The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Learning Locomotion over Rough Terrain using Terrain Templates

Mrinal Kalakrishnan*, Jonas Buchli*, Peter Pastor*, and Stefan Schaal*†‡

*Computer Science, University of Southern California, Los Angeles, CA 90089 USA
†Neuroscience and Biomedical Engineering, University of Southern California, Los Angeles, CA 90089 USA
‡ATR Computational Neuroscience Labs, Kyoto 619-0288, Japan
Email: {kalakris, buchli, pastorsa, sschaal}@usc.edu

***Abstract*— We address the problem of foothold selection in robotic legged locomotion over very rough terrain. The difficulty of the problem we address here is comparable to that of human rock-climbing, where foot/hand-hold selection is one of the most critical aspects. Previous work in this domain typically involves defining a reward function over footholds as a weighted linear combination of terrain features. However, a significant amount of effort needs to be spent in designing these features in order to model more complex decision functions, and hand-tuning their weights is not a trivial task. We propose the use of *terrain templates*, which are discretized height maps of the terrain under a foothold on different length scales, as an alternative to manually designed features. We describe an algorithm that can simultaneously learn a small set of templates and a foothold ranking function using these templates, from expert-demonstrated footholds. Using the LittleDog quadruped robot, we experimentally show that the use of terrain templates can produce complex ranking functions with higher performance than standard terrain features, and improved generalization to unseen terrain.**

I. Introduction

Traversing rough terrain with carefully controlled foot placement and the ability to clear major obstacles is what makes legged locomotion such an appealing, and, at least in biology, a highly successful concept. Surprisingly, when reviewing the legged locomotion literature, relatively few projects can be found that actually address walking over rough terrain. Most legged robots walk only over flat or at best slightly uneven terrain, a domain where wheeled systems are usually superior. Walking over rough terrain poses a variety of challenges. First, the walking pattern needs to be very flexible in order to allow close to arbitrary foothold selection – indeed, even the choice of which leg is the swing leg may have to be altered on the fly [1]. Second, balance control becomes crucial due to slipping and other mistakes, such that sole reliance on a stable walk pattern is insufficient [2]. And third, foothold selection for maximal robustness and speed is crucial. In previous work [1], [2], we have addressed the first two issues. In this paper, we consider the problem of foothold selection for locomotion over rough terrain.

Related work in the literature has used classifiers that classify footholds on the terrain as acceptable or unacceptable using terrain features like slope and proximity to cliffs [3]. Other work involves defining a reward function over footholds as a weighted linear combination of terrain features like slope and curvature on different length scales, and subsequently picking the foothold that maximizes the reward [4]. The weights on the features in the reward function in [4] are inferred using a learning procedure called *hierarchical apprenticeship learning* on footholds and body paths demonstrated by an expert. The performance of such a system, however, is critically dependent on the careful design of heuristic terrain features which are flexible enough to model the expert's training data.

Fig. 1. The LittleDog quadruped robot on rocky terrain

The contribution of this paper is the introduction of *terrain templates* (hereafter simply referred to as *templates*) as a tool for learning locomotion over rough terrain. The concept is partly inspired by template matching techniques widely used in computer vision [5]. A template is a discretized height map in a small area around a foothold that the robot encounters. We introduce an algorithm that can learn a set of templates from expert-demonstrated footholds, along with an associated set of weights, and use them to successfully navigate previously unseen terrain. We present results showing that the learnt templates alone can outperform multi-scale terrain features on complex terrain. We also show that the combination of features and templates performs the best, due to the broad generalization ability of features, and the specialization capability of templates.

The rest of this paper is laid out as follows. In Section II, we formulate the foothold selection problem and introduce an algorithm that learns a ranking function for foothold selection

from expert demonstrations, using an arbitrary set of features that represent a foothold. In Section III we introduce the concept of *terrain templates* as a flexible foothold feature representation, and propose an algorithm that learns a set of templates and a corresponding foothold ranking function from expert demonstrations. In Section IV, we introduce our experimental setup and describe our control architecture. In Section V we present results showing that the use of terrain templates for foothold selection allows navigation of more complex terrain than previously possible. Finally, we conclude the paper and discuss ideas for future work in Section VI.

II. Learning a Ranking Function for Foothold Selection

We develop our work in the context of quadruped locomotion, but this approach equally scales to any other legged locomotion system. We consider static or quasi-static locomotion, in which either the Center of Gravity (COG) or the Zero Moment Point (ZMP) [6], respectively, is always maintained within the support polygon. The task is to find a suitable foothold for the swing leg, given the positions of the stance legs. The chosen foothold must:

- Minimize slipping on terrain
- Not result in collisions with the environment
- Maximize forward progress towards the goal
- Be within the kinematic range of the robot
- Serve to maximize the area of future support polygons for maximum stability

Most often, these goals are conflicting, and trade-offs must be made between them. Specifying these trade-offs by hand is non-trivial, since it typically involves manual tuning of a large number of parameters. From our experience, it is nearly impossible to achieve a system that generalizes well to all kinds of terrain purely by manual parameter tuning, as the search space is prohibitively large.

We seek to learn the optimal trade-offs for foothold selection, using footholds that are demonstrated by an expert. To this end, we formally define the foothold selection problem below, describe the process of collecting expert demonstrations, and then introduce an algorithm that learns how to select good footholds.

A. Problem Formulation

The state of the robot is defined as the 3D position of all feet, 3D body position and orientation, and the knowledge of which leg is the swing leg. We define the set of n footholds reachable from this state by $\mathcal{F} = \{f_1 \dots f_n\}$. Each of these footholds f_i is described by a feature vector $\mathbf{x}_i \in \mathbb{R}^d$. We define this feature vector $\mathbf{x}_i$ as being composed of two groups: terrain features, which encode information about the terrain at the foothold, and pose features that quantify all other relevant information like progress towards the goal, stability margins, and collision margins.

We define a reward function R over footholds as a weighted linear combination of the features:

$$R(f_i) = \mathbf{w}^T \mathbf{x}_i, \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector. This reward function can then be used to rank all the footholds in $\mathcal{F}$, and the one with the highest reward selected as the target foothold[1]. The task of learning the trade-offs for foothold selection has thus been reduced to learning the weight vector $\mathbf{w}$.

B. Collecting Expert Demonstrations

An expert is shown logs of runs executed by the robot on different kinds of terrain. S/he inspects each run for sub-optimal foot placement choices made by the robot, and in each of those situations, labels the correct foothold f_c that s/he thinks is the best greedy choice from among the entire set of reachable footholds $\mathcal{F}$. Fig. 3(a) shows a screenshot of our teaching interface, where the red ball depicts the sub-optimal foot placement choice made by the robot, and the white ball below shows the optimal foot placement chosen by the expert.

C. Learning From Expert Demonstrations

When the expert labels the foothold f_c from the set $\mathcal{F}$ as the optimal foothold, s/he is implicitly providing information that the reward for the chosen foothold f_c is better than the reward for all others:

$$\mathbf{w}^T \mathbf{x}_c > \mathbf{w}^T \mathbf{x}_i \quad \forall i \in \mathcal{F};\ i \neq c. \tag{2}$$

This corresponds to the widely studied rank learning or preference learning problem in the machine learning literature [7]. This problem can be converted into a binary classification problem [8]. Rearranging Eq. (2), we get:

$$\mathbf{w}^T (\mathbf{x}_c - \mathbf{x}_i) > 0 \quad \forall i \in \mathcal{F};\ i \neq c. \tag{3}$$

The optimum value of $\mathbf{w}$ could be found by defining a suitable loss function that penalizes values of $\mathbf{w}^T(\mathbf{x}_c - \mathbf{x}_i)$ that are lower than zero. However, it is observed that such a loss function would be identical to that of a linear binary classifier, in which the inputs are the pairwise difference feature vectors $(\mathbf{x}_c - \mathbf{x}_i)$, all mapped to the target class $+1$. Hence, the weights $\mathbf{w}$ can be obtained by running a linear classifier[2] like Support Vector Machines (SVM) or logistic regression (LR) on the pairwise difference feature vectors $(\mathbf{x}_c - \mathbf{x}_i)$. The resulting weight vector $\mathbf{w}$ is used to evaluate the reward R in Eq. (1) on all candidate footholds $f_i \in \mathcal{F}$. The ranking imposed upon $\mathcal{F}$ by the reward R will satisfy all the training data in Eq. (2), if and only if the expert's ranking function is truly a linear function of the features. We show in the following section that this is usually not the case, especially with conventional feature representations of footholds.

[1]Maximizing the immediate reward for the current foothold is an inherently greedy approach. At the expense of higher computation time, one can achieve better performance in practice by choosing the foothold that maximizes a multi-step criterion, i.e., the value function, which is the sum of expected future rewards obtained by choosing a foothold, using standard heuristic graph search algorithms like A*.

[2]Our choice of a specific classifier is motivated in Section III.

III. Terrain Templates

Some of the terrain features previously used for the foothold selection problem are slope, curvature and standard deviation of the terrain on various spatial scales [4], [9]. However, since the rewards are defined to be linear in the features, such a feature set can only represent more complex decisions by careful engineering of nonlinear features. For example, Fig. 2 shows three footholds with increasing levels of curvature. The foothold illustrated in Fig. 2(b) is preferable to flat terrain (Fig. 2(a)), because it prevents foot slippage. The one in Fig. 2(c), however, is not, because the foot could get stuck in it. Such a ranking is hard to achieve with a reward function that is a linear function of curvature.

Although this specific example could possibly be handled by hand-crafting additional features, one can imagine that many more such situations exist which cannot be foreseen. The problem thus requires a much more general solution[3].

We propose the concept of a *terrain template*, which is a discretized height map of the terrain in a small area around the foothold. One could imagine an approach in which the robot maintains a library of such templates, each associated with a reward value. Subsequently, during execution, each candidate foothold can be assigned the reward from the closest matching template in the library (using an arbitrary similarity measure). In the context of Fig. 2, the robot would store a template for each of the three footholds shown, along with an appropriate reward for each. Assigning the highest reward to the template for Fig. 2(b) would allow the robot to select that foothold over the others.

Manual creation of such a library of templates would be too time-consuming, and it would be nearly impossible to attain good generalization performance, since the rewards for each template would have to be tuned carefully. Hence, we propose an algorithm that uses expert-demonstrated footholds to simultaneously learn a small set of templates, and a foothold ranking function that uses these templates. We first describe our methodology for template extraction from expert demonstrations, followed by the template learning algorithm.

[3]Non-linear classifiers, such as the use of radial basis function kernels in SVMs [10] can learn non-linear ranking functions [8], but are still limited by the original feature representation. Additionally, the decision function cannot be interpreted as easily as in the linear case.

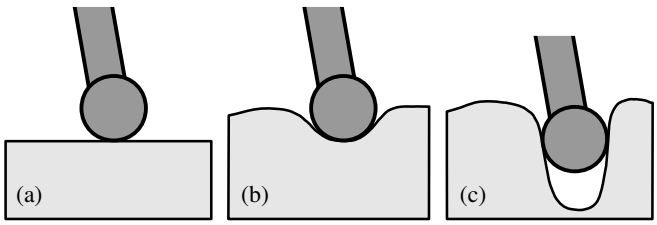

Fig. 2. Three footholds with increasing values of curvature: (a) Flat terrain; (b) Terrain with mild curvature, preferable to flat terrain since it prevents slipping; (c) Terrain with high curvature, which can cause the foot to get stuck, thus less preferred than (a) or (b).

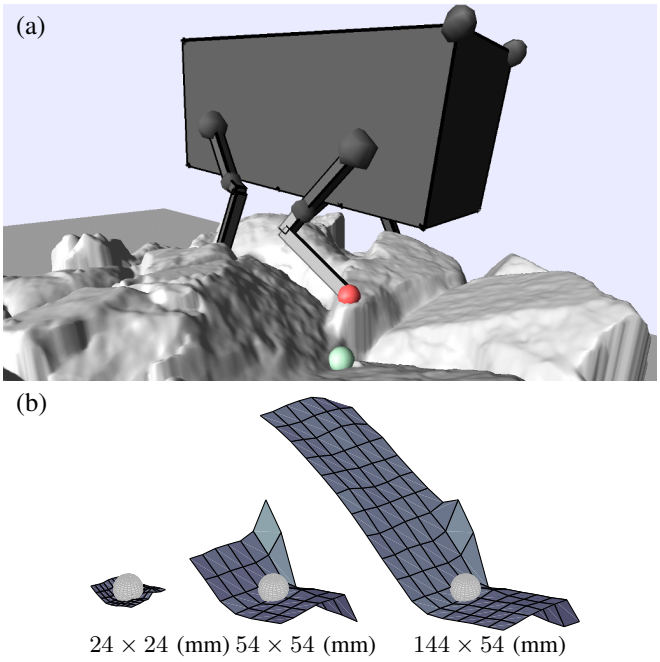

Fig. 3. (a) Teaching interface used to demonstrate footholds. Red ball at the foot indicates the chosen (dangerous) foothold, light green ball below indicates the demonstrated optimal (safe) foothold; (b) Terrain templates in multiple scales extracted from the demonstrated foothold. The white spheres indicate the position of the foot on each template.

A. Template Extraction

From each foothold demonstration $\langle \mathcal{F}, f_c \rangle$ made by the expert, we extract a set of templates on multiple scales. Fig. 3(b) shows the three scales of templates that were extracted from the demonstrated foothold (light green ball in Fig. 3(a)). The scales that we use are dictated by the geometry of our quadruped robot, and are designed to independently capture different properties of the terrain that make up its reward function. The smallest scale (24×24 mm) is on the order of the size of the foot, and encodes information about surface friction properties and micro-features that define the slip characteristics of the foothold. We intend to capture information about clearance from obstacles and drop-offs using the medium scale (54×54 mm). Finally, the large scale (144×54 mm) accounts for potential shin and knee collisions.

Templates on all scales are also extracted from all the other reachable footholds in the set $\mathcal{F}$, for every expert demonstration. This creates a large library of templates representing different kinds of terrain. This comprises the input to the following template learning algorithm, which selects a small subset of these templates and learns a foothold ranking function using them.

B. Template Learning

Each template in the library contributes a feature to the feature vector $\mathbf{x}_i$ of a foothold f_i. The feature value represents the similarity between the template and the candidate foothold. We choose a radial basis function similarity measure – a negative squared exponential kernel centered at the template, and evaluated at the candidate foothold. The

feature value is given by:

$$x = \exp\left(-h\sum_{i=1}^{n}(t_i - c_i)^2\right), \quad (4)$$

where x is the value of the feature, h is the bandwidth parameter of the kernel[4], n is the number of terrain points in the discretized height map of the template, t_i is the i-th height value of the template, and c_i is the i-th height value of the candidate foothold.

This feature set forms the input to the algorithm described in Section II, that learns a ranking function for footholds from expert demonstrations. This is done by converting the expert ranking data into a problem of linear binary classification, and using the resulting weights as a ranking function. However, since we have a potentially huge template library, selecting a small subset of these templates for use in the final ranking function is important for two reasons. Firstly, the use of the entire template set would allow the learning algorithm to trivially overfit all the training data, resulting in poor generalization. And secondly, the similarity measure for templates needs to be evaluated in real-time while the robot is walking, and these computations can be prohibitively expensive for a large template library.

We combine the steps of template subset selection and weight learning by using a linear classifier that promotes sparsity in feature space. We run an efficient implementation of l_1-regularized logistic regression (LR) [11] on the pairwise difference feature vector data ($\mathbf{x}_c - \mathbf{x}_i$). l_1- regularized LR is a linear classifier that can be defined as an optimization problem to minimize the following cost function:

$$J = \sum_{i=1}^{m} -\log\left(\frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x}_i y_i)}\right) + \lambda\|\mathbf{w}\|_1, \quad (5)$$

where m is the number of training examples, $\mathbf{x} \in \mathbb{R}^d$ is the i-th input vector, $y_i \in \{-1, 1\}$ is the label of the i-th input vector, $\mathbf{w} \in \mathbb{R}^d$ is the weight vector, and λ is the regularization parameter. The predicted label y_{test} for an input $\mathbf{x}_{test}$ is obtained using:

$$y_{test} = \text{sgn}(\mathbf{w}^T\mathbf{x}_{test}). \quad (6)$$

The second term in Eq. (5) is a regularization term that penalizes the l_1 norm of the weight vector. The use of an l_1 regularization term has been known to produce sparse solutions, i.e. a weight vector $\mathbf{w}$ which contains very few non-zero values [12]. It has been shown that l_1-regularized LR can outperform l_2- regularized LR, especially in cases where the number of features is much larger than the number of data points [12]. In our case, each expert demonstration produces a large set of training examples for the classifier. However, data from footholds that are very close to each other is likely to be very similar, so the true number of unique training examples that convey novel information is usually much lower.

The regularization parameter λ can be viewed as a control for the desired amount of sparsity – higher values of λ typically produce sparser weight vectors. We fix the value of λ by searching through a range of values and picking the one that minimizes the cross validation training error.

The sparse weight vector $\mathbf{w}$ that we obtain from the l_1-regularized LR allows us to discard all the templates from the library which have a zero weight. The reward for each foothold can now be evaluated by calculating the feature values for templates that remain in the library using Eq. (4), and evaluating the reward function in Eq. (1) using the learnt weight vector $\mathbf{w}$.

IV. Experimental Setup and Control Architecture

In this section we describe our experimental setup and provide an overview of our robot control architecture.

A. Experimental Setup

Our experimental setup consists of the LittleDog quadruped robot (Fig. 1) with ball-like feet that are close to point feet, manufactured by Boston Dynamics. It is about 0.3m long, 0.18m wide, and 0.26m tall, weighs approximately 2.5kg, and has 3 degrees of freedom per leg. Joint angle and velocity set-points are sent to the robot from a Linux host computer over a wireless connection at 100Hz, and the on-board low level controller servos each actuator using PD control at 400Hz. LittleDog has a 3-axis force sensor on each foot, position sensors to measure joint angles, and an on-board inertial measurement unit (IMU). An external motion capture system (VICON) provides information about the absolute world position and orientation of the robot.

We use a set of interchangeable terrain modules of size 61×61 cm. The terrain modules include flat modules, steps of various step heights, different sizes of barriers, slopes, logs, and rocky terrain of varying difficulty levels. Each terrain module is scanned by a laser scanning system beforehand, to produce a high (1mm) resolution 3D model. A unique arrangement of reflective markers is fixed on each terrain module, enabling the motion capture system to track the position and orientation of each module independently, thus providing complete information about the terrain in the world. This greatly simplifies the perceptual component of rough terrain locomotion, allowing us to focus on the planning and control problems.

B. Control Architecture

We use a hierarchical control scheme, similar to [9], briefly outlined below:

1) Approximate Body Path Planner: We first plan an approximate path for the robot body through the terrain, from the start to the goal. This is achieved by discretizing the entire terrain into 1 cm square grid cells, and assigning a reward to each cell. The cell rewards are obtained by assuming that the center of the robot is in the cell and the four feet are

[4]Currently, the bandwidth parameter h of the kernel is tuned manually per length scale. We keep the parameter fixed for all experimental results shown in this paper.

in their default stance positions. For each leg, the best five foothold rewards in a small search radius around the default position are summed up, along with some heuristic scores based on body and knee clearance, to form the final reward for the cell.

The final path is obtained by generating a policy in this grid-world using Dijkstra's algorithm. This approximate path is used to direct the lower level footstep planner. If the foothold reward function is changed, it usually results in a different body path. Note that for the purposes of evaluation in this paper, we kept the body path fixed so as to remove the influence of the body path planner on foothold selection.

2) Footstep Planner: A reward function over footholds on the terrain is defined as described in Sections II and III. The reward function chooses footholds that are biased towards the global body path plan. Using this reward function for footholds, we run a five-step look-ahead planner and choose the foothold that maximizes the reward over the next five steps.

3) Pose Finder: For every step generated by the footstep planner, we optimize the pose of the robot (namely, the COG height, roll, pitch and yaw angles) to maximize reachability of all footholds and minimize knee and body collisions.

4) Body Trajectory Generation: Using the locations of the next five footholds generated by the footstep planner, we generate a COG trajectory through the resulting support triangles which allows for continuous forward movement of the COG with optimal, velocity-dependent sway. This trajectory planner is described in detail in another contribution [2].

5) Foot Trajectory Generation: The trajectory for moving a foot from one location on the terrain to another is defined by finding the convex hull of the terrain points in between, and raising the height of the hull by a minimum clearance. The points on the hull are then connected using piecewise cubic splines optimized for minimum acceleration subject to monotonicity constraints.

6) Low level Control: The resulting trajectories are then converted into desired joint angles and velocities using a closed form inverse kinematics solution for each 3-DOF leg individually, which are then tracked by the robot using a PD controller at 400Hz. Additionally, the body and foot trajectories are controlled in closed loop at 100Hz (using the VICON motion capture information) to improve foot placement tracking performance.

V. Results

In order to evaluate the performance of our template learning approach, we collected around 100 expert foothold demonstrations across different kinds of terrain of varying difficulty levels. Three different foothold reward functions were learnt using the following feature sets: (a) multi-scale terrain features, (b) multi-scale terrain templates and (c) terrain templates + terrain features. 26 pose features were also used along with each of the above feature sets. The pose features included measures of progress towards the goal, stability margins, reachability of the foothold, differences in leg heights, and knee and body clearance from the terrain.

TABLE I

Number of templates learnt on each spatial scale, from a total of 946, with and without augmentation by features

	Small scale	Medium scale	Large scale
Templates + Features	1	19	23
Templates Only	10	24	28

The terrain features we used (on each scale) were slope and curvature along the x and y axes[5], and standard deviation of the terrain. This resulted in a total of 9 features being used per length scale. The same three length scales were used for both feature computation and template extraction, and are depicted in Fig. 3(b). The features and templates are mirrored in the y direction for the left and right legs, but independent functions were learnt for front and hind legs as their properties tend to be quite different.

A foothold ranking function was learnt for each of the three feature sets mentioned above, using the algorithms described in Sections II and III. The l_1 regularization parameter λ was set individually in each case by choosing the value that minimized cross validation error on the training data. Table I shows the number of templates selected on each scale, with and without the use of terrain features. Interestingly, almost no templates were needed on the smallest scale when features were used, however, most of the templates on the medium and large scales were still required. This suggests that the expert's ranking function contained a significant amount of non-linearities in the larger scales that terrain features were unable to capture. At the smallest scale, the information contained by the templates seems to be identical to that of the features computed at this scale.

We measured locomotion performance using two metrics: success rate, and average slip experienced by the robot at each foothold. A run was deemed a success if the robot crossed the terrain and reached the goal without falling over. Slip was measured as the distance between the position of a foot at touchdown and the position of the same foot before it swings again, i.e., one full walking cycle later. We averaged the slip only over successful runs, since including the slip experienced on runs that failed would render the statistic meaningless. Table II shows the results that were obtained by performing 21 runs using the three different ranking functions learnt from the three feature sets. These runs were performed on a test terrain module which the robot was not trained on. The use of templates is seen to improve performance over that of terrain features, in terms of both success rate and slip. It is also observed that combining templates and features results in the most robust performance and lowest average slip. The baseline slip performance of the robot walking on flat terrain is also shown, as a lower bound to the achievable amount of slip on our experimental setup[6].

[5]We split each slope and curvature feature into two distinct features for the positive and negative directions. The x and y axes are in body coordinates, with x pointing forward, and y to the left.

[6]The amount of slip experienced on flat terrain might seem rather high. This is largely because the spherical feet on the LittleDog robot roll forward as the robot moves forward, resulting in a systematic change in the perceived foot location.

Fig. 4. Sequence of snapshots of the quadruped robot LittleDog crossing the test terrain after template learning. (left to right, top to bottom)

TABLE II

SUCCESS RATES AND AVERAGE SLIP AT FOOTHOLDS USING DIFFERENT FEATURE SETS. EXPERIMENTAL DETAILS CAN BE FOUND IN SECTION V.

Method	Success rate (out of 21 runs)	Average slip on successful runs (mm)
Features Only	47.6%	24.8 ± 5.9
Templates Only	76.2%	20.2 ± 5.1
Templates + Features	100.0%	17.3 ± 3.3
Baseline (on flat terrain)	100.0%	13.4 ± 0.4

The results achieved using a combination of templates and features suggest that achieving broad generalizations using heuristic features, while simultaneously representing exceptions to those rules by using templates results in a highly performant system. We stress the fact that since the templates are learnt automatically, strong regularization during the learning process is a key component in achieving generalization and preventing overfitting to the training data.

Fig. 4 shows a series of snapshots of the robot crossing the test terrain. The video attached to this paper presents an overview of our template learning procedure, and provides a more life-like impression of the robot's motion over the test terrain.

VI. CONCLUSION AND FUTURE WORK

We presented an algorithm that can learn a foothold ranking function for locomotion over rough terrain. This was achieved by collecting expert foothold demonstrations and learning a ranking function by converting the ranking problem into one of linear classification. We introduced *terrain templates* as a flexible feature representation for footholds that can be used to learn more complex ranking functions. We proposed an algorithm that can simultaneously select a small subset of templates and learn a ranking function using these templates. This ranking function was shown to outperform the use of heuristic terrain features, in addition, the combination of features and templates was shown to achieve the most robust generalization performance.

This work on terrain templates was motivated by a grander vision of designing a rough terrain locomotion system that can learn without supervision or expert demonstrations, improving its own performance from trial to trial. Our initial experiments suggest that the use of templates provides a system that can quickly learn from experience. Learning a foothold ranking function through reinforcement can possibly result in performance superior to that achievable by learning from expert demonstrations, since the expert is not necessarily always optimal in her/his choices. We intend to address this problem in future work.

ACKNOWLEDGEMENTS

This research was supported in part by the DARPA program on Learning Locomotion, National Science Foundation grants ECS-0325383, IIS-0312802, IIS-0082995, ECS-0326095, ANI-0224419, NASA grant AC98-516, an AFOSR grant on Intelligent Control, the ERATO Kawato Dynamic Brain Project funded by the Japanese Science and Technology Agency, and the ATR Computational Neuroscience Laboratories. J.B. was supported by a prospective researcher fellowship from the Swiss National Science Foundation.

REFERENCES

[1] D. Pongas, M. Mistry, and S. Schaal, "A robust quadruped walking gait for traversing rough terrain," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1474–1479.

[2] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and S. Schaal, "Compliant quadruped locomotion over rough terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[3] J. R. Rebula, P. D. Neuhaus, B. V. Bonnlander, M. J. Johnson, and J. E. Pratt, "A controller for the LittleDog quadruped walking on rough terrain," in *IEEE International Conference on Robotics and Automation*, 2007.

[4] J. Z. Kolter, P. Abbeel, and A. Y. Ng, "Hierarchical apprenticeship learning, with application to quadruped locomotion," in *Neural Information Processing Systems*, vol. 20, 2007.

[5] J. Buhmann, J. Lange, and C. von der Malsburg, "Distortion invariant object recognition by matching hierarchically labeled graphs," in *International Joint Conference on Neural Networks*, 1989, pp. 155–159 vol.1.

[6] M. Vukobratovic and B. Borovac, "Zero-moment point - thirty five years of its life," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 157–173, 2004.

[7] W. W. Cohen, R. E. Schapire, and Y. Singer, "Learning to order things," *Journal of Artificial Intelligence Research*, vol. 10, pp. 243—270, 1999.

[8] T. Joachims, "Optimizing search engines using clickthrough data," in *Eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002.

[9] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng, "A control architecture for quadruped locomotion over rough terrain," in *IEEE International Conference on Robotics and Automation*, 2008, pp. 811–818.

[10] B. Schölkopf and A. Smola, *A Short Introduction to Learning with Kernels*. Springer, 2003, pp. 41–64.

[11] K. Koh, S. jean Kim, S. Boyd, and Y. Lin, "An interior-point method for large-scale l1-regularized logistic regression," *Journal of Machine Learning Research*, 2007.

[12] A. Y. Ng, "Feature selection, l1 vs. l2 regularization, and rotational invariance," in *International Conference on Machine learning*. ACM, 2004, p. 78.