Navigation Planning for Legged Robots

Thesis Proposal

Joel Chestnutt
The Robotics Institute
Carnegie Mellon University

Thesis Committee

James Kuffner (chair)
Chris Atkeson
Mike Erdmann
Hirohisa Hirukawa

November 15, 2006

Abstract

As legged robots gain the abilities to walk and balance on more than just flat, obstacle-free floors, they grow closer to fulfilling the potential of legged locomotion shown by biological systems. To truly fulfill this potential these robots must successfully traverse complicated, rough terrain, requiring the robots to step onto or over various features of the environment. Furthermore, when a robot must spend a significant amount of time supported by a single foot, the contact that foot makes with the ground is very important for stability, requiring that the robot properly assess foot placement to maintain balance during locomotion. This thesis will address the problem of navigation for legged robots by using a global planning approach built on top of existing walking and running controllers. The planning process will reason about foot placement and contact configurations within the terrain, and the connectivity of those contact configurations. This thesis will provide a general global navigation strategy for a wide range of legged robots through complicated environments, while utilizing the advantages of their legs, as well as ensuring their safety and stability during locomotion.

Contents

1 Introduction **3**
1.1 Problem Description 4

2 Background **5**
2.1 Biped Locomotion 5
2.2 Quadrupeds and Hexapeds 6
2.3 Navigation Planning 6
2.4 Planning in Graphics 7

3 Footstep Planning **8**
3.1 Choosing an action set 8
3.2 Connectivity of stances 9

4 Implementation for a Biped Robot **9**
4.1 Representations 10
4.2 State Evaluation 11
4.3 Online Execution 15

5 Dynamics **17**

6 Quadruped planning **21**

7 Adaptive action set **22**

8 An Intelligent Joystick **25**

9 Proposed Work **28**
9.1 Action model exploration 28
9.2 Intelligent joystick through rough terrain 29
9.3 Navigation planning for running robots 29

10 Contributions **29**

11 Proposed Schedule **30**

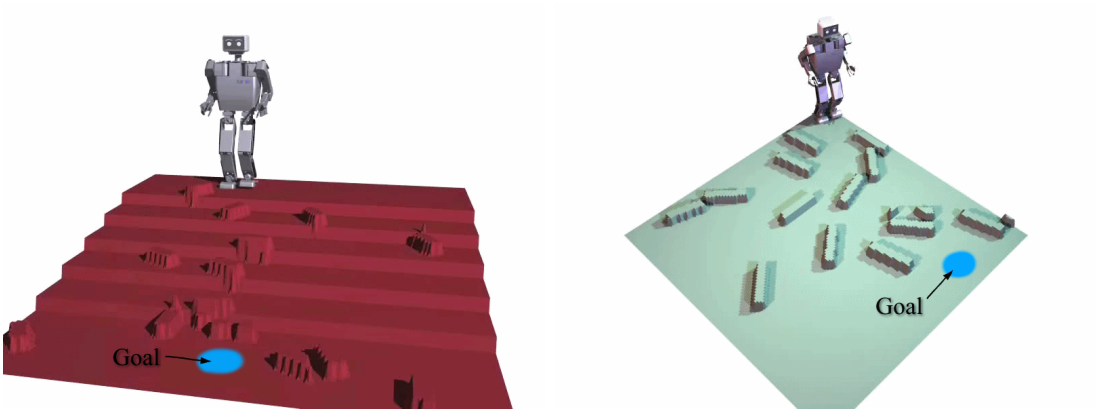


Figure 1: Example navigation problems for the robot to solve

1 Introduction

From examples in biology, we can see the great potential of legged systems. While they do not match the efficiency of wheeled robots in many environments, we can see many biological systems which exhibit great flexibility, agility and robustness in traversing extremely complicated terrains. Legged robotic systems, while they possess the ability to step onto or over obstacles, move in any direction, and balance autonomously, have not yet come close to the capabilities shown by biological systems. There are still many areas which need improvement for legged locomotion, including actuator and mechanism design, balance control and robustness to disturbances, sensing of the environment, and the ability for the robot to choose intelligent and appropriate actions for traversing terrain. The focus of this work is in the area of intelligently choosing actions to safely and efficiently traverse a given terrain. We build on top of existing balance controllers to safely navigate a robot through complicated environments. Some examples of the type of problem this thesis will attempt to solve are shown in Figure 1. For complex indoor environments designed for humans, this includes dealing with furniture, walls, stairs, doors, and previously unknown obstacles on the floor. For outdoor environments, this includes the ability to navigate on rough terrain and uneven surfaces. Because legged robots have the ability to step over and onto obstacles in their path, they are uniquely suited to overcoming these difficulties. However, existing navigation planning methods fail to consider these additional capabilities, because they were primarily designed for wheeled mobile robots.

Due to the many degrees of freedom present on legged robots, we focus on methods to reduce the dimensionality of the search space in order to find plans in a reasonable amount of time, while still retaining the capabilities that make legged robots interesting. This dimensionality reduction is accomplished by reasoning about the foot placement of the robot, discretizing our search along the discrete changes in the hybrid dynamics of the legged systems. This provides a natural way of breaking up the problem, allowing us to apply discrete planners to find a safe sequence of foot placements through the environment. The robot's locomotion controller can then take the robot through each stage of support, taking the robot from its start configuration to the goal.

1.1 Problem Description

We can formally define the input of our problem as

$$I = (x_{init}, x_{goal}, e, U),$$

where x_{init} is the initial state of the robot, x_{goal} is the goal state, e is the environment, and U is a set of control actions describing the actions of the robot. We can define a function over all state-action pairs, describing how the state of the robot evolves:

$$\dot{x} = f(x, u)$$

In this formulation, the problem that the robot faces at every instant in time is to select the "best" control action $u \in U$ to execute. Using a planning technique which guarantees optimality is one way to find an entire sequence of "best actions". While this planning is often computationally expensive, it provides a global value, which avoids the problem of getting stuck in local minimums. For the planning formulation, the problem is to find a path through the state space from the start state to the goal state, $\tau = (x_{init}, x_1, x_2, ..., x_{goal})$ and the corresponding series of control actions $\mathcal{P} = (u_0, u_1, u_2, ..., u_n)$, which if executed on the given terrain will cause the robot to follow the desired path. Given a cost function over these paths and actions $\psi(\tau, \mathcal{P}, e)$ which defines optimality, we want to find the path and associated control actions which minimize ψ.

Unfortunately, for a legged robot, this problem quickly becomes intractable, due to the number of degrees of freedom and the length of the path. Instead of planning at the lowest level of motor commands, we can create abstractions which encapsulate "chunks" of the robot's path. This abstraction is an action set $\mathcal{A}$, for which any action $a \in \mathcal{A}$ takes the system from one state to another as defined by the relationship

$$x_{i+1} = Succ(x_i, a, e).$$

Under this formulation, the problem is to find the sequence of actions from $\mathcal{A}$ which will take the robot from the initial state to the goal state in the given environment. The main difference between this formulation and the continuous formulation is that the output sequence of states is no longer a continuous path through the state space, but a discrete series of points which are connected through the use of our action model, $\mathcal{A}$. The implication of this difference is that performing collision checking on the individual states is not sufficient to ensure a safe path. An action that can safely take the robot between successive states in the given environment must be available for the path to be valid.

In order to apply this planning approach we need to make an important decision, namely, what are these actions, $\mathcal{A}$, that the robot can take? Are they trajectories, control policies, or something else? This choice of what the actions consist of and how they are chosen has serious consequences for both how difficult the planning problem becomes, as well as how much of the robot's capabilities are used in the resulting plan. As stated earlier, the full motion planning problem for all the degrees of freedom of a legged robot can quickly become intractable. However, we can break this problem down to a lower-dimensional subspace in an efficient way by looking at the structure of the

problem. After performing our planning in this low-dimensional space, we can turn our plan back into a full-body motion for the robot, allowing the robot to move to the goal. This thesis investigates constructing useful action models and planners for the particular problem of legged robots.

In choosing how to reduce our search space, we have several goals for our final action representation:

Re-use of existing control. A great deal of research has been performed on the subject of balance, walking, and running for legged robots. Ideally, our planning strategies can make use of the controllers that have resulted from that research, rather than requiring the planner to discover how to walk or run from scratch for each situation.

Planning simplicity. The fewer dimensions in our search space, the easier it is for a planner to explore. Simplifying the planning space has a large impact on real-time performance on a robot in real environments.

Capability usage. While we want to reduce the dimensionality of our problem and simplify the actions for our planner, another goal is to make sure that we do not sacrifice the unique abilities of legged robots in the process. We strive to find the right balance between utilizing as many capabilities of the robot as possible while at the same time not overwhelming the planner with too many details of the locomotion process.

Executability. In addition to having enough freedom in the planner to use the full capabilities of the robot, we need to have enough information in the plan to ensure that we are not exceeding the robot's capabilities. Depending on the details of the robot and its controllers, some dimensions may be safely ignored. However, it is important that we do not reduce our problem to the point where the planned action sequences exceed the abilities of the robot and controller.

2 Background

2.1 Biped Locomotion

Reliable walking biped robots have been developed only recently, although today there are several humanoid robots in use around the world, such as Johnnie[55], HRP-2[36] and HRP-3[2], H7[32], HUBO[71], WABIAN-2[68], Honda's ASIMO[81], Sony's QRIO[20], and Toyota's Partner robots. For these robots, comparatively little research attention has been focused on developing complete global navigation strategies. Instead, most research has focused on pre-generating stable walking trajectories (e.g. [27, 64, 93]), or on dynamic balance and control (e.g. [77, 89]). Recently, techniques have been developed to generate stable walking trajectories online[33, 65, 66, 67, 69, 72], though these results do not account for obstacles. Sensor-based obstacle-avoidance techniques have been developed for bipeds navigating in unknown environments[92, 57], which have allowed robots such as the Johnnie humanoid to adjust its path and step length in response to sensed

obstacles[56]. However, such reactive methods can become trapped in local loops or dead-ends, because they do not consider global information. In biomechanics, researchers have studied the problem of how humans perform local planning over irregular terrain based on visual feedback[73, 74].

There have been several running biped robots which store energy of the running motion in springs during stance, to be released on the next stride[1, 78]. For a planar running biped, control schemes were designed and implemented to enable placing the feet at desired locations[29], although the problem of how to choose those desired footholds was not addressed. The Bow-leg hopper[94] was constructed in such a way that most of the system parameters were fixed or self-stabilizing (body pitch, leg stiffness, energy insertion), and only one control parameter was free (leg angle at touchdown). The controller used a planner to perform a forward search using potential leg angles of future steps as its action set to find a sequence of actions that could be safely executed.

2.2 Quadrupeds and Hexapeds

For quadrupeds and hexapeds, the issue of balance is much simpler than the two-legged case for most situations, so more research effort has been put into negotiating difficult terrain than with biped robots. In fact, because of the increased inherent stability, many control schemes and robots have been developed which use open-loop gaits which self-stabilize over a wide region of the state space, allowing for blind locomotion over difficult terrain[13, 37, 82, 95]. To optimize these robots' motions when traveling over flat terrain, many researchers have studied gait generation for speed and efficiency[16, 39]. For quadruped robots, adaptive gait generation and control on irregular terrain and among obstacles has been previously studied [21, 28, 38], allowing reactive changes in gait for extra stability and speed over rough terrain. These approaches do not consider global information or produce global paths. In addition, there have been many mutli-legged robots which rely on static stability, which slowly and carefully move through rugged environments[4, 5, 8, 30, 59, 60, 63, 90].

2.3 Navigation Planning

Global path planning and obstacle avoidance strategies for mobile robots and manipulators have a large and extensive history in the robotics literature (e.g. see [31, 48, 51] for an overview).

In the path planning literature, related approaches using classical AI search or dynamic programming[9, 41] have been applied to finding collision-free motions. Some examples include car-like robots[7], kinodynamic planning[19], calculating sequences of pushing motions[58], planning for constrained systems[6], optimal control of manipulators[10, 83], and a variety of planning problems in a game-theoretic framework[50]. Ultimately, planning problems in robotics can be cast as discrete heuristic search problems involving cost metrics, sets of candidate actions, and goal regions.

Current planning approaches for legged robots lie along a spectrum based on how much of the robot's underlying details are considered. At one end of the spectrum, every detail is considered, and solving the navigation problem involves solving a giant motion planning problem for all degrees of freedom of the robot. This approach is used for

short-term motions, such as whole-body manipulation[44], but can quickly become too computationally expensive for locomotion problems. However, planning the details for the whole body has been used to connect different configurations as part of a locomotion plan[26, 34]. Other systems have used local planning on a step-by-step basis, allowing the robot to adjust its gait locally in response to the sensed terrain, usually in a statically stable manner[5, 42, 30, 91].

At the opposite end of the spectrum are planners which ignore all the details of the legs, and instead treat the robot as if it was a wheeled robot and "steer" it through the environment. Global navigation strategies for mobile robots can usually be obtained by searching for a collision-free path in a 2D environment. Because of the low-dimensionality of the search space, very efficient and complete (or resolution-complete) algorithms can be employed[85]. These techniques have been applied to biped humanoid robots, resulting in conservative global navigation strategies obtained by choosing an appropriate bounding volume (e.g. a cylinder), and designing locomotion gaits for following navigation trajectories computed by a 2D path planner[43, 76]. However, this always forces the robot to circumvent obstacles rather than using the ability to traverse obstacles by stepping over or onto them. For the QRIO robot, this approach has been augmented with additional actions such as stair climbing and descending, allowing the robot to use some more of its capabilites[80, 23]. Other applications of this approach use heuristics to generate a 2D body path for the environment, and then fill in the details along that path with local planning for the legs[54]. Another approach planned ways to adjust HRP-2's body posture to fit into the available free areas along a path[35].

Other approaches build action models which fall somewhere between these two extremes, trying to simplify the planning problem while still retaining the useful abilities of the robot. One action model uses straight sequences of footsteps to the edges of obstacles (similar to a visibility graph), combined with turning in place and stepping-over actions to cross through obstacle-filled environments[3]. Climbing robots have used reasoning about individual footholds combined with probabilistic motion planning to find motion plans for wall-climbing[12, 11, 25]. Recently, several approaches have used footsteps as an action model for moving through an environment for both bipeds[26, 70] and quadrupeds[52]. Finally, planning for multi-legged robots is a similar problem to that of finger-gaiting in manipulation[14, 15, 24], where instead of manipulating an object, the robot can be though of as manipulating the world underneath it. This thesis is based on previous research in footstep planning by Kuffner and colleagues[45, 47].

2.4 Planning in Graphics

Other related techniques in computer animation that use footprint placement for motion specification have been developed for bipeds[22, 88, 84], and quadrupeds[40, 87]. Large datasets of captured human motion are now available, and techniques have been developed to synthesize animations based on artist specified goals and splicing together segments of motion capture data [53, 49]. However, the focus in these works is on providing a realistic-looking motion, rather than ensuring that the path is executable by a physical robot.

3 Footstep Planning

3.1 Choosing an action set

To create an action set for legged robots that can use the full capabilities of the robot, yet still plan quickly, we examine the structure of the legged locomotion problem.

Let $\mathcal{S}$ be the set of all possible states of the robot, including the robot's joint positions, velocities, and world position and velocity. We can classify them by the support configuration, the set of contact points between all parts of the robot and the environment, which I will refer to in this document as a *stance*. Let $\mathcal{L}$ be the set of all stances the robot can have in the environment. For any given stance $\ell_i \in \mathcal{L}$, there exists some surface of associated robot states X_i that fit within the particular constraint. Any motion the robot takes will move through a sequence of these surfaces. For a walking biped, any possible path involves switching back and forth between various single and double support regions, the double support surface being the intersecting regions of adjacent single support surfaces. This classification provides a natural way to break up the problem into discrete chunks, in order to build our action set, $\mathcal{A}$. By planning a path first as a sequence of constrained surfaces through which the motion must pass, we can significantly reduce the dimensionality of the search and simplify the checking of environment constraints, without the need to sacrifice the robot's capabilities. For the walking biped example, every path will be made up of a sequence of double support phases, connected by paths through a single support phase. This thesis explores this action representation of planning for stances of the robot, studying the stability of individual stances, and their connectivity in the environment.

This breakdown also provides a very natural level of abstraction for behaviors/controllers. The behavior or controller can handle the unconstrained degrees of freedom of the robot to maintain balance, while the planner plans in the space of changing support constraints. In this document I will refer to the set of contact points between a foot (or other part of the robot) and the environment as a *foothold*. Thus the current stance is made up of the union of the current set of footholds. Finally, the motion the robot takes between one stance and another will be called a *footstep*.

In this way, our action set $\mathcal{A}$ becomes the set of possible footsteps the robot can make. Collision checking becomes a matter of evaluating footholds and stances at the border between actions, as well as the motion of the connecting footsteps. The planning process with this action model breaks the full motion planning problem into a planning problem in the reduced the dimensionality of relevant stances, $\mathcal{L}$, and then the problem of generating footsteps, paths through those constraint surfaces, X.

To re-use existing control strategies with this action set, we can use one of many previously developed locomotion controllers to solve the problem of connecting various stances. For our walking biped example, we can use the stances from the planner as input to a walking controller which then generates a dynamically stable motion taking the robot from one foothold to the next.

3.2 Connectivity of stances

One of our goals was to ensure the executability of generated plans. Adding a controller will add extra constraints to the valid robot states, depending on the particular capabilities of the controller or behavior. The use of particular controllers can also limit the capabilities of what the system can do. For our walking biped example, a controller may not be able to handle large step lengths, or stances involving the knees or hands, or airborne states. The planner's problem is then to find a sequence of stances in $\mathcal{L}$ which take the robot from the start to the goal, such that the stances and the surfaces connecting those stances lie within the capabilities of the available controllers. The resulting path through $\mathcal{L}$ will jump from one point in that space to another, not in a continuous path. Thus, we need ways to determine the connectivity of the points in $\mathcal{L}$ with regard to the system.

The dimensionality reductions possible are determined by the minimal information needed to determine the connectivity of stances for a given robot and controller. For example, some controllers (such as those for H7, HRP-2, and LittleDog) limit the body velocity enough that the connectivity of adjacent stances is independent of walking speed. Thus, for these systems, the planning state space can be reduced to just the space of stances, $\mathcal{L}$, significantly reducing the dimensionality of the space the planner must explore.

To ensure that our sequence of stances can be followed, we need to know two important pieces of information based on the robot and controllers. First, we need to know which stances are valid. This involves evaluating the terrain to determine if it is indeed a place to which the robot and controller can step. Second, we need to know how the potential trajectories of the robot affect the connectivity of the valid configurations. This is determined by both the kinematic reach of the robot, but more importantly by the limitations of the underlying controller which will be tasked with moving from one support state to another. The connectivity problem can be phrased as the question: given two support states and the environment, can the controller safely move from one to the other?

This second piece of information I will refer to as the action model of the robot. It encodes the capabilities of the robot and its controller, allowing the robot to generate plans which will be executable by the underlying system.

In this thesis we restrict contact with the environment to the robot's feet, so planning in the space of stances becomes the problem of choosing a sequence of footholds in the environment. Our action model thus describes what stepping actions can be made from the current state of the robot.

4 Implementation for a Biped Robot

Acting upon this idea of navigation planning based on footstep choice, we have constructed planners for various robots in different settings. These planners allow the robots to autonomously navigate through complicated environments. This section describes some early work involving the implementation of a basic footstep planner for the navigation of a biped robot.

The planner takes as input a map representing the terrain to plan over, an initial and goal state, and an action model consisting of a discrete set of possible footsteps that can be taken. This set represents a fixed sampling of the full space capabilities of the robot.

Algorithm 1: PLANPATH($x_{init}, x_{goal}, e, \mathcal{A}$)

```
   //Initialize search (state, cost, expected, parent);
1  Q.Insert(s_init, 0, 0, NULL);
2  while running_time < t_max do
3     x_best ← Q.ExtractMin();
4     if GoalReached(x_best, x_goal) then
5        return x_best;
      end
6     foreach a ∈ A do
7        x_next ← T(x_best, a, e);
8        c_l ← LocationCost(e, x_next);
9        c_s ← StepCost(e, a);
10       c_e ← ExpectedCost(e, x_next);
11       Q.Insert(x_next, x_best.cost + c_l + c_s, c_e, x_best);
      end
   end
```

If a path is found, the planner returns the solution as an ordered list of the footsteps that should be taken to reach the goal. The planner itself is implemented as an A* search over the possible footsteps of the robot to find the optimal sequence. The algorithm for this planning is given in Algorithm 1.

4.1 Representations

Biped Robot Representation: For evaluating footholds, we model the biped as its footprint rectangle (but any other polygon would be acceptable) at each step. We represent the state of the robot by

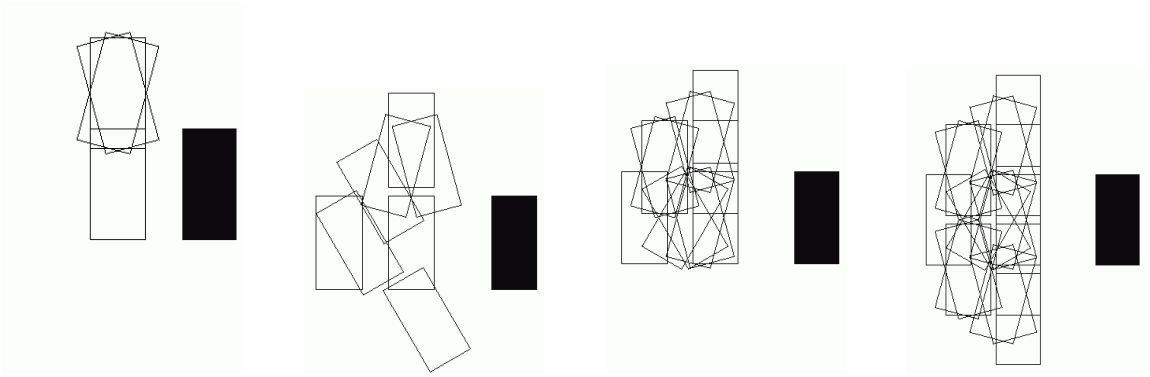
$$(x, y, \theta, s) \in \Re^2 \times [0, 2\pi) \times \{L, R\},$$

where x and y are the coordinates of the center of the rectangle in a fixed world coordinate system, θ is the angle between the y-axis of the world coordinate system and the forward direction of the foot, and s denotes the support foot (left or right). Notice that there is no joint information, and no velocities or body position included in this state representation. In this case our state representation is merely the foothold the robot will stand on during its footstep. As mentioned earlier, the path will jump between different points in this space, so the start and goal do not have to be in the same connected component of the (x, y, θ) space.

Environment: The terrain map M is represented by a grid of cells. Each cell c is represented by

$$(x, y, h, i) \in \Re^3 \times \{0, 1\},$$

where (x, y) is its location in the grid, h is the height of that cell, and i is an information value used to encode extra information about the validity of the terrain that may not



4 actions 8 actions 13 actions 20 actions

Figure 2: Footstep action sets. The actions displayed are only those for the left foot (relative to the right foot shown in black).

be apparent from its shape. Together, the cells create a height map representing the shape of the terrain the planner must overcome. The information values provide extra knowledge of the terrain, for places which appear to be safe to step on but should be treated as obstacles. This representation is easily generated from sensor data or other available representations of the terrain. It provides a simple way to represent many different kinds of environments, with the restriction that it cannot model certain details, such as overhangs or the areas underneath tables the way a full 3D representation can.

Action Model: The actions of the robot in this model are the footsteps which the robot can make, represented by

$$(x, y, \theta, c, h_{high}, h_{low}, h_{obst}) \in \Re^2 \times [0, 2\pi) \times \Re^4,$$

where x and y represent a relative displacement in the robot's current coordinate frame, θ represents the relative orientation change of the robot, and c represents the cost of making the step. h_{high} and h_{low} represent the allowable relative height change the action can make from one step to the next, and h_{obst} represents the maximum relative height of an obstacle that this action can step over. The set of actions the planner uses are sampled from the range of footstep location for which the robot's controller is capable of generating trajectories. The action set is constructed in such a way as ensure that all states generated by applying the actions are reachable for the robot. The parameters h_{high}, h_{low}, and h_{obst} are used to verify that connectivity in the presence of the terrain. Four examples of footstep action sets that we used in our experiments are illustrated in Figure 2. The actions are grouped into sets for the left and right feet, the two sets being mirror images of each other.

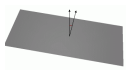
4.2 State Evaluation

The planner evaluates all transitions from a state, generating three costs for each one. First is the *location cost* $L(x)$, which evaluates the actions's destination state as a potential foothold. This cost uses a variety of metrics to quickly compute how viable a

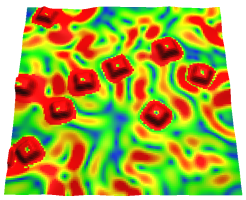
location is for stepping onto. Second is a *step cost* $S(x_{next}, a, x_{current})$, which computes the cost of reaching the state x_{next} by taking the action $a \in \mathcal{A}$ from the current state $x_{current}$. This cost describes the connectivity of x_{next} and $x_{current}$. This cost includes the action's associated cost, a penalty for height changes, as well as an obstacle clearance check of the terrain between the foot's last position and the new foothold. Finally, the third cost is a heuristic $R(x_{current}, x_{goal})$ which estimates the *remaining cost* to reach the goal state. This remaining cost can be computed in several ways, for example using the Euclidian distance to the goal, or the result of a traditional mobile robot planner. These three costs are then used by the planner to determine the cost of each node in an A* search. Note that the location cost is independent of the action or current state of the biped, and thus can be precomputed for all locations if desired, rather than computed for the needed states at run-time.

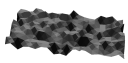
Location Metrics To evaluate a location's cost, we would like to know exactly how the foot would come to rest on the surface, which parts of the foot would be supported, and be able to build a support polygon of the foot based on which parts of the foot are touching the ground and evaluate how stable that support polygon is. Unfortunately, this is very expensive to exactly compute. Instead, we use a set of metrics which can be quickly computed and serve to approximate this ideal location cost. To be useful, a metric should be quick to compute, invariant to the resolution of the heightmap, and should eliminate or penalize an unwanted form of terrain while not eliminating or heavily penalizing good terrain.

To compute the metrics, we first determine the cells in the heightmap which will be underneath the foot for a particular step. This gives us a set of cells, $\mathcal{C}$, to use with each of the metrics defined below: Each of the metrics has both a weight and a cutoff value associated with it. If any metric passes its cutoff value, the location is discarded. Otherwise, the location's cost is the weighted sum of these metrics. Below are descriptions of the five metrics currently used for terrain evaluation. Each metric's scoring of an example terrain is shown to the right, with blue areas as the lowest cost, up to red as the highest. The weighted sum of these metrics is shown in Figure 3.



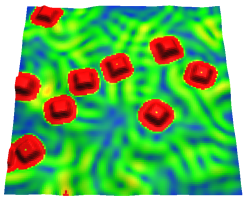
The slope angle of the surface at the candidate location. Perfectly horizontal surfaces are desired. The slope angle is computed by fitting a plane $h_{fit}(x, y)$ to the cells in the location.





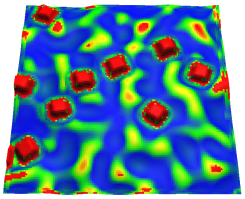
The "roughness" of the location. A measure of the deviation of the surface from the fitted plane. Computed by averaging the difference in height of each cell to the plane's height at the that cell.

$$\frac{1}{N} \sum_{(x,y,h,i) \in \mathcal{C}} |h - h_{fit}(x, y)| \qquad (1)$$



The "stability" of the location. This metric evaluates the curvature of the location. While perfectly flat is desired, curving down at the edges is penalized more than curving up at the edges. This metric is computed using a weighted sum of the heights (convolving with a dome-shaped filter).

$$\frac{1}{N} \sum_{(x,y,h,i)\in\mathcal{C}} ([h - h_{fit}(x,y)]g(x_f, y_f)) \qquad (2)$$



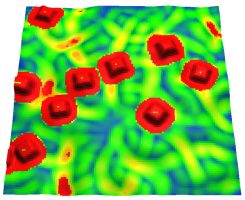
x_f and y_f are x and y in the foot's coordinate system. $g(x,y)$ is the dome-shaped filter. Restrictions on $g(x,y)$ are that it should be higher in the center than on the edges, and it should sum to zero over the area of the foot. The filter we used was:

$$g(x,y) = cos\left(\frac{2\pi x}{w}\right) + cos\left(\frac{2\pi y}{l}\right) \qquad (3)$$

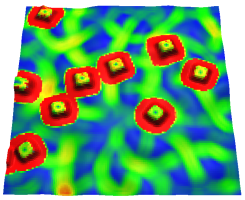
w and l are the length and width of the foot.

The largest bump of the location. Bumps above the fitted plane are much worse for a location than holes in the plane. This metric finds the largest deviation above the plane.

$$\max\{h - h_{fit}(x,y)\} \quad , \quad (x,y,h,i) \in \mathcal{C} \qquad (4)$$



The "safety" of the location. This refers to the area around the location. Its purpose is to take into account the possible inaccuracy of foot positioning. This can be computed using the roughness and largest bump metrics, using the cells around the foot location.



Step Cost The step cost computes the cost to the robot of making a particular footstep. In addition, it will determine if a particular footstep can be executed in the presence of obstacles (by returning infinite cost if it is unexecutable). The cost of taking a step is given by:

$$S(x_{next}, a, x_{current}) = c_a + w_h|H(x_{next}, x_{current})| \qquad (5)$$

c_a is the cost of the action a. $H(x_{next}, x_{current})$ is the height change between the state x_{next} and the current state of the robot $x_{current}$. w_h is the penalty for height changes, chosen manually based on the user's desire to try to avoid paths that go up or down. If the height change is outside the action's allowable range, x_{next} is not reachable from $x_{current}$, and the step is discarded. An obstacle collision check is also done to determine if

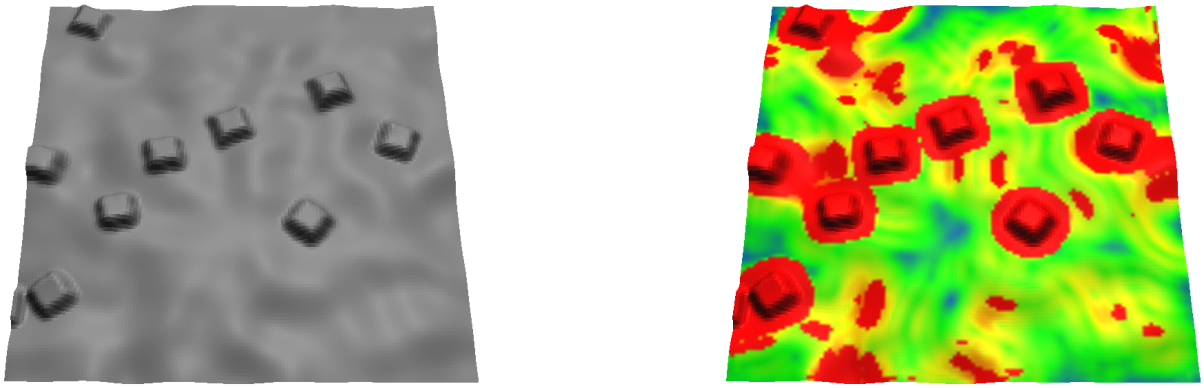


Figure 3: *Left:* an example terrain. *Right:* the weighted sum of the foothold metrics. Blue is for the lowest cost areas, up to red for the unsteppable areas.

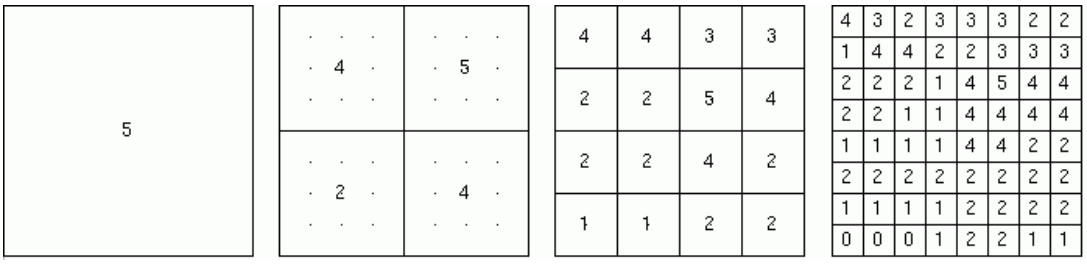


Figure 4: A quadtree built to encode the maximum height of the terrain in each region allows for fast computation of swing leg feasibility.

the foot can safely clear all the cells along the path from its previous location to its new location. Because this path may include many cells, quadtrees that encode the maximum height are used to quickly check for collisions (see Figure 4). If collisions are found, once again x_{next} is not reachable from $x_{current}$, and the step is discarded.

Estimated Cost Heuristic A mobile robot planner that plans outward from the goal state to the initial state provides a useful estimate of remaining cost, with the results stored in a grid which discretizes the workspace. During the footstep planning, the remaining cost can then be found in constant time. This heuristic takes more information about the environment into account than a Euclidean distance metric, but has several disadvantages besides the extra preprocessing time. Mobile robot planners look for a continuous path through the configuration space or workspace that connects the initial and goal states. Because the biped has the ability to step over obstacles, it does not require a continuous path through the workspace. The result of this difference is that the mobile robot planner can severely misjudge the cost of a location. In an environment with a long, low, thin obstacle, the mobile robot planner will provide lower cost to areas which send the biped the long way around instead of stepping over the obstacle, resulting in an overestimate. Also, it can underestimate when finding a path that one foot can fit though, but where there are not actually alternating footholds the robot can step on. In

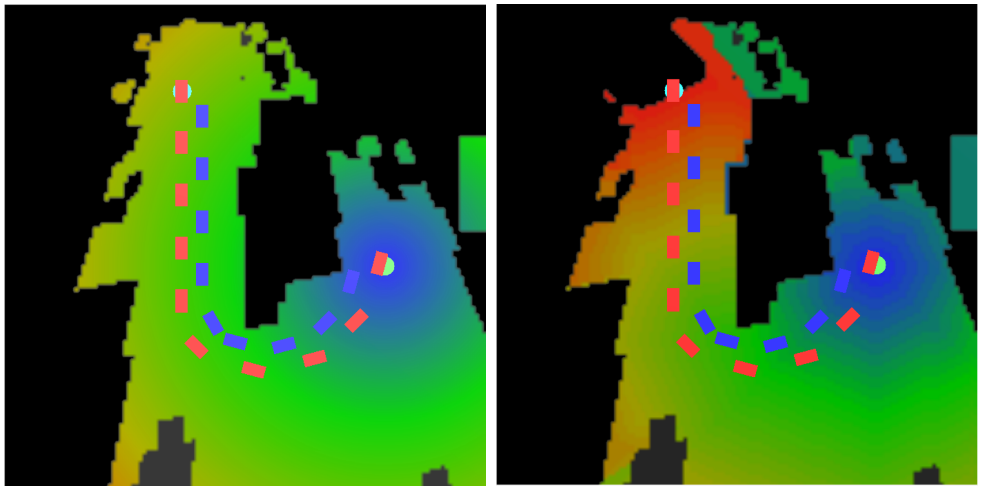


Figure 5: Examples of heuristics. *Left:* Euclidean heuristic. Notice that for the first half of the path, the heuristic does not provide useful information. *Right:* Heuristic from traditional mobile robot planner.

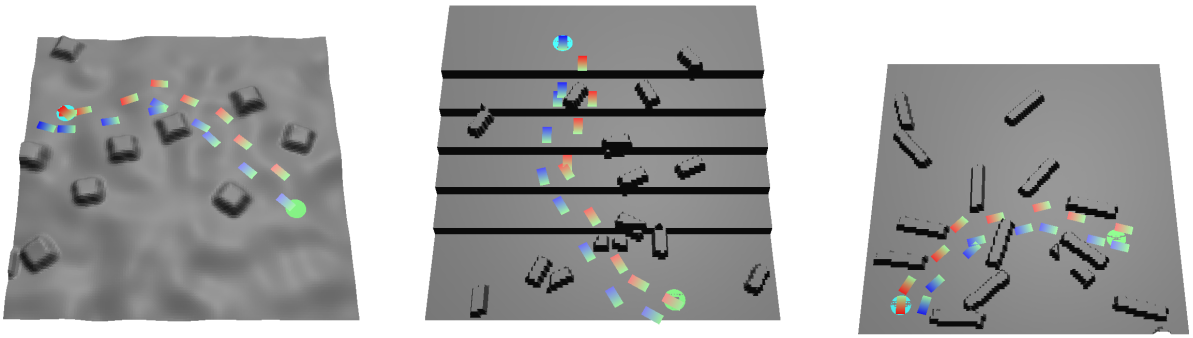


Figure 6: Example plans generated by the planner.

general, the time complexity of A* search is an exponential function of the error in the heuristic used [75]. So while in many environments, this heuristic performs much better than Euclidean distance, the worst case can be an arbitrarily large overestimate. Some examples of heuristics are shown in Figure 5. For the Euclidean heuristic, the first half of the path has approximately the same value, while the heuristic from the mobile robot planner pushes the planner in the correct direction from the start. In this example, the more informed heuristic planned the same path in one-fourth the time.

Using these representations, metrics, and heuristics, we now have the necessary components to search through the space of footholds, and plan paths through many different types of terrains. Examples of the paths generated for some environments are shown in Figure 6.

4.3 Online Execution

There are several additional components necessary to control a legged robot for real-time operation in a complicated environment. The complete system we use is shown

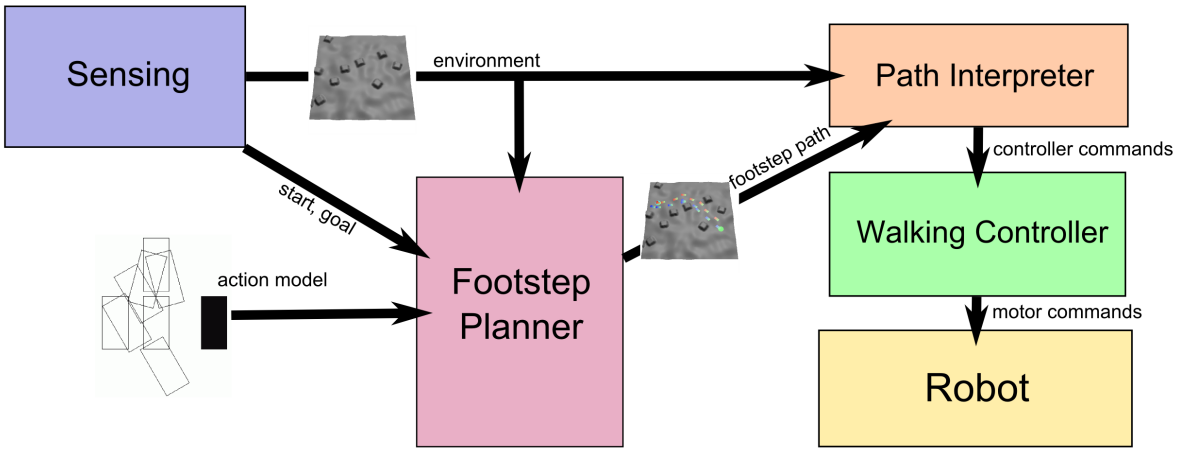


Figure 7: Overall system used to control a robot

in Figure 7. First, a sensing system must provide adequate information to the planner to initially find a path. Second, we need to translate the footstep plan into commands for the low-level locomotion controller which will generate the full motion for the robot. For example, when HRP-2 is walking up or down stairs, or stepping over obstacles on the floor, the walking controller must be told both the footstep locations and how to modify the swing leg trajectory to avoid collisions with the environment. For the ASIMO humanoid, the plan must be translated into the particular navigation commands for the robot that will cause the feet to reach the desired steps. Third, we must be able to adjust for new sensor data and changes in the environment which necessitate a change in the plan.

For sensing systems, we have used many different sensors and approaches to gather the needed data. Some sensor systems which have been used for this footstep planning include stereo vision[17], color segmentation[62], real-time motion capture data[86], and edge-based model tracking[61].

For determining the swing leg trajectory, our approach has been to move the foot directly from one support to the next, adjusting the height to avoid the terrain. By moving the foot along a straight line in the $x - y$ plane, the cells in the height map can be re-mapped to a representation of (d, h), where d is the distance along that line, and h is the height of the terrain in that cell. This representation is a one-dimensional height map from which we construct a convex hull of the terrain. This convex hull (combined with a safety margin) is then used to build a spline over the terrain for the foot to follow.

While executing the plan, the robot can run into many additional difficulties. Execution error can result in the robot deviating from the plan, sensor error can result in the initial plan being unexecutable, or the environment can change, rendering parts of the plan invalid. We deal with these errors in several ways. The first is to have the low-level execution constantly be eliminating error in the plan following to return to the correct sequence of stances. As long as the connectivity of the robot's state with the rest of the path is not broken, the robot can return to correct motion, even in the presence of execution errors. When the error is large enough that connectivity is not maintained, or when new knowledge of the environment invalidates part of the existing path, the plan must be

adjusted to remain feasible and still reach the goal. In many of our implementations, this adjustment is accomplished by constantly replanning while the robot walks, generating a new plan at every step. We limit the planning time to fit within one or two step cycles, which introduces a planning horizon. As a result, in difficult terrains we may not have a complete plan by the end of the cycle. But because we are replanning at every step, we can begin executing a partial path, and be assured of a new path by the end of the next step.

If the planner cannot plan all the way to the goal, it can no longer guarantee completeness and the quality of the partial path becomes crucial. The partial path should take the robot in the "right" direction, in order to find a complete path in the future. In addition, with a limited sensing horizon, planning all the way to the goal may not be possible or desirable. Instead, the robot must rely on other estimates for the areas that the planner has not yet explored, or where sensor data is lacking. This limited horizon places extra importance on the heuristic used, so that a partial path will still be taking the robot along a route from which it will be able to find a continuing path to the goal. The mobile robot planner heuristic discussed so far was tuned by hand to generate reasonable values, but recent research by Ratliff et al.[79] has allowed the heuristic to be trained from example footstep paths, resulting in speedups of over 100 times the basic Euclidean heuristic, and 20 times the hand-tuned heuristic for some sample terrains.

With all of these components assembled, we have been able to plan through obstacle-filled environments, with moving obstacles, a moving goal, and terrains that require the robot to step onto and over obstacles. An example trial with a moving goal and moving obstacles are shown in Figure 8 with the plans overlaid on the world.

5 Dynamics

In the previous algorithm, the state of the robot used in the planner only contained the support configuration. However, in many cases, the connectivity of two points in different support configurations is dependent on more of the full state of the robot. This section describes extensions made to the planning process to account for some vehicle and environment dynamics.

For the ASIMO robot, the effects of the commands that we send the robot varies with the body velocity of the robot. Figure 9 shows the difference between executing actions from a stand-still, and executing the same actions while already moving forward. In this case, the planner's representation of state must include some notion of body velocity, and the action model must describe how the connectivity of support configurations changes with the different possible velocities.

For ASIMO, there is a layer of control that we have cannot access. As a result, we do not have access to the full state information of the robot. However, we do know the history of commands we have sent. By measuring the robot's response to sequences of commands, we were able to build a forward model of the output motion of the robot. We found that augmenting our state representation with the previous two commands given to the robot was sufficient to describe the effect of the robot's velocity on the subsequent command. Thus, our state representation becomes

$$(x, y, \theta, s, a_1, a_2) \in \Re^2 \times [0, 2\pi) \times \{L, R\} \times Z^2$$

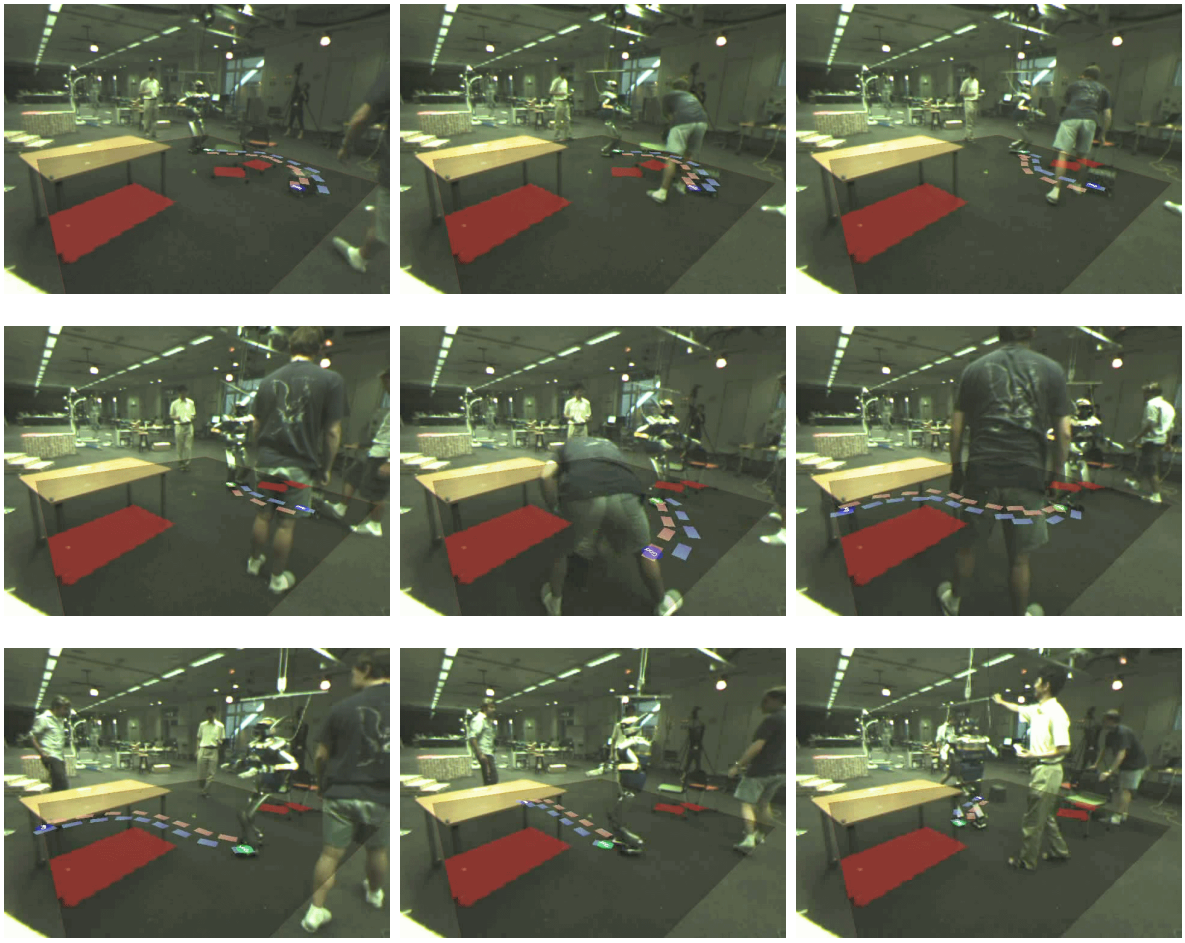


Figure 8: HRP-2 replanning through an environment with moving obstacles and a moving goal. The generated footstep path is overlaid onto the world.

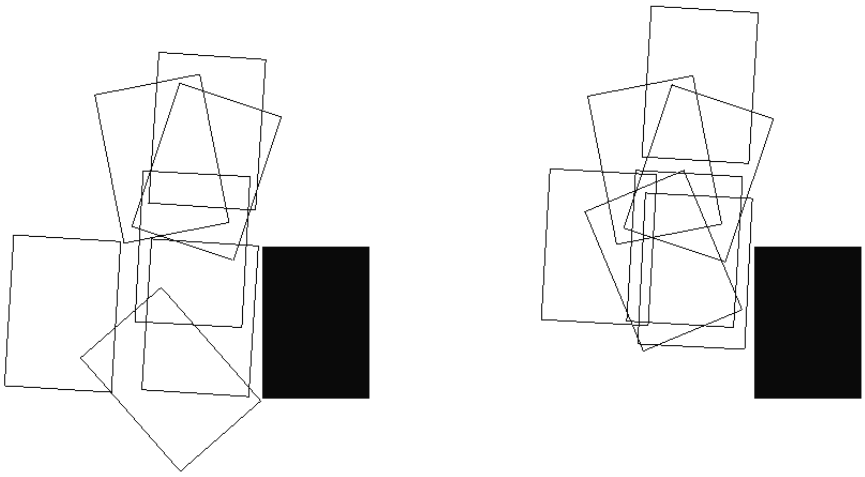


Figure 9: Body velocity dependence of the effects of the actions. The black blocks represent the location of the right foot, and the rectangles are the resulting locations of the left foot from each commanded action. The commands given to the robot are not shown. *Left:* Effects of commanded actions from standing still. *Right:* Effects of commanded actions from full speed forward walking.

, where a_1 and a_2 are the previous two commands sent to the robot.

The action model was created by having ASIMO perform sequences of commands, and recording the resulting motion. The motion was captured using the Vicon[1] optical system. Twelve cameras were used, and each one captures data with a frequency of 120 Hz and a resolution of 1000 x 1000. Six markers were placed on the robot's feet. From the positions of these markers, we determined the positions and orientations of both the left and right feet. We then computed the relative displacements of the feet from this data. We chose a set of seven actions for each foot, and with the displacements dependent on the previous two actions, we captured 343 sequences of commands to cover all possibilities. Parsing this data gave us a mapping from a state and action to the output state for the each of the seven selected actions, which was then used as a lookup table during planning. This modeling would not have been necessary if we had access additional to additional state information directly from the robot, but it does demonstrate that we can in some cases build compact forward models for complicated systems even when the underlying controller is not well-understood. Using this mapping meant that the robot dynamics were all accounted for during the initial planning, and that paths through a static environment could be executed with no replanning necessary.

A relatively straightforward extension allows us to plan for known environment dynamics. While the state representations discussed did not include time, it is a relatively easy task to augment the state representation with time, and modify the action model to include the duration of the various actions. Therefore, when changes in the environment can be predicted, the state evaluation for a particular state can be performed against

[1]Vicon is a trademark of Vicon Motion Systems, Ltd.

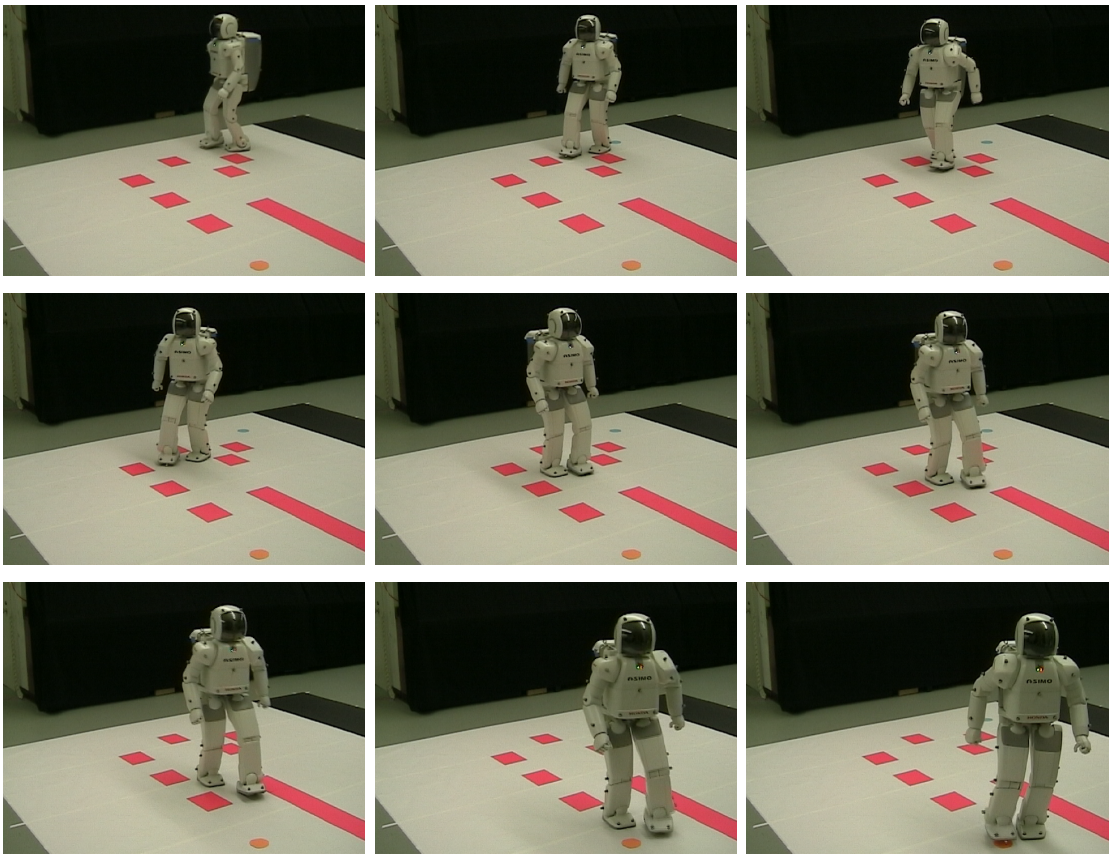


Figure 10: ASIMO navigating in an environment with static obstacles.

Figure 11: The LittleDog robot executing a plan over rough terrain.

the predicted future state of the environment, allowing the robot to navigate through dynamic environments without the need for replanning. This was also implemented for the ASIMO humanoid robot, in environments with predictably moving obstacles[18].

6 Quadruped planning

The algorithms described this far have all been using biped robot models and applied to humanoid robots. However, legged robots can come in many forms, with more than just two legs. This section describes some of our work on planning for a quadruped robot.

One significant change between a biped and a robot with 4 or more legs is that the extra legs mean that the robot no longer needs to spend time supported by a single foothold. For a biped, it is imperative to ensure that each foothold is secure, so that the robot can support itself solely from that particular location. However, with a quadruped robot, each foot is less important than the *combination* of feet which make up the total contact configuration with the environment. In addition, quadrupeds and hexapeds have much greater inherent stability due to their larger bases of support. This greater stability also changes the importance of individual footholds, as a small slip or stub for a quadruped is not the emergency it is for current biped robots. Finally, when moving to more legs, the feet are often much simpler, often nearly point feet, which changes the criteria for what constitutes a "good" location to step to.

However, having stated that, it is still possible to use the previous biped planner with few modifications and successfully plan for a quadruped robot. While it is the combination of footholds that is important for each step, a set of excellent of individually

excellent footholds will provide excellent support, with the additional constraint of a sufficient support polygon between the footholds. This allows us to still process location costs individually over the terrain and be able to traverse a wide range of environments.

In addition to treating the footholds in the same manner as the biped planner, the action model can still largely treat the robot in the same manner as a biped. In our current implementation, the planner uses the front legs similarly to how it planned for a biped, and then has the rear legs follow along through the terrain. While this is not as useful a model if the robot needs to move backwards or perform lots of turning maneuvers, it provides a simple action representation for walking across interesting terrains.

We have implemented a planner for the quadruped LittleDog (shown in Figure 11). This planner is based on the previously described biped algorithm. The state has been augmented to include the four legs of the robot, with each action changing the configuration of one of them. In addition, the location cost metrics changed to reflect the different needs of point feet. While a bump under the foot was a large problem for a biped, causing the foot to not lie flat, it is not such an important issue for point feet. Instead, the metric is reversed, penalizing holes and dropoffs from which the foot can slip. Also, a walking biped often can generate large forces and torques through friction, which prevents the feet from slipping. With many point feet the force is more distributed, so in more challenging environments the footholds need to provide a larger degree of shape closure to hold the feet in place during stepping motions.

Furthermore, for the kinematics of this robot and the terrains involved, the legs themselves can easily collide with the terrain in many cases. We are not currently considering support configurations with more than just the feet in contact, so these become invalid step locations. We add this to the location cost by computing the space the leg will need for the given location, leg, and direction, and make sure that area is clear of terrain features. All of these terrain location costs are precomputed over a discretization of position and orientation and used as a lookup table at runtime.

As a result of these modifications and extensions to the biped planning algorithm, The LittleDog robot has successfully traversed a variety of different terrain setups. Figure 12 shows a quadruped plan being generated over one rough terrain example, along with the results of the pre-computed location costs.

7 Adaptive action set

A large drawback to the action models described this far is the fact that the actions the robot can take are limited to a very small subset of the robot's potential. By using a fixed set of actions, the planner can only solve problems for which that particular action set "fits." We can choose that action set so that it can still traverse a wide variety of terrains, but ultimately, certain terrains will be untraversable simply because the needed action was not a part of our model. To illustrate this problem imagine sets of stairs, where the particular desired step length for climbing them might not be in the action set. An example is would be stepping stones across a river, where a very specific sequence of actions is required to negotiate the terrain. To remove this action set limitation, we need the action model to be capable of making any action that the robot and controller are capable of performing. This section describes some of our previous work extending our

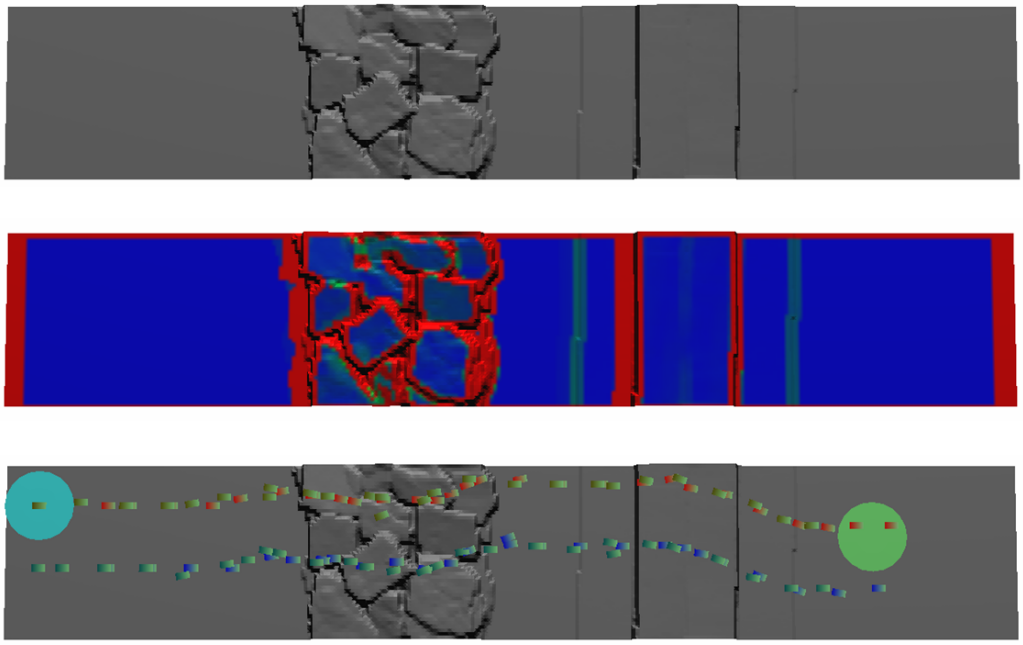


Figure 12: An example terrain with pre-computed location costs shown and the resulting footstep plan for a quadruped walking from the left to the right.

action model to provide more complete coverage of the true action space of the robot.

Because the robot and controller are generally going to have a continuous space of actions available to them, adding all possible actions or even a large fixed number of samples to one giant action set to try at run time is not practical, due to the increase in branching factor. To maintain a constant branching factor b, we would like to only use the best b actions available for a particular state taking into account obstacles in the environment. Unfortunately, we cannot know what the best actions are without solving the problem, and choosing the best b actions according to our heuristic will likely result in a clump of actions, precluding any significant search over the space.

Other than our heuristic value, we also want to maximize the reachability of the next step. If we want to be sure that we are not cutting off some part of the space from our search unnecessarily, we must make sure that the expansion in the search has the potential to reach as many states as possible. Unfortunately, finding an action set maximizing the reachability of the next step in the presence of arbitrary obstacles is itself a computationally difficult problem. Our current approach to this problem is to use a reference set of actions, which provide good reachability, and allow those actions to adjust to the particular terrain via a local search.

The new action model in our implementation has largely the same representation as our previous action models, with the exception of an explicit definition of the reachability of the robot and controller. Because the action may now adjust locally, the actions may leave the realm of executability which was previously built into the action model via the fixed sampling. The explicit definition of reachability allows the local search to move the action up to the edge of feasibility, but not beyond. During planning, when an action

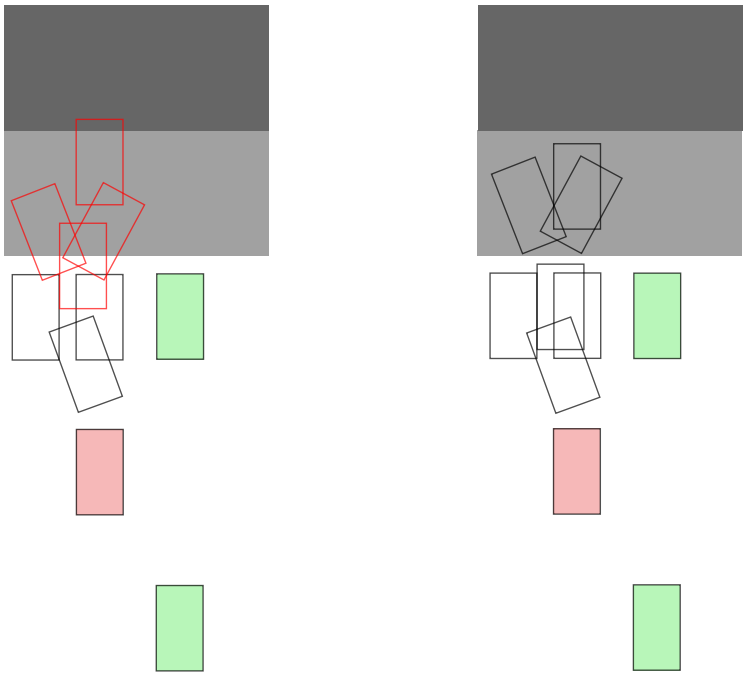


Figure 13: Adapting actions to the terrain with stairs.*Left:* Several actions from a fixed set of action samples (the open rectangles) provide invalid steps (marked red), with no valid action from the set climbing the stair. *Right:* The reference actions are adapted to find the closest fit in the terrain. With the result that the robot now has actions which can directly step up onto the stair.

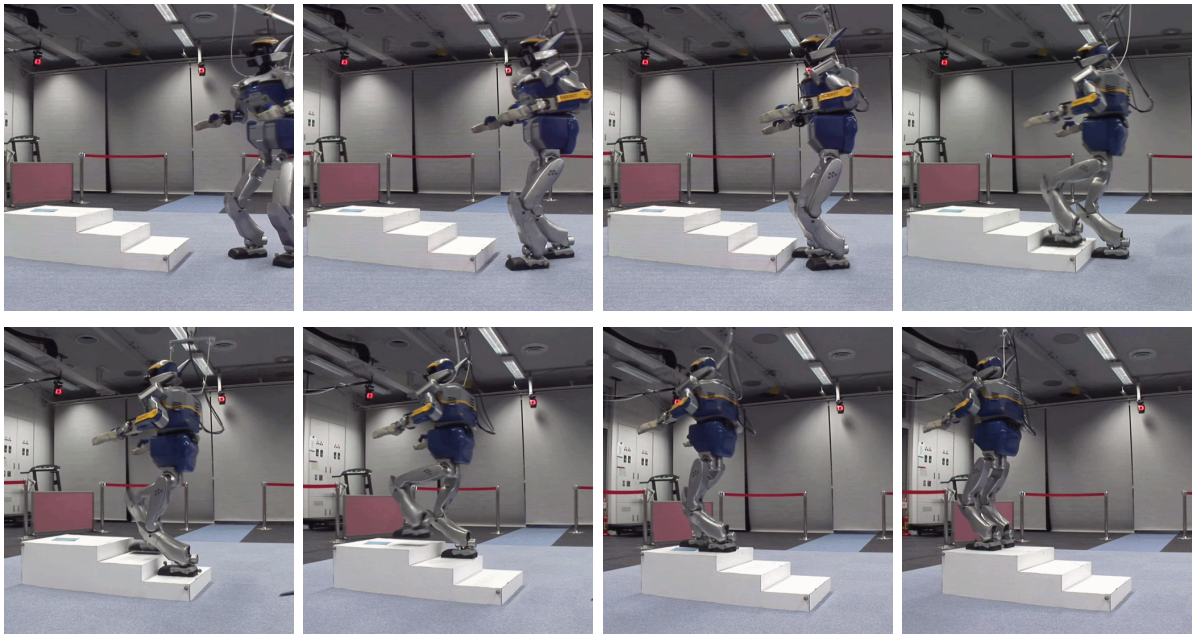


Figure 14: HRP-2 climbing stairs using adaptive actions

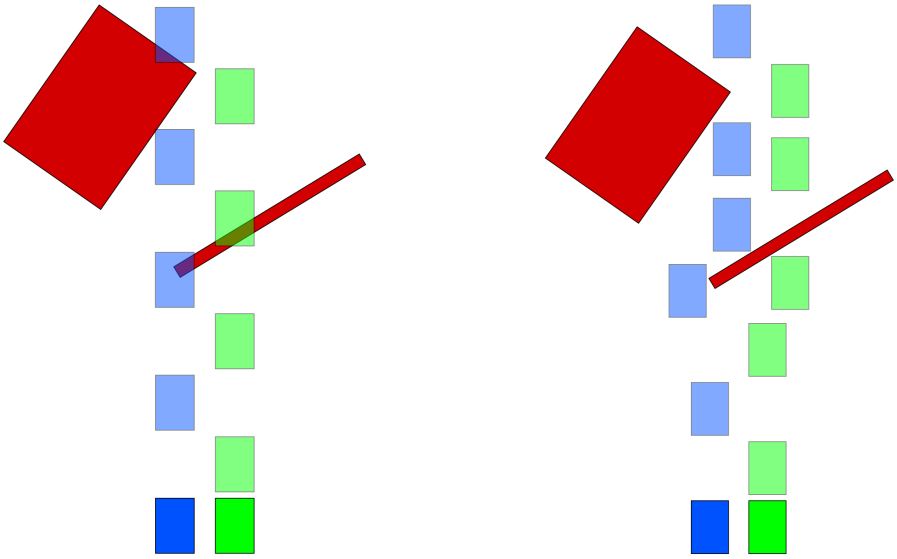
is applied, if it results in a valid footstep, the algorithm is unchanged. However, if the footstep is invalid for some reason, the algorithm begins successively testing the nearest locations in the environment up to a certain radius for a valid step. If one is found, it is inserted into the queue and the planner moves on to the next action. Figure 13 shows the adjustment performed as a result of the local search to one action for a particular terrain. This method provides a constant branching factor, while still allowing the planner to find paths where very specific steps are required.

For static environments, such as the quadruped example, the location costs are all pre-computed, so this local search can be performed at low computational cost. In fact, the ability to adapt the actions to the terrain became critical to allowing the quadruped to traverse some of the more difficult terrains, where valid footholds become sparse.

This same approach is usable in real-time planning, when the location costs have not been pre-computed. However, in this case, the system is limited by how many locations it can process in the given time limit, which restricts the environment complexity and length of the path with can be planned for. This system has been used on the humanoid HRP-2 to traverse environments in which the desired actions were not a part of the reference set. The robot, shown in Figure 14, had a desired step length of 27cm, while the stairs themselves were 30cm long. The planner adjusted the step location through its local search, allowing the robot to smoothly climb the stairs without explicitly needing the correct action for that staircase known in advance.

8 An Intelligent Joystick

A slight departure from the planners described thus far comes from incorporating human control to pilot the robot, instead of using full autonomy. Joystick control has previously been used in positioning and navigation for current humanoid robots[46], although this



(a) Conventional joystick (b) Intelligent joystick

Figure 15: Comparison of Conventional vs. Intelligent Joystick output given a command to walk forward in the presence of obstacles.

work does not account for obstacles and ammounts to "steering" the robot. More direct control of individual legs has been implemented as far back as the GE truck[63], which used force feedback for the operator to indivually control the legs. The Adaptive Suspension Vehicle used several different operating modes[90], one allowing the operator to directly control foot placement, a low-speed mode which progressed by "feeling" the terrain, a more autonomous mode where the joysticks control body position and orientation and foot locations are determined more autonomously, and a higher speed walk which did not account for obstacles. The framework described in this proposal can also be used for human control by replacing the planning portion with a value function generated provided by the user. The location and step evaluation can remain the same, with the choice of the "best" action resulting from how well each possible action matches the user's input. This section describes some of our previous work developing a control scheme to provide user control over the path the robot should take. The result of this system is an "intelligent" joystick control, which can safely avoid obstacles and choose footholds while obeying the user's navigational choices.

The idea of an intelligent joystick can be compared to riding a horse: the rider provides high-level control inputs about which direction to travel, but the horse handles all of the details of locomotion, including the complexities of selecting suitable foot placements and the overstepping of obstacles along the way. In the case of a legged robot, the joystick controls the overall movement direction of the robot, but the system autonomously selects foot placements and trajectories which best conform to the user's command given the constraints of balance and terrain characteristics. Figure 15 demonstrates how the intelligent joystick modifies the foot locations during a command to walk forward. A naive joystick controller generates the same walking pattern, regardless of obstacles in the path. An intelligent joystick will place the feet at the most suitable locations it can

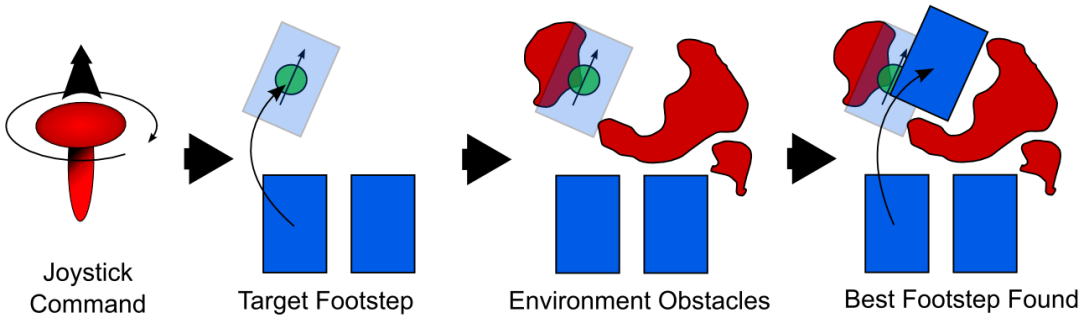


Figure 16: Foot placement selection for a joystick command of forward while turning to the right

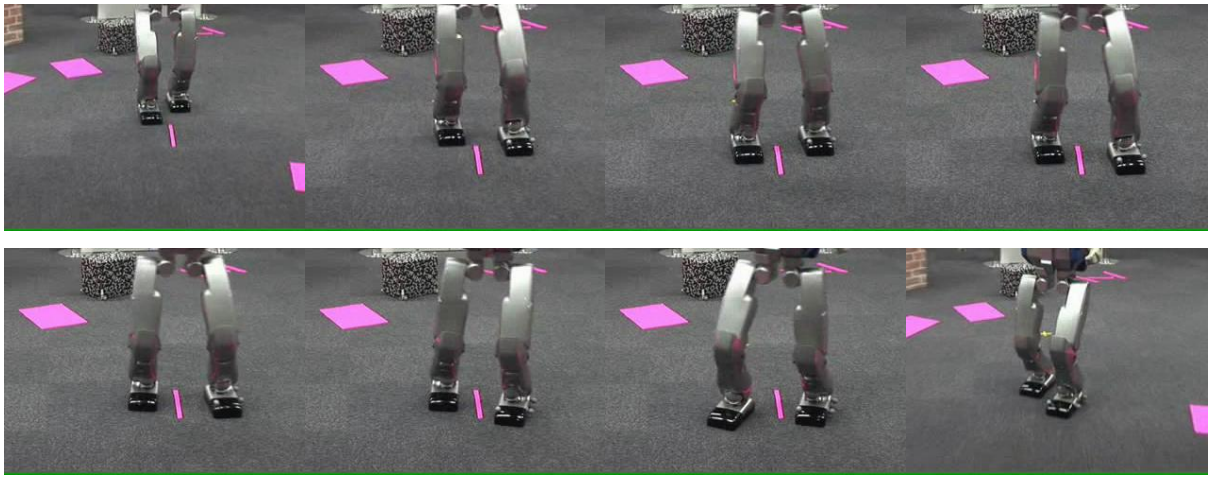


Figure 17: Robot controlled via intelligent joystick. The given command is "forward and turning to the right". The intelligent joystick executes the command while splaying its feet outward to avoid the small obstacle.

find while still making forward progress as commanded.

For controlling the humanoid robot in our experiments, we use a 3-axis joystick. This provides a simple mechanism to command forward motion, sideways motion, and rotation simultaneously through one interface.

Given a robot state $x_{current}$, an environment, e, and a joystick command $(\dot{x}, \dot{y}, \dot{\theta})$, the system's task is to determine the best walking action to follow the user's command that will still maintain balance when the environment is taken into account. In other words, determine the next stance state x_{next}, which satisfies balance requirements with respect to the environment, e, and brings the robot's velocity as close to the commanded $(\dot{x}, \dot{y}, \dot{\theta})$ as possible. To accomplish this, the system chooses a target location where it would like the next foot to land based on the commanded velocity and step period, and then evaluates that location and the locations nearby in the same manner as the adaptive actions described earlier to determine the closest location to the desired target location that is most suitable for stepping onto. Figure 16 illustrates the selection of a target foot placement from a joystick command. This location is then sent to the walking control subsystem of the robot, for walking trajectory generation. The overall control algorithm is described in Algorithm 2.

Algorithm 2: Joystick Control algorithm

```
previous ← initial_value;
step ← initial_value;
while still walking do
    // Gather information
    env ← GetEnvironment();
    robot ← GetRobotLocation();
    joy_cmd ← GetJoystickCommand();
    // Compute start, target locations
    start ← ComputeStanceLocation(robot, step);
    target ← ComputeTargetLocation(start, joy_cmd);
    // Get step from Footstep Server
    step ← GetNextStep(env, start, previous, target);
    // Update robot command
    if step = NULL then
        StopWalking();
    else
        SendRobotCmd(step);
        WaitForNextStep();
        previous ← start;
    end
end
```

This form of control has been implemented for the humanoid HRP-2, allowing for easy human control and steering of the robot through environments with obstacles, without requiring the human to specify low-level control of foot placement or leg trajectories. A trial involving colored-paper obstacles on the floor is shown in Figure 17.

9 Proposed Work

For this thesis I propose to develop a general framework for planning for a wide variety of legged robots through rough terrain. This framework is based on reasoning about the successive support configurations in a robot's motion, determining the suitability and connectivity of various stances in the environment.

9.1 Action model exploration

I propose to explore strategies for creating, representing, and efficiently planning with footstep-based action models. To date, we have developed action models to to efficiently navigate four different legged robots. These action models allow the robots to use their legged capabilities to traverse obstacle-filled environments, step onto and over obstacles, climb and descend stairs, and handle some vehicle and environment dynamics. We have extended these action models to plan for a quadruped robot through a series of rough terrain setups. Furthermore, we have designed action sets which can adapt to the terrain during the planning process.

Adapting the available actions to the terrain is an important part of using more of the robot's total action capabilities while keeping the branching factor of the search small. However, the current implementation involves a blind local search of the terrain near the reference action's foothold. This is not very expensive when foothold costs have been precomputed, but when those costs are computed online, they become a computational bottleneck in rough terrain. To speed this process, I propose to develop "smarter" metrics, which can provide not only information about the foothold cost, but additional information about how the footstep could be adjusted to improved that cost. This additional information can be used to improve the local search to find valid footholds with fewer terrain evaluations.

We will experimentally evaluate the benefits of this approach and all the previous action models on the HRP-2 humanoid in environments with long paths and dynamic obstacles.

9.2 Intelligent joystick through rough terrain

I propose to explore methods of allowing humans to provide high-level control of legged robot navigation, while hiding the complexities of balance, joint movement and foot placement from the user. We have developed a joystick-based control which has allowed HRP-2 to be manually driven in the presence of obstacles, intelligently placing its feet and maintaining balance while following the user's commands. The current implementation of the intelligent joystick has only been tested on flat terrain. Together with the new adaptive actions described above, I propose to extend the intelligent joystick work to traverse rough terrain with a low response time, potentially adjusting several times during each step to both avoid dynamic obstacles and respond quickly to user input.

This control will be evaluated on the HRP-2 humanoid and the Toyota iFoot walking chair.

9.3 Navigation planning for running robots

I propose to extend the models currently used to allow for more complicated vehicle dynamics. While the planning for Honda's ASIMO had to account for body velocities, the final motions were not highly dynamic. We would like to extend this planner to handle highly dynamic actions, such as running. To handle these actions, we will use two approaches. The first approach, similar to the Bow-leg hopper and ASIMO planner, is to plan for a sampling of the control space of the robot, predicting the next foothold from the current state and the set of chosen control actions. The second approach is to encode how much the body velocity can be modified by control actions during a step, and choose desired footholds in the computed reachable region. From the footholds, we then computed the desired control parameters for following the desired path. These approaches will be compared on a robot running in simulation.

10 Contributions

The contributions of this thesis are a framework for approaching legged locomotion navigation, a series of planners for the navigation of legged robots through complicated

environments. Another contribution is the exploration of various methods of action representation to reduce the planning dimensionality for various legged robots and controllers, and strategies for evaluating foot placement and compactly representing the connectivity of those foot placements. These contributions are immediately applicable to existing humanoids, and other legged robots. Additionally, this thesis will explore a high-level joystick interface to controlling legged robots in real-time. This interface can allow for easy manual control of legged robots in the presence of obstacles, as well as an interface for recently developed "walking chairs."

11 Proposed Schedule

Task	Date
Planning for biped navigation in 2D and 3D	done
Planning using additional velocity dependencies	done
Online replanning for biped	done
Planning for quadruped navigation in 2D and 3D	done
Adaptive local search for foot placement	Winter 2006
Human-controlled adaptive search in 3D	Spring 2007
Planning for running biped	Spring 2007
Defend and Write thesis	Summer 2007

References

[1] M. Ahmadi and M. Buehler. The ARL monopod II running robot: Control and energetics. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1689–1694, May 1999.

[2] K. Akachi, K. Kaneko, N. Kanehira, S. Ota, G. MiyaMori, M. Hirata, S. Kajita, and F. Kanehiro. Development of humanoid robot hrp-3p. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2005.

[3] Y. Ayaz, K. Munawar, M. B. Malik, A. Konno, and M. Uchiyama. Human-like approach to footstep planning among obstacles for humanoid robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[4] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. R. L. Whittaker. Ambler: An autonomous rover for planetary exploration. *IEEE Computer*, 22(6):18–26, June 1989.

[5] J. E. Bares and D. S. Wettergreen. Dante ii: Technical description, results, and lessons learned. *International Journal of Robotics Research*, 18(7):621–649, July 1999.

[6] J. Barraquand and P. Ferbach. Path planning through variational dynamic programming. Technical Report 33, Digital - PRL Research Report, sep 1993.

[7] J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. *Algorithmica*, 10:121–155, 1993.

[8] T. Bartholet. Odetics develops the worrld's first functionoid. *Nuclear Engineering International*, 30:37–38, 1985.

[9] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, New Jersey, 1957.

[10] J. Bobrow, S. Dubowsky, and J. Gibson. Time-optimal control of robotic manipulators. *International Journal of Robotics Research*, 4(3), 1985.

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

[12] T. Bretl, S. Rock, and J.-C. Latombe. Motion planning for a three-limbed climbing robot in vertical natural terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 2003.

[13] J. G. Cham, S. A. Baily, and M. R. Cutcosky. Robust dynamic locomotion through feedforward-preflex interaction. In *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*, Orlando, FL, November 2000.

[14] M. Cherif and K. Gupta. Planning quasi-static motions for re-configuring objects with a multi-fingered robotic hand. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997.

[15] M. Cherif and K. K. Gupta. Planning quasi-static fingertip manipulations for reconfiguring objects. *IEEE Transactions on Robotics and Automation*, 15(5):837–848, Oct 1999.

[16] S. Chernova and M. Veloso. An evolutionary approach to gait learning for four-legged robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 2004.

[17] J. Chestnutt, J. Kuffner, K. Nishiwaki, and S. Kagami. Planning biped navigation strategies in complex environments. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, Karlsruhe, Germany, October 2003.

[18] J. Chestnutt, M. Lau, G. Cheng, J. Kuffner, J. Hodgins, and T. Kanade. Footstep planning for the Honda ASIMO humanoid. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[19] B. Donald, P. Xavier, J. Canny, and J. Reif. Kinodynamic motion planning. *Journal of the ACM*, 40(5):1048–1066, Nov. 1993.

[20] M. Fujita, K. Sabe, Y. Kuroki, T. Ishida, and T. Doi. SDR-4X II: A small humanoid as an entertainer in home environment. In *Proceedings of the International Symposium on Robotics Research*, 2003.

[21] Y. Fukuoka, H. Kimura, and A. H. Cohen. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. *International Journal of Robotics Research*, 22(3-4):187–202, 2003.

[22] M. Girard. Interactive design of computer-animated legged animal motion. *IEEE Computer Graphics & Applications*, 7(6):39–51, June 1987.

[23] J.-S. Gutmann, M. Fukuchi, and M. Fujita. A floor and obstacle height map for 3d navigation of a humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[24] L. Han and J. C. Trinke. Dextrous manipulation by rolling and finger gaiting. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1998.

[25] K. Hauser, T. Bretl, and J.-C. Latombe. Learning-assisted multi-step planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[26] K. Hauser, T. Bretl, and J.-C. Latombe. Non-gaited humanoid locomotion planning. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, Tsukuba, Japan, 2005.

[27] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of Honda humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1321–1326, May 1998.

[28] S. Hirose. A study of design and control of a quadruped walking vehicle. *International Journal of Robotics Research*, 3(2):113–133, Summer 1984.

[29] J. K. Hodgins and M. H. Raibert. Adjusting step length for rough terrain locomotion. *IEEE Transactions on Robotics and Automation*, 7(3):289–298, June 1991.

[30] R. Hodoshima, T. Doi, Y. Fukuda, S. Hirose, T. Okamoto, and J. Mori. Development of TITAN XI: a quadruped walking robot to work on slopes - design of system and mechanism. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[31] Y. K. Hwang and N. Ahuja. A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32, Feb. 1992.

[32] S. Kagami, K. Nishiwaki, J. J. Kuffner, Y. Kuniyoshi, M. Inaba, and H. Inoue. Design and implementation of software research platform for humanoid robots: H7. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, Tokyo, Japan, November 2001.

[33] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, and H. Hirukawa. A realtiine pattern generator for biped walking. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002.

[34] M. Kallmann, R. Bargmann, and M. Mataric'. Planning the sequencing of movement primitives. In *Proceedings of the International Conference on Simulation of Adaptive Behavior*, 2004.

[35] F. Kanehiro, T. Yoshimi, S. Kajita, M. Morisawa, K. Fujiwara, K. Harada, K. Kaneko, H. Hirukawa, and F. Tomita. Whole body locomotion planning of humanoid robots based on a 3d grid map. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[36] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot hrp-2. In *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 2004.

[37] S. Kim, J. Clark, and M. Cutkosky. isprawl: Design and tuning for high-speed autonomous open-loop running. *International Journal of Robotics Research*, 25(9), September 2006.

[38] H. Kimura, Y. Fukuoka, and K. Konaga. Adaptive dynamic walking of a quadruped robot by using neural system model. *Advanced Robotics*, 15(8):859–876, 2001.

[39] N. Kohl and P. Stone. Machine learning for fast quadrupedal locomotion. In *Proceedings of the National Conference on Artificial Intelligence*, July 2004.

[40] E. Kokkevis, D. Metaxas, and N. I. Badler. Autonomous animation and control of four-legged animals. In *Proc. Graphics Interface*, pages 10–17, May 1995. ISBN 0-9695338-4-5.

[41] R. Korf. Artificial intelligence search algorithms. In M. Atallah, editor, *CRC Handbook of Algorithms and Theory of Computation*, pages 36.1–20, Boca Raton, FL, 1998. CRC Press.

[42] E. Krotkov, J. Bares, T. Kanade, T. Mitchell, R. Simmons, and W. R. L. Whittaker. Ambler: a six-legged planetary rover. In *Fifth International Conference on Advanced Robotics, 1991, Robots in Unstructured Environments (ICAR '91)*, volume 1, pages 717 – 722, June 1991.

[43] J. Kuffner. Goal-directed navigation for animated characters using real-time path planning and control. In *Proc. CAPTECH '98 : Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, pages 171–186, 1998.

[44] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue. Dynamically-stablemotion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, January 2002.

[45] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. Footstep planning among obstacles for biped robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 500–505, 2001.

[46] J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue. Self-collision detection and prevention for humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, D.C., May 2002.

[47] J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue. Online footstep planning for humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003.

[48] J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[49] M. Lau and J. Kuffner. Behavior planning for character animation. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2005.

[50] S. M. LaValle. *A Game-Theoretic Framework for Robot Motion Planning*. PhD thesis, University of Illinois, Urbana, IL, July 1995.

[51] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[52] H. Lee, Y. Shen, C.-H. Yu, G. Singh, and A. Y. Ng. Quadruped robot obstacle negotiation via reinforcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[53] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, 2002.

[54] T.-Y. Li, P.-F. Chen, and P.-Z. Huang. Motion planning for humanoid walking in a layered environment. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003.

[55] K. Löffler and M. Genger. Sensors and control concept of walking 'Johnnie'. *International Journal of Robotics Research*, 22(3-4):229–240, 2003.

[56] O. Lorch, A. Albert, J. Denk, M. Gerecke, R. Cupec, J. Seara, W. Gerth, and G. Schmidt. Experiments in vision-guided biped walking. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[57] O. Lorch, J. Denk, J. F. Seara, M. Buss, F. Freyberger, and G. Schmidt. ViGWaM - an emulation environment for a vision guided virtual walking machine. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2000.

[58] K. M. Lynch and M. T. Mason. Stable pushing: Mechanics, controllability, and planning. *Int. J. Robotics Research.*, 15(6):533–556, Dec. 1996.

[59] R. B. McGhee. Robot locomotion with active terrain accommodation. In *Proceedings of the National Science Foundation Robotics Research Workshop*, 1980.

[60] R. B. McGhee and G. I. Iswandhi. Adaptive locomotion of a multilegged robot over rough terrain. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:176–182, 1979.

[61] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade. GPU-accelerated real-time 3D tracking for humanoid locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007.

[62] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade. Vision-guided humanoid footstep planning for dynamic environments. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, December 2005.

[63] R. S. Mosher. Exploring the potential of a quadruped. International Automotive Engineering Congress, January 1969.

[64] K. Nagasaka, M. Inaba, and H. Inoue. Walking pattern generation for a humanoid robot based on optimal gradient method. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, 1999.

[65] K. Nishiwaki and S. Kagami. High frequency walking pattern generation based on preview control of ZMP. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[66] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue. Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 96–101, 2002.

[67] K. Nishiwaki, T. Sugihara, S. Kagami, M. Inaba, and H. Inoue. Online mixture and connection of basic motions for humanoid walking control by footprint specification. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001.

[68] Y. Ogura, H. Aikawa, K. Shimomura, H. Kondo, A. Morishima, H. ok Lim, and A. Takanishi. Development of a new humanoid robot WABIAN-2. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[69] Y. Ogura, H. ok Lim, and A. Takanishi. Stretch walking pattern generation for a biped humanoid robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 352–357, 2003.

[70] K. Okada, T. Ogura, A. Haneda, and M. Inaba. Autonomous 3D walking system for a humanoid robot based on visual step recognition and 3D foot step planner. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

[71] I.-W. Park, J.-Y. Kim, J. Lee, and J.-H. Oh. Mechanical design of humanoid robot platform KHR-3. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2005.

[72] I.-W. Park, J.-Y. Kim, J. Lee, and J.-H. Oh. Online free walking trajectory generation for biped humanoid robot KHR-3(HUBO). In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[73] A. Patla, A. Adkin, C. Martin, R. Holden, and S. Prentice. Characteristics of voluntary visual sampling of the environment for safe locomotion over different terrains. *Exp. Brain Res.*, 112:513–522, 1996.

[74] A. Patla, E. Niechwiej, and L. Santos. Local path planning during human locomotion over irregular terrain. In *Proc. AMAM2000*, 2000.

[75] J. Pearl. *Heuristics*. Addison-Wesley, Reading, Ma., 1984.

[76] J. Pettre, J.-P. Laumond, and T. Simeon. A 2-stages locomotion planner for digital actors. In *Proc. SIGGRAPH Symp. on Computer Animation*, 2003.

[77] J. Pratt and G. Pratt. Exploiting natural dynamics in the control of a 3d bipedal walking simulation. In *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR99)*, Portsmouth, UK, September 1999.

[78] M. H. Raibert. *Legged Robots that Balance*. MIT Press, 1986.

[79] N. Ratliff, D. Bradley, D. Bagnell, and J. Chestnutt. Boosting structured prediction for imitation learning. In *Proceedings of Neural Information Processing Systems*, Vancouver, Canada, December 2006.

[80] K. Sabe, M. Fukuchi, J.-S. Gutmann, T. Ohashi, K. Kawamoto, , and T. Yoshigahara. Obstacle avoidance and path planning for humanoid robots using stereo vision. In *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, April 2004.

[81] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent ASIMO: System overview and integration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2478–2483, 2002.

[82] U. Saranli, M. Buehler, and D. Koditschek. RHex: A simple and highly mobile hexapod robot. *International Journal of Robotics Research*, 20(7):616–631, July 2001.

[83] Z. Shiller and S. Dubowsky. On computing time-optimal motions of robotic manipulators in the presence of obstacles. *IEEE Transactions on Robotics and Automation*, 7(7), Dec. 1991.

[84] Z. Shiller, K. Yamane, and Y. Nakamura. Planning motion patterns of human figures using a multi-layered grid and the dynamics filter. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001.

[85] A. Stentz. Optimal and efficient path planning for partially-known environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3310–3317, 1994.

[86] M. Stilman, P. Michel, J. Chestnutt, K. Nishiwaki, S. Kagami, and J. Kuffner. Augmented reality for robot development and experimentation. Technical Report CMU-RI-TR-05-55, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2005.

[87] N. Torkos and M. van de Panne. Footprint-based quadruped motion synthesis. In *Proc. Graphics Interface*, pages 151–160, 1998.

[88] M. van de Panne. From footprints to animation. *Computer Graphics Forum*, 16(4):211–223, October 1997.

[89] M. Vukobratovic, B. Borovac, D. Surla, and D. Stokic. *Biped Locomotion: Dynamics, Stability, Control, and Applications*. Springer-Verlag, Berlin, 1990.

[90] K. J. Waldron and R. B. McGhee. The adaptive suspension vehicle. *IEEE Control Systems Magazine*, 6:7–12, 1986.

[91] D. Wettergreen and C. Thorpe. Developing planning and reactive control for a hexapod robot. In *Proceedings of ICRA '96*, volume 3, pages 2718 – 2723, April 1996.

[92] M. Yagi and V. Lumelsky. Biped robot locomotion in scenes with unknown obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 375–380, Detroit, MI, May 1999.

[93] J. Yamaguchi, S. Inoue, D. Nishino, and A. Takanishi. Development of a bipedal humanoid robot having antagonistic driven joints and three dof trunk. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 96–101, 1998.

[94] G. Zeglin and H. B. Brown. Control of a bow leg hopping robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998.

[95] Z. Zhang, H. Kimura, and Y. Fukuoka. Self-stabilizing dynamics for a quadruped robot and extension towards running on rough terrain. *Journal of Robotics and Mechatronics*, 19(1), 2007.