An Adaptive Action Model for Legged Navigation Planning

Joel Chestnutt† Koichi Nishiwaki‡ James Kuffner†‡ Satoshi Kagami‡

†Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{chestnutt,kuffner}@cs.cmu.edu

‡Digital Human Research Center
AIST Waterfront 3F
2-41-6,Aomi,Koto-ku,Tokyo
135-0064, Japan
{k.nishiwaki,s.kagami}@aist.go.jp

***Abstract*— Navigation planning for legged robots via foot placement planning has enabled several humanoids to traverse interesting environments autonomously. In this paper we explore methods of adapting foot placement actions to the terrain during the search process, allowing for fuller use of the robot's capabilities, and better resulting paths. We show the results of these adaptive action models for both the humanoid HRP-2 and the quadruped LittleDog.**

I. Introduction

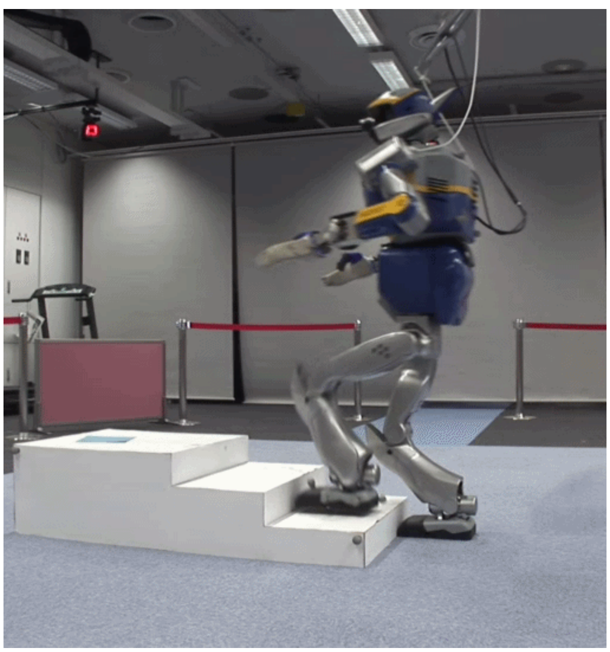

Fig. 1. HRP-2 humanoid autonomously walking up a set of stairs.

The design of algorithms to compute robust goal-directed navigation strategies for legged robots is an important area of research. For complex indoor environments designed for humans, this problem 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, many existing navigation planning methods fail to consider these additional capabilities, because they were primarily designed for wheeled mobile robots.

This paper presents an extension to our previous work, in which we have approached the problem of computing global navigation strategies for biped humanoids as one involving an iterated discrete search over a set of valid foot placements [5]. The result of the computation is a sequence of footstep placements that reach a goal region while minimizing encoded heuristics for effort, risk, or the number and complexity of the steps taken. As with other large search domains, computing true optimal solutions for biped navigation is computationally intractable. We have taken the approach of reducing the dimensionality of our search space by reasoning about the interaction with the environment, the step locations, and allowing the robot's locomotion controller to handle the motion of the the robot's many degrees of freedom within those stepping constraints. We have demonstrated the practicality of this approach on several robots, allowing these humanoids to traverse complicated and dynamic environments while maintaining balance and safety [5], [6], [20]. One drawback to the search approach used on these robots is that the stepping actions the planner considers are a discrete sampling of the robot's capabilities, fixed throughout the planning process. A result of this fixed sampling is that if the situation requires a particular action which is not present in the action sampling, the planner will be unable to find a solution. In addition, the fixed sampling may cause the robot to take several steps to reach a state which the robot was capable of reaching in a single step, but was not included in the action sampling.

In this paper we present methods for adapting the action set to the terrain, allowing it to change for each step the planner considers. This method uses a fixed sampling of the action space as a set of *reference actions*, and performs a local search around those actions to fit them to the terrain. This allows the search to maintain a constant branching factor, as well as other desired properties of the reference action set, while allowing it to use the full action capabilities of the robot and its locomotion controller.

II. BACKGROUND

Reliable walking biped robots have been developed only recently, although today there are several humanoid robots in use around the world. For these robots, comparatively little research attention has been focused on developing complete global navigation strategies. Sensor-based obstacle-avoidance techniques have been developed for bipeds navigating in unknown environments [19], [28], which have allowed robots such as the Johnnie humanoid to adjust its path and step length in response to sensed obstacles [18]. However, such reactive methods can become trapped in local loops or dead-ends, because they do not consider global information.

Current planning approaches for legged robots lie along a spectrum based on how much of the robot's underlying details are considered during the planning process. 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 [15], 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 [9], [11]. 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 [2], [10], [13], [27].

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 were 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 [25]. 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 [14], [23]. 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 capabilities [7], [24]. 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 [17]. Another approach planned ways to adjust HRP-2's body posture to fit into the available free areas along a path [12].

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 [1]. Climbing robots have used reasoning about individual footholds combined with probabilistic motion planning to find motion plans for wall-climbing [3], [4], [8]. Recently, several approaches have used footsteps as an action model for moving through an environment for both bipeds [9], [22] and quadrupeds [16].

III. PROBLEM DESCRIPTION

At a high level, we wish to navigate a legged robot through a complicated, real-world environment, utilizing the robot's ability to step over obstacles, and move between surfaces of different heights. To accomplish this goal, we perform an A* search through the space of possible footstep actions the robot can perform. The cost of a path is a combination of action costs and terrain costs, encoding a combination of the safety of the steps, and the number and type of stepping motions.

In this paper, the problem we are interested in is the choice of stepping actions used for expanding nodes in the A* search. An action a from the set of all actions of the robot, $\mathcal{A}$, is a mapping from one robot state, $x \in \mathcal{X}$, to another, corresponding to a particular motion of the robot:

$$x' = a(x, e) \tag{1}$$

where e is the environment, and $x' \in \mathcal{X}$ is the new state of the robot after performing action a. In our previous work [5], [6], we have manually designed a set of stepping actions which were applied during every node expansion. In constrained situations, the action needed to make progress from a particular state may not be present in that set of actions. Additional actions can be added to the action set to ameliorate this problem. However, this will increase the branching factor, and the extra actions may still not provide the needed action during planning. If we have a desired branching factor, b, for the node expansion, the problem we wish to solve can be stated as follows: What are the best b actions to perform from state x in environment e ?

In order to answer this question, we have to define what we mean by the "best" b actions. If we knew the value function for the navigation problem, the solution would be simple: perform the action the takes you to the best value in the value function. In that case, there would be no need to search at all, because the value function encodes within it the solution for every state. However, all that we have during the planning process is a cost function, and a heuristic estimate of the true value function. Because we do not know the true value function, simply stepping to where the cost function and heuristic are a minimum will likely not result in a solution. So while choosing actions that are low in cost is important, we want to make sure that the actions we take allow us to reach a large number of other states, allowing us to explore the state space of the problem more efficiently. This reasoning provides the basis for one useful definition of the "best" b actions: The actions which will allow us to reach the largest amount of the state space from the b resulting states.

The reachable space from a state and environment $R(s, e)$ can be given as the set

$$R(x, e) = \{x' \in \mathcal{X} \mid \exists a \in \mathcal{A} \,.\, x' = a(x, e)\}. \quad (2)$$

Therefore, the reachable states from state x after performing the actions $a_1, a_2, ...a_b$ in our action set $\mathcal{S}$are

$$R(a_1(x, e), e) \bigcup R(a_2(x, e), e) \bigcup ... \bigcup R(a_b(x, e), e) \quad (3)$$

Thus we wish to find the action set $\mathcal{S}$which maximizes this region. Unfortunately, computing this region for all possible $\mathcal{S}$is computationally intractable. Additionally, to pre-compute the necessary information for lookup would require computing the best $\mathcal{S}$for all possible states in all possible terrains, which would be infeasible to store.

IV. Algorithm

Because it is infeasible to compute the best $\mathcal{S}$for a given state and environment, we will settle for a suboptimal $\mathcal{S}$. To find this $\mathcal{S}$, we will first start from a reference action set, $\mathcal{S}_r$, which has the good reachability property that we desire. On obstacle-free, flat ground, we can use this reference action set directly as our set of actions for node expansion. In the presence of rough terrain or obstacles, a state x' that an action in $\mathcal{S}_r$ maps to may be invalid. In this case we perform a local search around x' to find a new state, x'', and compute a new action to reach that state, a'. Pseudo-code for this approach is shown in Algorithm 1. Through this approach, we maintain our branching factor, replacing invalid actions with nearby valid ones. In addition, we retain the good reachability property that the reference action set had, although the reachable region may be slightly reduced by the modification of the result states of our action set.

Algorithm 1: ApplyActions($\mathcal{S}_r$, x , e)

```
foreach a ∈ S_r do
    x' ← a(x, e)
    if ValidState(x', e) then
        AddToQueue(x', a)
    else
        x'' ← LocalSearch(x', e)
        a' ← ComputeAction(x, x', e)
        AddToQueue(x'', a')
    end
end
```

In order to implement this approach, we need two extra pieces of information to model the robot and its walking controller. First, we need an explicit representation of the boundaries of $R(x, e)$. When using a fixed action set, we can be certain that our actions are drawn from the robot action space, $\mathcal{A}$. However, if we are allowing the resulting state to be adapted to the terrain, the local search needs to remain within the reachable region, so that an action exists which can move the robot to that state. The second new requirement is an inverse of the action set mapping, such that we can find the action which will move us from one state to another within its reachable range. Previously, we only required a forward mapping of the robot's actions. This inverse mapping is what allows us to search in the state space to adapt to the terrain.

A. Blind Local Search

One simple local search method is to begin testing all states (at some discretization) near the resulting state of an invalid reference action, terminating the search when the nearest valid state is found, or when no valid location has been found within the specified search radius. This is implemented by pre-computing an ordered list of nearest relative states, and then iterating through those states for a given robot state until a valid step is found. This method will find the closest state at the chosen resolution, but because it is merely iterating through nearby states, it may need to perform many state evaluations to find a valid location.

B. Informed Local Search

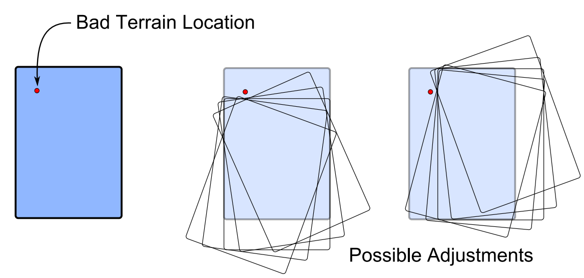

Fig. 2. Possible adjustments to avoid a bad terrain location, shifting so that location has either moved off to the side, or the front or back of the foot.

The current bottleneck in footstep planning is evaluating the terrain to determine the suitability of each step. Therefore, while the blind local search allows for the planning process to use more of the robot's capabilities and find its way through tighter areas, it may be evaluating many locations to find a suitable step to add to the search queue. Because we have access to the cost functions themselves, we can compute extra information about them to guide the local search to valid configurations. We propose an approach in which the cost function identifies the worst location underneath the foot for a given step. The search then tries alternate step positions which can avoid that bad location. This approach allows the search to very quickly adjust around bad terrain features to a suitable step position, with few terrain evaluations needed.

For bipeds, one aspect of the terrain cost involves fitting a plane to the terrain under the foot's location. A useful measure of the worst terrain location is the position of the maximum deviation of the terrain from the fitted plane. From this position, we can calculate candidate alternate foot locations that avoid this particular undesirable feature, as shown in Figure 2.

C. Notes on Optimality

When using a fixed action set and an admissible heuristic, we can claim that the path found by A* search is optimal by the metrics used, *with respect to the given action set*. No better path exists, which can be executed using just those actions. However, by adapting the reference actions to the terrain, we are allowing the robot to use more of its full capability. We cannot claim that the path returned by this search is optimal with respect to the robot's *full capability*. After adapting the actions to the terrain, we are still left with a sampling of the full capability, which will likely not contain the exact step needed for the optimal path. However, while this algorithm does not guarantee the true optimal path, its cost will be less than or equal to the cost of using the same reference actions as a fixed action set. This property is due to the fact the the actions applied when using the adaptive action set are a superset of the reference actions. So the optimal path found when using fixed actions is also a path in the adaptive-action-set search tree. Therefore, the A* search will find a path of equal or lesser cost, depending on whether or not the adapted actions allow for a lower cost solution.

D. Local Minima Search

In addition to reachability, we also would like our action set to provide low-cost actions, rather than expensive ones. Instead of modifying a reference set of actions, an alternate approach is to compute the cost function and heuristic value function over the reachable region of the terrain, and select the local minima of the cost + value as the action set to use. This has the advantage of selecting some of the lowest cost available steps in rough terrain environments, but does not provide any guarantees about the reachability of that action set, and can be extremely computationally expensive to compute.

V. Case Studies

To date, these adaptive action models have been applied to two robots: the humanoid HRP-2, and the quadruped LittleDog.

A. HRP-2

For the humanoid HRP-2 we have used indoor environments, involving obstacles such as chairs and tables, and terrain features such as stairs and platforms. The robot navigating through one of these environments is shown if Figure 1. The sensing of the environment for these trials was provided by real-time motion capture data [26], as well as by on-board vision [21]. Visual processing and navigation planning was performed off-board, with the resulting path transmitted back to the robot for walking motion generation.

Figure 3 shows a comparison of the paths generated for walking up a set of stairs from the different local search methods. The same reference action set, containing 8 actions, was used for all examples. Notice that the two paths from the adaptive local search methods are similar, and contain far fewer steps than the path from the fixed action set. In this particular case the reference action set contained a forward step that was close to the length of the stairs, allowing the fixed action set to find a solution. However, due to the limited number of actions, many steps were necessary to put the robot into a position where its actions would fall correctly on the stairs.

Specific data on the planning processes for this example environment are provided in Table I. An interesting point in this data is that using the fixed action set took significantly longer to plan and had to expand many more nodes. This extra search was necessary to find the particular sequence of approach actions that allow the fixed action sequence to climb the stairs. Another interesting detail is the fact that while the blind local search found a shorter path that required expanding fewer nodes to find, it took longer to plan that path than when using the informed local search. First, the reason that the blind search finds a better path is due to the fact that it iterates over a large number of possible locations to adapt to the terrain. Thus, it can find step locations that the informed local search will miss when the heuristic informing the local search is does not guide it in the correct direction. However, while it may miss some locations, the informed local search generally finds a valid location much faster than the blind search with fewer terrain cost evaluations needed. Table I shows that the blind local search performs far more terrain evaluations than the informed local search, even though it expands fewer nodes during the search.

To further illustrate this point, Figure 4 shows a histogram of the search depth to find a valid step location when the reference location is invalid. For the blind local search, the search depth is proportional to the distance from the reference location to the nearest valid footstep, resulting in a fairly random distribution of search depths. In contrast, the informed local search terminates very quickly, usually with fewer than ten terrain evaluations required.

These plans were generated using a Euclidean distance heuristic to estimate the remaining cost for the A* search. This admissible heuristic provides a fair comparison, but for actual robot operation, a more informed (but inadmissible) heuristic is used, described in previous work [6]. In the same example environment, using the informed local search results in a 10 step path in under 200 milliseconds.

B. LittleDog

LittleDog, shown in Figure 5, is a small quadruped robot, tasked to crawl over rough terrain environments as quickly and reliably as possible. To use the footstep planning framework with this quadruped robot, several changes must be made from the biped planning version. First, the reachability model of the robot is very different, and must take into account all three stance legs. Second, the terrain evaluation function changes. What constitutes good stepping terrain for LittleDog is very different from what constitutes good stepping terrain for HRP-2. Finally, due to the roughness of the terrain involved, and the limited degrees of freedom of the robot's legs, the planner must perform extra checking of stepping trajectories, to ensure

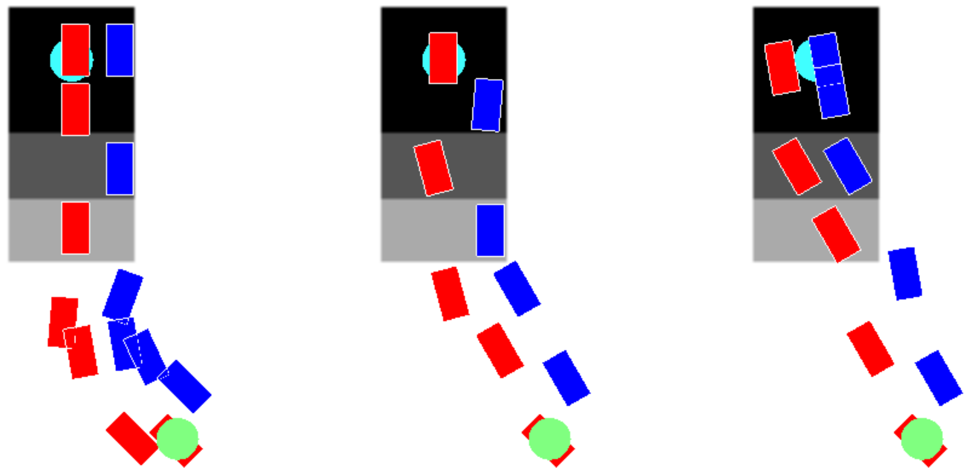

Fig. 3. Comparison of plans generated for walking up stairs. *Left:* Fixed action set. *Center:* Blind local search. *Right:* Informed local search.

Method	Planning Time	Path Length	Path Cost	Nodes Expanded	Cost Evaluations	Evaluations per Action
Fixed	34.6 s	12 steps	13.0022	204071	1632561	1
Blind Search	6.7 s	8 steps	9.00163	2485	934879	47.0
Informed Search	2.5 s	9 steps	9.80248	6145	57143	1.16

TABLE I
COMPARISON OF PLANNING COSTS

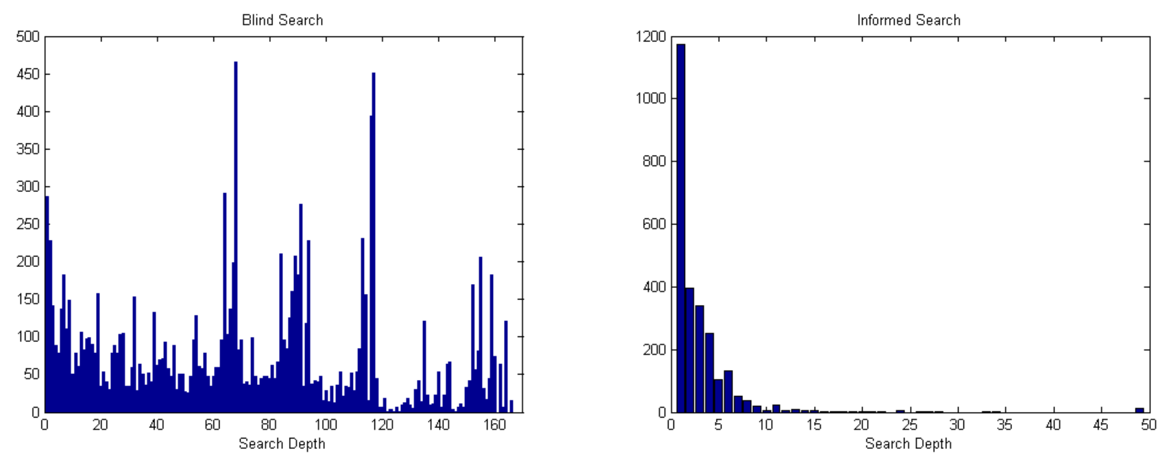

Fig. 4. Comparison of search depth for blind vs. informed local search in the stairs environment

that there is enough clearance for the knees/elbows of the robot during the traversal.

However, once these modifications are made, we can plan using the same framework as for bipeds. In the rough terrain environments that LittleDog must traverse, the fixed action set model rarely can find a path, so the adaptive models become necessary. In addition, for the particular trials LittleDog must perform, the environment is known in advance and is known to be unchanging during the trial. These facts allow us the opportunity to pre-compute terrain cost evaluations. With this pre-computation, terrain evaluations are no longer a bottleneck, and the blind local search becomes very fast. In addition, fast terrain evaluations allow us to compute the cost + heuristic value over the entire reachable region of the robot, and choose local minima of that function as our actions.

On the test environments, using solely local minima actions in the search does not provide good exploration of the space. Thus for planning paths, the local minima actions were com-

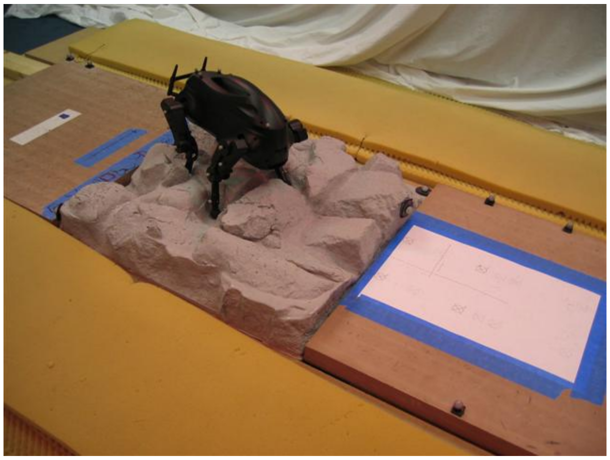

Fig. 5. LittleDog robot crawling over rough terrain.

bined with a small set of reference actions, adapted through blind local search. This allowed the search to maintain some reachability, while still choosing low-cost actions over difficult terrain. Table II shows that the path cost when using the combination of local minima actions and reference actions adapted to the terrain provided significantly lower cost paths than using the adapted reference actions alone. This lower cost resulted in a higher percentage of successful runs over the terrain, and faster execution of those paths, due to fewer slips and re-plans when walking along the planned paths.

Method	Avg. Path Cost	Success	Avg. Run Time
Reference	2716	50%	172 s
Reference & Local Min	1058	80%	94 s

TABLE II

LITTLEDOG PLANNING IN A DIFFICULT TERRAIN SETUP

VI. CONCLUSION AND FUTURE DIRECTIONS

This paper presents methods for adapting robot actions to the local terrain during legged navigation planning. This adaptation provides lower cost paths that can use more of the robot's full action space, and allow the navigation planner to find paths through environments that would be unsolvable with a fixed action set. Furthermore, this adaptation of actions often results in significantly faster planning times, by simplifying sequences of actions needed to fit through narrow passages in the search space.

These adaptive action models have been applied to a humanoid and quadruped robot with very promising results, allowing these robots to successfully and reliably navigate environments in which previously no path could be found with a fixed action set.

Reachability of the overall action set, as well as choosing low-cost actions are both important for finding a high-quality solution. The methods presented in this paper focus on just reachability (adapting reference actions) or low-cost (using local minima). Future work will investigate providing a more continuous trade-off between the two objectives, rather than just optimizing for one or the other.

ACKNOWLEDGMENTS

This research was supported in part by the DARPA Learning Locomotion Program.

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade. Online environment reconstruction for biped navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2006.

[21] 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/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[22] 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.

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

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.