Terrain Segmentation and Roughness Estimation using RGB Data: Path Planning Application on the CENTAURO Robot

Vivekanandan Suryamurthy[1]*, Vignesh Sushrutha Raghavan[2]*, Arturo Laurenzi[2], Nikos G. Tsagarakis[2], and Dimitrios Kanoulas[3]

***Abstract*— Robots operating in real world environments require a high-level perceptual understanding of the chief physical properties of the terrain they are traversing. In unknown environments, roughness is one such important terrain property that could play a key role in devising robot control/planning strategies. In this paper, we present a fast method for predicting pixel-wise labels of terrain (stone, sand, road/sidewalk, wood, grass, metal) and roughness estimation, using a single RGB-based deep neural network. Real world RGB images are used to experimentally validate the presented approach. Furthermore, we demonstrate an application of our proposed method on the centaur-like wheeled-legged robot CENTAURO, by integrating it with a navigation planner that is capable of re-configuring the leg joints to modify the robot footprint polygon for stability purposes or for safe traversal among obstacles.**

I. Introduction

Disaster scenarios which pose severe danger to the rescue efforts by human operators have highlighted the need for more autonomously functioning wheeled/legged robots capable of navigating and operating in such environments. A first step towards such a direction comprises of robust environmental perception. Perceptual methods in structured indoor environments have achieved impressive levels of reliability over the past years [1]. On the other hand, uncertain outdoor varying environments increase the challenge for robot navigation.

Humans are capable of reasoning about navigation and contacts in various environments using their visual cues. In particular, it has been shown that humans usually operate by recognizing materials and their properties (e.g. roughness or slippage) [2]. Material recognition enhanced with implicit quantitative information such as the slipperiness of ice is thus, an important aspect for high-level environment understanding. Similarly, robot perception for navigation in any environment, should include both terrain recognition and characterization, as discussed in [3] for planetary exploration and in [4] for material perception.

Recently, state-of-the-art performance on visual terrain recognition has shown robust results using deep learning, such as Convolutional Neural Networks (CNNs) [5]. However, inferring properties from vision is a complex task, given that true values of a property may be unknown. This statement was explored in [2], claiming that human brain is able to understand variations based on appearance-based features, such as reflections and lighting. Based on this rationale, we simultaneously estimate roughness and roughness properties of the terrain around the robot. We focus on roughness, given its distinct role in path-planning, traversability, and safety analysis during navigation. Such navigation decisions involve speed adjustments [6] or robot path-planning reconfiguration [7]. Notice that even though roughness is the focus of this work, we believe that the same principle holds for other visual terrain properties.

In this work, we firstly focus on the joint problem of terrain recognition and roughness estimation. We then create wheeled-legged robot navigation plans and execute motions based on this information. In particular, we train a single network, which includes a primary CNN for the terrain recognition and a secondary module for estimating the terrain roughness variations, using low-level RGB features (Fig. 1). Even though the technique of sharing layers and parameters for multiple tasks is well known and used to improve the computational complexity [8], in this work we focus on pixel-wise label prediction on one side (terrain type) and continuous values estimation on the other side (roughness calculation). The application of this combined information on robotic planning/navigation, contributes to the novelty of this work. For the network training we use transfer learning and data augmentation to comply with limited training databases. RGB data are used as input to the network. Even though this aligns with the current state-of-the-art in terrain recognition, it was only recently used for the determination of terrain properties, such as depth [9]. The reasoning behind this direction of research, lies to the time efficiency advantage of using CNNs on RGB data and also to the fact that RGB cameras are cheap and light (versus lidar scanners), sunlight independent (versus RGB-D cameras), and provide dense sensing, which make them ideal for any type of robot. Having a method for accurate terrain properties computations (versus stereo cameras), as proposed in this work, makes them ideal for fast computations. Moreover, the RGB-based method allows the enhancement or replacement of any network module (e.g. roughness) with another that could recognize terrain properties which are not strictly connected to depth information, such as slipperiness.

The simultaneous inference of terrain labels and pixel-wise roughness from an input RGB image, is experimentally used on the centaur-like robot CENTAURO [10] for driving

[1]Department of Cognitive Robotics, TU Delft. V.Suryamurthy@student.tudelft.nl

[2]Humanoids and Human-Centered Mechatronics Department, Istituto Italiano di Tecnologia (IIT), Via Morego 30, 16163, Genoa, Italy. {Vignesh.Raghavan, Arturo.Laurenzi, Nikos.Tsagarakis, Dimitrios.Kanoulas}@iit.it

[3]Department of Computer Science, University College London, Gower Street, WC1E 6BT, UK. d.kanoulas@ucl.ac.uk

* equal contribution to this work

leg-reconfiguration wheeled navigation in rough terrains. The network, dataset, and code can be found under: `https://sites.google.com/site/tsrenet`

Next, we review the related work (Sec. II) and describe the methodology (Sec. III). Then, we experimentally validate our algorithm (Sec. IV) and apply the methodology on CENTAURO robot navigation system (Sec. V). Last, we conclude with the future work (Sec. VI).

II. RELATED WORK

Terrain segmentation emerged in the beginning as a process of image representation, either using color [11], texture [12], or features [13], [14] for patch-based classification. The use of hand-based image features for representing an image to perform real-time segmentation has been eliminated with end-to-end learning, an important aspect of deep learning. Thus, deep learning methods such as CNN classifiers belong to the state-of-the-art in image-based segmentation, compared to traditional techniques that use multi-class detectors [15]. Terrain and material perception were studied with the same underlying principle. Bell et al. [16] used CNN and Conditional Random Field (CRF) as a segmentation architecture that labels (i.e. predicts independently) material patches in the image with a sliding window approach. Wang et al. [17] fine-tuned a Fully Convolutional Network (FCN) [18] to weigh similar patch-trained CNN on 4D light-field data. Schwartz et al. [19] improved material recognition by integrating global context. The aforementioned methods were trained on image patches, whereas our segmentation approach is closer to that of the study in [5] where the FCN model was trained on the cityscapes dataset [20]. The model of the local terrain is then fine-tuned using the pre-trained weights of hand-labelled images. Instead, we fine-tune the pre-trained model on the specific terrain dataset (see Sec. IV-A.1).

Roughness prediction was broadly used for terrain traversability in the past [21], [22]. The study in [23] used an image-based approach to detect rock concentration as a roughness measure. Recently, the study [24] used a CNN architecture for image and roughness-based material identification, based on reflectance. The study [25] used SegNet [26] for a fast terrain roughness prediction, discretized into four classes. Contrary to that approach, our regression framework estimates roughness as a continuous value. Similar to our approach, there are methods to predict properties such as friction [27] or slippage [28]. The joint friction-material distribution was experimentally determined on a humanoid robot for different terrains using a SegNet CNN, while slippage was determined through appearance and geometry properties acquired by stereo vision. Terrain recognition based on appearance enables non-linear map learning between slopes and slip. Our RGB-based deep neural network method differs from these approaches by using a unified architecture to determine terrain classes and regress their roughness. Sharing layers [8] in a single network for joint learning tasks were used in the past, for example to perform semantic segmentation and depth prediction [29], 3D reconstruction [30], or intrinsic properties estimation [31]. Compared to these works, our proposed method is an end-to-end solution to perform segmentation and curvature estimation (that, as far as we are concerned, was not proposed before), for which each module is trained with different datasets. We use the predicted values to perform robot planning on our wheeled-legged robot.

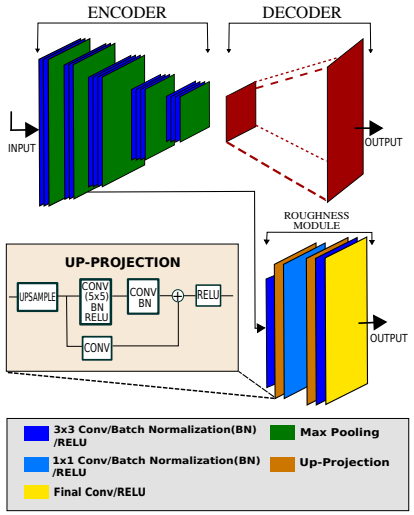

Fig. 1. The proposed architecture: the encoder-decoder system to predict the pixel-wise terrain type labels and the module to regress the roughness.

III. METHODOLOGY

In this section, we present our unified terrain segmentation labeling and pixel-wise roughness estimation method. The proposed architecture is visualized in Fig. 1.

A. Terrain Segmentation

We treat terrain segmentation as a semantic pixel-wise labeling problem, using a deep convolutional encoder-decoder architecture. The encoder abstracts features through a pooling operation and the decoder is upsampling. Real-time performance for driving robot actions require computational and memory efficient predictions. We have implemented our method using the SegNet [26] and the recently introduced ERFNet [32] architectures. Compared to other networks, such as FCN-based ones (see Sec. II), these offer a better trade-off between inference time and performance. Using techniques, such as storing the max-pooling indices for upsampling in SegNet as opposed to encoder feature maps in FCNs, and factorized convolutions in ERFNet, results in better memory efficiency. The inference time is reduced also by merging the batch normalization and dropout layers after training.

To avoid training the networks from scratch, which require several man-hours and big training data (see Sec. IV-A.1), we utilize transfer learning. In particular, we fine-tune the weights of a pre-trained model of our networks on the cityscapes dataset [20]. In this way, we achieve fast and generalized results versus freshly trained models [33]. Cityscapes, being an outdoor-based dataset, is closely related to the task of terrain segmentation and hence, suitable for initialization.

Outdoor environments often suffer from ambiguity, which requires the model to generalize fairly to unforeseeable terrains. To solve this problem in training, we enable data augmentation [34], such as random horizontal/vertical flips and contrast/brightness changes with a 0.25 probability. We do not crop the images, due to sparse label annotations.

Fine-tuning is performed on RGB images with resized dimensions of 256×512. Class imbalance causes the networks to minimize a weighted cross-entropy loss by gradient descent with a constant learning rate of 10^{-3}, the momentum is set to 0.9, and weight decay to 0.0005. Median frequency balancing [35] is used to compute the weights for the loss function. Finally, an experimentally determined confidence measure of 0.4 is used to threshold the model softmax output, labeling predictions to "unlabelled" if it is not reached. Post-processing steps (e.g. CRF) could have further improved the results, but were avoided, keeping the computational efficiency high.

B. Roughness Estimation

In deep learning, feature space specificity increases with the CNN depth. Thus, to estimate the roughness we use the bottom layers (closer to the input image) of the trained network which correspond to primitive appearance features. In particular, given an input image I and the features $F = f_1, f_2, ...f_n$ (where n is the number of features), the network's bottom layers provide the probability distribution $P(F|I)$ from the initial fine-tuning and the roughness estimation module computes the probability distribution $P(R|F, I)$ to estimate roughness R. The mathematical formulation is given by:

$$P(R, F|I) = P(R|F, I)P(F|I).$$

As the bottom layers of SegNet provide sufficient feature abstraction, the introduced roughness module serves as a decoder (decodes features to roughness) only. Up-projection blocks are used [36], to increase the spatial resolution of the downsampled feature maps. Thus, memory-intensive regression of fully connected layers are eliminated and the network converges faster. The roughness module includes: 1) a 3×3 kernel convolutional layer, 2) two up-projection blocks separated by a 1×1 convolution (i.e. feature pooling), and 3) convolutional layers with ReLU activation, which estimate the pixel-wise roughness values. Batch-normalization follows the convolutions during up-projection.

During training, terrain segmentation weights are frozen and the second pooling layer of the model is appended with the new roughness module branch. Notice that, adding the module at a deeper pooling layer did not improve the regression performance. We used the same training setting, only with a lower learning rate of 10^{-7} and an increased batch size of 25. During regression we experimentally tried both the conventional Euclidean (L_2 norm) and the berHu [37] loss. The latter switches between the L_1 and L_2 norms depending on a threshold c:

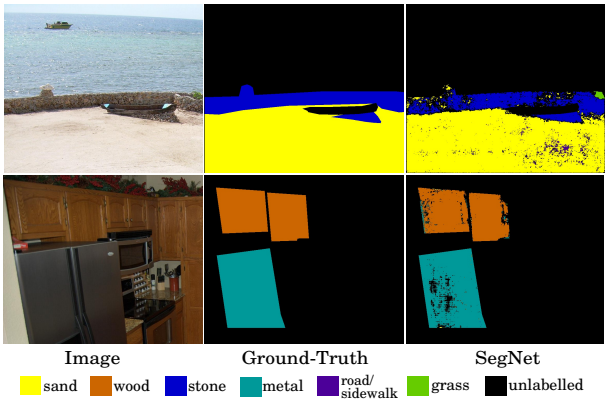

Fig. 2. Terrain segmentation on validation set images: from the ADE20K (top) and from the OpenSurfaces (bottom) datasets.

$$B(x) = \begin{cases} |x| & |x| \leq c \\ \dfrac{x^2 + c^2}{2c} & |x| > c \end{cases}$$

where x is the error between the prediction and the ground-truth, and $c = \frac{1}{5} max_k(|x_k|)$ is computed for every step of gradient descent for an output with k pixels. BerHu outperforms Euclidean loss, due to small errors being given higher weightage.

IV. Experiments

In this section, we present the terrain and roughness datasets (available on our web-page) and the visual results of the introduced method.

A. Datasets

1) Terrain Dataset: Segmentation has been developed on pixel-wise annotated image datasets, such as ADE20K [38], which usually include very sparse sets of terrains. On the other side, datasets focused on materal segmentation, such as OpenSurfaces [39], include only indoor images. The classes of our interest, such as rocks and sand, are distributed among these datasets. Thus, we formed a new terrain dataset by extracting only related images and annotations. In this way, we were able to use available datasets and eliminate the need for hand-labelled annotations.

We generate a dataset with six terrain classes that occur frequently: sand, stone/gravel, wood, metal, road/sidewalk, grass. We tried to include varying images that balance between terrain classes (imbalances generate biases). For the construction of our dataset we first select an image subset from ADE20K, which includes only a sparse and minimum set of wood and metal surfaces. We balance these type of terrains from OpenSurfaces. In total, our dataset includes 1380 images, from which 65% are used for training and the remaining for evaluation. In Sec. V-B, we collect RGB images from RGB cameras and RGB-D sensors mounted on the CENTAURO robot and uses these images for the validation of the terrain segmentation module.

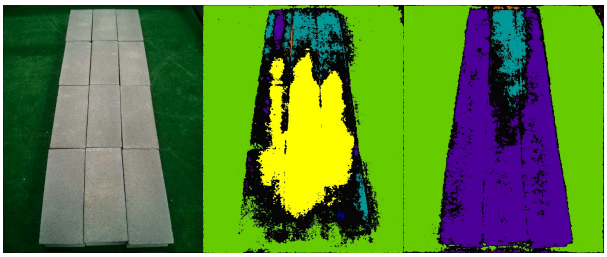

Fig. 3. Data augmentation: an RGB image (left), prediction without (middle), and with (right) brightness and contrast tweaks.

2) Roughness Dataset: In this work, for time efficiency we attempt to estimate roughness (surface irregularities) using RGB images, compared to traditional techniques [22]. For the training, we use the GeoMat [40] dataset which includes terrain image patches with the associated 3D depth data, generated by Structure from Motion (SfM) and interpolation. The depth data are transformed to point clouds [41]. The roughness r_i at each point (x_i, y_i, z_i) is then calculated by fitting a local plane $(ax + by + cz + d = 0)$ in the 3D point cloud data using the least squares method [21]:

$$r_i = \frac{|-d - ax_i - by_i - cz_i|}{\sqrt{a^2 + b^2 + c^2}}.$$

From the dataset we use the patches that have sizes 400×400 and 800×800 pixels, given that in 3D, these dimensions approximate to the size of robot parts such as the wheel/footpad surfaces. We use 1375 images for training and 880 for evaluation.

B. Visual Experimental Results

In this section, we experimentally validate the visual terrain and roughness output of the networks implemented in Caffe [42] (the weights provided for fine-tuning can be found in the caffe model zoo).

1) Terrain Segmentation: The average accuracy and the mean Intersection over Union (IoU) metrics [43] are used for the assessment of the terrain segmentation performance. Being focused on real-time applications, inference time is crucial and thus we compare three networks, i.e., ENet [44], SegNet, and ERFNet. When evaluated on images of 256×512 pixels, compared to the latter two, ENet is found to be the fastest (10.25ms per frame), ERFNet is slightly slower (3.86ms per frame slower than ENet), while SegNet is the slowest over all (8.29ms per frame slower than ENet). Moreover, evaluation on the validation set showed that the class average accuracy of ENet is 54%, SegNet 64%, and ERFNet 65%. Similarly, the mean IoU of ENet is 27%, SegNet 43%, and ERFNet 49%. All network architectures were trained in the same dataset for comparison purposes, while for ENet and ERFNet we merged the batch normalization and dropout.

After the assessment on our dataset, it is clear that the state-of-the-art ERFNet is more accurate than the other two networks and also has the best segmentation performance. Moreover, we noticed that the predictions improve when we freeze the first layer of the convolution in the model, and given the small size of the dataset, we train only for 50 epochs to avoid overfitting. Fig. 2 illustrates terrain segmentation example results on the validation images. Data augmentation aids with biases and ambiguous labelling in the dataset, e.g. the "path" class in the ADE20K dataset could be classified both as sand or road. As illustrated in Fig. 3 random variations in contrast and brightness moderate labeling ambiguity, and thus make the model more general and accurate.

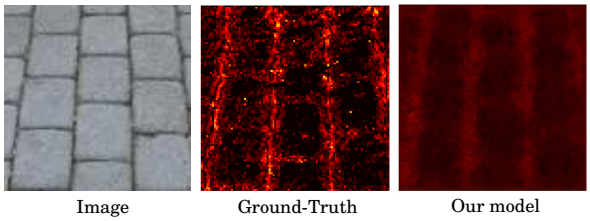

Fig. 4. Roughness estimation on the transformed GeoMat dataset (for visualization purposes the heatmaps are scaled equally).

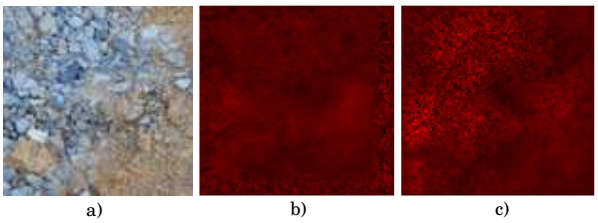

Fig. 5. An RGB image (a), the roughness estimation from the freshly trained model (b) and by using the appearance-based features (c).

2) Terrain Roughness: We also assess the pixel-wise roughness regression, using two metrics: the Root Mean Square Error (RMSE) and Root Mean Square Log Error (RMSLE). Table I includes our model's error using the two loss functions (Euclidean and berHu).

TABLE I
ERROR ASSESSMENT FOR EUCLIDEAN AND BERHU LOSS FUNCTIONS

Our model	RMSE	RMSLE
Euclidean	0.4350	0.1123
berHu	0.4335	0.1119

As can be seen, the two loss functions have similar estimation errors. Furthermore, we observed that berHu produces better results on our validation dataset. As illustrated in Fig. 4, the quality of the roughness estimation is good, despite the variations due to ground-truth depth data noise from the SfM. The flat surfaces (inner area of brick) are estimated to have low or no roughness while the boundaries (between the bricks) have higher roughness. Notice that during regression the estimated roughness magnitude values were downscaled to $0.05 - 1.4$cm range compared to the ground-truth $0-3.7$cm. This is possibly due to biases towards almost flat surfaces that dominate most of the dataset. We do not consider this as a problem since human vision system acts similarly, by capturing relative variations of roughness to drive human actions [2]. During training, it appears that the gradient descent calculates low magnitude values conservatively. These biases are less when the model is trained with appearance-based features compared to freshly trained models (Fig. 5).

V. Robotics Applications

In this section, we present two applications of terrain segmentation and roughness estimation on the centaur-like robot CENTAURO (Fig. 6). Both of them utilize roughness as a risk measure for terrain traversability of non-deformable terrains (stone/road/sidewalk). In particular, in the first set of experiments the robot detects rough surfaces of the upcoming terrain to modify its Center-of-Mass (CoM) position for stability purposes, while in the second set of experiments the robot plans paths around risky surfaces (of high roughness) if possible.

A. The Robotic Platform

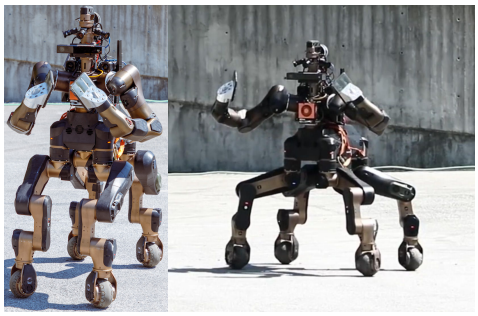

Fig. 6. The centaur-like CENTAURO robot.

CENTAURO [45] is a 42-DoF wheeled-legged robot of 93kg and 1.5m height. In this work, we have used a monocular RGB camera (PointGrey BlackFly) mounted on its head. The lower body of the robot includes four 7-DoF legs with wheels for end-effectors. We control the robot in real-time through a Robot Operating System (ROS) interface named CartesI/O [46] to drive Cartesian references. In particular, we customize a module of CartesI/O to drive hybrid wheeled-legged locomotion actions on the robot, that allow full 6D pose control of its CoM by wheel steering and spinning.

B. Online Segmentation and Roughness Estimation

First, using images from CENTAURO's camera published at 30 FPS, we evaluated the real-time visual terrain segmentation and roughness estimation. We vary the roughness of the surface manually to assess the estimations. In Fig. 7 we illustrate some of the experimental outcomes of terrain and roughness estimation. We notice that even though we have focused on the model generalization, there are some segmentation failures (outliers in the last row) where some parts of large rocks may be confused with road/sidewalk surfaces due to similar appearance. In these images of 256×512 size, the inference time is on average 15.37ms (using the ERFNet architecture), as previously shown in Sec. IV-B. We have tested the time efficiency of the method for cropped 192×384 images in the robot's closest observable area (of width slightly bigger than the distance between the two front wheels), which improves the inference time by roughly 5ms. Moreover, this proves the generalization and flexibility of the network with respect to different image sizes. Lastly, it is noticeable that the camera's motion blur affects the results and we leave the use of cameras with minimum blurring during motion as future work.

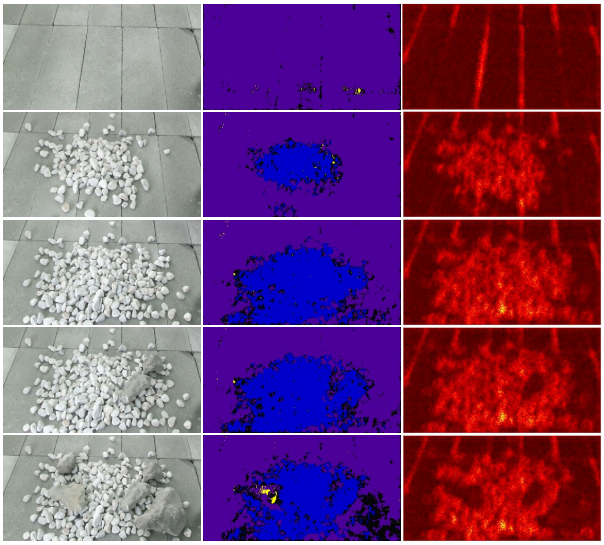

Fig. 7. Terrain segmentation and roughness estimation results (for better visualization, the heatmaps are downscaled).

C. CoM Modification for Balancing

Rough surfaces may be risky for robot navigation, due to the possibility of the robot loosing contact with the environment which may result in balancing instabilities that can cause falls or motor failures. A common technique used from animals/humans to achieve better balance is to increase the support polygon area formed from their feet and lower their CoM height. Similar to these strategies, we allow CENTAURO lower its pelvis height while increasing its support polygon by extending its wheeled feet end-effectors. Experimentally, we have realized the shaky instabilities of the robot rolling while keeping its legs straight over rough surfaces as illustrated in Fig. 8.

When the robot expands its support polygon by bending its legs the joint motors increase their torques, which results to higher energy consumption, unlike the case of straight legs, where the support polygon is much smaller but the produced torques are also minimal. Thus, in the first experiments we allow the robot to move with stretched legs over flat terrain, forming a support polygon of 0.36m^2 with the pelvis height being 0.95m, while when very rough terrain is detected, we let the robot modify its support polygon using the CartesI/O module for better stability, with a 0.63m^2 support polygon and 0.75m pelvis height from the ground. We also allow an intermediate level with pelvis height of 0.85m.

To drive the aforementioned actions, a threshold of the average roughness of the upcoming terrain is required (we experimentally set this to 0.3cm). Even though the roughness magnitude is approximated by our model, similar to humans perception, the relative roughness is enough to drive robot navigation. Notice that the robot requires some rolling time while changing its configuration, and thus we drive the support polygon change well before the front wheels reach the rough terrain.

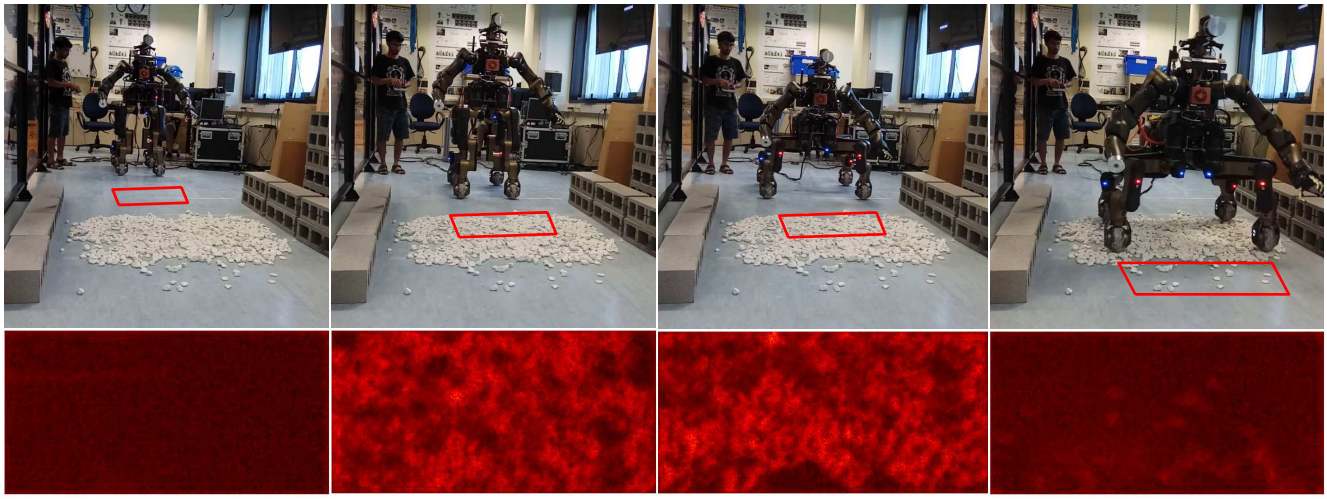

Fig. 8. Rough terrain experiments, with the estimated roughness. a) CENTAURO with stretched legs on a flat surface, b) detection of the upcoming rough terrain from our network, c) CENTAURO while bending its legs, and d) traversing the rough surface.

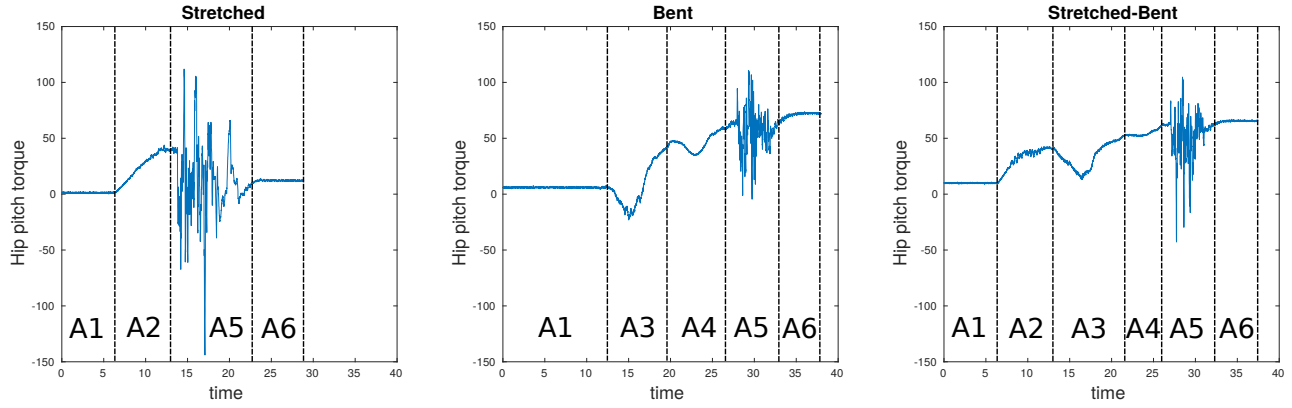

Fig. 9. The hip pitch joints torques for three experimental CENTAURO terrain navigation: 1) traversing the surface with stretched legs, 2) traversing the surface with bent legs, and 3) traversing the surface with stretched legs over the flat terrain and bents them for the detected rough terrain. The splits for each figures illustrate: [**A1**] static stretched legs configuration; [**A2**] acceleration with stretched legs configuration; [**A3**] transition from stretched to bent legs; [**A4**] acceleration with bent legs; [**A5**] traversing the rough surface with bent legs; and [**A6**] deceleration.

We run, three experiments with CoM variation being high (stretched), low (bent), and adaptable (stretched-bent). Fig. 9 illustrates the hip pitch torques of a leg during the experiments. We focused on hip joints since the knee and ankle are aligned with the ground reaction forces and do not contribute much in the torque generation. In the first experiment, the robot with stretched legs produces almost zero torques. However, when vibrating due to rough terrain instabilities, there are high torque fluctuations (torque sign alterations indicate loss of leg contact with the ground). In the second experiment, the extended support polygon of the robot results in bigger energy consumption due to increased torques (see [A6] in Fig. 9), but torque fluctuations over the rough surface are smaller, which results in rare loss of contacts with the ground. As a result, it is clear that when in less rough terrain it is preferable to have the legs of the robot stretched and bend them for the upcoming rough terrains. In this way, maximum stability and minimum energy consumption is possible on the robot.

D. Re-configurable Navigation Planning

A second application that uses the roughness information estimated from our network, involves the re-configurable path planning algorithm for CENTAURO, using the same principle of modifying the robot's footprint polygon and the pelvis height. This allows the robot to expand over certain wide obstacles and narrow its polygon in tight spaces. To plan for re-configurations over rough surfaces in difficult scenarios, we modify the planner introduced in [7]. We use roughness information in addition to the low-frequency ($\sim$0.3Hz) lidar data based heightmap used in the original algorithm in [7], which is obtained from the data of the rotating (2rad/s) Velodyne VLP-16 lidar sensor mounted on the head of the robot. Since they are parallel processes and the roughness estimation runs in real-time, no extra calculations or computing power are needed and the two modules can be interfaced directly. In this section, we briefly explain the three steps of the path planning algorithm and the conducted experiments.

In the first step of the planning, a segmented map is generated using the point cloud from the lidar sensor. The map classifies points in the robot surrounding environments into three categories: 1) short obstacle points (with height from the ground lower than 0.4m); these areas can be easily cleared or negotiated by the robot, 2) high obstacles (with height greater than 0.4m); these areas are non-traversable areas, and 3) completely free space points.

In the second step of the planning, the aforementioned segmented map acts as input to the proposed A*-based re-configurable planner. Firstly, a point in the map is checked for robot polygon collisions with obstacles. In case a collision with a negotiable obstacle is exits, the planner selects from three strategies: 1) avoid the obstacle, 2) expand over the obstacle to encompass it, such that the wheels roll on the sides of the obstacle, or 3) narrow into a polygon that doesn't collide with the obstacle. In case of a high obstacle, the robot avoids or narrows the polygon to fit into the free space. The aforementioned decisions are integrated into A* [47] using the following cost functions:

$$g(x_c, y_c) = g(x_p, y_p) + W_t \times \frac{|\theta_{ent} - \theta_{ext}|}{2\pi} + W_c \times (|\delta w|)$$
$$\theta_{ent} = acos\left(\frac{x_p - x_c}{\sqrt{(x_p - x_c)^2 + (y_p - y_c)^2}}\right) \quad (1)$$

$$h(x_c, y_c) = \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} + W_g \times \frac{|\theta_{goal} - \theta_{ent}|}{2\pi}$$
$$\theta_{goal} = acos\left(\frac{x_g - x_c}{\sqrt{(x_g - x_c)^2 + (y_g - y_c)^2}}\right) \quad (2)$$

where (x_p, y_p) is the parent node (point in 2D segmented map) and (x_c, y_c) is the target node currently being evaluated. As in standard A*, the exact cost moving from the starting point to (x_c, y_c) is given by $g(x_c, y_c)$ (Eq. 1) and the heuristic estimated cost of moving to (x_c, y_c) with respect to the goal node (x_g, y_g) is given by $h(x_c, y_c)$ (Eq. 2). W_t is the weight on the cost of turning, W_c is the weight on the cost of changing configurations, $|\delta w|$ is the change of the width of the robot polygon and W_g is the cost of turning away from the goal.

The third step of the planning involves the estimated roughness from our model. In particular, when our A* planner is unable to find a valid path, the roughness information of the low obstacle points in the segmented map are obtained. If these points that obstruct the robot's path to the goal have small roughness (experimentally set to 0.3cm), then we change the class of these points in the segmented map to free space points. In this way the A* planner can plan paths through them, with an inferred lower velocity when traversing them.

To demonstrate the application of the roughness application for navigation planning, three experiments depicting the usefulness of readily available information, are illustrated in Fig. 10. The left column depicts the case where the robot chose to avoid the rough obstacles as it had enough space, as observed from only the lidar point cloud data. The middle column depicts the case where the segmented map from only the lidar point cloud data shows that the robot is further limited in space and hence has to narrow to safely pass through the line of small rocks. The right column represents the situation where the planner has no possible path to reach the goal and hence, it uses roughness information of the low points and chooses to roll straight over the rough obstacles with low speed as the roughness is within traversable limits. The re-configurable planner on an average takes 20ms to give a valid plan or indicate the nonexistence of a safe valid plan. With the roughness module providing roughness estimates every 15.35ms, it was possible to re-plan almost instantaneously without needing any extra processes, for the third case. This instant availability of roughness is bound to aid in the flexibility and dynamism for future planners.

VI. Conclusions and Future work

In this work, we have utilized end-to-end deep convolutional neural networks to segment terrains and estimate their roughness. For training the models we customized a dataset (provided under our web-page) of several labeled terrain images with their roughness. Finally, we evaluated the visual method on the CENTAURO robot by performing real-time tasks, such as leg reconfiguration for planning and balancing purposes during navigation. We plan to test the roughness model (in an extended dataset) with more state-of-the-art segmentation networks to investigate their accuracy and computational complexity. Moreover, we plan on improving the robotic applications by driving various actions, such as speed regulation, for path planning. Last but not least, we plan in fusing the vision-based terrain and roughness computation with other terrain estimation, such as force/torque-based ones [48], to enable autonomous planning and control during robot navigation.

Acknowledgment

This work is supported by the CENTAURO (no 644839) EU project. The Titan Xp was donated by the NVIDIA Corporation, while we appreciate the help from Prof. Martijn Wisse on the draft and Joseph DeGol on the GeoMat dataset.

References

[1] P. Papadakis, "Terrain Traversability Analysis Methods for Unmanned Ground Vehicles: A Survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1373–1385, 2013.

[2] R. W. Fleming, "Visual Perception of Materials and their Properties," *Vision Research*, vol. 94, pp. 62–75, 2014.

[3] E. Tunstel and A. Howard, "Sensing and Perception Challenges of Planetary Surface Robotics," in *IEEE Sensors*, vol. 2, 2002, pp. 1696–1701.

[4] E. H. Adelson, "On Seeing Stuff: the Perception of Materials by Humans and Machines," in *Human Vision and Electronic Imaging V*, ser. SPIE Proceedings, 2001, pp. 1–12.

[5] F. Schilling, X. Chen, J. Folkesson, and P. Jensfelt, "Geometric and Visual Terrain Classification for Autonomous Mobile Navigation," in *IEEE/RSJ IROS*, 2017, pp. 2678–2684.

[6] M. Castelnovi *et al.*, "Reactive Speed Control System Based on Terrain Roughness Detection," in *IEEE ICRA*, 2005, pp. 891–896.

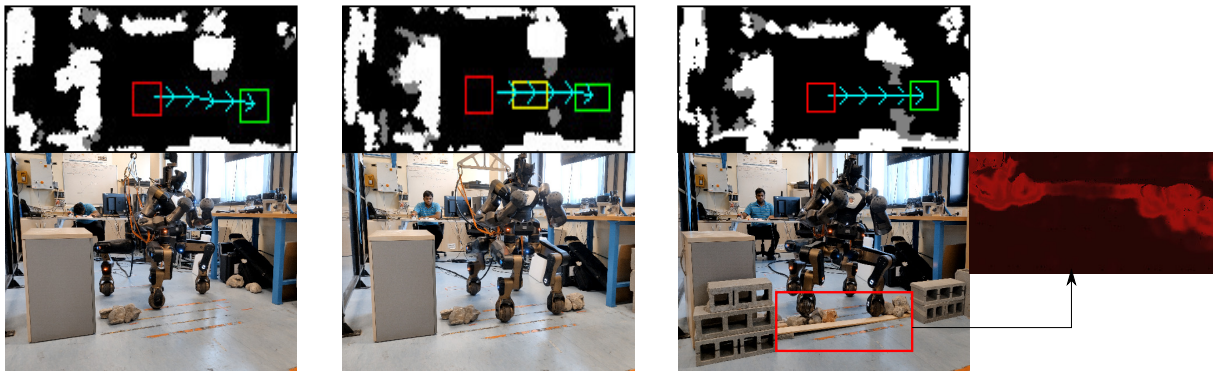

Fig. 10. Three re-configurable planner experiments. The top row images depict the planner output, where yellow rectangles represent polygon change, red rectangles represent the starting point, and green rectangles represent the goal point. The bottom row images represent the key points during navigation for the three experiments. For the third column the red rectangle and the inset image on the right represent the area of evaluation and the result of roughness estimation performed for the third case respectively.

[7] V. S. Raghavan, D. Kanoulas, A. Laurenzi, D. G. Caldwell, and N. G. Tsagarakis, "Variable Configuration Planner for Legged-Rolling Obstacle Negotiation Locomotion: Application on the CENTAURO Robot," in *IEEE IROS*, 2019.

[8] C. Szegedy *et al.*, "Going Deeper with Convolutions," in *IEEE CVPR*, 2015, pp. 1–9.

[9] R. Garg, B. V. Kumar, G. Carneiro, and I. Reid, "Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue," in *ECCV*. Springer, 2016, pp. 740–756.

[10] L. Baccelliere *et al.*, "Development of a Human Size and Strength Compliant Bi-Manual Platform for Realistic Heavy Manipulation Tasks," in *IEEE/RSJ IROS*, 2017, pp. 5594–5601.

[11] I. L. Davis, A. Kelly, A. Stentz, and L. Matthies, "Terrain Typing for Real Robots," in *Intelligent Vehicles Symposium*, 1995, pp. 400–405.

[12] M. Marra, R. Dunlay, and D. Mathis, "Terrain Classification Using Texture For The ALV," *Cambridge SPIE*, vol. 1007, pp. 1–8, 1989.

[13] P. Filitchkin and K. Byl, "Feature-based Terrain Classification for LittleDog," in *IEEE/RSJ IROS*, 2012, pp. 1387–1392.

[14] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Fast Terrain Classification Using Variable-Length Representation for Autonomous Navigation," in *IEEE CVPR*, June 2007, pp. 1–8.

[15] E. Romera, L. M. Bergasa, and R. Arroyo, "Can we Unify Monocular Detectors for Autonomous Driving by using the Pixel-Wise Semantic Segmentation of CNNs?" *CoRR*, vol. abs/1607.00971, 2016.

[16] S. Bell *et al.*, "Material Recognition in the Wild with the Materials in Context Database," in *IEEE CVPR*, 2015, pp. 3479–3487.

[17] T.-C. Wang *et al.*, "A 4D Light-Field Dataset and CNN Architectures for Material Recognition," in *ECCV*, 2016, pp. 121–138.

[18] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *IEEE CVPR*, 2015, pp. 3431–3440.

[19] G. Schwartz and K. Nishino, "Material Recognition from Local Appearance in Global Context," *CoRR*, vol. abs/1611.09394, 2016.

[20] M. Cordts *et al.*, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *IEEE CVPR*, 2016, pp. 3213–3223.

[21] D. B. Gennery, "Traversability Analysis and Path Planning for a Planetary Rover," *Aut. Robots*, vol. 6, no. 2, pp. 131–146, 1999.

[22] S. B. Goldberg, M. W. Maimone, and L. Matthies, "Stereo Vision and Rover Navigation Software for Planetary Exploration," in *IEEE Aerospace Conference*, vol. 5, 2002, pp. 2025–2036.

[23] A. Howard and H. Seraji, "Vision-based Terrain Characterization and Traversability Assessment," *Journal of Robotic Systems*, vol. 18, no. 10, pp. 577–587, 2001.

[24] J. Kim *et al.*, "RGBD Camera Based Material Recognition via Surface Roughness Estimation," in *IEEE WACV*, 2018, pp. 1963–1971.

[25] S. Ram, "Semantic Segmentation for Terrain Roughness Estimation Using Data Autolabeled with a Custom Roughness Metric," Master's thesis, 2018.

[26] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE TPAMI*, vol. 39, no. 12, pp. 2481–2495, 2017.

[27] M. Brandao, Y. M. Shiguematsu, K. Hashimoto, and A. Takanishi, "Material Recognition CNNs and Hierarchical Planning for Biped Robot Locomotion on Slippery Terrain," in *16th IEEE-RAS ICHR*, 2016, pp. 81–88.

[28] A. Anelia, M. Larry, H. Daniel, and P. Pietro, "Learning and Prediction of Slip from Visual Information," *JFR*, vol. 24, no. 3, pp. 205–231.

[29] O. H. Jafari *et al.*, "Analyzing Modular CNN Architectures for Joint Depth Prediction and Semantic Segmentation," in *IEEE ICRA*, 2017, pp. 4620–4627.

[30] L. Ladický *et al.*, "Joint Optimization for Object Class Segmentation and Dense Stereo Reconstruction," *International Journal of Computer Vision*, vol. 100, no. 2, pp. 122–133, Nov 2012.

[31] A. S. Baslamisli *et al.*, "Joint Learning of Intrinsic Images and Semantic Segmentation," in *ECCV*, 2018, pp. 289–305.

[32] E. Romera, J. M. lvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, Jan 2018.

[33] J. Yosinski *et al.*, "How Transferable Are Features in Deep Neural Networks?" in *NIPS*, 2014, pp. 3320–3328.

[34] A. Krizhevsky *et al.*, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012, pp. 1097–1105.

[35] D. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture," in *IEEE ICCV*, 2015, pp. 2650–2658.

[36] I. Laina *et al.*, "Deeper Depth Prediction with Fully Convolutional Residual Networks," in *3DV*, 2016, pp. 239–248.

[37] A. B. Owen, "A Robust Hybrid of Lasso and Ridge Regression," Tech. Rep., 2006.

[38] B. Zhou *et al.*, "Semantic Understanding of Scenes Through the ADE20K Dataset," *arXiv preprint arXiv:1608.05442*, 2016.

[39] S. Bell *et al.*, "OpenSurfaces: A Richly Annotated Catalog of Surface Appearance," *ACM SIGGRAPH*, vol. 32, no. 4, 2013.

[40] J. DeGol, M. Golparvar-Fard, and D. Hoiem, "Geometry-Informed Material Recognition," in *IEEE CVPR*, 2016, pp. 1554–1562.

[41] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE ICRA*, 2011.

[42] Y. Jia *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[43] M. Everingham *et al.*, "The Pascal Visual Object Classes (VOC) Challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[44] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation," *CoRR*, vol. abs/1606.02147, 2016.

[45] E. Rolley-Parnell, D. Kanoulas, A. Laurenzi, B. Delhaisse, L. Rozo, D. G. Caldwell, and N. G. Tsagarakis, "Bi-Manual Articulated Robot Teleoperation using an External RGB-D Range Sensor," in *15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018, pp. 298–304.

[46] A. Laurenzi, E. M. Hoffman, L. Muratore, and N. G. Tsagarakis, "CartesI/O: A ROS Based Real-Time Capable Cartesian Control Framework," in *IEEE ICRA*, 2019.

[47] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[48] K. Walas, D. Kanoulas, and P. Kryczka, "Terrain Classification and Locomotion Parameters Adaptation for Humanoid Robots using Force/Torque Sensing," in *16th IEEE-RAS ICHR*, 2016, pp. 133–140.