this path with a fixed gait (Kuffner 1999; Kovar et al. 2002; Pettr e et al. 2003; Kron and Shin 2005). A similar method is used to plan the motion of non-holonomic wheeled vehicles (Laumond 1987; Laumond et al. 1994). A related strategy plans the motion of key points on a robot or digital actor (such as the CM or related ground reference points (Popovic et al. 2005)), tracking these points with an operational space controller (Sentis and Khatib 2005). These approaches work well only when it does not matter much where a robot or digital actor contacts its environment.

- *Bias inverse kinematic solutions.* Like model reduction, this strategy first plans the motion of key points on a robot or digital actor, such as the location of hands or feet. However, instead of a fixed controller, a search algorithm is used to compute a pose of the robot or actor at each instant that tracks these points by selecting an inverse kinematic solution. One approach is to choose an inverse kinematic solution according to a probability density function learned from high-quality example motions (Grochow et al. 2004; Yamane et al. 2004; Liu et al. 2005; Meredith and Maddock 2005). The set of examples gives the resulting pose a particular "style". We take a similar approach in this paper, planning steps for a legged robot by sampling waypoints in a growing distribution around high-quality nominal paths.

1.5. Limitations

Our work still has many limitations. In particular, we do not consider dynamic equilibrium, closed-loop control, visual feedback, robustness to modeling errors, error recovery or deformable terrain. We briefly address some of these limitations in the conclusion. For example, the use of motion primitives to generate more natural motions could be extended to handle dynamic constraints. Some robustness to modeling errors can be achieved by requiring that the robot's CM stays well within the support polygon. Deformable terrain would require a more general contact model than pure frictional contact. Despite such current limitations, we believe that the planning methods presented in this paper can be useful in practice.

1.6. Organization of the Paper

Section 2 presents our footfalls-before-motion planning approach. Section 3 describes more specific algorithmic tools that are designed to deal with the difficult computational issues raised by robots such as ATHLETE and HRP-2. Section 4 discusses the generation, use and selection of motion primitives to help plan higher-quality motion. Section 5 presents simulation results for both ATHLETE and HRP-2, demonstrating that

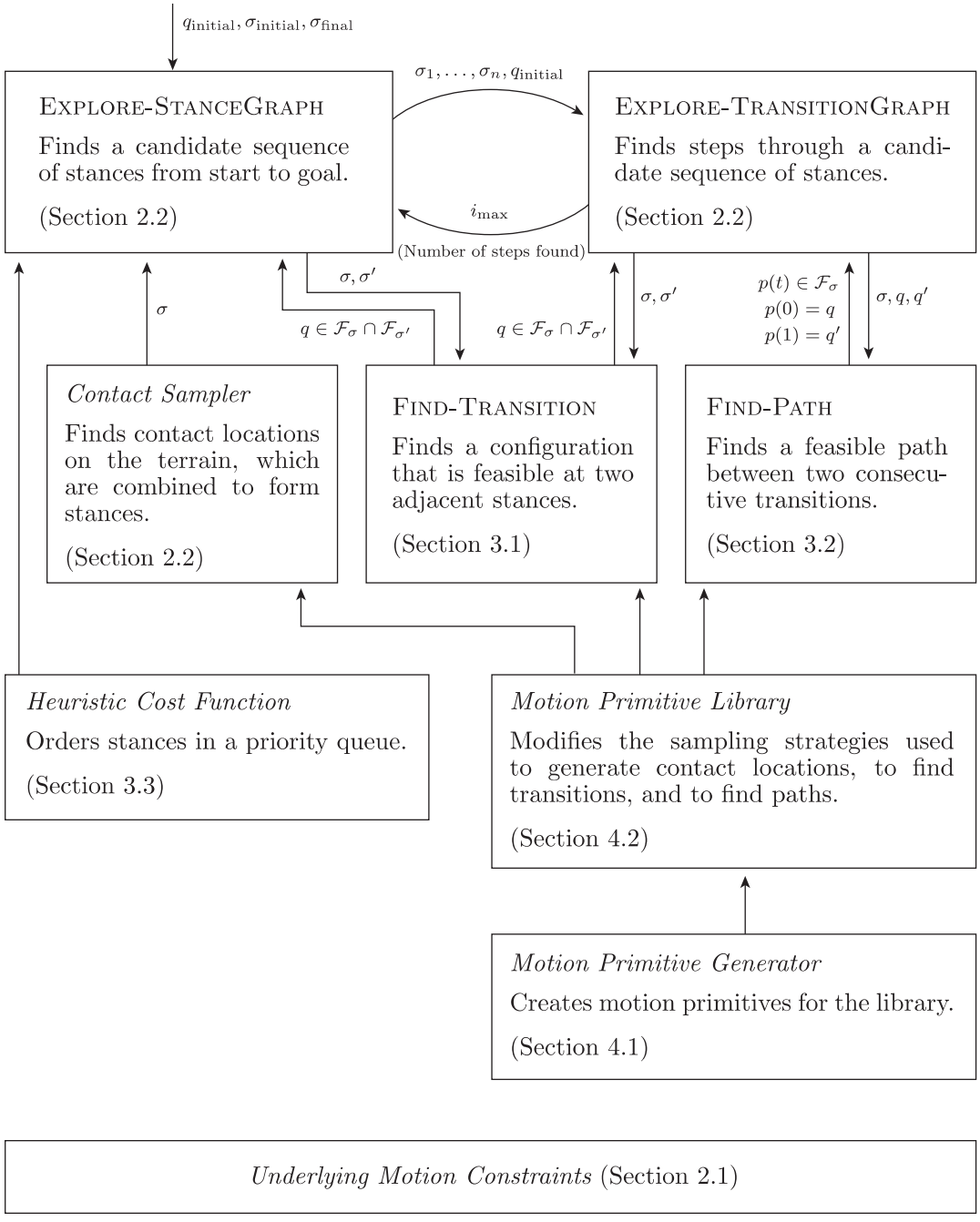


Fig. 4. Overall structure of the motion planner.

our planner enables these robots to traverse terrain where fixed gaits would fail. These results also show that the use of motion primitives makes it possible to plan motions of significantly higher quality, particularly when these primitives are also used to select contacts with the terrain. All computation times reported in this paper were obtained by running our software on a 1.8 GHz PC.

2. Design of the Motion Planner

Our planner extends a similar approach for humanoid robots (Hauser et al. 2005), which was based on earlier work for a free-climbing robot (Bretl 2006). Figure 4 shows the overall structure of this planner. Here, we summarize our basic approach.

2.1. Motion constraints

A *configuration* of a robot, here either ATHLETE or HRP-2, is a parameterization q of the robot's placement in three-dimensional space. For ATHLETE, q consists of six parameters defining the position and orientation of the hexagonal chassis and a list of 36 joint angles (since each leg has six actuated, revolute joints). For HRP-2, q consists of six parameters defining the position and orientation of the torso and a list of 30 joint angles. The set of all such q is the *configuration space*, denoted $\mathcal{Q}$, of dimensionality 42 for ATHLETE and 36 for HRP-2.

We consider terrain that may include an arbitrary mixture of flat, sloped or irregular ground. We assume that this terrain and all robot links are perfectly rigid. We also assume that we are given a set of robot links in advance that are allowed to touch the terrain. For ATHLETE, this set includes only its six wheels, to which brakes are applied (to prevent rolling) while the robot is walking. For HRP-2, this set may include the hands and the knees, in addition to the feet. We call the placement of a link on the terrain a *contact*, and fix the position and orientation of the link while the contact is maintained. We call a set of simultaneous contacts a *stance*, denoted by σ. Consider a stance σ with $N \geq 1$ contacts. The *feasible space* $\mathcal{F}_\sigma$ is the set of all feasible configurations of the robot at σ. To be in $\mathcal{F}_\sigma$, a configuration q must satisfy several constraints.

- *Contact.* The N contacts form a linkage with multiple closed-loop chains. So, q must satisfy inverse kinematic equations. Let $\mathcal{Q}_\sigma \subset \mathcal{Q}$ be the set of all configurations q that satisfy these equations. This set $\mathcal{Q}_\sigma$ is a submanifold of $\mathcal{Q}$ of dimensionality $42-6N$ for ATHLETE and $36-6N$ for HRP-2, which we call the *stance manifold*. This manifold is empty if it is impossible for the robot to achieve the contacts specified by σ, for example if two contact points are farther apart than the maximum span of the robot.

- *Static equilibrium.* To remain balanced, both ATHLETE and HRP-2 must apply forces at contacts in σ that compensate for gravity without slipping. For valid forces to exist, the robot's CM must lie above its *support polygon*. However, on irregular and sloped terrain, the support polygon does not always correspond to the base of the robot's feet. For example, both ATHLETE and HRP-2 will slip off a flat and featureless slope that is too steep, regardless of their CM position. To compute the support polygon, we model each contact as a frictional point in the case of ATHLETE and a set of frictional points (the vertices of the convex hull of the contact area) in the case of HRP-2. Let $r_1, \ldots, r_n \in \mathbb{R}^3$ denote the position of all points modeling the N contacts. Similarly, let $\nu_i \in \mathbb{R}^3$, $\mu_i \geq 0$, and $f_i \in \mathbb{R}^3$ for $i = 1, \ldots, n$, be the normal vector, the static coefficient of friction, and the reaction force acting on the robot at each point, respectively. We decompose each force f_i into a component $\nu_i^T f_i \nu_i$ normal to the terrain surface and a component $(I - \nu_i \nu_i^T) f_i$ tangential to the surface. Let $c \in \mathbb{R}^3$ be the position of the robot's CM (which varies with the configuration q). Let m be the robot's mass and $g \in \mathbb{R}^3$ be the acceleration due to gravity. All vectors are defined with respect to a global coordinate system with axes e_1, e_2, e_3, where $g = -\|g\| e_3$. Then the robot is in static equilibrium if:

$$\sum_{i=1}^{n} f_i + mg = 0 \quad \text{(force balance)}, \tag{1}$$

$$\sum_{i=1}^{n} r_i \times f_i + c \times mg = 0 \quad \text{(torque balance)}, \tag{2}$$

$$\left\|(I - \nu_i \nu_i^T) f_i\right\|_2 \leq \mu_i \nu_i^T f_i$$

$$\text{for all } i = 1, \ldots, n \quad \text{(friction cones)}. \tag{3}$$

These constraints are jointly convex in $f_1, \ldots, f_n$ and c. In particular, Equations (1) and (2) are linear and Equation (3) is a second-order cone constraint. The set of jointly feasible contact forces and CM positions is a high-dimensional convex set (Bretl et al. 2003; Bretl 2006; Bretl and Lall 2006, 2008). Since

$$c \times mg = m \|g\| \begin{bmatrix} -c \cdot e_2 \\ c \cdot e_1 \\ 0 \end{bmatrix},$$

Equations (1) and (2) do not depend on $c \cdot e_3$ (the CM coordinate parallel to gravity), so the support polygon is the projection of this convex set onto the coordinates e_1, e_2. One way to compute this projection and to test the membership of c is described by Bretl and Lall (2008) and Bretl and Lall (2006).

- *Joint torque limits.* The above equilibrium test assumes the robot is a rigid body, "frozen" at configuration q. In reality, to stay in this configuration each joint must exert a torque, which in turn must not exceed a given bound. Let τ be the vector of all joint torques exerted by the robot. These torques must satisfy

$$\tau = G(q) - \sum_{i=1}^{n} J_i(q)^T f_i, \tag{4}$$

where $G(q)$ is the generalized gravity vector and $J_i(q)$ is the Jacobian of the point on the robot touching r_i. Let $\|\cdot\|_\infty$ be a scaled L_∞ norm where $\|\tau\|_\infty < 1$ implies that each joint torque is within bounds. We check joint torque

limits by solving a second-order cone program (Boyd and Vandenberghe 2004) to compute contact forces that satisfy Equation (1)–(4) with minimum $\|\tau\|_\infty$ and verify $\|\tau\|_\infty < 1$.

- *Collision.* In addition to satisfying joint angle limits, the robot must avoid collision with the environment (except at contact points) and with itself. We use techniques based on pre-computed bounding volume hierarchies to perform collision checking, in a similar way to Gottschalk et al. (1996) and Schwarzer et al. (2002).

The static equilibrium and torque limit conditions, as we state them above, are only necessary for stability. Additional assumptions, for example, resolving contact force indeterminacy, are required for these conditions to be sufficient (Pang and Trinkle 2000; Greenfield et al. 2005; Rimon et al. 2006).

2.2. Two-stage Search

Both ATHLETE and HRP-2 move from one place to another by taking a sequence of *steps*. Each step is a continuous motion at a fixed stance that ends by making or breaking a single contact to reach a new stance. Suppose that the robot begins a step at configuration $q \in \mathcal{F}_\sigma$ at stance σ. Let σ' be a stance *adjacent* to σ, that is, that either adds one contact to σ or removes one contact from σ. A single step from σ to σ' is a feasible path in $\mathcal{F}_\sigma$ from the initial configuration q to some final configuration $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$, which we call a *transition*, that is feasible at both σ and σ'.

As an example, consider HRP-2 in Figure 7(a). The stance σ contains one contact, corresponding to the fixed placement of the robot's right foot. The adjacent stance σ' contains two contacts, the second corresponding to the fixed placement of its left foot. The initial configuration $q \in \mathcal{F}_\sigma$ is shown in the first frame of Figure 7(a); the transition $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ is shown in the last frame. At the transition, the robot can reach both contacts in σ', but can support itself only with forces applied at the contact in σ. The step is the path between q and q' shown in the intermediate frames of Figure 7(a).

The planner samples contact locations on the terrain either at random or using a distribution taking into account the terrain's local geometry, in a similar manner to Chestnutt et al. (2003). These contact locations, along with the given set of robot links that are allowed to touch the terrain, then determine the set of all possible stances (usually a huge set) that may be considered by the planner. We say that two adjacent stances in this set are *connected* if the robot can take a step from one to the other. We encode necessary conditions for connectivity as a *stance graph*. Each node of this graph is a stance. Two nodes σ and σ' are connected by an edge if there is a transition between $\mathcal{F}_\sigma$ and $\mathcal{F}_{\sigma'}$. The existence of this transition is a necessary condition for the robot to take a step from some configuration at one stance to some configuration at an adjacent stance. Note, however, that for any two connected stances σ and σ', both $\mathcal{F}_\sigma$ and $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ may contain several components. So, we encode both necessary and sufficient conditions for connectivity as another graph, the *transition graph*. Each node of this graph is a transition. Two nodes $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ and $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma''}$ are connected by an edge if there is a continuous path between them in $\mathcal{F}_\sigma$.

Neither the stance graph nor the transition graph are constructed in their entirety, both because these graphs are very large and because a search algorithm is required to verify the existence of graph edges. Moreover, verifying the *non*-existence of graph edges is prohibitively difficult. For this reason, our planner interweaves exploration of the stance graph and the transition graph, based on the method of Bretl (2006). The algorithm EXPLORE-STANCEGRAPH searches the stance graph (Figure 5). It maintains a priority queue Q of nodes to explore. When it unstacks σ_{final}, it has found a candidate sequence of adjacent stances from σ_{initial} to σ_{final}. The algorithm EXPLORE-TRANSITIONGRAPH then verifies that this candidate sequence corresponds to a feasible motion by searching a subset of the transition graph (Figure 5). It explores a transition $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ only if (σ, σ') is an edge along the candidate sequence, and a path between $q, q' \in \mathcal{F}_\sigma$ only if σ is a node along this sequence. We say that EXPLORE-TRANSITIONGRAPH has *reached* a stance σ_i if some transition $q \in \mathcal{F}_{\sigma_{i-1}} \cap \mathcal{F}_{\sigma_i}$ is connected to q_{initial} in the transition graph. The algorithm returns the index i of the farthest stance reached along the candidate sequence. If this is not σ_{final}, then the edge (σ_i, σ_{i+1}) is removed from the stance graph, and EXPLORE-STANCEGRAPH resumes exploration.

The important effect of this two-stage search strategy is to postpone the generation of one-step paths (a relatively costly computation) until after generating transitions. It works well because, as we mentioned in Section 1, both ATHLETE's and HRP-2's motion on irregular and steep terrain is most constrained just as either robot places or removes a foot. In our experiments we have observed that if we can find $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ and $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma''}$, then a path between q and q' likely exists in $\mathcal{F}_\sigma$. For example, in Section 5.1 we present experiments with ATHLETE on a variety of terrain. In these experiments, there was a 60–75% chance of finding a feasible path between randomly sampled $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ and $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma''}$. Moreover, even if we could find no feasible path from q to q', nearly 100% of the time we could find a feasible path from q to a different configuration in $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma''}$. So after sampling even a small number of transitions, we can be reasonably sure to also find a one-step path.

Two key tools are embedded in this framework: the subroutines FIND-TRANSITION and FIND-PATH, and a heuristic for ordering Q. We present these tools in the following section.

```
EXPLORE-STANCEGRAPH(q_initial, σ_initial, σ_final)
1  Q ← {σ_initial}
2  while Q is non-empty do
3      unstack a node σ from Q
4      if σ = σ_final then
5          construct a path [σ_1, ..., σ_n] from σ_initial to σ_final
6          i ← EXPLORE-TRANSITIONGRAPH(σ_1, ..., σ_n, q_initial)
7          if i = n then
8              return the multi-step motion
9          else
10             delete the edge (σ_i, σ_{i+1}) from the stance graph
11     else
12         for each unexplored stance σ' adjacent to σ do
13             if FIND-TRANSITION(σ, σ') then
14                 add a node σ' and an edge (σ, σ')
15                 stack σ' in Q
16 return "failure"

EXPLORE-TRANSITIONGRAPH(σ_i, ..., σ_n, q)
1  i_max ← i
2  for q' ← FIND-TRANSITION(σ_i, σ_{i+1}) in each component of F_{σ_i} ∩ F_{σ_{i+1}} do
3      if FIND-PATH(σ_i, q, q') then
4          i_cur ← EXPLORE-TRANSITIONGRAPH(σ_{i+1}, ..., σ_n, q')
5          if i_cur = n then
6              return n
7          elseif i_cur > i_max then
8              i_max = i_cur
9  return i_max
```

Fig. 5. Algorithms to explore the stance graph and the transition graph.

3. Tools to Support the Motion Planner

3.1. Generating Transitions

Both EXPLORE-STANCEGRAPH and EXPLORE-TRANSITIONGRAPH require the subroutine FIND-TRANSITION to generate transitions $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ between pairs of stances σ and σ'. To implement FIND-TRANSITION, we use a sample-based approach. The basic idea is to sample configurations randomly in the robot's configuration space $\mathcal{Q}$ and reject those which are not in $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$. However, because of the contact constraint (Section 2.1), $\mathcal{Q}_\sigma$ is a sub-manifold of $\mathcal{Q}$, and in particular has lower dimension than $\mathcal{Q}$. As a result, $\mathcal{Q}_\sigma$ has zero measure in $\mathcal{Q}$, and so random sampling in $\mathcal{Q}$ would never generate a feasible transition. So, like Cortés et al. (2002), Wang and Chen (1991) and Yakey et al. (2001), we spend more time trying to generate configurations that satisfy the contact constraint at σ (or at σ' if $\sigma \subset \sigma'$) before rejecting those that do not satisfy other constraints. Like Hauser et al. (2005), we do this in two steps.

1. *Create a candidate configuration that is close to $\mathcal{Q}_\sigma$.* This step is tailored to the particular legged robot.

   ATHLETE: Each contact in the stance σ corresponds to the placement of a foot at a footfall in the terrain. First, we create a nominal position and orientation of the chassis: (i) we fit a plane to the footfalls in σ in a least-squares sense; (ii) we place the chassis in this plane, minimizing the distance from each hip to its corresponding footfall; and (iii) we translate the chassis a nominal distance perpendicular to the plane-fit and away

from the terrain. Then, we sample a position and orientation of the chassis in a Gaussian distribution about this nominal placement. Finally, we compute the set of joint angles that cause each foot to either reach or come closest to reaching its corresponding footfall. Note that here a footfall fixes the intersection of the ankle pitch and ankle roll joints relative to the chassis (Figure 1). The hip yaw, hip pitch and knee pitch joints determine this position. There are up to four inverse kinematic solutions for these joints or, if no solutions exist, there are two configurations that are closest (straight knee and completely bent knee). The knee roll, ankle roll and ankle pitch determine the orientation of the foot, for which there are two inverse kinematic solutions. We select a configuration that satisfies joint-limit constraints; if none exist, we reject the sample and repeat.

HRP-2: Each contact in the stance corresponds to the placement of a robot link on the terrain. We select one of these links as a root and fix its location. Then, starting from the root link, we incrementally sample joint angles (satisfying joint angle limits) along each closed-loop kinematic chain using a bounding-volume technique similar to Cortés et al. (2002). Finally, we use cyclic coordinate descent (Wang and Chen 1991) to adjust these joint angles so that every contact in the stance is approximately achieved.

2. *Repair the candidate configuration using numerical inverse kinematics.* We move the candidate configuration to a point in $\mathcal{Q}_\sigma$ using an iterative Newton–Raphson method. For each contact $i = 1, \ldots, N$ in σ, the error in position and orientation of the corresponding robot link is a differentiable function $\varepsilon_i(q)$ of the configuration q. Let

$$E(q) = \begin{bmatrix} \varepsilon_1(q) \\ \vdots \\ \varepsilon_N(q) \end{bmatrix}.$$

The contact constraint is the equality $E(q) = 0$. At each iteration k of the Newton–Raphson method, we transform the current configuration by taking the step

$$q_{k+1} = q_k - \alpha_k \nabla E(q_k)^{-\dagger} E(q_k),$$

where q_1 is the initial candidate configuration generated at step 1, $\nabla E(q_k)^{-\dagger}$ is the pseudo-inverse of the gradient of the error function, and α_k is the step size (computed using backtracking line search). The algorithm terminates with success if at some iteration $\|E(q_k)\| < \eta$ for some tolerance η, or with failure if a maximum number of iterations is exceeded.

The first step rarely generates configurations in $\mathcal{Q}_\sigma$, but quickly provides configurations that are close to $\mathcal{Q}_\sigma$. On the other hand, the primary cost of the second step is in computing $\nabla g(q_k)^{-\dagger}$ at every iteration, but few iterations are necessary when candidate configurations are sufficiently close to $\mathcal{Q}_\sigma$. So, it is the combination of these two steps that makes our sampler fast. For ATHLETE, the experiments corresponding to Figure 11 show that the repair step increases the fraction of feasible configurations from 1.9% to 18.4% and reduces the average time to generate each feasible sample from 0.64 to 0.24 s. For HRP-2, the experiments corresponding to Figure 13 show that the repair step increases the fraction of feasible configurations from 0.4% to 31.9% and reduces the sampling time from 0.74 to 0.06 s.

The above method quickly samples configurations in $\mathcal{Q}_\sigma$. However, the other constraints (equilibrium, torque, collision) restrict the feasible space $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ to a subset of $\mathcal{Q}_\sigma$. As the relative volume of this subset decreases, the rejection rate of the method increases. This problem arises in particular for HRP-2, which in general has a smaller support polygon and tighter joint limits than ATHLETE. To reduce the rejection rate, we write the equilibrium and joint-limit constraints as differentiable inequalities, and enforce them together with the contact equality when we repair a candidate configuration. We include these inequalities by combining the Newton–Raphson procedure with an active-set method in a similar manner to Byrd et al. (2002).

Note that EXPLORE-TRANSITIONGRAPH additionally requires that we sample a single transition in each connected component of $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$. Our approach is not guaranteed to do this, but the probability that it samples at least one in each component increases with the number of samples. If it eventually misses a component, this component is likely to be small, hence sensitive to modeling uncertainty and not useful in practice.

3.2. Generating Paths between Transitions

EXPLORE-TRANSITIONGRAPH requires FIND-PATH to generate paths in $\mathcal{F}_\sigma$ between pairs of transitions $q \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ and $q' \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma''}$. We use the PRM planner called SBL (Sánchez and Latombe 2002), which grows two trees of milestones rooted at q and q'. To sample configurations in $\mathcal{F}_\sigma$, we face a similar challenge to that discussed in the previous section (that a random configuration has zero probability of being in $\mathcal{Q}_\sigma$), and so we use a similar approach. However, in this case, to grow the two trees of milestones, SBL only requires sampling each new configuration in a neighborhood of an existing milestone. Close to this milestone (call it q_0), the shape of the manifold Q_σ is approximated well by the hyperplane

$$\left\{ p \in \mathcal{Q} \nabla E(q_0)^T p = \nabla E(q_0)^T q_0 \right\}.$$

```
FREE-PATH(q, q')
 1  if the distance from q to q' is less than ε then
 2      return TRUE
 3  q_mid ← (q + q')/2
 4  if Newton–Raphson from q_mid results in q_mid ∈ Q_σ then
 5      if q_mid ∈ F_σ then
 6          return (FREE-PATH(q, q_mid) & FREE-PATH(q_mid, q'))
 7      else
 8          return FALSE
 9  else
10      return FALSE
```

Fig. 6. Algorithm to connect close configurations with a local path.

So, before applying the iterative method to repair the sampled configuration, we first project it onto this hyperplane (in a similar way to Yakey et al. (2001)).

To connect milestones with local paths, we face again a similar challenge, since the straight-line path between any two configurations q and q' will not, in general, lie in $\mathcal{Q}_\sigma$. We deform this straight-line path into $\mathcal{Q}_\sigma$ using the bisection method FREE-PATH (Figure 6). At each iteration, FREE-PATH first applies Newton–Raphson (see Section 3.1) to the midpoint of q and q' to generate $q_{\text{mid}} \in \mathcal{Q}_\sigma$, then it checks that $q_{\text{mid}} \in \mathcal{F}_\sigma$. If both steps succeed, the algorithm continues to recurse until a desired resolution has been reached; otherwise, the algorithm returns failure. The advantage of this approach is that it does not require a direct local parameterization of $\mathcal{Q}_\sigma$, as it may be difficult to compute such a parameterization that covers both q and q'.

3.3. Ordering the Graph Search

Our two-stage search strategy can be greatly improved by ordering the stances in the priority queue Q used by EXPLORE-STANCEGRAPH in increasing order of a heuristic cost function $g(\sigma) + h(\sigma)$.

The term $g(\sigma)$ assigns a cost to the path from σ_{initial} to σ in the stance graph. First we associate a cost with each edge (σ, σ') in the stance graph. Each edge cost is initialized to 1, but may be modified later as indicated below. Then, we associate a cost with each path in the stance graph as the sum of its edge costs. Finally, we define $g(\sigma)$ as the minimum path cost required to reach σ from σ_{initial} in the stance graph.

The term $h(\sigma)$ assigns a cost to the stance σ itself. We define $h(\sigma)$ as a weighted sum of several criteria:

- *Planning time.* We increase the cost of σ proportional to the amount of time spent trying to sample a transition $q \in \mathcal{F}_{\sigma'} \cap \mathcal{F}_\sigma$ to reach it (Nielsen and Kavraki 2000).
- *Distance to goal.* We increase the cost of σ proportional to the distance between the centroid of its contacts and those of the goal stance σ_{final}.
- *Footfall distribution.* We increase the cost of σ proportional to the difference (in a least-squares sense) between its contacts and those of a nominal stance on flat ground (for example, with feet directly under each hip).
- *Equilibrium criteria.* We increase the cost of σ inversely proportional to the area of its support polygon.

This heuristic reduces planning time and improves the resulting motion. It also allows us to relax an implicit assumption, that FIND-TRANSITION and FIND-PATH always return "failure" correctly. As we implement these subroutines using a probabilistic sampling approach, they are unable to distinguish between impossible and difficult queries. So, on failure of FIND-TRANSITION in EXPLORE-STANCEGRAPH, we still add σ to the stance graph but give σ a high cost. Likewise, rather than delete (σ, σ') on failure of FIND-PATH, we increase the step cost of (σ, σ').

4. Improving Motion Quality

The motions planned using the methods described so far are feasible (given an accurate terrain model) but not necessarily of high quality. In particular, on easy terrain the motion of ATHLETE and HRP-2 is lightly constrained. Then our planner may generate paths that contain erratic, unnecessary movements of the legs, arms and torso. To improve the result, we apply a post-processing method of smoothing similar to those

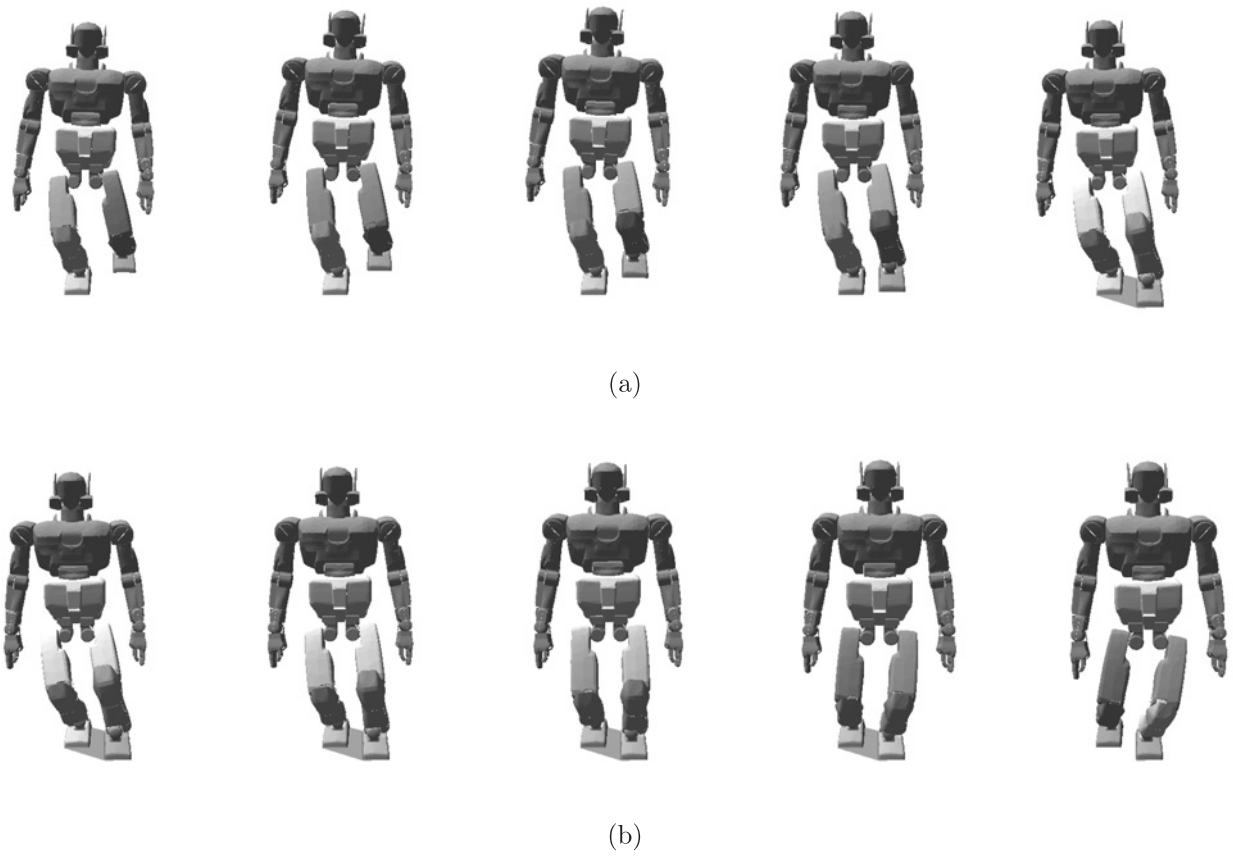


Fig. 7. Two primitives on flat ground to (a) place a foot and (b) remove a foot. The support polygon, here just the convex hull of supporting feet, is shaded blue. See also Extensions 1 and 2.

of Geraerts and Overmars (2004) and Vougioukas (2005), who used gradient descent to achieve criteria such as the minimum path length and maximum clearance (or safety margin). However, such a method (even a more computationally intensive one that optimizes dynamic criteria, such as energy spent) often fails to eliminate all needless motion, so does not transform any inefficient motion into an efficient motion.

Moreover, because we sample each contact location on the terrain (see Section 2.2), we might end up trying difficult steps when simpler steps (with slightly different contact locations) would also have led to the goal. For example, the robot might reach a stance σ associated with a feasible space $\mathcal{F}_\sigma$ containing a narrow passage, forcing the planner to compute a contrived motion, while a small perturbation of the contact locations at σ may have eliminated this passage (Hsu et al. 2005). To address the motion-quality issue, we provide our planner with a small library of motion primitives, each being a single step of very high quality. We do not restrict motion to these primitives, for example by sequencing, superimposing or transforming them (Section 1.4). Instead, we apply these primitives to bias the sampling strategy used to find transitions, to find paths, and to find contact locations on the terrain. So, motion primitives do not limit the region of configuration space explored by our planner; rather, they influence which regions of configuration space are explored first. Below we describe (1) how we generate these primitives, (2) how we use a given primitive to plan a better one-step motion and (3) how we decide which primitive to use when planning this motion.

4.1. Generating Motion Primitives

Given two adjacent stances σ and σ' and an initial configuration $q \in \mathcal{F}_\sigma$, our planner uses the methods described in Sections 2 and 3 to generate a large number of paths to final configurations randomly sampled in $\mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$. Then, it runs a non-linear optimization package (Lawrence et al. 1997) to optimize each path with respect to a given objective function (Harada et al. 2006) that combines path length, torque, energy and the amount of deviation from an upright posture. This entire process is an off-line pre-computation; several hours were required to generate the two example primitives in Figure 7.

In our current implementation, the user selects the triplets (σ, σ', q) given as input to the planner. These triplets should correspond to steps that are similar those likely to be needed in the type of terrain considered. The user must also choose which optimized motions generated by the planner are retained in the library of primitives, since objective functions may not be guaranteed to correspond to our esthetic notion of what is "natural" or "good-looking".

The generation of motion primitives has not been the main focus of our work so far, so many improvements are possible. For example, better results might be obtained with the method of optimization proposed by (Bobrow et al. 2001), which uses a B-spline representation of the trajectory and a different performance metric. Likewise, we might use a learned classifier to decide, without supervision, whether candidate primitives look natural, in a similar manner to Ren et al. (2005). Finally, we might automate the selection of triplets (σ, σ', q) by learning a statistical model of importance (similar to location-based activity recognition (Liao et al. 2005)) or applicability after perturbation (similar to PRM planning with model uncertainty (Missiuro and Roy 2006)).

We record each primitive as a nominal path

$$u : t \in [0, 1] \rightarrow u(t) \in \mathcal{Q}$$

in configuration space. Each recorded primitive either adds a contact (if $\sigma \subset \sigma'$), or breaks a contact (if $\sigma \supset \sigma'$). We denote the initial and final stances for each primitive u by σ_u and σ'_u, respectively.

4.2. Using Primitives for Planning

We use motion primitives to help our planner generate each step. We do this at three levels:

1. finding a path between a given configuration and a given transition in an adjacent stance;
2. finding a transition between two given stances (here, no transition is given);
3. finding a new contact given a stance (in order to define the step's final stance).

In each case, first we transform the primitive to better match the step we are trying to plan, then we use the transformed primitive to bias the sampling strategy used by our planner.

4.2.1. Finding Paths

Consider the robot at an initial configuration $q_{\text{initial}} \in \mathcal{F}_\sigma$ at some stance σ. Assume that we are given an adjacent stance σ' and a transition $q_{\text{final}} \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$. Also assume that we are given a primitive $u \subset \mathcal{Q}$. We want to use u to guide the search of the PRM planner for a path from q_{initial} to q_{final} in $\mathcal{F}_\sigma$. We still use SBL (see Section 3.2) to grow trees from root configurations, but rather than root these trees only at q_{initial} and q_{final}, we now root them at additional configurations (in a similar manner to Akinc et al. (2003)) sampled according to the primitive u.

(a) Transforming the Primitive to Match q_{initial} and q_{final}. Although we expect u to be somewhat similar to the step we are trying to plan, it will not be identical in general. Therefore, we first transform u so that it starts at q_{initial} and ends at q_{final}. We use an affine transformation

$$\hat{u}(t) = A\,(u(t) - u(0)) + q_{\text{initial}} \tag{5}$$

that maps the straight-line segment between $u(0)$ and $u(1)$ to the segment between q_{initial} and q_{final}. Hence,

$$\begin{aligned}
\hat{u}(0) &= A\,(u(0) - u(0)) + q_{\text{initial}} \\
&= 0 + q_{\text{initial}} \\
&= q_{\text{initial}} \\
\hat{u}(1) &= A\,(u(1) - u(0)) + q_{\text{initial}} \\
&= (q_{\text{final}} - q_{\text{initial}}) + q_{\text{initial}} \\
&= q_{\text{final}}.
\end{aligned}$$

We select A closest to the identity matrix, by minimizing

$$\min_A \sum_{i,j} (A_{ij} - \delta_{i,j})^2$$

$$\text{such that } A\,(u(1) - u(0)) = q_{\text{final}} - q_{\text{initial}},$$

where $\delta_{ij} = 1$ if $i = j$ and zero otherwise. We compute A in closed form as

$$A = I + \frac{((q_{\text{final}} - q_{\text{initial}}) - (u(1) - u(0)))\,(u(1) - u(0))^T}{\|u(1) - u(0)\|_2^2}.$$

We depict this transformation in Figure 8(a). First, u is translated to start at q_{initial}. Then, the farther we move along u (the more we increase t), the closer $\hat{u}$ is pushed toward the segment from q_{initial} to q_{final}.

(b) Sampling Root Milestones. Let $q_1, \ldots, q_s$ be configurations evenly distributed along $\hat{u}$ from $q_1 = q_{\text{initial}}$ to $q_s = q_{\text{final}}$ (Figure 8(b)). For each $i = 1, \ldots, s$, we test if $q_i \in \mathcal{F}_\sigma$. If so, the PRM planner adds q_i as a root milestone in the roadmap. If not, it samples other configurations in a growing neighborhood of q_i until it finds some feasible $q'_i \in \mathcal{F}_\sigma$, which is added as a root milestone instead of q_i.

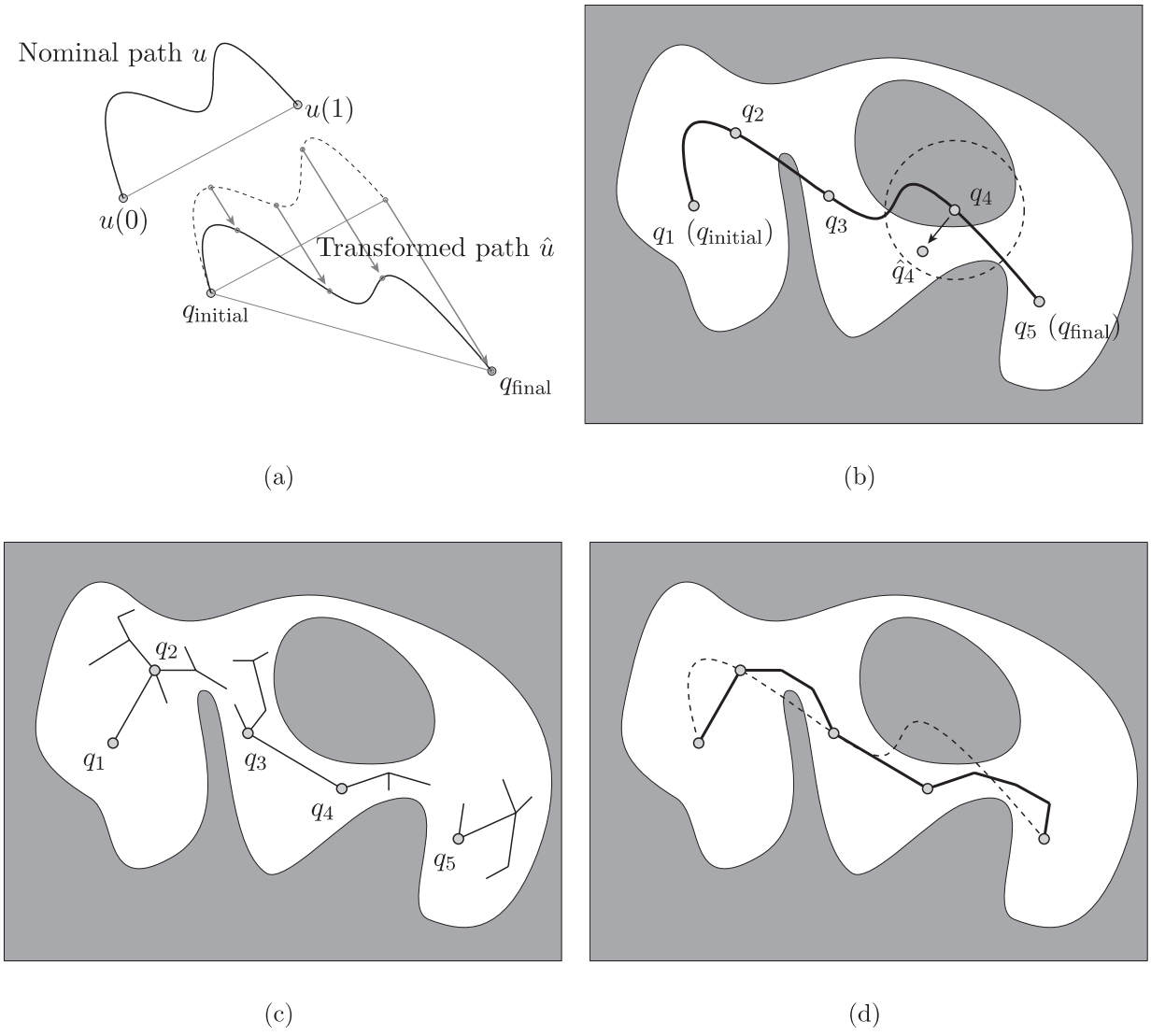


Fig. 8. Using a primitive to guide path planning. (a) Transforming a motion primitive to start at q_{initial} and end at q_{final}. (b) Sampling root milestones in $\mathcal{F}_\sigma$ near equally spaced waypoints along $\hat{u}$. (c) Growing trees to connect neighboring roots. (d) The resulting path, which if possible is close to $\hat{u}$ (dotted).

(c) Connecting Neighboring Roots with Sampled Trees. For $i = 1, \ldots, s-1$, we check whether the root milestone q_i can be connected to its neighbor q_{i+1} with a feasible local path (in a similar manner to Hauser et al. (2005)). If not, we add the pair of roots (q_i, q_{i+1}) to a set $\mathcal{R}$. Then, we grow trees of milestones between every pair in $\mathcal{R}$. For example, in Figure 8(c) we add (q_2, q_3) and (q_4, q_5) to $\mathcal{R}$ and grow trees to connect both q_2 with q_3 and q_4 with q_5. We process all trees concurrently. At every iteration, for each pair $(q_i, q_{i+1}) \in \mathcal{R}$, we first add ℓ milestones to the trees at both q_i and q_{i+1} (in our experiments, we set $\ell = 5$). Then, we find the configurations q connected to q_i and q' connected to q_{i+1} that are closest. If q and q' can be connected by a local path, we remove (q_i, q_{i+1}) from $\mathcal{R}$. When we connect all neighboring roots, we return the resulting path; if this does not happen after a fixed number of iterations, we return failure. Just like our original implementation, this approach will find a path between q_{initial} and q_{final} whenever one exists (given enough time). However, since we seed our roadmap with milestones that are close to u, we expect the resulting motion to be similar (and of similar quality) to this primitive whenever possible (Figure 8(d)), deviating significantly from it only when necessary.

4.2.2. Finding Transitions

Again consider the robot at configuration $q_{\text{initial}} \in \mathcal{F}_\sigma$ at a stance σ. But now, assume that we are only given stance σ'. We want to use a primitive u to guide the search for a transition $q_{\text{final}} \in \mathcal{F}_\sigma \cap \mathcal{F}_{\sigma'}$ before we plan a path to reach it (as described in Section 4.2.1). We expect a well-chosen transition to further improve the quality of this path.

(a) Transforming the Primitive to Match σ and σ'. Since we do not know q_{final}, we can not use the same transformation defined by Equation (5) that we used in Section 4.2.1(a). Instead, we use a rigid-body transformation of the form

$$\hat{u}(t) = Au(t) + b \tag{6}$$

that maps the stances σ_u and σ'_u associated with the primitive u as closely as possible to the stances σ and σ'.

Recall from Section 2.1 that a stance consists of several contacts, each placing a link of the robot on the terrain, and that we model a contact made by a robot link as a finite set of frictional points. Let $r_i \in \mathbb{R}^3$ for $i = 1, \ldots, m$ be the set of all points defining the contacts in both σ_u and σ'_u, and let $s_i \in \mathbb{R}^3$ for $i = 1, \ldots, m$ be the set of all points defining the contacts in both σ and σ'. (We assume that u has been chosen so that both sets have the same number of points.) We specify the positions of the contact points relative to the robot, and we choose the rotation matrix A and translation b in Equation (6) that minimize

$$\min_{A,b} \sum_i \|Ar_i + b - s_i\|_2^2 .$$

We can compute A and b in closed form (Arun et al. 1987), but we only consider rotations A about the gravity vector to avoid tilting the robot into an orientation that would violate the static equilibrium constraint.

(b) Sampling a Transition. We use the same two-step method as in Section 3.1. However, rather than sampling candidate configurations close to $\mathcal{Q}_\sigma$ at random, we sample them in a growing neighborhood of $\hat{u}(1)$.

4.2.3. Finding Contacts

Once more, consider the robot at configuration $q_{\text{initial}} \in \mathcal{F}_\sigma$ at stance σ, but now assume that we are given neither an adjacent stance σ' nor a transition, only a primitive u. If u removes a robot link from the terrain, then we immediately generate a final stance σ' by removing the corresponding contact from σ, and we apply the above techniques to compute a transition and a path to reach it. However, if u creates a new contact between a link and the terrain, we may use it to guide our search for a new contact during the construction of the stance graph.

(a) Transforming the Primitive to Match σ. We use the same transformation defined by Equation (6) to construct $\hat{u}$ as for finding transitions, but here we compute A and b to map only σ_u to σ, since we do not know σ'. We then use this transformation to adjust the placement of the new contact given by u. Let $r_i \in \mathbb{R}^3$ for $i = 1, \ldots, m$ be the set of points defining this contact. Then the transformed contact is given by $\hat{r}_i = Ar_i + b$ for $i = 1, \ldots, m$.

(b) Sampling a Contact. We define a sphere of radius δ, centered at $(1/m)\sum_i \hat{r}_i$. We increase δ until the intersection of this sphere with the terrain is non-empty (initially, we set δ to be approximately the size of either ATHLETE's or HRP-2's foot). We randomly sample a placement of the points $\hat{r}_i$ on the surface of the terrain inside the sphere, by first sampling a position of their centroid $s \in \mathbb{R}^3$ on the surface, then sampling a rotation of $\hat{r}_i$ about the surface normal at s. We check that the contact defined by this placement has similar properties (normal vector, friction coefficient) to the contact defined by u. If so, we add it to σ to form σ'. If not, we reject it and sample another placement.

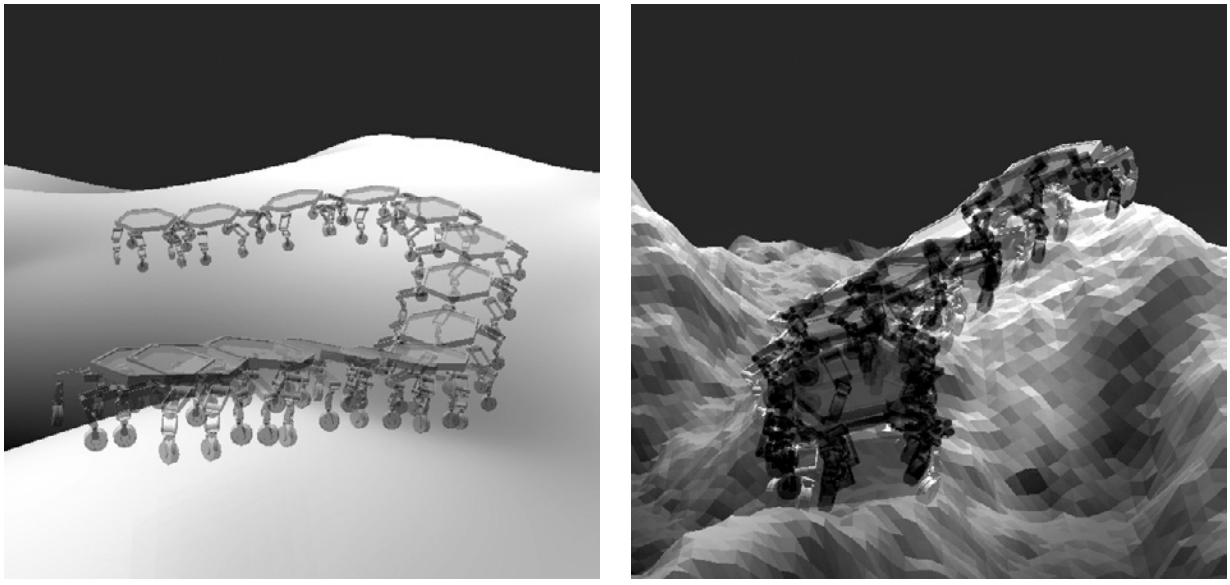
4.2.4. Deciding Which Primitive to Use

It only remains to decide which primitive u should be used, if any, given an initial stance σ and configuration q_{initial}. We have experimented with a variety of heuristics. For example, we may pick the primitive that most closely matches σ_u with σ (in other words, that minimizes the error in a transformation of the form (6)). Likewise, we may pick the primitive that most closely matches σ'_u with the actual terrain. If no primitives match well, we use the basic method from Section 3 instead. However, the best approach is still not clear, and this issue remains an important area for future work.

5. Results

5.1. Application to ATHLETE

We tested the planner in simulation on several example terrains. Our main goal was to demonstrate that the planner enables ATHLETE to traverse terrain where fixed gaits would fail. In particular, we tested the planner on terrains generated to simulate a range of Lunar surfaces. Using a fractal generation method, we created height maps of the form $z = f(x, y)$ as triangle meshes, where each triangle is about half the size of ATHLETE's wheels. All contacts were modeled with the same coefficient of friction. Figure 9(a) shows an alternating-tripod gait applied to smooth, undulating terrain. The gait can traverse the terrain freely. However, on irregular and steep ground (Figure 9(b)), the gait does not work at all: it results in ATHLETE losing balance or exceeding torque limits at several locations. We applied our planner to the same terrains, setting



(a) (b)

Fig. 9. Walking with an alternating tripod gait is (a) feasible on smooth terrain but (b) infeasible on uneven terrain due to violations of equilibrium and torque constraints. Extensions 3 and 4 show exactly when these constraints are violated.

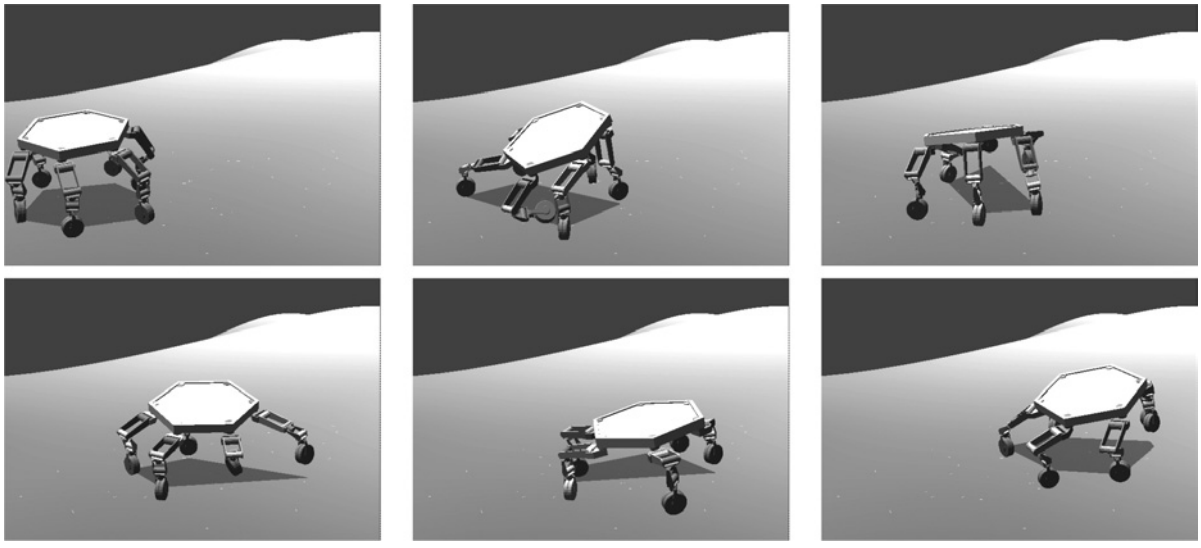


Fig. 10. Walking on smooth, undulating terrain with no fixed gait (see Extension 5).

the initial and final stances at a distance of about twice the diameter of ATHLETE's chassis, and sampling 200 contacts in the terrain to use for creating stances. Figure 10 shows motion on smooth ground, computed in 14 min and consisting of 66 steps. Figure 11 shows a feasible motion on irregular and steep ground, computed in 26 min and consisting of 84 steps.

We also performed more quantitative tests on simpler terrains, namely on a series of stair steps. The results are summarized in Table 1. The stairs range from 0.2 to 0.5 times the diameter of ATHLETE's chassis, and require moving about two body lengths. Alternating tripod, four-legged and six-legged gaits were able to traverse the lowest stair (after some recover-

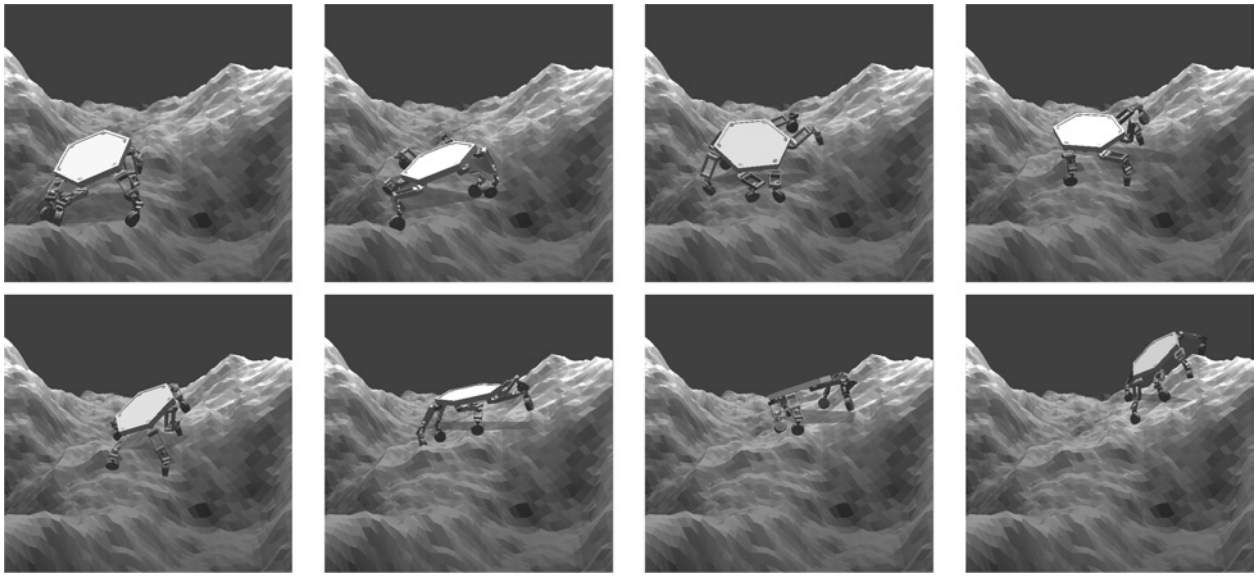


Fig. 11. Walking on steep, uneven terrain with no fixed gait (see Extension 6).

Table 1. Stair steps planned with various methods. Dashes indicates failure. Planner times are averaged over four runs.

	Gait				
Height	Tripod	Four	Six	Planner	Manual
0.2	✓	✓	✓	8 min	5 min 40 s
0.3	—	—	—	8 min 30 s	14 min
0.4	—	—	—	16 min 15 s	—
0.5	—	—	—	15 min 15 s	—

able slippage), but failed on all others. The planner, however, was able to reliably plan motions over all stairs, after sampling 200 footfalls at random in each terrain and relying on the search heuristic (Section 3.3) to identify which stances are useful. We compared the planner with footsteps chosen manually. A human operator used a point-and-click interface to place and break contacts. Motions to achieve the commanded contact changes were planned automatically with the one-step PRM planner. Manual operation was straightforward for the 0.2-unit stair, but the 0.3-unit stair required a large amount of trial-and-error and backtracking. An attempt to plan the 0.4-unit stair was stopped in frustration after about 30 min.

In another series of experiments, we demonstrated that the planner is flexible enough to handle different robot morphologies. Figure 12 shows motion to descend irregular and steep terrain at an average angle of about 60°. In this example, ATHLETE is rappelling, using a tether (anchored at the top of the cliff) to help maintain equilibrium. We included the tether with no modification to our planner, treating it as an additional leg with a different kinematic structure. The resulting motion consisted of 32 steps. Total computation time was 16 min.

All of these examples were generated without the use of motion primitives. The use of primitives becomes more important with HRP-2, as described in the following section.

5.2. Application to HRP-2

With HRP-2, the use of motion primitives was critical to plan motion paths of reasonable quality. We first demonstrate this point in detail on a stair-climbing example. Then we show several other experimental examples.

(a) An Example of Climbing a Single Stair.

In this example, we show that using a simple primitive can significantly help to improve motion quality, and that using the primitive to also sample transitions and contacts successively adds to the quality of the result. Figure 13 shows a two-step motion of HRP-2 to climb a single stair of height 0.3 m (just below the knee). This motion was planned without primitives (i.e. only with the methods described in Sections 2 and 3). As the motion is lightly constrained, the planner creates poorly chosen and superfluous movements (even after we apply a post-processing method of smoothing). In particular, the robot's arm and leg motions are erratic, and its right foot is placed too far on the stair. To improve this motion, we applied the two primitives shown in Figure 7 (steps on flat ground). First, we used these primitives only to help the PRM planner

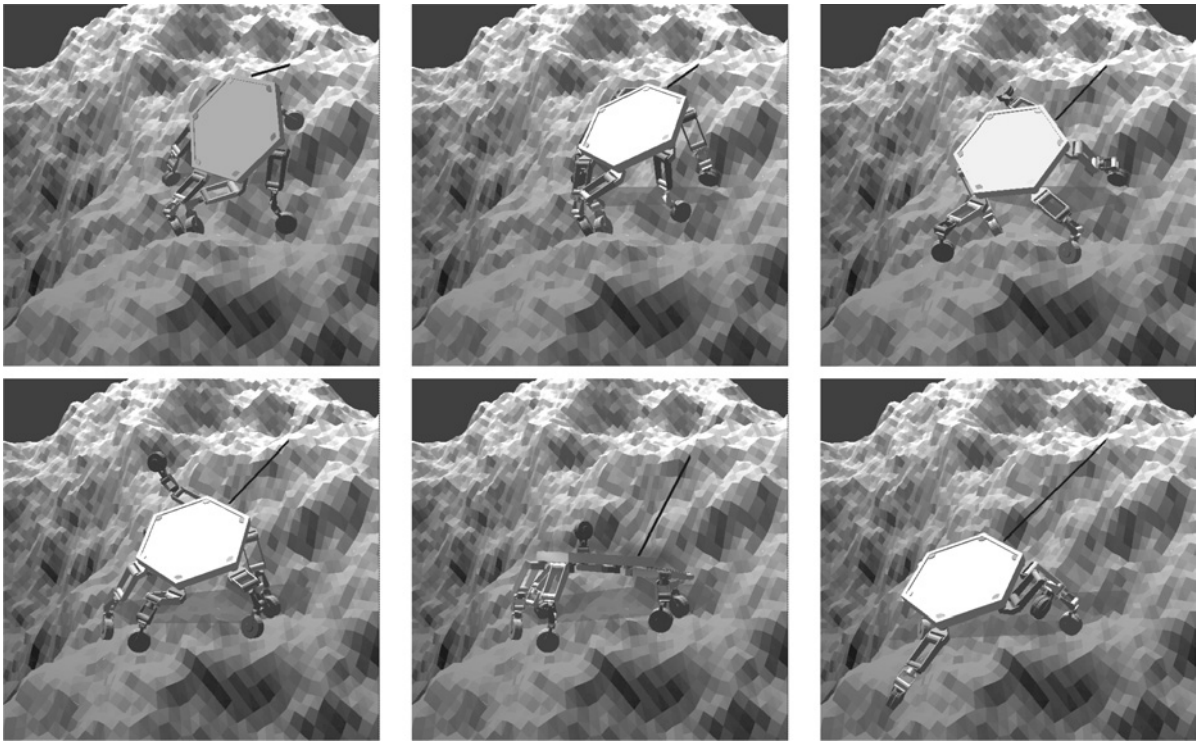


Fig. 12. Rappelling down an irregular 60° slope with no fixed gait (see Extension 7).

generate each one-step path, as described in Section 4.2.1. Figure 14 shows the resulting motion. Some erratic leg motions are eliminated, such as the backward movement of the leg in the second frame. The erratic arm motions remain, however, in particular because the transition in the fourth frame is the same (still randomly sampled). Figure 15 shows the result of using the primitives to adjust this transition as well as to plan paths, as described in Section 4.2.2, eliminating most of the erratic arm motions. However, the extreme lean in the fifth frame remains, due to the fact that the right foot is placed too far on the stair. Finally, Figure 16 shows the result of using the primitives to select contacts, sample transitions and plan paths, as described in Section 4.2.2. The chosen contact resulted in an easier step, eliminating the lean in the fifth frame. The foot is placed halfway off the step, something that people often do but that is potentially less stable. Here, we consider only static equilibrium (Section 2.1), a consideration of stability is beyond the scope of this paper.

(b) Motion quality and planning times for stairs of different heights.

We have observed that high-quality motion can be generated even when we use a primitive to plan a step that is significantly different. For example, we applied the same two primitives shown in Figure 7 to climb stairs of height 0.2, 0.3 and 0.4 m. Figure 17 shows the results. The quantitative results in the table were averaged over five runs. Quality is measured by an objective function that penalizes both path length and deviations from an upright posture (lower values indicate higher quality). For comparison, we report the minimum objective value achieved after a lengthy off-line optimization. These results demonstrate that using primitives both significantly improves motion quality and provides some reduction in planning time. Note also that both quality and time degrade gracefully as the stair gets higher, hence the steps deviate further from the primitives.

(c) Other examples.

We have tested our planner with HRP-2 on many other examples. Figure 18 shows a motion of HRP-2 on slightly uneven terrain where the highest and lowest points differ by 0.5 m; the motion was planned using the primitives given in Figure 7. In Figure 19 HRP-2 climbs a ladder with rungs that have non-uniform spacing and deviate from the horizontal by up to 15°. The primitives used for planning this motion were generated on a ladder with horizontal, uniformly spaced rungs. In Figure 20 HRP-2 walks sideways on a sloped terrain among boulders, using the hands to maintain equilibrium. Here, the primitives used by the planner were generated to step sideways on

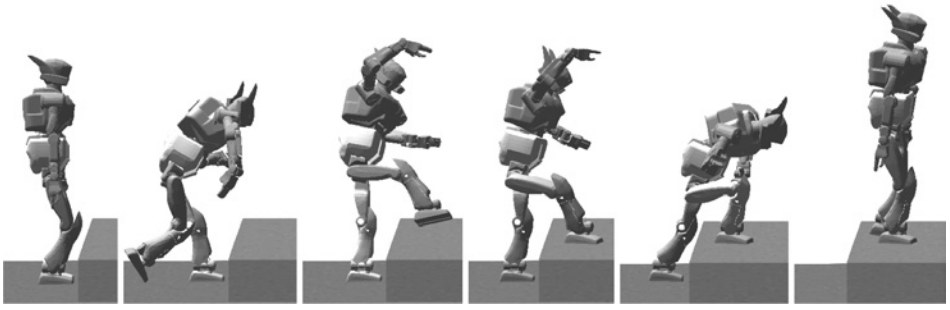


Fig. 13. Stair step planned entirely from scratch.

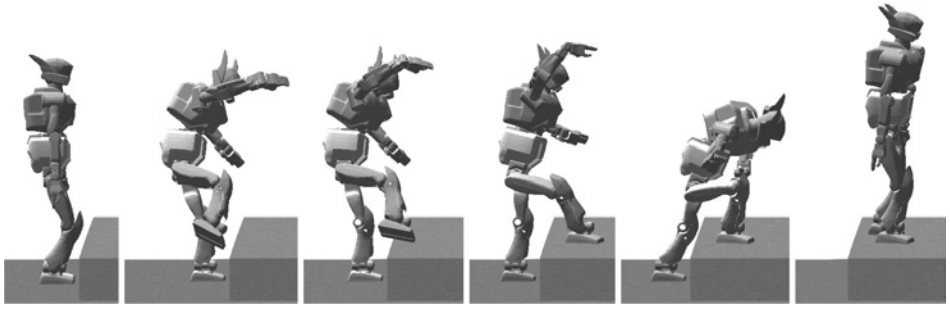


Fig. 14. Primitives guide path planning, reducing unnecessary leg motions.

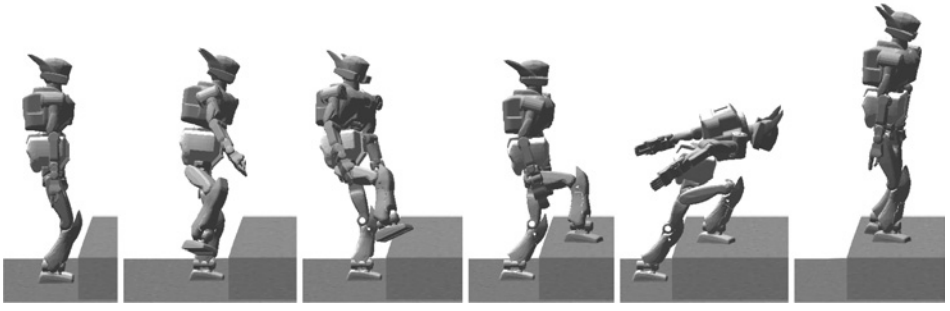


Fig. 15. Primitives guide transition sampling, reducing unnecessary arm motions.

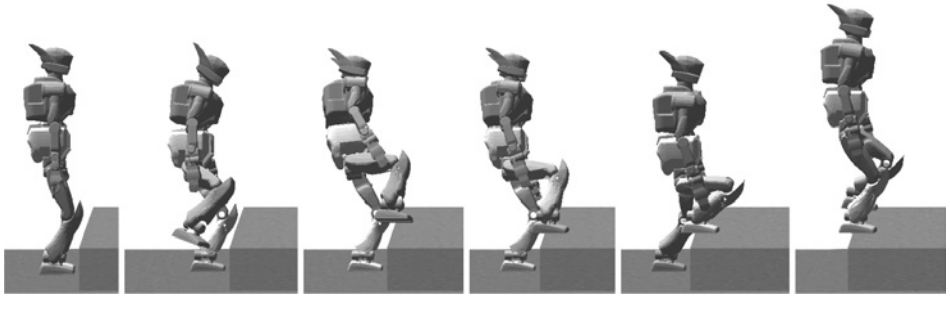
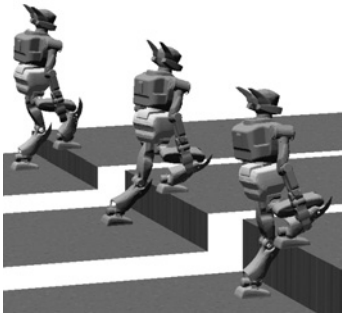


Fig. 16. Primitives guide the choice of contact, resulting in an easier step.



Stair	From scratch		Adapt primitive		Optimal
height	Time	Objective	Time	Objective	objective
0.2 m	8.61	5.03	5.42	3.04	2.19
0.3 m	10.3	4.67	4.08	2.31	2.17
0.4 m	12.2	5.15	10.8	3.27	2.55

Fig. 17. Planning time and objective function values for stair steps, averaged over five runs.

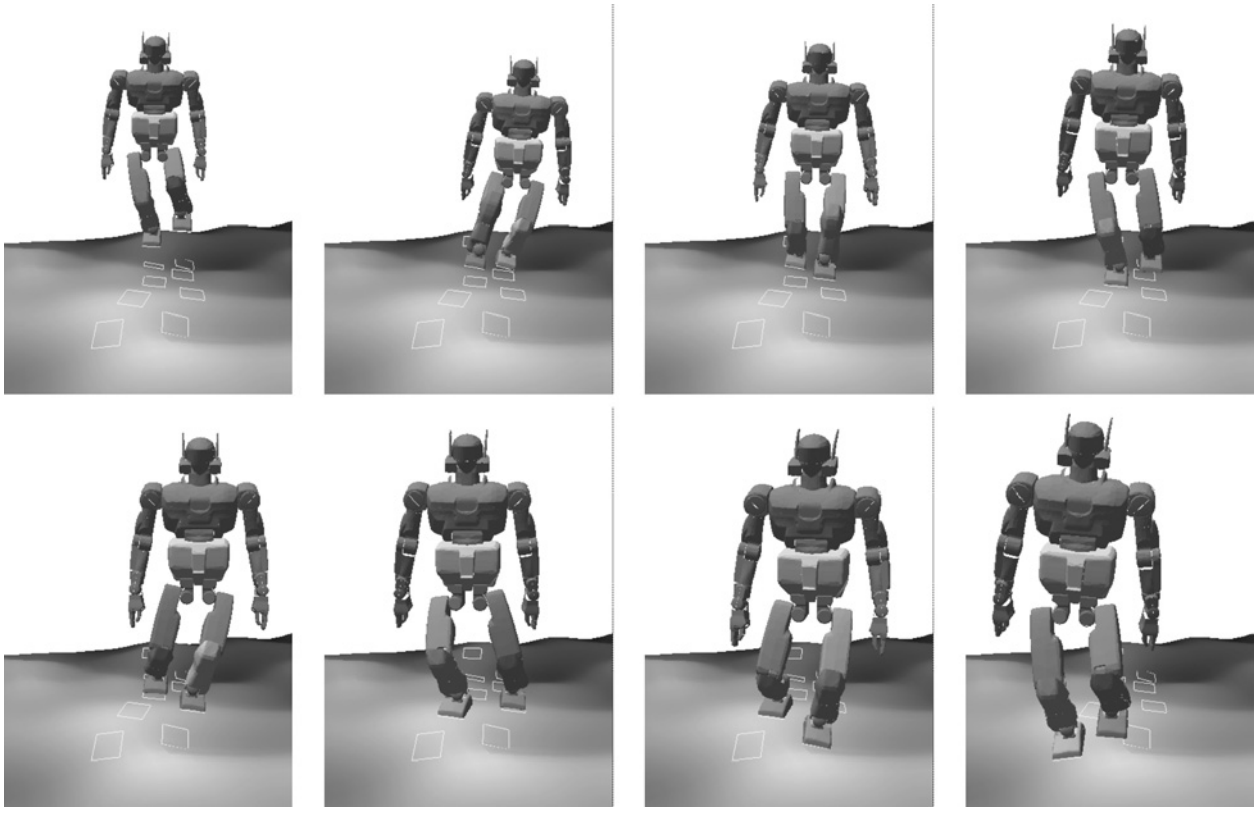


Fig. 18. A planar walking primitive adapted to slightly uneven terrain (see Extension 8).

flat ground while pushing against a vertical wall. Figure 21 shows HRP-2 traversing very rough terrain with slopes up to 40°. This motion was generated using a larger set of primitives, including steps of several heights, a pivot step and a high step using a hand contact for balance. In all of these examples, contact sampling was guided by motion primitives, as described in Section 4.2.2. Planning for the first three examples took about 1 min each. The fourth example took about 8 min.

6. Conclusion

In this paper we have described the design and implementation of a motion planner that enables legged robots with many DOFs to walk safely across rough, irregular terrain. We focused on the application of this planner to two such robots: the six-legged Lunar vehicle ATHLETE (which has wheels on the end of each leg, but can fix these wheels to walk), and the humanoid HRP-2. These robots are mechanically capable of

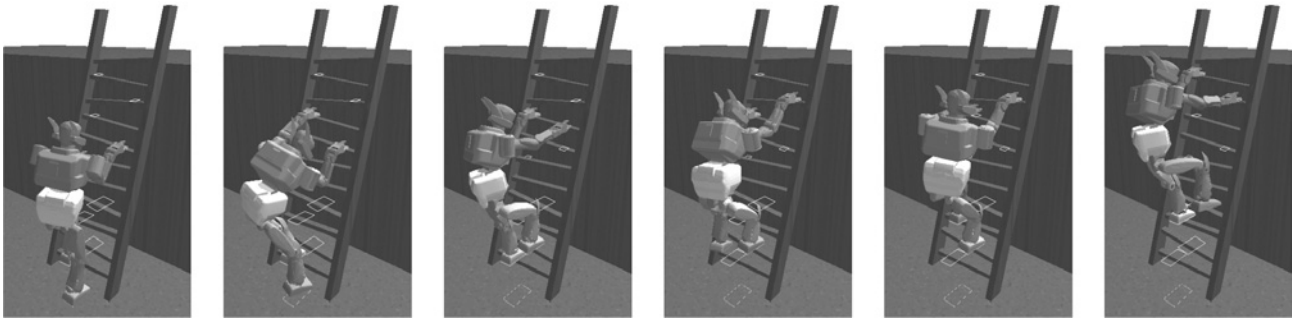


Fig. 19. A ladder-climbing primitive adapted to a new ladder with uneven rungs (see Extension 9).

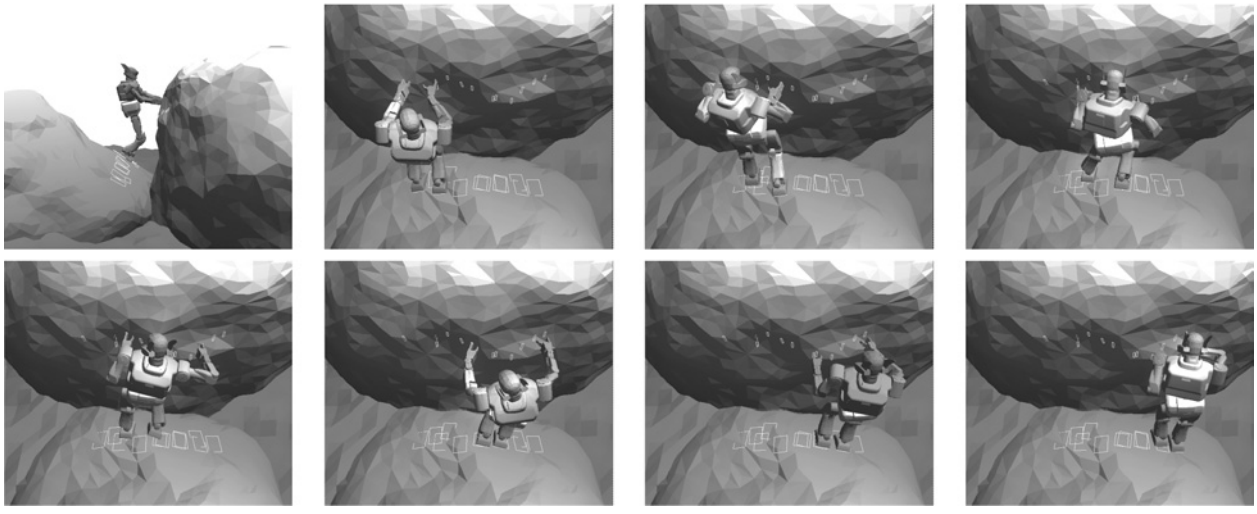


Fig. 20. A side-step primitive using the hands for support, adapted to a terrain with large boulders. Hand support is necessary because the robot must walk on a highly sloped boulder (see Extension 10).

walking carefully over terrain so rough that a fixed gait is insufficient, but an adequate motion planner is needed to take advantage of this ability. Our planner is based on a key design choice, choosing contacts and stances before computing motions, because on rough, irregular terrain, a legged robot's motion is most constrained just when it makes a new contact or breaks an existing contact (transitions). We extended previous techniques with several algorithmic tools to deal with difficult computational issues raised by robots such as ATHLETE and HRP-2: sampling feasible configurations (in particular, transitions), generating feasible local paths and searching the huge stance and transition graphs. To improve motion quality, we also described how to derive a probabilistic sampling strategy from a small library of pre-computed motion primitives. We demonstrated the flexibility of our planner with simulation results for both ATHLETE and HRP-2 on a variety of terrains, ranging from slightly uneven to very irregular and steep. Our planner can be applied directly to any other legged robot that is modeled as a collection of rigid links connected by actuated revolute joints and that makes contact with the environment at frictional points. The only details that may change are the method of sampling candidate configurations when generating transitions (Section 3.1) and the method of generating motion primitives (Section 4.1).

Our work still has many limitations that present opportunities for future work. Along a parallel line of research, a method of closed-loop control has been designed to execute motion plans generated by our planning approach (Miller et al. 2006), by adjusting the robot configuration to the forces applied at the contacts. This controller makes it possible to reliably execute motion plans despite modeling errors. We are currently working on integrating our planner and this controller. For example, by running a planned motion on a dynamic simulator, it is possible to determine how fast the controller can reliably

Fig. 21. A motion on steep and uneven terrain generated from a set of several primitives. A hand is being used for support in the third configuration (see Extension 11).

execute the motion. The planner would also benefit from better methods to generate high-quality motion primitives and to select which primitive is most appropriate to help plan each step. Planning dynamically stable motions might be too hard in general, but well-designed primitives could enable such planning when dynamic moves are critical to reaching a goal. Finally, our planner could help to design better legged robots by facilitating the study of their inherent capabilities. For certain applications, such as space exploration, it could help human teleoperators to design difficult motions more quickly. A similar approach was used to plan motions for the recent Mars rovers.

Acknowledgments

This work was partially supported by NSF grant 0412884 and by the RTLSM grant from NASA JPL, specifically for the ATHLETE project. K. Hauser is supported by a Thomas V. Jones Stanford Graduate Fellowship.

Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	HRP-2 taking a single step to place a foot
2	Video	HRP-2 taking a single step to remove a foot
3	Video	ATHLETE walking with an alternating tripod gait on smooth terrain (feasible).
4	Video	ATHLETE walking with an alternating tripod gait on uneven terrain (infeasible). When the chassis is green, the configuration is feasible. When the chassis changes color, one or more constraints have been violated.
5	Video	ATHLETE walking with no fixed gait on smooth terrain.
6	Video	ATHLETE walking with no fixed gait on uneven terrain.
7	Video	ATHLETE rappelling down an irregular 60° slope with no fixed gait on smooth terrain.
8	Video	HRP-2 on slightly uneven terrain (planar walking primitive).
9	Video	HRP-2 climbing a ladder with uneven rungs (ladder-climbing primitive).
10	Video	HRP-2 traversing large boulders (side-step primitive).
11	Video	HRP-2 on steep and uneven terrain (multiple primitives).

References

Akinc, M., Bekris, K. E., Chen, B. Y., Ladd, A. M., Plaku, E. and Kavraki, L. E. (2003). Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps. *Proceedings of the International Symposium on Robotics Research*, Siena, Italy.

Alami, R., Laumond, J.-P. and Siméon, T. (1995). Two manipulation planning algorithms. *Algorithmic Foundations of Robotics*, Goldberg, K., Halperin, D., Latombe, J.-C. and Wilson, R. (eds). Wellesley, MA, A.K. Peters, pp. 109–125.

Arun, K., Huang, T. and Blostein, S. (1987). Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**(5): 698–700.

Bares, J. E. and Wettergreen, D. S. (1999). Dante II: technical description, results and lessons learned. *The International Journal of Robotics Research*, **18**(7): 621–649.

Bevly, D., Farritor, S. and Dubowsky, S. (2000). Action module planning and its application to an experimental climbing robot. *Proceedings IEEE International Conference on Robotics and Automation*, pp. 4009–4014.

Bicchi, A. and Kumar, V. (2000). Robotic grasping and contact: a review. *Proceedings IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 348–353.

Bobrow, J., Martin, B., Sohl, G., Wang, E., Park, F. and Kim, J. (2001). Optimal robot motions for physical criteria. *Journal of Robotic Systems*, **18**(12): 785–795.

Boissonnat, J.-D., Devillers, O. and Lazard, S. (2000). Motion planning of legged robots. *SIAM Journal of Computing*, **30**(1): 218–246.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge, Cambridge University Press.

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.

Bretl, T. and Lall, S. (2006). A fast and adaptive test of static equilibrium for legged robots. *Proceedings IEEE International Conference on Robotics and Automation*, Orlando, FL.

Bretl, T. and Lall, S. (2008). Testing static equilibrium for legged robots. *IEEE Transactions on Robotics*, to appear.

Bretl, T., Latombe, J.-C. and Rock, S. Toward autonomous free-climbing robots. *Proceedings of the International Symposium on Robotics Research*, Siena, Italy.

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

Byrd, R. H., Gould, N. I. M., Nocedal, J. and Waltz, R. A. (2002). An active-set algorithm for nonlinear programming using linear programming and equality constrained subproblems. *Technical Report OTC 2002/4*, Optimization Technology Center, Northwestern University, Evanston, IL.

Chestnutt, J., Kuffner, J., Nishiwaki, K. and Kagami, S. (2003). Planning biped navigation strategies in complex environments. *Proceedings IEEE International Conference on Humanoid Robots*, Munich, Germany.

Choset, H., Lynch, K., Hutchinson, S., Kanto, G., Burgard, W., Kavraki, L. and Thrun, S. (2005). *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA, MIT Press.

Cortés, J., Siméon, T. and Laumond, J.-P. (2002). A random loop generator for planning the motions of closed kinematic chains using PRM methods. *Proceedings IEEE International Conference on Robotics and Automation*, Washington, DC.

Eldershaw, C. and Yim, M. Motion planning of legged vehicles in an unstructured environment. *Proceedings IEEE International Conference on Robotics and Automation*, Seoul, South Korea.

Estier, T., Crausaz, Y., Merminod, B., Lauria, M., Pguet, R. and Siegwart, R. (2000). An innovative space rover with extended climbing abilities. *Proceedings of Space and Robotics*, Albuquerque, NM.

Frazzoli, E., Dahleh, M. A. and Feron, E. (2002a). Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, **25**(1): 116–129.

Frazzoli, E., Dahleh, M. A. and Feron, E. (2002b). Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transactions on Robotics*, **25**(1): 116–129.

Gavrilets, V., Frazzoli, E., Mettler, B., Peidmonte, M. and Feron, E. (2001). Aggressive maneuvering of small autonomous helicopters: a human-centered approach. *The International Journal of Robotics Research*, **20**(10): 795–807.

Geraerts, R. and Overmars, M. (2004). Clearance based path optimization for motion planning. *Proceedings IEEE International Conference on Robotics and Automation*, New Orleans, LA.

Gleicher, M. (1998). Retargetting motion to new characters. *Proceedings of ACM SIGGRAPH'98*, pp. 33–42.

Gottschalk, S., Lin, M. and Manocha, D. (1996). OBB-tree: a hierarchical structure for rapid interference detection. *Computer Graphics*, **30**: 171–180.

Greenfield, A., Saranli, U. and Rizzi, A. A. (2005). Solving models of controlled dynamic planar rigid-body systems with frictional contact. *The International Journal of Robotics Research*, **24**(11): 911–931.

Grochow, K., Martin, S. L., Hertzmann, A. and Popović, Z. (2004). Style-based inverse kinematics. *ACM Transactions on Graphics*, **23**(3): 522–531.

Harada, K., Kajita, S., Saito, H., Kanehiro, F. and Hirukawa, H. (2004). Integration of manipulation and locomotion by

a humanoid robot. *Proceedings of the International Symposium on Experimental Robotics*, Singapore.

Harada, K., Hauser, K., Bretl, T. and Latombe, J.-C. (2006). National motion generation for humanoid robots. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China.

Hauser, K., Bretl, T. and Latombe, J.-C. (2005). Non-gaited humanoid locomotion planning. *Proceedings of Humanoids*, Tsukuba, Japan.

Heiken, G. H., Vaniman, D. T. and French, B. M. (1991). *Lunar Sourcebook: A User's Guide to the Moon*. Cambridge, Cambridge University Press.

Hirose, S. and Kunieda, O. Generalized standard foot trajectory for a quadruped walking vehicle. *The International Journal of Robotics Research*, **10**(1): 3–12.

Hirose, S., Yoneda, K. and Tsukagoshi, H. Titan VII: quadruped walking and manipulating robot on a steep slope. *Proceedings IEEE International Conference on Robotics and Automation*, Albuquerque, NM, pp. 494–500.

Hsu, D., Latombe, J.-C. and Kurniawati, H. (2005). On the probabilistic foundations of probabilistic roadmap planning. *Proceedings of the International Symposium on Robotics Research*, San Francisco, CA.

Iagnemma, K., Genot, F. and Dubowsky, S. (1999). Rapid physics-based rough-terrain rover planning with sensor and control uncertainty. *Proceedings IEEE International Conference on Robotics and Automation*, Detroit, MI.

Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K. and Isozumi, T. (2004). Humanoid robot HRP-2. *Proceedings IEEE International Conference on Robotics and Automation*, New Orleans, LA, pp. 1083–1090.

Kavraki, L. E., Svetska, P., Latombe, J.-C. and Overmars, M. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, **12**(4): 566–580.

Koga, Y. and Latombe, J.-C. (1994). On multi-arm manipulation planning. *Proceedings IEEE International Conference on Robotics and Automation*, San Diego, CA, pp. 945–952.

Kovar, L., Gleicher, M. and Pighin, F. (2002). Motion graphs. *Proceedings of ACM SIGGRAPH'02*, San Antonio, TX, pp. 473–482.

Kron, T. and Shin, S. Y. (2005). Motion modeling for online locomotion synthesis. *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, Los Angeles, CA, pp. 29–38.

Krotkov, E. and Simmons, R. (1996). Perception, planning, and control for autonomous walking with the ambler planetary rover. *The International Journal of Robotics Research*, **15**: 155–180.

Kuffner, J. J. Jr (1999) Autonomous agents for real-time animation. *Ph.D. Thesis*, Stanford University.

Kuffner, J. J. Jr, Nishiwaki, K., Kagami, S., Inaba, M. and Inoue, H. (2003). Motion planning for humanoid robots. *Proceedings of the International Symposium on Robotics Research*, Siena, Italy.

Laumond, J.-P. (1987). Finding collision-free smooth trajectories for a non-holonomic mobile robot. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1120–1123.

Laumond, J.-P., Jacobs, P., Taix, M. and Murray, R. (1994). A motion planner for nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, **10**(5): 577–593.

Lauria, M., Piguet, Y. and Siegwart, R. (2002). Octopus: an autonomous wheeled climbing robot. *Proceedings of CLAWAR*.

Lawrence, C., Zhou, J. and Tits, A. (1997). User's guide for CFSQP version 2.5: a C code for solving (large scale) constrained nonlinear (minimax) optimization problems, generating iterates satisfying all inequality constraints. *Technical Report TR-94-16r1*, Institute for Systems Research, University of Maryland, College Park, MD.

Lee, H., Shen, Y., Yu, C.-H., Singh, G. and Ng, A. Y. (2006). Quadruped robot obstacle negotiation via reinforcement learning. *Proceedings IEEE International Conference on Robotics and Automation*.

Liao, L., Fox, D. and Kautz, H. (2005). Location-based activity recognition. *Advances in Neural Information Processing Systems*, http://www.cs.washington.edu/ai/Mobile_Robotics/postscripts/places-nips-05.pdf.

Liu, C. K., Hertzmann, A. and Popović, Z. (2005). Learning physics-based motion style with nonlinear inverse optimization. *ACM Transactions on Graphics*, **24**(3): 1071–1081.

Meredith, M. and Maddock, S. (2005). Adapting motion capture data using weighted real-time inverse kinematics. *Computer Entertainment*, **3**(1): 5.

Miller, T., Bretl, T. and Rock, S. (2006). Control of a climbing robot using real-time convex optimization. *Proceedings of the IFAC Symposium on Mechanical Systems*, Heidelberg, Germany.

Missiuro, P. E. and Roy, N. (2006). Adapting probabilistic roadmaps to handle uncertain maps. *Proceedings IEEE International Conference on Robotics and Automation*.

Mumm, E., Farritor, S., Pirjanian, P., Leger, C. and Schenker, P. (2004). Planetary cliff descent using cooperative robots. *Autonomous Robots*, **16**: 259–272.

Ng, A. Y., Kim, H. J., Jordan, M. and Sastry, S. (2004). Autonomous helicopter flight via reinforcement learning. *Neural Information Processing Systems 16*.

Nielsen, C. L. and Kavraki, L. E. (2000). A two level fuzzy prm for manipulation planning. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Takamatsu, Japan, pp. 1716–1721.

Okamura, A., Smaby, N. and Cutkosky, M. (2000). An overview of dexterous manipulation. *Proceedings IEEE International Conference on Robotics and Automation*, pp. 255–262.

Pai, D. K., Barman, R. A. and Ralph, S. K. (1995). Platonic beasts: spherically symmetric multilimbed robots. *Autonomous Robots*, **2**(4): 191–201.

Pang, J.-S. and Trinkle, J. (2000). Stability characterizations of rigid body contact problems with coulomb friction. *Zeitschrift für Angewandte Mathematik und Mechanik*, **80**(10): 643–663.

Pettré, J., Laumond, J.-P. and Siméon, T. (2003). A 2-stages locomotion planner for digital actors. *Proceedings of the Eurographics/SIGGRAPH Symposium on Computer Animation*.

Popovic, M. B., Goswami, A. and Herr, H. (2005). Ground reference points in legged locomotion: Definitions, biological trajectories and control implications. *The International Journal of Robotics Research*, **24**(12): 1013–1032.

Popović, Z. and Witkin, A. (1999). Physically based motion transformation. *Proceedings of ACM SIGGRAPH'99*, pp. 11–20.

Ren, L., Patrick, A., Efros, A. A., Hodgins, J. K. and Rehg, J. M. (2005). A data-driven approach to quantifying natural human motion. *ACM Transactions on Graphics*, **24**(3): 1090–1097.

Rimon, E., Burdick, J. W. and Omata, T. (2006). A polyhedral bound on the indeterminate contact forces in planar quasi-rigid fixturing and grasping arrangements. *IEEE Transactions on Robotics*, **22**(2): 240–255. DOI: 10.1109/TRO.2005.862478.

Sahbani, A., Cortés, J. and Siméon, T. (2002). A probabilistic algorithm for manipulation planning under continuous grasps and placements. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, pp. 1560–1565.

Sánchez, G. and Latombe, J.-C. (2002). On delaying collision checking in PRM planning: application to multi-robot coordination. *The International Journal of Robotics Research*, **21**(1): 5–26.

Schwarzer, F., Saha, M. and Latombe, J.-C. (2002). Exact collision checking of robot paths. *Proceedings of WAFR*, Nice, France.

Sentis, L. and Khatib, O. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robotics*, **2**(4): 505–518.

Shapiro, A. and Rimon, E. (2003). PCG: A foothold selection algorithm for spider robot locomotion in 2D tunnels. *Proceedings IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, pp. 2966–2972.

Shin, H. J., Lee, J., Shin, S. Y. and Gleicher, M. (2001). Computer puppetry: an importance-based approach. *ACM Transactions on Graphics*, **20**(2): 67–94.

Song, G., Miller, S. and Amato, N. M. (2001). Customizing PRM roadmaps at query time. *Proceedings IEEE International Conference on Robotics and Automation*, Seoul, Korea, pp. 1500–1505.

Song, S.-M. and Waldron, K. J. (1989). *Machines that Walk: the Adaptive Suspension Vehicle*. Cambridge, MA, MIT Press.

Stilman, M. and Kuffner, J. J. Planning among movable obstacles with artificial constraints. *Proceedings of WAFR*, New York, NY.

Vougioukas, S. G. (2005) Optimization of robot paths computed by randomized planners. *Proceedings IEEE International Conference on Robotics and Automation*, Barcelona, Spain.

Wang, L. and Chen, C. (1991). A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Transactions on Robotics and Automation*, **7**(4): 489–499.

Wettergreen, D., Thorpe, C. and Whittaker, W. (1993). Exploring Mount Erebus by walking robot. *Robotics and Autonomous Systems*, **11**: 171–185.

Wilcox, B. H., Litwin, T., Biesiadecki, J., Matthews, J., Heverly, M., Morrison, J., Townsend, J., Ahmad, N., Sirota, A. and Cooper, B. (2007). ATHLETE: a cargo handling and manipulation robot for the Moon. *Journal of Field Robotics*, **24**(5): 421–434.

Witkin, A. and Popović, Z. (1995). Motion warping. *Proceedings of ACM SIGGRAPH'95*, Los Angeles, CA, pp. 105–108.

Yakey, J. H., LaValle, S. M. and Kavraki, L. E. (2001). Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, **17**(6): 951–958.

Yamane, K., Kuffner, J. J. and Hodgins, J. K. (2004). Synthesizing animations of human manipulation tasks. *ACM Transactions on Graphics*, **23**(3): 532–539.

Yoneda, K., Ito, F., Ota, Y. and Hirose, S. (1999). Steep slope locomotion and manipulation mechanism with minimum degrees of freedom. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1897–1901.

Zheng, Y. F. and Shen, J. (1990). Gait synthesis for the SD-2 biped robot to climb sloping surface. *IEEE Transactions on Robotics and Automation*, **6**(1): 86–96.