# Single-Image Footstep Prediction for Versatile Legged Locomotion

Wuming Zhang[1] and Kris Hauser[2]

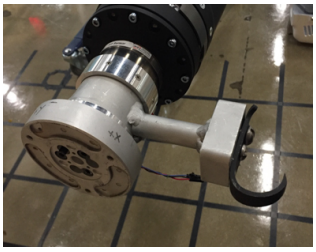
***Abstract*— Walking and climbing robots need to plan long-term routes on both horizontal and vertical terrain, but on-board sensors take images from vantage points that provide strongly foreshortened images that cause the appearance of terrain features to vary greatly by distance and viewing angle. This paper presents a convolutional neural network (CNN) method for predicting valid handhold and foothold locations from single RGB+D images taken at arbitrary tilt angles. Experiments show that the method predicts holds more accurately than comparable learning techniques, and that a route planner based on these predictions generates plausible plans for flat ground, stairs, and walls in rock climbing gyms.**

## I. INTRODUCTION

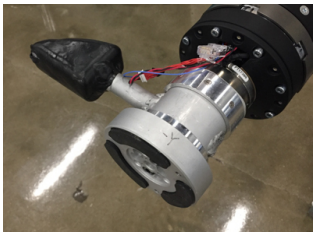
Footstep planning and route planning are essential components for legged robots to navigate natural terrain. As part of the Versatile Locomotion project, we are investigating strategies for the JPL Robosimian robot (Fig. 1a) to autonomously plan walking and climbing motions on outdoor, rugged terrain. Ultimately we intend for the robot to navigate a wide variety of environments ranging from horizontal terrain to vertical rock walls, as well as man-made terrain like the interior of buildings and vehicles. To navigate a complex environment, a robot may need to exploit contact with different parts of its body such as hands, feet, knees, and even the torso. As a human hiker or rock climber discovers via experience, each body part is appropriate for use on different terrain features. For example, a whole-hand hold may be appropriate for grasping a bulging overhead rock, while a secure toe hold can be obtained on a small protrusion. We call the problem of identifying potential contacts *footstep prediction*, even if the contact is not made with a foot.

This paper presents a novel footstep prediction method for versatile legged robots based on a deep convolutional neural network (CNN) architecture. We present two technical contributions to achieve high accuracy in varied terrain. First, because terrain images are usually highly foreshortened, our method generates perspective-corrected image patches with a consistent absolute size to feed into the CNNs. Second, the robot can capture terrain images from a variety of camera tilt angles using its arm-mounted and body-mounted RGB-D cameras. Because the utility of the feature depends on inclination relative to world coordinates, we incorporate the tilt angle as input into the network. (Note that camera yaw is irrelevant to footstep feasibility, and roll can be corrected by image rotation.) Given an intensity/depth image of the terrain, as well as the camera orientation, the method predicts candidate footstep locations for each part of the robot's body that is available to make contact.

(a) RoboSimian



(b) Hook end effector ("Hand")



(c) Toe end effector ("Foot")

Fig. 1: RoboSimian posed on climbing wall, and its two end effectors.

Experiments demonstrate that this method predicts foothold locations more accurately than comparable machine learning techniques, and is able to generalize between different camera tilt angles and between environments. We also apply our predictor to generate coarse route plans, similar to how a hiker or climber can observe a terrain from a long distance and mentally plan a general route by observing the arrangement of terrain features. This route planner can distinguish between plausible and implausible routes on novel climbing gym walls, stairs, and various indoor environments.

## II. RELATED WORK

Route planning for offroad vehicles has often been performed in image-space by predicting terrain traversability from the appearance of small image patches [1], [2]. Other authors have built elevation maps from onboard lidar taking vantage point into account [3], and have developed probabilistic methods for estimating terrain geometry behind occluding features like ridges [4].

Unlike vehicles, in legged locomotion a robot must construct a route from a potentially scattered distribution of terrain features rather than localized ones. Legged robot route

*This work is supported by NSF grant NRI-#1527826

[1]W. Zhang is with the Department of Computer Science and Department of Statistical Science, Duke University, Durham, NC 27708, USA. wuming.zhang@duke.edu

[2]K. Hauser is with the Department of Electrical & Computer Engineering and Department of Mechanical Engineering & Materials Science, Duke University, Durham, NC 27708, USA. kris.hauser@duke.edu

planning is typically performed on a 2.5D height map of the environment that is primarily horizontal but uneven. Given a pre-scanned terrain height map the suitability of a given footstep location can be taught by a human expert as a learned combination of height map features, such as flatness and convexity [5], [6]. Other work addresses the fact that on-board sensors provide an oblique vantage point of the terrain [7]. Several authors have noted that data quality tends to be very sparse and noisy at longer distance and so past work has developed techniques for filling in occluded areas in height maps [8] and incorporating sensing uncertainty into height maps [9], [10], [11].

With the versatility of the RoboSimian, we are afforded a larger range of traversable environments, including overhanging terrain. The wide range of camera vantage points and terrain slopes encountered when switching between walking and climbing makes 2.5D heightmap approaches inappropriate for route planning in full 3D. Past authors have built 3D terrain models using multiple laser or stereo vision scans [12], [13], [14], [15]. However, these maps are represented by occupancy grids, which are projected down onto a heightmap for footstep planning. Occupancy grids are too coarse for reasoning about the small terrain features upon which hooks and pointed toes can make contact. Furthermore, to climb vertically with hooks or hands the robot must reason about the occluded shape of handholds above the robot. Object recognition techniques that operate on point clouds are typically tuned for indoor scenes [16], and would fail to identify terrain features in occluded regions or distant areas where the point cloud is sparse.

Our work, like many others, relies on a human expert to define "good" footholds for learning. This is not necessarily easy to do for a robot with unusual or non-human-like physiology. There are other approaches for evaluating foothold quality, such as unsupervised tactile exploration [17], wherein the robot automatically touches sample terrains to discover terrain shapes that supply desired support forces. However, data generation is far more time consuming using this approach.

Convolutional neural networks (CNNs) have recently gained substantial attention for their state-of-the-art performance in object recognition. Region-CNN (R-CNN) is a powerful model for object bounding box estimation and identification [18], [19], [20]. It follows a pipeline of 1) region proposal, 2) classification, and 3) bounding box regression. Our method follows a similar pipeline but uses a dense region proposal method using depth information, and uses camera angle for prediction, which greatly improves accuracy. Other recent work has used CNNs for object recognition in 3D depth maps or point clouds, but do require identification of a bounding box or an isolated image of an object [21], [22]. In contrast, in rough terrain the objects that we seek are not visually distinct from the "background."

## III. SINGLE-IMAGE FOOTSTEP PREDICTION

We assume 2 body part types are allowed to make contact with the terrain. In our work we mounted 2 end effector types on the RoboSimian as shown in Fig. 1. On the front limbs ("hands") we mounted hooks used by human aid climbers that can securely grasp onto protruding cracks. On the rear limbs ("feet") we mounted rubberized "toes" modeled after human rock climbing shoes.

Our RoboSimian has 3 stereo camera pairs on the torso and one RGB-D camera on each "forearm." The RGB-D camera used in this project is the Orbbec Astra Pro, which has a depth range from $0.4$ to $8$m. The input to our system is an intensity / depth image from the camera as well as the estimated tilt angle of the camera's viewing direction. We discard color information because our training set includes brightly colored climbing gym holds, and we wish to avoid overfitting on color. Specifically, we preprocess color images by preserving intensity only. The input to our perception system is a $640 \times 480$ image with 8-bit intensity and 16-bit depth channels. Both channels are rectified and aligned.

The overall pipeline of our method consists of the following main steps:

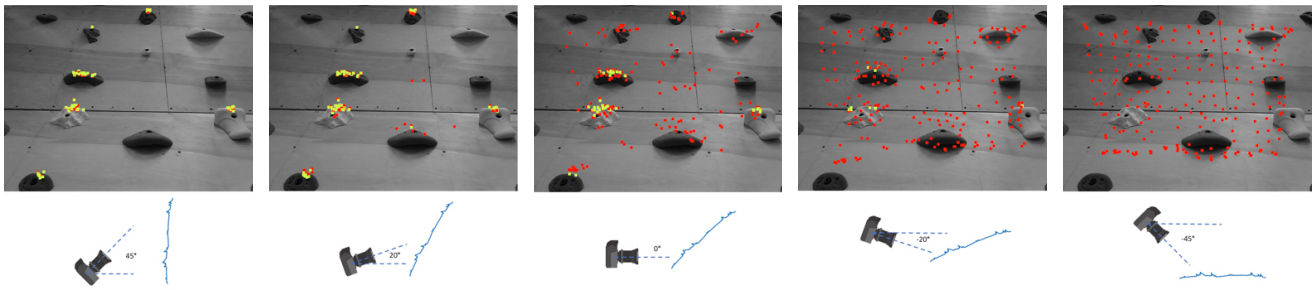
- An intensity / depth image is captured, and a windowing approach is used to uniformly sample perspective-corrected patches across the image. The camera orientation is also captured at this point.
- A CNN predicts whether a proposed patch contains a good foothold & a good handhold and the most likely pixel position for the best foothold & handhold.
- The CNN training set consists of images of various terrains labeled with footholds and handholds as determined by a human expert.

The dramatic effect of camera angle on footstep utility is illustrated in Fig. 2. Terrains that have identical appearance but different slopes should be climbed differently, and our model successfully makes predictions in these scenarios.

### A. Region Proposal and Perspective Correction

We first generate perspective-corrected image patches to be used as input for foothold prediction. This procedure is performed to enforce distance invariance so that a hold farther away is at the same size in the image patch as the same hold close by. To do so we place an $m \times n$ grid on the intensity and depth images (a $14 \times 19$ grid is used in our implementation, giving a 32 pixel spacing). An image patch is placed at each vertex $p_i$, $i = 1, \ldots, mn$ with its width and height $s_i$ determined as follows. First, we estimate the depth $d_i$ at $p_i$ by averaging a $50\times$ 50 pixel patch in the depth image, centered at $p_i$. Then, to calculate $s_i$ we take a nominal minimum patch size $s_{far}$ at the farthest detectable distance $d_{far}$, and set $s_i = round(s_{far} \cdot d_{far}/d_i)$. We choose $s_{far} = 10$ pixels. Finally, we rescale all patches to a consistent size, which is $60 \times 60$ pixels for our network. Examples of perspective-corrected regions are shown in Fig. 3.

Before the prediction step, we normalize depth and intensity in each region to make our model more robust to depth and lighting variation. For the depth channel, we first normalize by the absolute depth range and then subtract the mean depth of pixels in the patch. We also normalize the intensity channel to the range $[-1, 1]$.



(a) Pitch $= 45°$ (b) Pitch $= 20°$. (c) Pitch $= 0°$. (d) Pitch $= -20°$. (e) Pitch $= -45°$.

Fig. 2: The utility of a terrain feature is dependent on appearance as well as camera pitch. If this image were seen looking up, then the hands and feet must make contact with protruding terrain features. If the same image were seen looking down, then the ground is nearly flat and only feet should be used. The foot- and hand-holds predicted by our method are shown in red and yellow, respectively (best viewed in color).

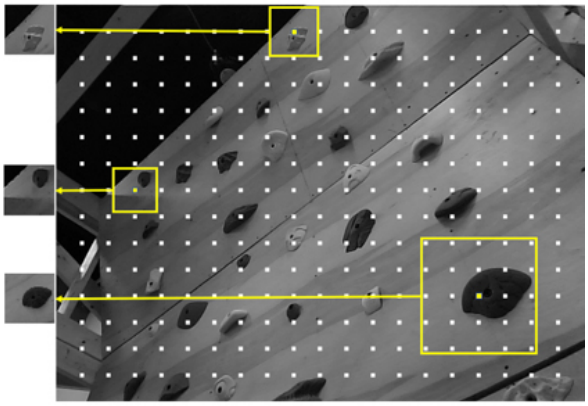


Fig. 3: Perspective-corrected region proposal on a vertical climbing gym terrain. The center positions of all proposed regions are shown in white. Three image patches and their normalized versions are shown in yellow.

### *B. Foothold and Handhold Prediction with CNN*

Our predictor is a unified variant of a CNN that performs classification and regression for all body parts using a multi-task loss function. The overall CNN architecture is shown in Fig 4. Given a normalized image region $I$ and camera tilt angle $\theta$, we classify whether a hold exists in $I$ for each body part and find the most likely pixel location of the hold $(x, y)$. We train a CNN that takes as input the image $I$ (size $2 \times 60 \times 60$) and three *side-channel* values $(\cos\theta, \sin\theta, \theta/90°)$ giving a viewing-angle dependency to the predictions. (The division in the last side-channel value rescales the angle to be in the range $[-1, 1]$. ) The output of our network is a 6-tuple $(p_f, p_h, x_f, y_f, x_h, y_h)$, where $p_f$ indicates that a foothold exists, $p_h$ indicates that a handhold exists, and $(x_f, y_f)$ and $(x_h, y_h)$ are the coordinates of the holds, normalized to the range $[0, 1]^2$.

To include the side-channel information with the main image channel, we explored three recently proposed architectures: Mix-in, Stacking, and Activation Map [23]. Experiments found that the Mix-in has the best performance by a small margin. The convolutional stage of our neural network uses four convolutional layers, with 30, 30, 60 and 60 kernels respectively. Each convolutional layer is followed by an activation layer with ReLU function. There is a pooling layer with $2 \times 2$ max-pooling after the first, second and fourth convolutional layer. For the fully connected stage, we flatten the output of the convolutional layers into a single vector, concatenate the side-channel information, and use two fully connected layers with each containing 500 neurons. Again, an ReLU activation layer follows each fully connected layer. There are six output channels, where the first two are thresholded to give classification results $p_f$ and $p_h$, and the other four perform regression to generate the $(x_f, y_f)$ and $(x_h, y_h)$ coordinates.

Network hyperparameters were chosen by tuning. We experimented with deeper neural networks with up to 9 convolutional layers and up to 5 fully-connected layers. We also explored kernel sizes from 30 to 120 with a spacing of 30, keeping the first and second convolutional layers have the same kernel number and the last two have the same kernel number. The structure described above was found to perform the best.

To train the network to perform classification and regression simultaneously, we define a multi-task loss function [19], [24] as

$$L(p_f, p_h, x_f, y_f, x_h, y_h) = L_{cls}(p_f, p_f^*) + L_{cls}(p_h, p_h^*) + \lambda[p_f^* L_{reg}((x_f, y_f), t_f^*) + p_h^* L_{reg}((x_h, y_h), t_h^*)]. \quad (1)$$

Here, $p_f^*$ are $p_h^*$ are 1 if the ground truth label for corresponding hold (food or hand, respectively) is positive, and 0 otherwise; $t_f^*$ and $t_h^*$ are the ground truth $(x, y)$ coordinates of their corresponding holds relative to the image patch. Like the regression output $(x_f, y_f)$ and $(x_h, y_h)$, $t_f^*$ and $t_h^*$ are both scaled to $[0, 1]$. $L_{cls}$ is the negative log likelihood of the two classes and $L_{reg}$ is the smooth $L_1$ loss, as defined in [19]. The significance of this loss function is that the regression loss terms are only activated when the corresponding classification label is positive. The parameter $\lambda$ trades off between classification and regression accuracy.

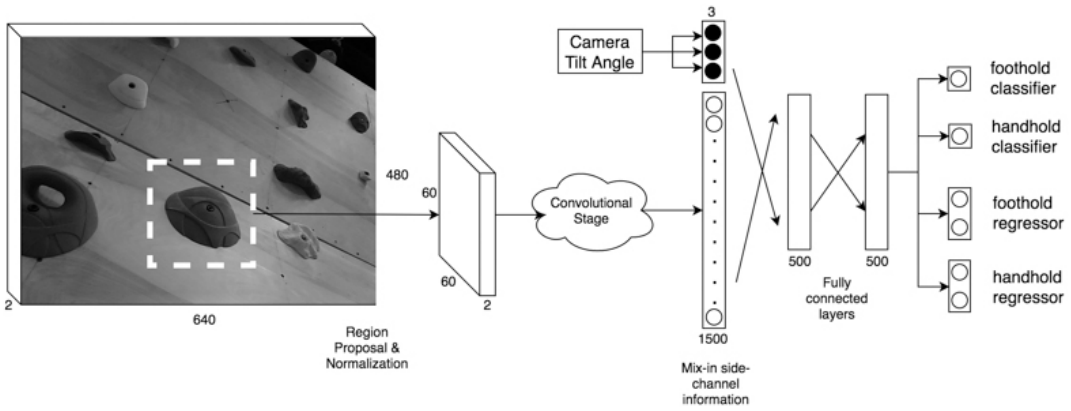



Fig. 4: CNN architecture illustration

Experimental performance was relatively insensitive to the value of $\lambda$, so we used $\lambda = 1$.

### C. Training and labeling ground truth

For training we captured 138 images of several terrains — indoor climbing walls, floors, rotated-down indoor climbing wall (with synthetically adjusted tilt angle to suggest that the holds are on the floor) and stairs (Fig. 5). For each image a human expert "paints" good foothold positions in a distinctive color for each available body part used for contact. Blue is used for footholds, green for handholds, and red for both hand and footholds, as illustrated in Fig. 5. Although human intuition does not necessarily provide perfect "ground truth" for whether a foothold is useful for a robot, we allowed the expert to manually touch the terrain with the end effectors to build a mental model of what makes a secure hold.

After labeling, we then used region proposal to generate 266 observations per RGB-D image. Each observation is labeled as a positive foothold if the center 80% of the region contains more than 20 blue or red pixels, and a positive handhold if the center contains more than 20 green or red pixels. We ignore holds at the patch border since the patch contains insufficient information to determine its suitability. This does not significantly increase false negative rate because neighboring patches do overlap. We then augment the data to $266 \times 9 = 2{,}394$ observations by randomly adjusting brightness and contrast, and mirroring the image horizontally. In total, we have $138 \times 2{,}394 = 330{,}372$ training observations. Both CNN training and testing were performed on an Nvidia GeForce GTX 1080 using Theano in Python. We considered training to converge when testing error did not decrease by more than $0.5\%$ after $10$ iterations through the entire training data. This process takes approximately 2 hours.

### D. Experiments on Novel Terrains

For testing, we captured another 40 images of six terrain types: 1) the same indoor climbing wall as in training, but with new holds, 2) indoor climbing wall, 3) rotated-down indoor climbing wall, 4) stairs, 5) hallways, 6) office furniture. Novel environments were used for types 2–6, while in type 1 we used the same wall but a entirely different set of 42 holds from the 42 holds in the training set. For ground truth we follow the same labeling procedure as performed for the training set.

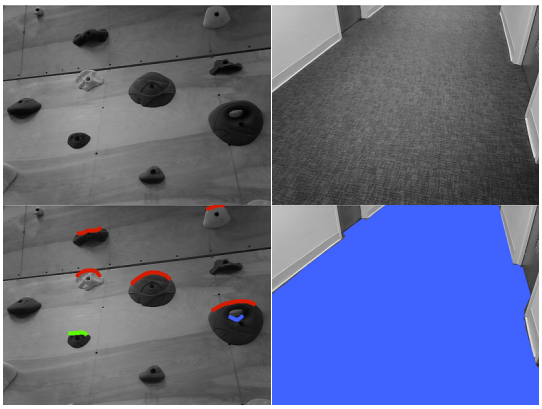
TABLE I: Overall classification error (c) and regression error in pixels (r) for different terrains

| | Foot (c) | Hand (c) | Foot (r) | Hand (r) |
|---|---|---|---|---|
| New holds | 12.71 % | 11.95 % | 8.88 | 8.86 |
| Climbing wall | 14.43 % | 13.08 % | 9.33 | 11.10 |
| Rotated-down wall | 1.82 % | 1.25 % | 4.07 | 1.16 |
| Stair | 1.28 % | 11.13 % | 6.08 | 8.07 |
| Hallway | 9.02 % | 1.88 % | 7.35 | 2.20 |
| Furniture | 18.05 % | 23.68 % | 10.47 | 9.83 |
| Range | [0, 100%] | [0, 100%] | [0, 85px] | [0, 85px] |

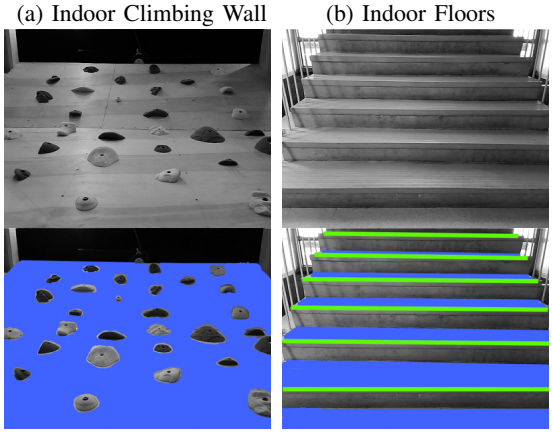
Tab. I shows that our model achieves high accuracy across multiple terrain types. Region proposal and footstep prediction across an entire image takes around 2 seconds in unoptimized Python code on a 4.0 GHz Intel Core i7-6700K Quad-Core CPU. Fig. 6 shows some examples of predicted footholds on the testing set. We observe that it performs quite well at predicting that hands should not be used on flat ground and the synthetic "uneven terrain" of the rotated-down wall. Surprisingly, it also produces fairly reasonable estimates on the Furniture environment, which is drastically different from anything seen in training. We also observe that the predictions are more accurate for nearby terrain, which is possibly caused by the smaller region size and greater noise in depth at longer distance.

### E. Comparisons with Other Models

Tab. II shows comparisons between our model and others on the overall testing set performance. *No angle* gives results with our same CNN model, but trained and tested without the camera tilt angle side-channel. Classification performance drops drastically, with error increasing by approximately a factor of two. *I Only* and *D Only* measure the effect of restricting our network to only intensity or depth channels, respectively. It can be seen that keeping both intensity and depth indeed helps achieve better performance.

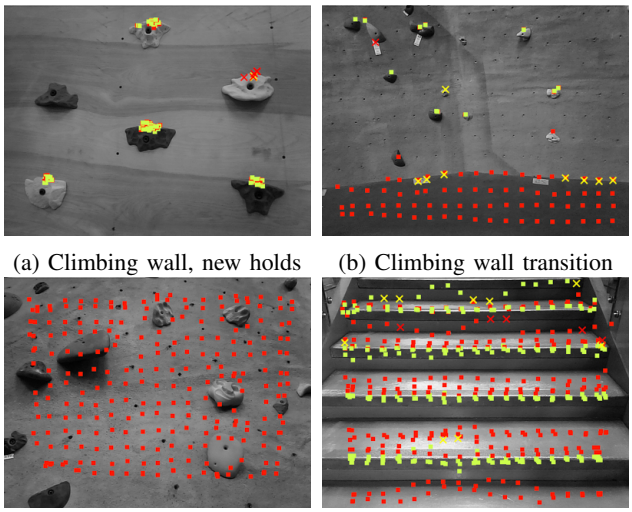


(a) Indoor Climbing Wall (b) Indoor Floors



(c) Rotated-down Wall (d) Indoor Stairs

Fig. 5: Example of human expert labeling by painting. Blue: good for foot only. Green: good for hand only. Red: good for both foot and hand (best viewed in color).



(a) Climbing wall, new holds (b) Climbing wall transition

(c) Rotated-down wall (d) Stairs

(e) Hallways (f) Office furniture

Fig. 6: Hold prediction on test terrains. Predicted locations of footholds are plotted in red and handholds in yellow (dot: true positive, cross: false positive). (best viewed in color)

Next we compare to features used by prior authors to estimate foot cost for quadruped robots [25], [26], [27]. In each of these methods, the foot cost is assumed to be a weighted linear combination of local terrain features extracted from the height map at multiple scales. The most commonly used features are:

- Standard deviation of the heights in the cell
- Maximum and minimum height in the cell relatively to height at the center position
- Average slope in $x, y$ directions
- Average curvature in $x, y$ directions

The weight vector is usually derived by experimentally tuning [27] or some learning technique such as hierarchical pprenticeship learning [25]. To directly compare this feature extraction method to our method, for each of our $60 \times 60$ proposed region, we extract these 7 features at three different scales of the depth image (a sub-image of size $10\times10$, $30\times30$ and $50 \times 50$ centered at the original region center), and add the camera angle as an additional feature to form a feature vector $f \in \mathbb{R}^{22}$. Given this feature space, we derive the best weight vector by fitting a linear SVM classifier and a regression model. The SVM row of Tab. II shows the test set error. We also compared radial basis function kernels with similarly poor results. This suggests that CNN is a much more powerful tool in extracting terrain image features than manually defined feature extraction.

TABLE II: Classification error (c) and regression error (r) compared to other models

| | Foot (c) | Hand (c) | Foot (r) | Hand (r) |
|---|---|---|---|---|
| Our method | **8.84%** | **7.07%** | **6.45** | **8.20** |
| No angle | 23.41% | 14.08% | 12.24 | 10.34 |
| I Only | 12.84% | 10.15% | 8.32 | 9.20 |
| D Only | 10.82% | 9.35% | 7.34 | 10.00 |
| SVM | 17.60% | 21.90% | 7.87 | 14.94 |
| Fast R-CNN | 22.5% | 24.6% | 13.7 | 16.5 |
| Range | [0, 100%] | [0, 100%] | [0, 85px] | [0, 85px] |

We also compare to Fast R-CNN [19], in which we use ground-truth labels with normalized proposal regions for training. For testing, we consider true positives to be any ground-truth hold within a predicted region, false positives as any region that contains no hold, and false negatives as any ground truth hold outside of any predicted region. Tab. II shows that our method outperforms R-CNN by a large margin, most likely because 1) that there is no clear "region" in our locomotion problem that separates visually distinct foreground and background objects, and 2) camera angle information is not used in R-CNN.

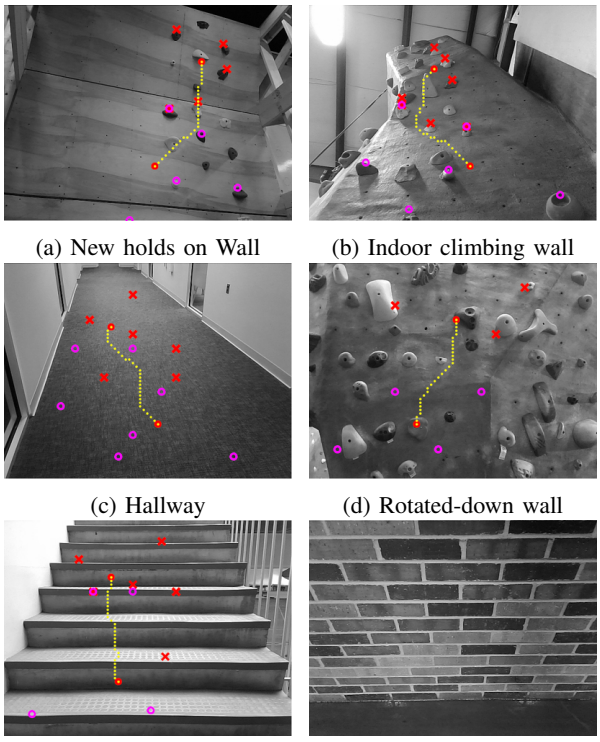
## IV. Application to Coarse Route Planning

We illustrate how long-range footstep predictions can be used for locomotion by suggesting coarse routes for a legged robot. Human hikers and climbers develop general, imperfect routes from long-distance observations, and then follow these routes using individual footsteps. Due to occlusions and low resolution at far distances, a route may be imperfect, and it should be adjusted throughout navigation as new information becomes available. Nevertheless a route should not, for example, suggest for the robot to climb a featureless cliff, jump over a wide chasm, or climb strenuous terrain when easy terrain suffices.

Let $H$ denote the set of predicted hold positions as computed by our footstep predictor, and projected into 3D space according to estimated depth and camera orientation. Our route planner calculates a path for the robot's body position such that there is an available hold within reach of each limb. We require that at each body position $(x, y, z)$ the robot can reach a valid *stance* $\sigma \equiv (h_l, h_r, f_l, f_r)$. A stance is defined as any subset of $H$ with size 4, which assigns hand or footholds $h_l$ and $h_r$ to the left and right front limbs, respectively, (these can be used as "hands" or "feet") and assigns footholds $f_l$ and $f_r$ to the left and right rear limbs (which can only be used as "feet"). To improve the precision of footstep predictions and improve computational performance, we first employ k-means clustering algorithm to cluster both foothold and handhold predictions (we use k = 30). Precision for handhold and foothold predictions improve by around 0.6% and 2.6% respectively.

We require all four holds in a stance to be mutually reachable by each end effector, where each end effector has an approximate workspace that is a sphere of radius 0.7 m centered at its shoulder. To avoid limb interference we filter out any stances in which two holds are too close (in our implementation, $< 0.3$ m apart). For each position-stance pair we then define a cost term that prioritizes flexibility of movement. Define the movement flexibility score $Flex(v, \sigma) \in [0, 1]$ of a position-stance pair is defined as the fraction of the robot's height and yaw workspace in which $h_l, h_r, f_l$, and $f_r$ are reachable. This score is approximated by defining a nominal pose about the terrain plane estimated by MSAC [28], and by testing hold reachability for each body pose over a grid in height and yaw space.

We use Dijkstra's algorithm to search for a body path on a $60 \times 80$ grid corresponding to pixels of a downsampled RGB-D image. We construct a directed graph $G = (V, E)$ over this image in which each vertex contains the coordinates of a pixel back-projected into 3D space using the estimated depth. Each vertex is connected to its 8 neighbouring pixels. We define the optimal stance at a vertex $\sigma^*(v)$ as the one with largest value of $Flex(v, \sigma)$, and then assign the cost of an edge $(u, v)$ as $Cost[u, v] = \|u - v\| \cdot (1 - Flex(\sigma^*(v)))$. Given the optimal body path $v_1, \ldots, v_n$ the corresponding optimal stances $\sigma^*(v_1), \ldots, \sigma^*(v_n)$ is retrieved as the footstep plan. The stance sequence is then refined to avoid excessive numbers of footsteps, by not moving a limb if its next hold is within a threshold of 0.3 m.



(a) New holds on Wall (b) Indoor climbing wall

(c) Hallway (d) Rotated-down wall

(e) Stairs (f) Empty wall (no route)

Fig. 7: Examples of planned routes in several terrains. Cross: steps for front limb. Circle: steps for rear limb.

Fig. 7 shows the results on some test terrains. We manually set a starting position at the bottom of the image and the goal to arrive anywhere above a given horizontal line. The planner takes around 30 seconds using unoptimized MATLAB code, with the running time dependent on the total number of predicted footholds. Observe especially in Figs. 7a, 7b, and 7f that the route correctly avoids areas that are impassable. In Figs. 7c, 7d, and 7e, there are many routes available, and the planned routes are relatively short.

## V. Conclusion

This paper presented a single-image footstep predictor and route planner for versatile legged robots based on perspective- and tilt-corrected deep-learning on RGB-D image patches. Trained on only 138 human-labeled images, experiments show that our model outperforms competing methods, and demonstrate that our model can generalize footstep predictions to novel terrains. Camera pitch is also shown to be essential for prediction performance.

Ongoing work is studying how to integrate our footstep predictor onto the RoboSimian to validate whether the predicted holds work well in practice. We are also investigating how to integrate footstep prediction with coarse route planning and full-body motion planning on the physical robot. We are interested in predicting footholds in larger terrain maps stitched from multiple views and heterogeneous imaging modalities (e.g., stereo, structured light, lidar). It

may also be possible to use locomotion experience or tactile exploration data as feedback into learning to improve prediction accuracy.

## References


[1] M. Suzuki, E. Terada, T. Saitoh, and Y. Kuroda, "Vision based far-range perception and traversability analysis using predictive probability of terrain classification," in *41st Int. Symp. Robotics*, 2010, pp. 50–55.

[2] M. Bajracharya, B. Tang, A. Howard, M. Turmon, and L. Matthies, "Learning long-range terrain classification for autonomous navigation," in *IEEE Int. Conf. Rob. Aut.*, 2008, pp. 4018–4024.

[3] E. Krotkov and R. Hoffman, "Terrain mapping for a walking planetary rover," *IEEE T. Rob. Aut.*, vol. 10, no. 6, pp. 728–739, 1994.

[4] R. Hadsell, J. A. Bagnell, D. Huber, and M. Hebert, "Space-carving kernels for accurate rough terrain estimation," *Int. J. Rob. Res.*, vol. 29, no. 8, pp. 981–996, 2010.

[5] J. Z. Kolter, P. Abbeel, and A. Y. Ng, "Hierarchical apprenticeship learning with application to quadruped locomotion," in *Neur. Inf. Proc. Sys.*, 2008, pp. 769–776.

[6] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Fast, robust quadruped locomotion over challenging terrain," in *IEEE Int. Conf. Rob. Aut.*, 2010, pp. 2665–2670.

[7] J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, and S. Kagami, "Biped navigation in rough environments using on-board sensing," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys*, 2009, pp. 3543–3548.

[8] J. Z. Kolter, Y. Kim, and A. Y. Ng, "Stereo vision and terrain modeling for quadruped robots," in *IEEE Int. Conf. Rob. Aut.*, 2009, pp. 1557–1564.

[9] D. Belter, P. Łabeçki, and P. Skrzypczyński, "Map-based adaptive foothold planning for unstructured terrain walking," in *IEEE Int. Conf. Rob. Aut.*, 2010, pp. 5256–5261.

[10] P. Łabęcki, D. Rosiński, and P. Skrzypczyński, "Terrain map building for a walking robot equipped with an active 2d range sensor," *J. Aut., Mobile Rob., Intel. Sys.*, vol. 5, no. 3, pp. 67–87, 2011.

[11] C. Plagemann, S. Mischke, S. Prentice, K. Kersting, N. Roy, and W.Burgard, "Learning predictive terrain models for legged robot locomotion," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys*, 2008, pp. 3545—3552.

[12] C. Mastalli, I. Havoutis, A. W. Winkler, D. G. Caldwell, and C. Semini, "On-line and on-board planning for quadrupedal locomotion, using practical, on-board perception," in *IEEE Int. Conf. Tech. Practical Robot Applications*, May 2015.

[13] R. B. Rusu, A. Sundaresan, B. Morisset, K. Hauser, M. Agrawal, J.-C. Latombe, and M. Beetz, "Leaving flatland: Efficient real-time three-dimensional perception and motion planning," *J. Field Robotics*, vol. 26, no. 10, pp. 841–862, 2009.

[14] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *IEEE Int. Conf. Rob. Aut.*, 2015, pp. 5148–5154.

[15] A. Stelzer, H. Hirschmüller, and M. Görner, "Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain," *Int. J. Rob. Res.*, vol. 31, no. 4, pp. 381–402, 2012.

[16] L. Bo, X. Ren, and D. Fox, "Depth kernel descriptors for object recognition," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys*, Sept 2011, pp. 821–826.

[17] M. A. Hoepflinger, M. Hutter, C. Gehring, M. Bloesch, and R. Siegwart, "Unsupervised identification and prediction of foothold robustness," in *IEEE Int. Conf. Rob. Aut.*, 2013, pp. 3293–3298.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conf. Comp. Vis. Pat. Rec.*, June 2014.

[19] R. Girshick, "Fast r-cnn," in *IEEE Int. Conf. Comp. Vis.*, 2015, pp. 1440–1448.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conf. Comp. Vis. Pat. Rec.*, 2016, pp. 779–788.

[21] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys*, 2015, pp. 922–928.

[22] M. Schwarz, H. Schulz, and S. Behnke, "Rgb-d object recognition and pose estimation based on pre-trained convolutional neural network features," in *IEEE Int. Conf. Rob. Aut.*, 2015, pp. 1329–1335.

[23] Y. Zhou and K. Hauser, "Incorporating side-channel information into convolutional neural networks for robotic tasks," in *IEEE Int. Conf. Rob. Aut.*, 2017.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Neur. Inf. Proc. Sys.*, 2015, pp. 91–99.

[25] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng, "A control architecture for quadruped locomotion over rough terrain," in *IEEE Int. Conf. Rob. Aut.*, 2008, pp. 811–818.

[26] M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal, "Learning locomotion over rough terrain using terrain templates," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys*, 2009, pp. 167–172.

[27] A. Winkler, I. Havoutis, S. Bazeille, J. Ortiz, M. Focchi, R. Dillmann, D. Caldwell, and C. Semini, "Path planning with force-based foothold adaptation and virtual model control for torque controlled quadruped robots," in *IEEE Int. Conf. Rob. Aut.*, 2014, pp. 6476–6482.

[28] P. H. Torr and A. Zisserman, "Mlesac: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.