Single-shot Foothold Selection and Constraint Evaluation for Quadruped Locomotion

Single-shot Foothold Selection and Constraint Evaluation for Quadruped Locomotion

Dominik Belter[1], Jakub Bednarek[1], Hsiu-Chin Lin[2], Guiyang Xin[2], Michael Mistry[2]

***Abstract*— In this paper, we propose a method for selecting the optimal footholds for legged systems. The goal of the proposed method is to find the best foothold for the swing leg on a local elevation map. First, we evaluate the geometrical characteristics of each cell on the elevation map, checks kinematic constraints and collisions. Then, we apply the Convolutional Neural Network to learn the relationship between the local elevation map and the quality of potential footholds. During execution time, the controller obtains the qualitative measurement of each potential foothold from the neural model. This method evaluates hundreds of potential footholds and checks multiple constraints in a single step which takes 10 ms on a standard computer without GPU. The experiments were carried out on a quadruped robot walking over rough terrain in both simulation and real robotic platforms.**

I. Introduction

Locomotion in challenging terrain requires careful selection of footholds. The robot should select stable support for each foot to avoid slippages and falls. This strategy is crucial when the robot needs to deal with highly irregular terrain. A challenging example is maneuvering in extreme environments such as a mine, or the aftermath of a natural disaster where the robot can find only a few acceptable footholds. A poor foothold selection method means that the robot may fall and cannot execute the mission.

In contrast, other types of locomotion assume that the robot walks dynamically on rough terrain and stabilizes its posture using fast control algorithms and compliant legs [1], [2]. Stable locomotion relies on the capability of the controller to cope with disturbances (e.g. slippages) resulting from unstable footholds. However, this approach is only efficient on moderately rough terrain and it will fail when the robot has to face extreme environments such as a mine, or the aftermath of a natural disaster.

To efficiently navigate in a more extreme environment, perception systems are required to detect high obstacles so the robot can avoid them while walking [3]. A full 3D model of the environment can be obtained using terrain mapping methods, such as OctoMap [4], Normal Distribution Transform Occupancy Maps (NDT-OM) [5], or elevation map [6], [7]. Among all of the above, elevation map sufficiently represents the terrain, guarantees quick access to each cell, and can be directly transformed to grayscale image to feed the Convolutional Neural Network [8] which we use in this research.

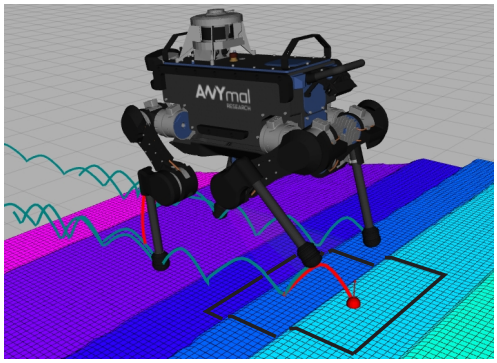

Fig. 1. The foothold selection problem for a quadruped robot. The region of the elevation map below the i-th leg is evaluated to find the best foothold. Each candidate position of the foot has to be kinematically feasible and collision-free.

The problem of foothold selection is presented in Fig. 1. The robot evaluates the region (elevation map) to select the best foothold that fulfills a set of kinematics (e.g., workspace limit, self-collision, etc) and environmental constraints (e.g., avoiding sharp edges). In the classical approach, all constraints are verified sequentially by the controller of the robot during walking [9]. On the other hand, online optimization is time-consuming and not feasible for real-time control.

In this research, we propose a computationally efficient solution for foothold selection. We applied a neural network to learn a model (off-line) that maps the properties of the terrain to the quality of a potential foothold while excluding footholds which are risky or kinematically infeasible. During execution time, we efficiently predict the quality of a potential foothold from the learned model. The proposed method is verified on a quadruped robot walking over rough terrain, in both simulation and real robot platforms.

II. Related Work

The problem of foothold selection is similar to the problem of multi-finger grasping and was studied widely by the robotics community. Recent development in this field includes the method which use local geometrical properties of the objects to find the acceptable positions of the fingertips on the object's surface [10]. The grasp con-

*This research was supported by EU Horizon 2020 project THING. This work has been conducted as part of ANYmal Research, a community to advance legged robotics. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

[1]Institute of Control, Robotics and Information Engineering, Poznan University of Technology, Poznan, Poland dominik.belter@put.poznan.pl

[2]School of Informatics, University of Edinburgh, Edinburgh, UK

figurations are trained from real examples. The collision and kinematic constraints are taken into account during the inference procedure. Recently, deep neural network, such as the Convolutional Neural Networks (CNN) gained high popularity in robotics applications. In grasp, the CNN is applied to select feasible grasp and robotics finger positions on the object's surface using point cloud [11] or depth images [12].

Most approaches for the foothold selection are based on the local features computed for the terrain surface, such as the inclination of the terrain, roughness, and local curvature from the elevation maps [13]. These features are provided to the input of the simple neural network which was trained on the data provided by human experts. Another approach takes the elevation map and estimates a probability map that is related to the capability of each cell to provide stable support for the robot's feet [14]. The StarlETH robot is equipped with the haptic device on the feet, which explores and evaluates the potential footholds without human supervision [15]. The HyQ robot focuses more on the reflexes which stabilize the robot [16], but it also uses visual information about the terrain to avoid risky footholds [17]. The robot corrects the nominal foothold positions according to the output from the visual pattern classifier applied on the terrain patches.

Great progress in the field of autonomous legged locomotion on rough terrain was done on the quadruped robot LittleDog [18]. The authors proposed a terrain scorer which computes the spatial relationship between a considered point and its neighboring points and then rejects points which are located on edges, large slope, the base of a cliff, or inside of a hole.

A learning-based method was proposed to evaluate terrain templates based on the human demonstration [19]. The terrain scorer approach is also adapted in [20], where the weights of geometric features of the terrain are obtained during training and then used for the footsteps planning. Very recently, the CNN classifier for the footholds has been proposed [21].

The foothold selection method for a six-legged robot is represented by the method implemented on the Lauron IV robot [22]. The foothold selection module considers points around initial foothold and takes into account elevation credibility, the mean height, and the height variance of the cells. The six-legged Messor robot learns which points on the elevation map can provide stable support from simulation data [9]. Then, the trained Gaussian Mixture is used to select the footholds in the RRT-based motion planner [23]. The kinematic and self-collision constraints are also taken into account. However, this process significantly slow-downs the foothold selection process.

A. Approach and Contribution

In this paper, we propose a novel method to evaluate potential footholds for the quadruped robot in a single step using CNN. We collect data for training the network using the kinematic model of the robot, elevation map of rough terrain and a reference foothold scorer [9]. The learnt network implicitly stores information about the kinematic and collision model of the leg and rejects footholds which are outside the workspace. The main contributions of this paper include the following:

1) We are the first who show that the CNN can be used to evaluate geometrical properties of the potential footholds and simultaneously consider all kinematic constraints which are related to the model of the robot. Comparing to formal work which optimizes constraints on-line, this significantly reduces the online computational time.
2) A transfer learning approach that learns a model from data gathered in the simulation. With this approach, we provide a sufficient number of examples to train a neural network without the need for using the real robot. We also show that the obtained neural model is successfully transferred to the real robot.

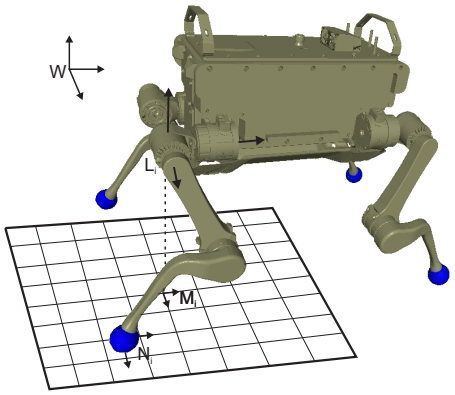

Fig. 2. Center of the local map M_i used for foothold selection is located below the i-th leg joint L_i. The foothold selection algorithm considers also the nominal position of the foot N_i.

III. Foothold Selection Module

We propose a foothold selection module to evaluate potential footholds which are inside the local map extracted from the global elevation map.

A. Problem Definition

A *global map* is an elevation map built online by the robot and aligned with the world coordinate system W, where the center of this map is the center of the robot projected on the ground. The size of the global map is $6 \times 6m$ and the size of each cell is 2×2 cm.

A *local map* is a subset of the global elevation map where the center M_i is a point below the hip joint of the considered leg (presented in Fig. 2). The size of the local map, which is 40×40 cells, covers the kinematic range of the leg.

A *nominal foothold* N_i is the desired position of the foot for a given step length and assuming that the robot is walking on flat terrain. Lastly, a *potential foothold* is a cell inside of the local map.

Our goal is to select an optimal foothold from a set of potential footholds online using the information from the local map that satisfies the following constraints. First, the robot should avoid selecting footholds on sharp edges and/or steep slopes because they are potentially risky. Second, the selected foothold should be within the kinematic limit of the robot (inside the workspace of the leg). Also, the robot should avoid self-collisions and check whether the thigh or shank collides with the terrain.

B. Dataset

Training a model from the robot is expensive, time-consuming, and dangerous since sensor data is prone to noise. Thus, we took the *transfer learning* approach, where data are gathered in simulation but used on the real robot.

To train the neural network we collect the samples on the $12 \times 12m$ elevation map presented in Fig. 3. The map was created offline by composing maps obtained during various experiments on the robot. We also added the flat region, steps with various height, concavities, and bumps to increase the variation of foothold examples.

Since the robot is symmetrical, we train the models for the right legs and adapt it for the left legs. To this end, we have to flip horizontally the input terrain map and after inference, we flip horizontally the obtained cost map. Therefore, we only need to collect data for two legs and train two separate models.

To generate training data, we randomly select the position of the robot on the map (horizontal position and distance to the ground). The orientation of the robot on the horizontal plane (yaw angle) is randomly selected from four main orientations: $n \cdot \frac{\Pi}{2}$, for $n = 0,1,2,3$. For the obtained pose of the robot, we compute the pose of the i-th leg and extract a 40×40 local map. For each cell of the map, we quality the cost of a foothold $c_f \in [0,255]$ based on a set of constraints. This quantity is first evaluated based on the following *hard constraints*:

1) *kinematic range of the leg*: If the given position of the foot is outside the workspace of the considered leg, we set the cost to the maximal value $c_f = 255$, and we do not check other constraints.
2) *self-collisions and collision with the ground*: We use Flexible Collision Library [24] to determine whether there is any collision between any pairs of rigid bodies and with the ground, except the collision between the foot and the terrain. A foothold is rejected (i.e., $c_f = 255$) if there is a collision, and we do not check further.

If a given potential foothold is collision-free and the foot is within leg's workspace, the quality of selecting a foothold is evaluated using the following evaluation criteria:

1) *kinematic margin*: The kinematic margin c_k is the distance between the current position of the foot and the border of the workspace. The maximal value of c_k means that the leg has the maximal motion range. The c_k value is normalized.

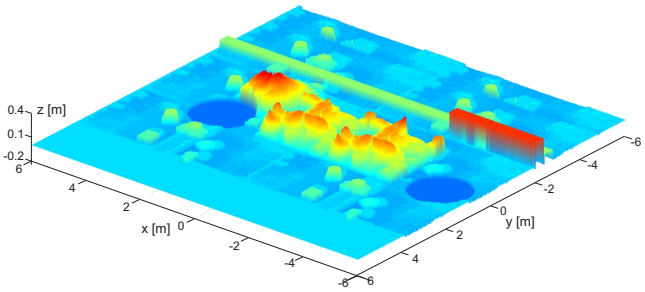

Fig. 3. Elevation map created offline to collect data for training the neural network.

2) *terrain cost*: The terrain cost c_m evaluates the property of the local map. We use the same cost function with the hexapod Messor robot [9] since both robots have a similar hemispherical foot. The c_m value is normalized.

Finally, we compute the final cost of the considered foothold c_f and scale the cost to the range $[0,255]$:

$$c_f = \frac{c_k + 2 \cdot c_m}{3} \cdot 255. \quad (1)$$

For each cell, we repeat the above procedure and save the input (elevation map) and the output (terrain cost). We collected 20000 training pairs for each leg.

Examples of training data are presented in Fig. 4. The first two columns present the computed cost maps for the flat terrain. In this case, the output depends on the leg's workspace and the kinematic margin. The distance between the terrain and the robot is larger on the map in Fig. 4a than in Fig. 4b. The output cost map also differs. The obtained cost maps (Fig. 4f and Fig. 4g) represents the horizontal cross-section over the workspace of the robot's leg. The yellow cells represent positions of the foot which are outside the workspace and are inaccessible for the robot ($c_f = 255$). Collisions, the edges on the obstacles or slopes also increase significantly the cost of footholds and the neural network classifies them as inaccessible (yellow color). The red cells correspond to acceptable footholds. In the following examples in Fig. 4c–e the terrain is irregular and we can observe how the workspace of the robot is limited by the terrain shape.

C. Convolutional Neural Network

We aim to learn a mapping between the features extracted from a local elevation map and the quality of potential footholds. In this work, we choose Convolutional Neural Network (CNN), since this architecture runs in real-time on machines without GPUs.

Because CNN is much more efficient in solving classification than regression, we discretize the terrain costs into C different classes. In this work, C is set to 14 since it is sufficient to distinguish between weak and good footholds and we can easily provide a sufficient number of training samples for each class.

The proposed CNN architecture is an Efficient Residual Factorized ConvNet (ERF) first introduced in [25]. The

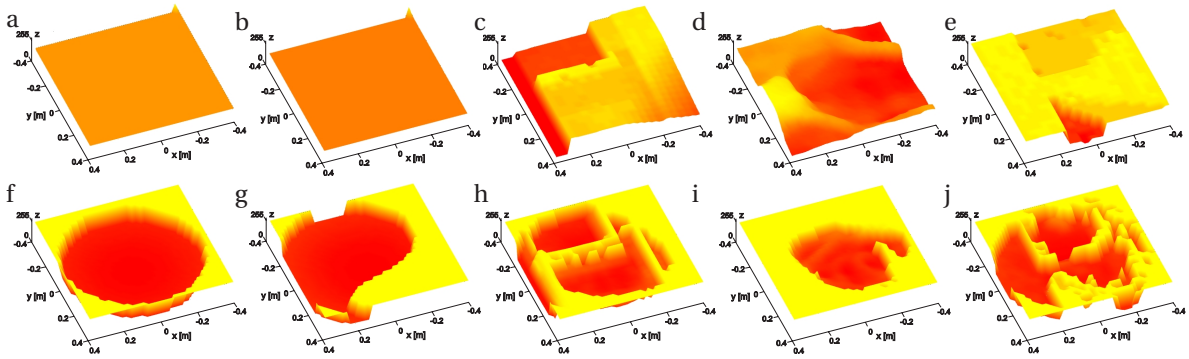

Fig. 4. Example training data: local elevation maps (a,b,c,d,e), and corresponding terrain cost (f,g,h,i,j) (red color – acceptable footholds, yellow – unacceptable footholds). Note that both subfigures a and b represent flat terrain but elevation is different. Thus, the acceptable region (red area which corresponds to the leg's workspace) obtained from the neural network is different in subfigures f and g.

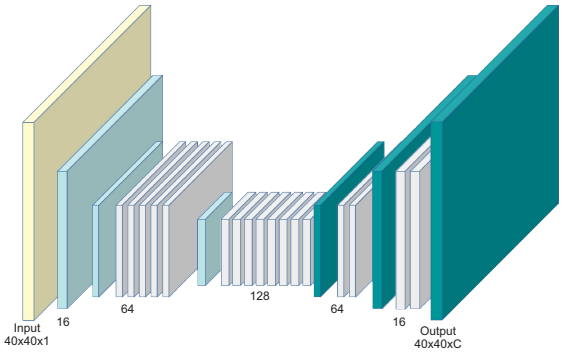

Fig. 5. Model of the ERF network. Light blue blocks represent downsampling, dark blue - upsampling by transposed convolution and white blocks show residual layers. Numbers below blocks describes the number of feature maps used in specific levels. C denotes the number of classes (14 in the current implementation).

Leg	Accuracy [%]	IoU
front leg	82.61	49.9
rear leg	82.61	49.88

TABLE I

ACCURACY AND INTERSECTION OVER UNION (IOU) OBTAINED ON VALIDATION SET FOR FRONT AND REAR LEG MODELS

characteristics of this model is the modification of the *residual layer* [26] called *residual non-bottleneck 1D layer*. The 2D convolution with a 3×3 filter is replaced by two 2D convolutions with filters shape 3×1 and 1×3. This approach reduces the number of variables and complexity.

The ERF model is shown in Fig. 5. First, the input data is processed twice by downsampling blocks. The downsampling blocks are created from the concatenation of the max pooling and 2D convolution with a 3×3 filter and a stride of 2. The concatenation is followed by the activation function. Then, five residual layers and another downsample block are added. The output of the encoder part is processed by eight residual layers which are interwoven with different dilation rate applied to the convolutions. The decoder part of the model consists of two series of convolutional upsampling and two residual layers. The upsampling is performed by transposing convolution with a stride of 2. The output of the model is produced by upsampling convolution with 2×2 filters and strides of 2, where the number of filters is equal to the number of classes. The Activation function used in each nonlinear layer is a rectified linear unit (*ReLU*).

The optimized objective of the model is composed of cross-entropy loss and regularization loss. In the training dataset, the most examples are provided for the class which represents footholds inaccessible for the robot. To handle unbalanced data, the cross-entropy is additionally weighted [27] based on the frequency of occurrence. Namely, the weight of the i^{th} class w_i is defined by

$$w_i = \frac{1}{\log(c + p_i)} \tag{2}$$

where $c = 1.08$ is a constant and p_i is a probability of the occurrence of the i^{th} class in the entire training dataset.

We use the method presented in [28] for training the model with an initial learning rate of 5e-4. Additionally, the exponential decay was applied after each epoch to the learning rate with a factor of 0.98. Because of the nature of the training examples, we can't use any of the known data augmentation methods.

In order to measure the quality of models accuracy, an Intersection over Union (IoU) metrics were calculated. The learning process took place in 500 epochs. The results obtained by two ERF models for front and rear legs are shown in Tab. I. The IoU value is higher than 82%. Note that, this value does not represent directly the quality of the foothold selection module. Although the learnt model misclassifies 22% of the footholds, most of the errors are between neighboring classes, which is not a crucial problem. For example, if the foothold is classified as a class number 13 instead of the class number 14 it is still considered as a very weak foothold.

D. Inference procedure

The inference procedure is presented in Fig. 7. In the first step, we get submap from the global map built by

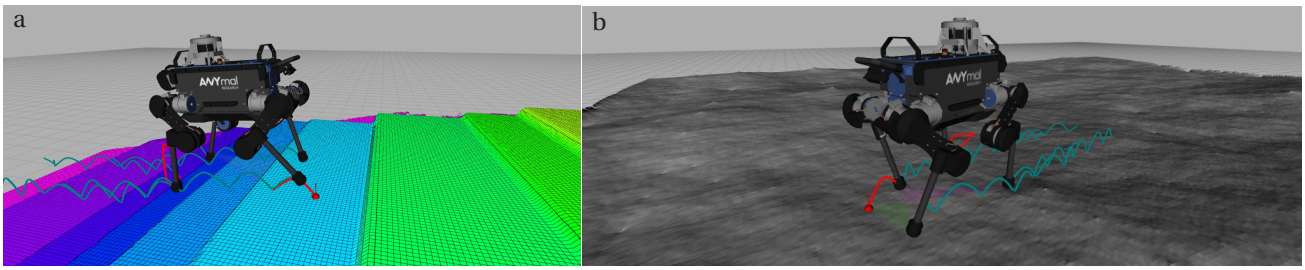

Fig. 6. Environment configuration during experiment with the ANYmal robot on stairs (a) and on rough terrain (b) in the Gazebo simulator. Blue lines represent feet trajectories.

the robot. The obtained map is aligned with the world coordinate system W but our neural network uses the elevation map which is aligned with the robot coordinate system. Thus, we rotate the obtained local map by the current orientation of the robot on the horizontal plane (yaw angle). Some information about cells at the corners is lost during this rotation, therefore, we take a slightly larger map for rotation purpose. Before rotation, the size of the local map is 51×51 cells and after rotation, we crop the map to size 40×40 cells.

In the next step, we convert the obtained elevation map to the image. To this end, we compute the distances between the i-th leg coordinate system L_i and each cell of the map. We use 8-bit grayscale images as an input to the network so the obtained distance values are fitted into range 0–255. We use a constant normalization factor (0.85 m) for each leg of the robot. The obtained image which represents the terrain patch around the consider leg is the input to the neural network model.

The network classifies each pixel on the image, and the cost at each pixel corresponds to the cost of taking that foothold. The example inference results for the input image representing stairs are presented in Fig. 7. The pixels which are located on the edges between steps on the output image are brighter which means that the robot should avoid these footholds. At the same stage of the inference procedure, we compute the distance from the nominal foothold d_n. Then, we compute the final cost c_{final} for each pixel (foothold):

$$c_{\text{final}} = c_f + k \cdot d_n, \tag{3}$$

where c_f is the cost computed by the neural network and k is the constant value which determines the influence of the distance from the nominal foothold on the final cost of the potential foothold. In the experiments presented in the paper the k value is set to 140. We compute the final cost c_{final} for each pixel on the image (c.f. Fig. 7) and we find the minimal value. Then, the pixel with the minimal cost in image coordinates is converted into the 3D point in the world coordinate system. The obtained value is sent to the controller which executes the motion for the given foothold.

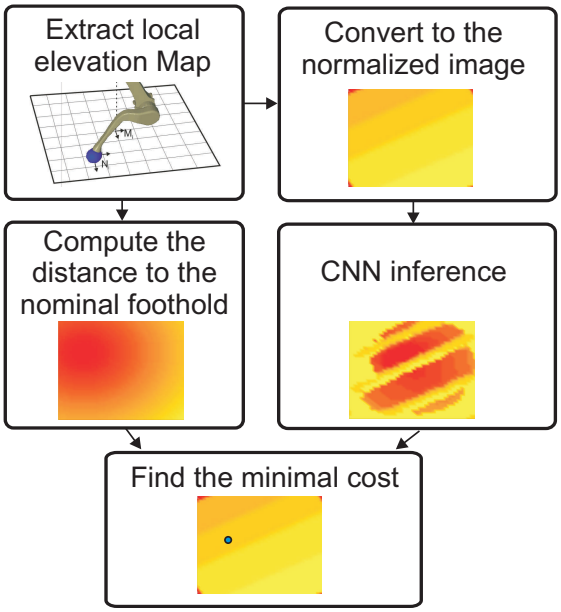

Fig. 7. Foothold selection procedure on the local elevation map

IV. Results

First experiments are performed in the Gazebo simulator. We verified the proposed foothold selection method on the ANYmal robot walking on stairs (Fig. 6a) and on rough terrain (Fig. 6b). The robot uses a simulated Intel RealSense D435 RGB-D sensor to build a map of the environment [7]. We use the controller presented in [29] to plan the foot trajectories above the obstacles, estimate the state of the robot and execute planned trajectories. We only replaced the foothold selection model.

The example inference results are presented in Fig. 8. We provide the terrain patches extracted from the global elevation map, the distance between potential footholds and the nominal foothold, and the output from the CNN. It is clearly visible from the result obtained on the stairs that the robot avoids placing its feet on the edges. These regions are classified by the neural network as risky and rejected by the foothold selection module. Similar behavior can be observed in the results obtained on rough terrain. In this case, the obstacles are more irregular. For both patches obtained on rough terrain, the region in the center of the workspace has a similar cost. In this case, the distance from the nominal foothold plays an important

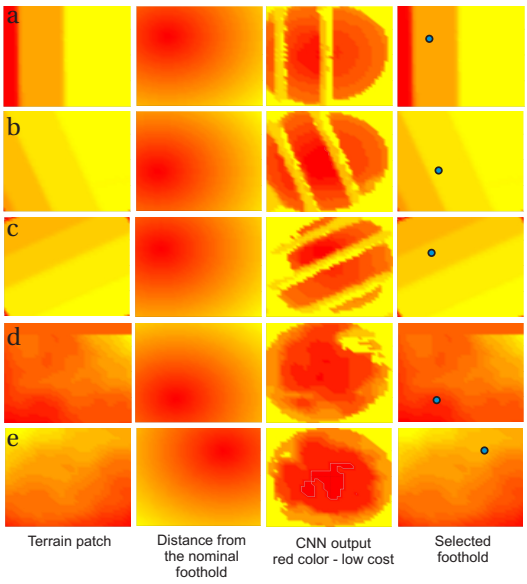

Fig. 8. Example inference results obtained during the experiment on stairs (a,b,c) and on rough terrain (d,e). The stairs and rough terrain are presented in Fig. 6a and Fig. 6b, respectively.

role. The selected foothold is close to the nominal foothold but still on the position with acceptable foothold cost predicted by the CNN.

We also compare our method with the foothold selection method proposed in [29]. The average foothold selection time for [29] is 3.44 ms per 10 samples, while our method produces the output in 151.93 ms (the CNN inference takes 10 ms on the Intel CPU i7-2640M and the rest of the time is consumed by getting data from the global elevation map and preparing input data for the CNN). Although our method may not be the most time efficient one, we outperform the previous work in the choice of footholds. We validated both methods on a stair with 8 steps, each step is 0.18 m high and 0.29 wide. In average, the robot reached 5 steps (in 10 trials) using our method while the robot failed after 2.6 steps using method from [29]. Note that, the failures in both methods are not caused by improper foothold selection but by the stability controller. In all unsuccessful cases, the robot fails because of the lack of stability. Our method performs better because it takes into account the workspace of the leg and selects footholds which are far from the border of the leg's workspace. As a result, the posture of the robot is more stable during walking.

To demonstrate the benefit of learning a model instead of optimizing all constraints online, we compared our method with the method proposed in [9]. We applied the method from [9] to the ANYmal robot and we evaluated the same constraints. Instead of using CNN we use the kinematic model of the robot to check the workspace of the legs and we use Flexible Collision Library [24] to detect collisions. In this case, the foothold evaluation in the 40×40 window takes 2008.81 ms which is more than 10 times slower than the proposed application of CNN.

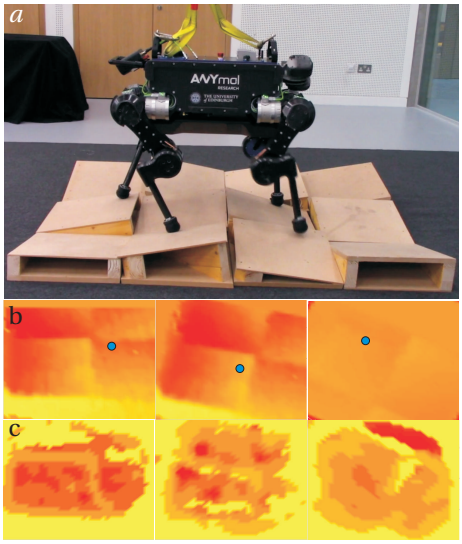

Fig. 9. Experiment with the ANYmal robot on the rough terrain mockup (a): example terrain patches (b) and CNN output (c)

In contrast to previous work based on manually computed features of the terrain [13], the method based on CNN extracts features automatically from data. Our method evaluates 1600 potential footholds and checks constraints in a single inference step. In contrast to previous work, our new approach only needs approximately 10 ms on the CPU to infer from the input elevation map.

Finally, we performed the experiments on the real robot walking over a customized rough terrain consists of 12 blocks with different slopes and orientation. The robot should avoid stepping on the tips or the edges. The example results are presented in Fig. 9. The obtained elevation map (Fig. 9b) is less accurate than the map obtained in the simulation experiments due to noise, but the robot can still identify risky edges and place its feet on the stable positions (see supplementary video).

V. Conclusions and Future Work

In this paper, we propose a novel foothold selection method for legged systems. In contrast to methods known from the literature, the proposed method learns a model that evaluates the terrain patches and checks all constraints in a single step. The time complexity for the inference is significantly reduced. With the proposed method, the robot avoids placing its feet on the edges or steep slopes. The neural network also implicitly takes into account the kinematic range of the leg and detects self-collisions and collisions with the ground. The proposed foothold selection module is integrated with the controller of the robot. In the simulation and experiments with the real robot, we present the properties and the efficiency of the proposed method.

In the future, we plan to use the neural network to optimize simultaneously the foothold position and the posture of the robot.

REFERENCES

[1] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fánkhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, M. Hoepflinger, ANY-mal - a highly mobile and dynamic quadrupedal robot, Proc. IEEE International Conference on Intelligent Robots, pp. 38–44, 2016

[2] D. Hyun, D. Jin, S. Seok, J. Lee, S. Kim, High speed trot running: Implementation of a hierarchical controller using proprioceptive impedance control on the MIT Cheetah, The International Journal of Robotics Research, Vol. 33(11), pp. 1417–1445, 2014

[3] D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A.A. Rizzi, M. Raibert, Autonomous navigation for BigDog, 2010 IEEE International Conference on Robotics and Automation, pp. 4736–4741, 2010

[4] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Autonomous Robots, Vol. 34(3), pp. 189–206, 2013

[5] J. Saarinen, H. Andreasson, T. Stoyanov, A.J. Lilienthal, 3D normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments, International Journal of Robotics Research, Vol. 32(14), pp. 1627–1644, 2013

[6] D. Belter, P. Łabecki, Fánkhauser, R. Siegwart, RGB-D terrain perception and dense mapping for legged robots, International Journal of Applied Mathematics and Computer Science, Vol. 26(1), pp. 81–97, 2016Automation, pp. 5761–5768, 2018

[7] P. Fánkhauser, M. Bloesch, C. Gehring, M. Hutter, R. Siegwart, Robot-Centric Elevation Mapping with Uncertainty Estimates, International Conference on Climbing and Walking Robots, pp. 433–440, 2014

[8] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, Proc. of the International Conference on Neural Information Processing Systems, pp. 1097–1105, 2012

[9] D. Belter, P. Skrzypczyśki, Rough terrain mapping and classification for foothold selection in a walking robot, Journal of Field Robotics, Vol. 28(4), pp. 497-528, 2011

[10] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis, J.L. Wyatt, One shot learning and generation of dexterous grasps for novel objects, International Journal of Robotics Research, pp. 959–976, Vol. 35(8), 2015

[11] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. Aparicio, K. Goldberg, Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics, Robotics Science and Systems, 2017

[12] M. Gualtieri, A. ten Pas, K. Saenko, R. Platt, High precision grasp pose detection in dense clutter, IEEE/RSJ International Conference on Inteligent Robots and Systems, pp. 598–605, 2016

[13] E. Krotkov, R. Simmons. Perception, planning, and control for autonomous walking with the Ambler planetary rover. International Journal of Robotics Research, Vol. 15(2), pp. 155–180, 1996

[14] C.-H. Chen, V. Kumar. Motion planning of walking robots in environments with uncertainty. In Proc. IEEE Int. Conf. on Robotics and Automation, pp. 3277–3282, 2009

[15] M. A. Hoepflinger, M. Hutter, C. Gehring, M. Bloesch, and R. Siegwart, Unsupervised identification and prediction of foothold robustness, IEEE International Conference on Robotics and Automation, pp. 3293–3298, 2013

[16] M. Focchi, R. Orsolino, M. Camurri, V. Barasuol, C. Mastalli, D.G. Caldwell, C. Semini, Heuristic Planning for Rough Terrain Locomotion in Presence of External Disturbances and Variable Perception Quality, arXiv:1805.10238v3, 2018

[17] V. Barasuol, M. Camurri, S. Bazeille, D.G. Caldwell, C. Semini, Reactive Trotting with Foot Placement Corrections through Visual Pattern Classification, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5734–5741, 2015

[18] J. Rebula, P. Neuhaus, B. Bonnlander, M. Johnson, and J. Pratt. A controller for the LittleDog quadruped walking on rough terrain. In Proc. IEEE Int. Conf. on Robotics and Automation, pp. 1467–1473, 2007

[19] M. Kalakrishnan, J. Buchli, P. Pastor, S. Schaal, Learning locomotion over rough terrain using terrain templates. In Proc. IEEE Int. Conf. on Intelligent Robots and Systems, pp. 167–172, 2009

[20] J. Kolter, M. Rodgers, A. Ng, A control architecture for quadruped locomotion over rough terrain. In Proc. IEEE Int. Conf. on Robotics and Automation, pp. 811–818, 2008

[21] O. Villarreal, V. Barasuol, M. Camurri, M. Focchi, L. Franceschi, M. Pontil, D.G. Caldwell, C. Semini, Fast and Continuous Foothold Adaptation for Dynamic Locomotion through Convolutional Neural Networks, arXiv, 2019

[22] A. Roennau, T. Kerscher, M. Ziegenmeyer, J. M. Zöllner, and R. Dillmann. Six-legged walking in rough terrain based on foot point planning. In O. Tosun, M. Tokhi, G. Virk, and H.L. Akin (Eds.), Mobile Robotics: Solutions and Challenges, World Scientific, pp. 591–598, 2009

[23] D. Belter, P. Łabecki, P. Skrzypczyński, Adaptive Motion Planning for Autonomous Rough Terrain Traversal with a Walking Robot, Journal of Field Robotics, Vol. 33(3), pp. 337–370, 2016

[24] J. Pan, S. Chitta, D. Manocha, FCL: A General Purpose Library for Collision and Proximity Queries, IEEE International Conference on Robotics and Automation, pp. 3859–3866, 2012

[25] E. Romera, J.M. Alvarez, L.M. Bergasa, R. Arroy, ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. IEEE Transactions on Intelligent Transportation Systems Vol. 19, pp. 263–272, 2018

[26] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, IEEE Conference on Computer Vision and Pattern Recognition, pp. 837–840, 2017

[27] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation, arXiv:1606.02147, 2016

[28] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, International Conference on Learning Representations, 2014

[29] P. Fankhauser, M. Bjelonic, C.D. Bellicoso, T. Miki, M. Hutter, Robust Rough-Terrain Locomotion with a Quadrupedal Robot, IEEE International Conference on Robotics and and Systems, Daejeon, South Korea, pp. 38–44, 2016