Hindawi Publishing Corporation
Journal of Sensors
Volume 2015, Article ID 425732, 14 pages
http://dx.doi.org/10.1155/2015/425732

Research Article

Locomotion Strategy Selection for a Hybrid Mobile Robot Using Time of Flight Depth Sensor

Artur Saudabayev, Farabi Kungozhin, Damir Nurseitov, and Huseyin Atakan Varol

Department of Robotics, School of Science and Technology, Nazarbayev University, Astana 010000, Kazakhstan

Correspondence should be addressed to Huseyin Atakan Varol; ahvarol@nu.edu.kz

Received 29 November 2014; Accepted 22 March 2015

Academic Editor: Andreas Schütze

The performance of a mobile robot can be improved by utilizing different locomotion modes in various terrain conditions. This creates the necessity of having a supervisory controller capable of recognizing different terrain types and changing the locomotion mode of the robot accordingly. This work focuses on the locomotion strategy selection problem for a hybrid legged wheeled mobile robot. Supervisory control of the robot is accomplished by the terrain recognizer, which classifies depth images obtained from a commercial time of flight depth sensor and selects different locomotion mode subcontrollers based on the recognized terrain type. For the terrain recognizer, a database is generated consisting of five terrain classes (Uneven, Level Ground, Stair Up, Stair Down, and Nontraversable). Depth images are enhanced using confidence map based filtering. The accuracy of the terrain classification using Support Vector Machine classifier for the testing database in five-class terrain recognition problem is 97%. Real-world experiments assess the locomotion abilities of the quadruped and the capability of the terrain recognizer in real-time settings. The results of these experiments show depth images processed in real time using machine learning algorithms can be used for the supervisory control of hybrid robots with legged and wheeled locomotion capabilities.

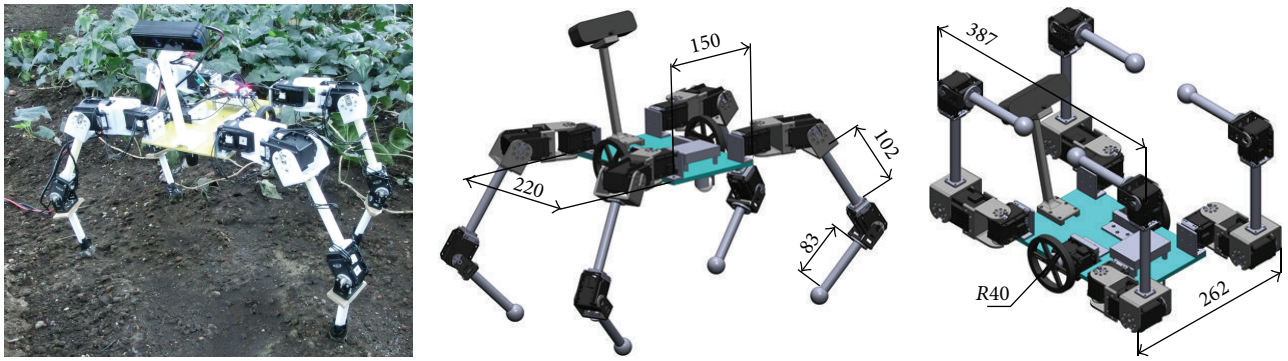
1. Introduction

Locomotion is a fundamental problem of mobile robotics. Legged, wheeled, and articulated bodies are three primary types of locomotion. Working environment, stability, system complexity, and cost are some of the primary factors to consider in choosing a task-specific locomotion configuration for a mobile robot. In such systems wheels are employed in first place thanks to their higher indices of stability, efficiency, and increased payload capacity. However, ubiquitous implementation of this configuration is prevented by the substantial limitations of these robots on unstructured environments. Despite the significant effort to tackle the problem [1–5], wheels are still far from being a panacea for locomotion in all types of environment.

Inspired from the nature, legged locomotion offers robust mechanisms to overcome the difficulties presented by the rough terrain [6]. Capable in navigating on even terrain, legs could have been claimed a universal solution for mobile robot locomotion. However, there are disadvantages of this configuration as well. Static and dynamic stability is one of the major challenges for legged systems. To solve these, engineers tend to create complex solutions with increased costs [6].

Major drawbacks of legged and wheeled locomotion remain unresolved. Hybrid robots emerged to find the best combinations of these two configurations. Generally, hybrid mobile robots can be divided into two types [7]. First type is articulated-wheeled robots with wheels mounted at the end of the legs. Wheels can be active or passive to provide more options for terrain-specific mobility [8–10]. Second type of hybrid robots [7, 11] is designed with legs and wheels separated, but it is always acting synergistically during locomotion.

One of the primary requirements for a field robot is the capability to operate in different terrain conditions. Employment of an appropriate locomotion strategy for a certain terrain condition necessitates the recognition of the environment and execution of a corresponding high level control action. Terrain classification has been approached from various perspectives using different sensors such as RGB cameras and range imaging devices [12–15]. Several examples of using various sensors for mobile robot navigation and



(a) (b) (c)

Figure 1: Hybrid quadruped test platform during legged operation (a), and the solid models with dimensions in legged (b) and wheeled (c) configurations. Dimensions are given in mm.

traversability analysis can be studied in [16–19]. High classification accuracy and time efficiency is reported on seven-class recognition problem [20], where researchers build class representations of nonuniform complexity based on the fact that some classifications can be successfully done with cheap classifiers while others might require much complex ones and incur a lot of penalty for errors. Accuracy of up to 90% was observed with feature-based color images classification on eight-class recognition problem; authors used terrain classification to select an energy efficient gait for a hexapod robot [21]. To continue, an approach to analyze environment traversability by processing depth images acquired from a mobile robot was presented in [22]. For this, authors introduce the Unevenness Point Descriptor (UPD), which encompasses information on both inclination and roughness of surface.

In this work we present a novel approach for locomotion strategy selection of a hybrid mobile robot based on depth images acquired from an on-board time of flight depth sensor, part of a low-cost lightweight compact RGB-Depth camera. The framework requires a supervisory controller which contains a terrain recognizer as the main element used for decision making. Specifically, a hybrid quadruped capable of legged, wheeled, and synergistic legged-wheeled locomotion is designed and built as a test bed (see Figure 1). With respect to the locomotion, both configurations are separated from each other and do not operate collaboratively, except for Stair Ascent and descent. Rather than looking for the most efficient combination of legged and wheeled configurations, our system applies locomotion switching for utilizing the configuration associated with a particular terrain type. To the best of the authors' knowledge, this work is the first to utilize depth sensor for creating a depth image based terrain classification framework enabling mobile robot locomotion strategy selection. Another contribution is the analysis of the effects of confidence based filtering to depth image classification for real-time robot locomotion problem.

2. Time of Flight Range Imaging

Time of flight (ToF) is a range imaging technology which dates back to early 2000s. Distinct from other range imaging techniques, ToF allows the capture of the entire target scene at once using only one device. Standard configuration of a ToF device allows acquisition of range data and is comprised of modulated infrared light source, a sensor (e.g., CCD/CMOS), and an optical system. Time of flight sensors employ the principle of calculating the time of flight of an infrared light emitted from a source to the destination (Figure 2). Specifically, the method calculates the phase shift of the signal and, knowing the speed of light, obtains the distance to the point in the scene from which the light was reflected. By measuring the phase shift of the signal at each pixel of the sensor and knowing the modulation frequency, it is possible to calculate the time delay using (1) and distance using (2):

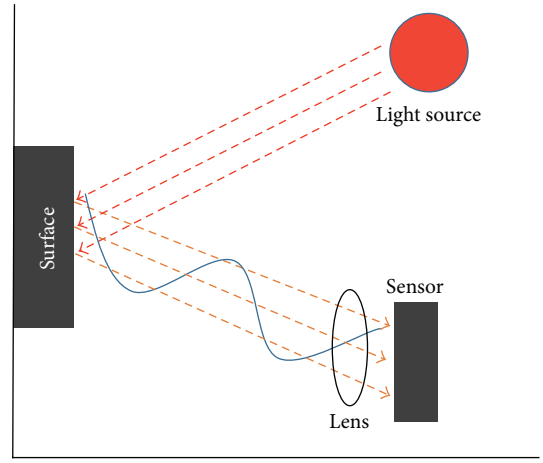


Figure 2: Time of flight sensor operation principle.

$$\Delta t = \frac{\varphi}{2\pi f_m}, \tag{1}$$

$$d = c\Delta t, \tag{2}$$

where φ is the phase shift, f_m is the modulated frequency, d is the distance, and c is the speed of light. Given these and

(a) (b) (c) (d) (e) (f)

Figure 3: Commercial ToF sensors: (a) Fotonic E70. (b) MESA SR4000. (c) Brainvision BV-TOF. (d) Argos 3D p100. (e) Microsoft Kinect. (f) SoftKinetic DS325.

the fact that we measure half the distance the light travels (Figure 2), the depth is calculated as follows:

$$d = \frac{c\varphi}{4\pi f_m}. \quad (3)$$

Additionally, the system measures the amplitude of the incident infrared light at each pixel and provides the corresponding map, which is often called the "confidence" map. The values of the confidence map can be used to describe the reliability of the measurements at each pixel (the lower the amount, the less reliable the measurement).

A relatively young technology, ToF, outperforms common range imaging systems in certain aspects. A comprehensive survey of time of flight cameras and comparative analysis of the technology with alternative methods is given in [23].

(i) ToF cameras require only a single image for depth map acquisition; they do not impose additional processing for solving correspondence problem and are robust to illumination change and partial scene occlusion, compared to passive triangulation methods such as stereo vision.

(ii) Active triangulation light techniques (e.g., structured light) on the other hand also eliminate the problem of extensive computation and use a single camera. However, there is a high power demand from the light source and the requirement of a controlled light environment. The latter problem is also valid for ToF devices (in terms of infrared noise), however in a much lesser degree.

(iii) Finally, laser based systems (e.g., LIDAR), a widely used range imaging technique, outperform ToF cameras in terms of measurement range, accuracy, and reliability and have been successfully used for years in numerous application areas including mobile robotics. Nonetheless, laser based systems look less attractive when compared from weight, size, and power consumption point of view. Most importantly, the nature of laser based systems assumes cross-sectional scans performed in sequence, which limits the capability of the devices in real time and within dynamic scenes. In this relation ToF devices offer advantageous sampling rates of 30–60 Hz; they provide depth data as well as confidence information which can be used for pixel-wise depth measurement evaluation.

Nowadays, ToF devices are broadly utilized in human-computer interaction, computer games and graphics, robotics, activity recognition, and so forth [24–28]. In [29], Alenyà et al. review applications of ToF sensors for vision systems in robotics pointing out the distinctive features ToF sensors for each of the discussed works. General characteristics of Fotonic E70 [30], MESA SR4000 [31], Brainvision BV-TOF [32], Argos 3D p100 [33], Microsoft Kinect [34], and SoftKinetic DS325 [35], commercial off-the-shelf depth sensors, are given in Table 1 with their illustrations in Figure 3. These characteristics need to be evaluated to choose the most appropriate ToF for specific robotic applications.

3. Hybrid Quadruped with ToF Sensor

Hybrid quadruped test bed capable of implementing two locomotion configurations separately is comprised of a main platform, four legs, and four wheels (see Figure 1). The length, width, and height of the textolite main platform are 220 mm, 150 mm, and 3 mm, respectively. This part accommodates the embedded system of the robot comprised of CM-700 Dynamixel controller, RGB-Depth camera (DepthSense DS325), and Zigbee receiver-transmitter (RX-TX) device (Zig 110A). Total weight of the robot with all of the listed components is 2.65 kg.

Every leg of the robot consists of shoulder, elbow, and carpal joints. Legged locomotion of the hybrid quadruped is achieved using eight Dynamixel MX-28 [36] (elbow and carpal joints) and four Dynamixel MX-106 [36] servo motors (shoulder joints). Shoulder joint of each of the four legs is attached to the L-bracket fixed to the main platform. Total length of the leg is 440 mm and its structure is made of caprolon.

Wheels of the quadruped robot are placed in a rhomboid form, similar to one of the configurations proposed in [37]. Castor wheels were placed at the front and rear, while two MX-28 actuated wheels were in-lined in the middle of the platform form differential drive mechanism. For wheeled locomotion, the robot lifts the legs above the main platform as shown in Figure 1. On the other hand, for the legged mode the robot "stands up" leaving the wheels in the air without ground contact. Stair Ascent and Descent are implemented by combinatory motions.

The hardware-software interface block diagram of the hybrid quadruped is shown in Figure 4. The computational elements of the robot consist of the on-board CM-700 controller, and the personal computer (PC) running Windows 7 operating system. PC implements high level system control, whereas CM-700 carries both middle and low level logic.

Table 1: Time of flight depth sensor major specifications.

Sensor model	Manufacturer	Range	FOV ($H \times V$)	Resolution	Dimensions ($H \times W \times L$) mm	Weight (g)	Power (W)	Interface
Fotonic E70	Fotonic	0.1–10 m	70 × 53	160 × 120	80 × 80 × 86	800	16	Ethernet
MESA SR4000	MESA Imaging	Up to 5 or 10 m (optional)	44 × 35	176 × 144	65 × 65 × 76	470	24	USB 2.0 or Ethernet
BV-TOF	Brainvision Inc.	Up to 2 m	60 × 60	128 × 120	50 × 50 × 60	180	12	USB 2.0
Argos 3D p100	BLUETECHNIX Products	0.1–3 m	90 × 90	160 × 120	27 × 75 × 57	140	15	USB 2.0
Kinect	Microsoft	0.8–4 m 0.5–4 m (near mode)	57 × 43	640 × 480	73 × 280 × 73	1360	12	USB 2.0
SoftKinetic DS325	SoftKinetic	0.15–1 m (nominal)	74 × 58	320 × 240	30 × 105 × 23	120	<2.5	USB 2.0

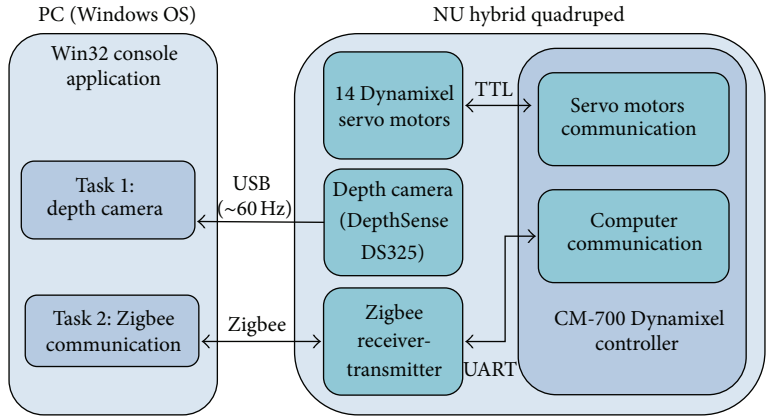


Figure 4: NU hybrid quadruped hardware-software block diagram.

Communication between processing units is accomplished with Zigbee RX-TX devices. Provided with control decisions from PC, CM-700 is responsible for maintaining middle level locomotion controllers and sending corresponding commands to actuators. The system is powered by lead-acid battery (Shimastu NP9-12, 12V9Ah) connected to CM-700 and carried by human operator.

DS325 from DepthSense is selected as the perception sensor for the hybrid quadruped due to its low weight, compact size, low power consumption, and relatively high depth image acquisition rate. DS325 is connected to the computer via universal serial bus (USB). The sensor is fixed to the front of the main platform making an angle of 60 degrees with the ground normal.

4. Control System Design

4.1. Control System Architecture. The three-level control system of the hybrid quadruped is shown in Figure 5. High level controller consists of the user direction reference, m_{ref}, terrain recognizer, and the associated state chart, shown in Figure 6, which depicts the transitions between locomotion states. These states, Level Ground, Nontraversable, Stair Descent, Stair Ascent, and Uneven Terrain, represent corresponding terrain type locomotion controller and comprise middle level controllers of the system. Finally, the low level controller consists of the closed-loop position and velocity controllers implemented internally at the Dynamixel actuators.

Even though our work is applicable to autonomous robots, the present implementation assumes that a user is providing the high level motion references to the hybrid quadruped (i.e., forward, right/left). Additionally, the control of the robot is delegated to the user, when the terrain recognizer identifies a Nontraversable terrain. Specifically, at the instant when Nontraversable terrain is detected, the robot invokes the Stop subcontroller (Nontraversable state) and then switches to the full manual control. The operator directs the hybrid quadruped away from the Nontraversable region and returns the control to the system. Provided with directions and while navigating traversable environments, the robot independently executes its control algorithms to perform locomotion mode selection based on terrain classification results.

4.2. Wheeled and Legged Locomotion. As discussed in Section 1, wheeled and legged locomotion configurations are advantageous for navigation in structured and unstructured environments, respectively. Therefore, as shown in Figure 6, locomotion controller of Level Ground is contained within wheeled mode section, whereas Uneven Terrain is part of

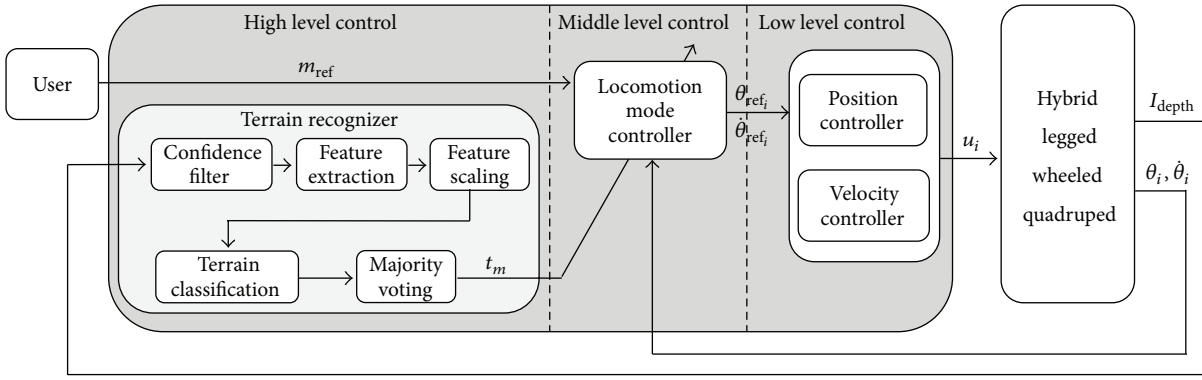


Figure 5: Hybrid quadruped robot control system consisting of three hierarchical levels.

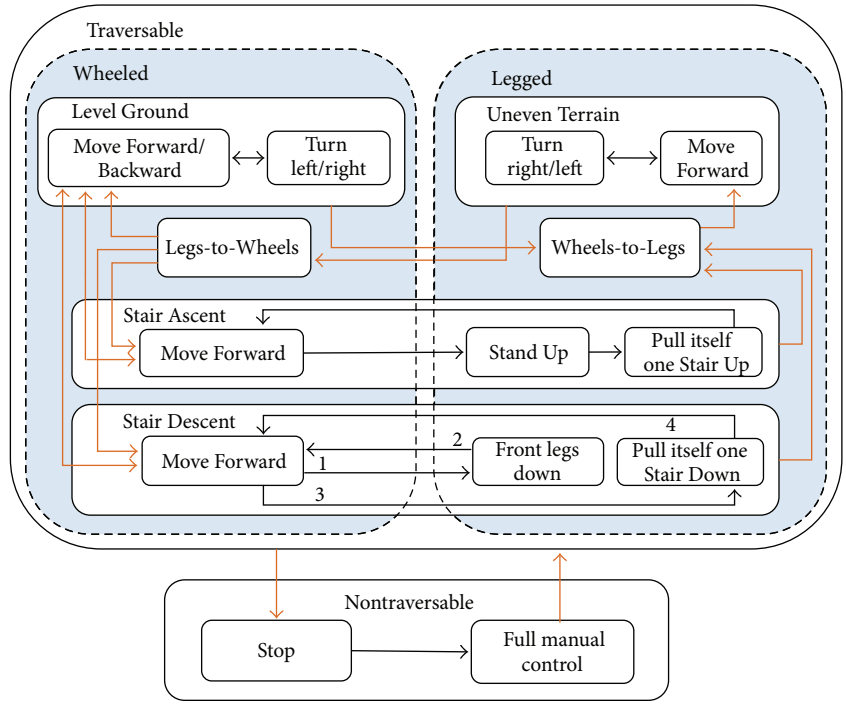


Figure 6: State chart of the middle level locomotion mode controllers.

the legged mode. Internal subcontrollers of the states as well as arrow lines connecting them describe particular set of motions implementation. For example, Move Forward/Backward and turn left/right are the subcontrollers of Level Ground state which generates trajectories for forward and backward movements, as well as turning clockwise and counter clockwise using differential drive. Similarly, corresponding pair of subcontrollers maintains robot's gait on unstructured grounds. However, as it can be seen from *Uneven Terrain* state in Figure 6, legged locomotion is lacking backward movement.

Change of the environment requires a robust locomotion configuration switching procedure. For this, *Legs-to-Wheels* and *Wheels-to-Legs,* two transitional controllers, are programmed to connect states and enable seamless navigation in mixed terrain environments. For instance, the transition from *Level Ground* into *Uneven Terrain* state goes through *Wheels-to-Legs* controller as shown in Figure 6.

4.3. Mixed Locomotion: Stair Ascent and Descent. Hybrid quadruped utilizes a collaborative locomotion which employs both wheels and legs for Stair Ascent and Descent. When terrain recognizer classifies the environment as *Stair Down* or *Stair Up*, the control system starts analyzing subsequent depth images in order to detect declivity and acclivity, respectively. Specifically, for each newly acquired depth image during either of the modes, the depth values of each row for specific range of columns (shaded region shown in Figure 7) are summed. If the difference between two consecutive rows exceeds a threshold, then a step is detected. The size and location of the depth image analyzed for step detection, as well as the numerical difference in depth between rows required for decision making, were derived empirically by analyzing number of factors, such as step dimensions, mobile robot speed, and terrain recognition sampling time.

Both of the stair controllers start in wheeled mode, where the robot drives forward until stair detection occurs. For

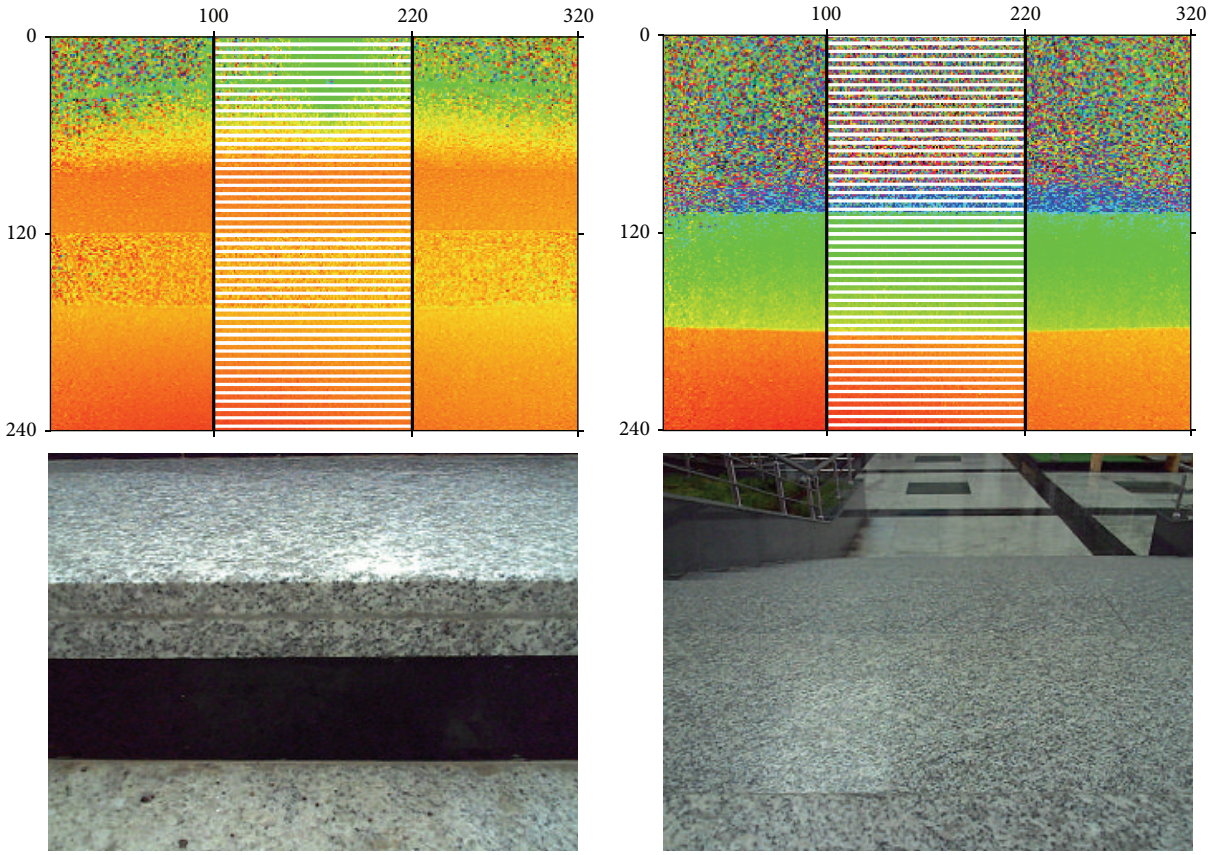


Figure 7: Portions of the depth images analyzed for step detection in Stair Ascent (a) and Stair Descent (b) controllers with corresponding RGB images shown on the bottom row.

ascent, the robot implements the *Stand Up* subcontroller which is equivalent to *Wheels-to-Legs* transition. On the legs, the robot pulls itself onto the stair; it then changes back to wheeled configuration by lifting the legs up and invokes *Move Forward* subcontroller. Thus, in the cyclic manner the robot climbs one stair after another upon detection of one, until the supervisory controller classifies different environment and commands to change locomotion mode.

Likewise, the cyclic manner of subcontroller invocation and transition between locomotion modes is utilized for Stair Descent. After moving forward on wheels (*Move Forward*), the robot puts the pair of frontal legs down one stair; subsequent wheel motion shifts the robot to the edge until active middle wheels hang in the air and the hybrid quadruped stops resting the backside of the platform on previous stair. The robot then pulls itself down onto the next stair and lands onto wheels lifting legs up and searching for a new step until different terrain is recognized. Notice that the hybrid platform motion is hard-programmed for proof of concept for the depth image based supervisory controller implementation. Thus, the platform has a limitation regarding the locomotion versatility and robustness. Specifically, the subcontrollers are programmed with the knowledge of the target environment and under ideal assumptions. For example, we assume that a robot can approach a step only facing it in a straight direction with little deviation from this direction allowed. The operations of both Stair Ascent and Stair Descent subcontrollers are illustrated in Figure 8.

5. Supervisory Control: Terrain Recognizer

The main part of the supervisory controller is the terrain recognizer. Terrain recognizer classifies acquired depth images to five terrain types (classes). Depth image processing, feature extraction, and classification are the main aspects which determine the performance of the terrain recognizer. The block diagram of the terrain recognizer is shown in Figure 5. Terrain recognition starts with depth image acquisition, proceeds with filtering, feature extraction and scaling, and classification, and finishes with majority voting filter. This section describes each of these steps in detail.

5.1. Database Generation. Nazarbayev University atrium (large indoor area containing various ground types) was selected for the database generation, terrain recognizer implementation, and subsequent proof of concept experiments. The database contains five ground types as follows: *Level Ground*, *Nontraversable*, *Stair Down*, *Stair Up,* and

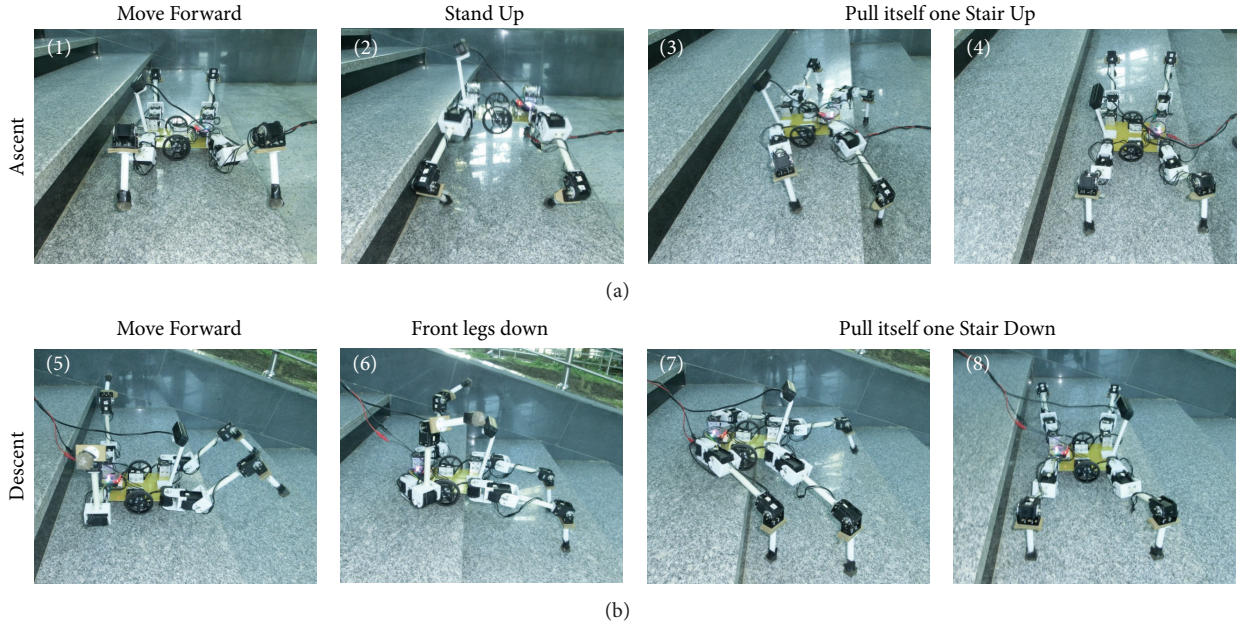


Figure 8: Snapshots of the NU hybrid quadruped in different subcontroller modes ascending (a) and descending (b) stairs.

Uneven. Three datasets, each comprised of 9000 depth images, are used for classifier training, testing, and voting filter implementation, respectively. Depth images of 320×240 pixel resolution were acquired at 30 Hz sampling rate using DepthSense DS325 camera. During database generation the robot was always using a locomotion type corresponding to the present terrain, that is, legged locomotion for uneven ground, wheeled mode for Level Ground, and mixed locomotion for Stair Ascent and Descent. Three datasets were captured in different locations of respective terrains to avoid including same regions multiple times in the database. Ten scans of 180 images (around 6 second segments per scan) are captured per each class for every dataset. Database generation took approximately three hours. Depth and corresponding RGB samples for all terrain types are shown in Figure 9.

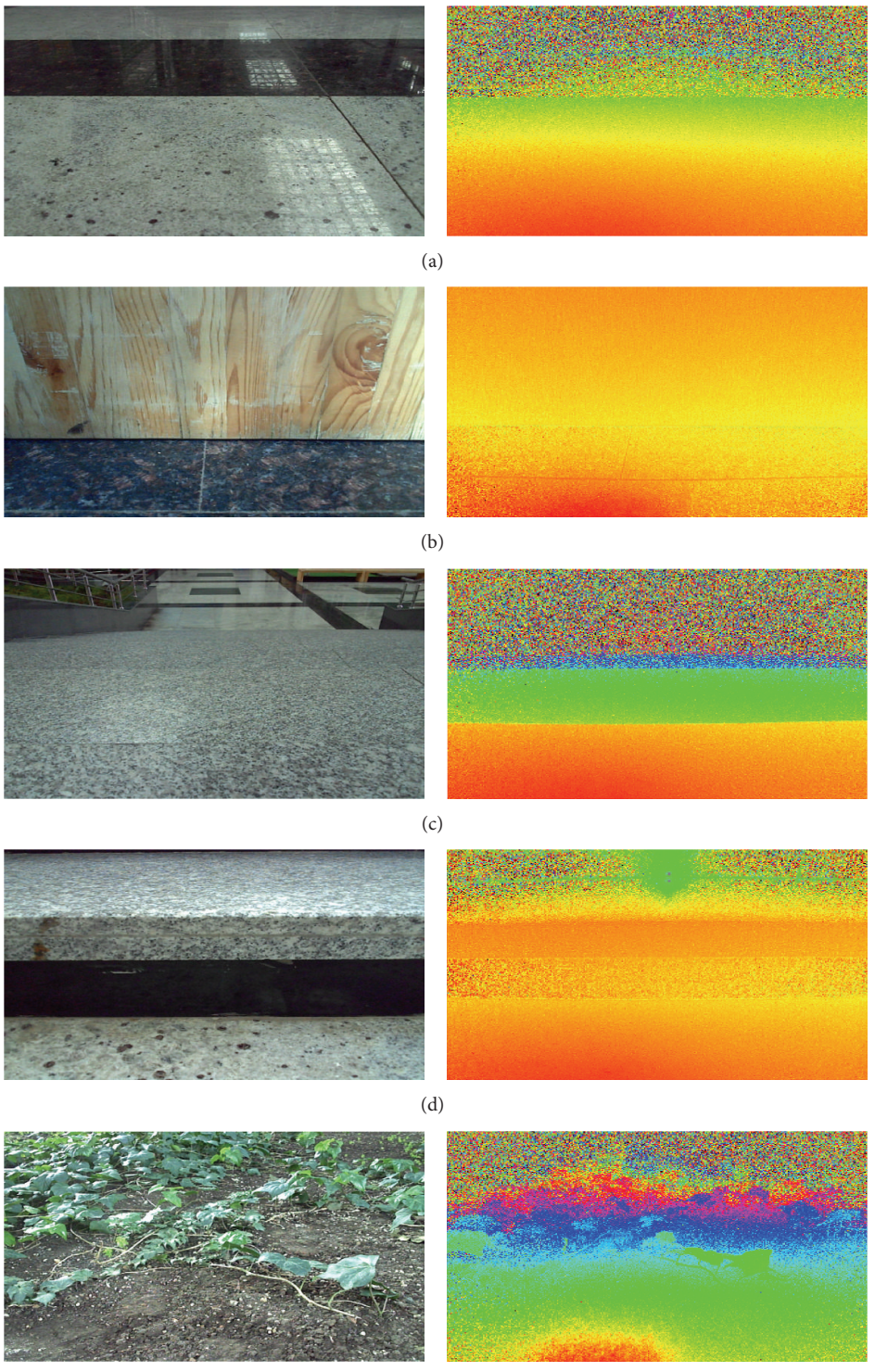
5.2. Depth Image Filtering. The quality of the depth images can be improved using filtering. Consumer depth cameras, to which DepthSense DS325 belongs, have a limited operational range. Frequently, objects and surfaces in the scene are out of the camera's working range which lowers the data reliability and consistency. Moreover, the infrared reflectivity and light scattering are factors, which introduce noise to the depth image acquisition process. Many depth sensors provide confidence maps along with depth data. These maps contain the modulated intensity of infrared light reflected back onto the sensor. In other words, when associated with depth images, confidence values can be employed as a measure of reliability for the acquired range pixels.

Related work in the field of depth image filtering shows variety of approaches to the problem. For example, authors in [38] discard depth data by setting an amplitude threshold; that is, range information for which the corresponding confidence fails to exceed the threshold value is not taken into account. Other researchers state that this approach might result in the loss of important data and instead introduce more complex filtering algorithms [39]. A comprehensive work on denoising of continuous-wave depth images acquired using time of flight depth sensors [40] studied the effect of applying several filtering techniques. Namely, specific adaptive and nonadaptive variations of normalized convolutions and median filtering using confidence values were evaluated on real-world depth data acquired under various conditions. Later, Kim et al. reported on a novel mixed and noisy pixel filtering algorithm utilizing both depth and color data from RGB-D cameras [41]. To continue, a self-localization system for a mobile robot which utilizes depth data enhanced with PCA-based signal reconstruction functionality is presented in [42].

In this work, we used the normalized confidence map values as weights for pixels during the filtering process to improve the depth image denoising. Our choice for this rather simple approach is based on the tight computational requirements of real-time operation. The pixel (i, j) of the filtered depth image D_{filt} is computed using the original depth image D and the corresponding confidence map C using

$$D_{\text{filt}}(i,j) = \frac{\sum_{k_i=-1}^{1}\sum_{k_j=-1}^{1} D\left(i+k_i, j+k_j\right)\cdot C\left(i+k_i, j+k_j\right)}{\sum_{k_i=-1}^{1}\sum_{k_j=-1}^{1} C\left(i+k_i, j+k_j\right)}. \quad (4)$$

Computation of (4) requires the availability of confidence map C corresponding to each depth image present in the database. Thus, during the database generation confidence maps were also collected for all the samples in the datasets. The effects of the filter application are presented and discussed in Section 6.



(a)

(b)

(c)

(d)

(e)

Figure 9: Intensity and depth image samples for different terrains: (a) Level Ground, (b) Nontraversable, (c) Stair Down, (d) Stair Up, and (e) Uneven Terrain. (Note there is the horizontal shift of 25 mm between RGB and depth sensors on DepthSense DS325 camera, so the image pairs do not fully overlap.)

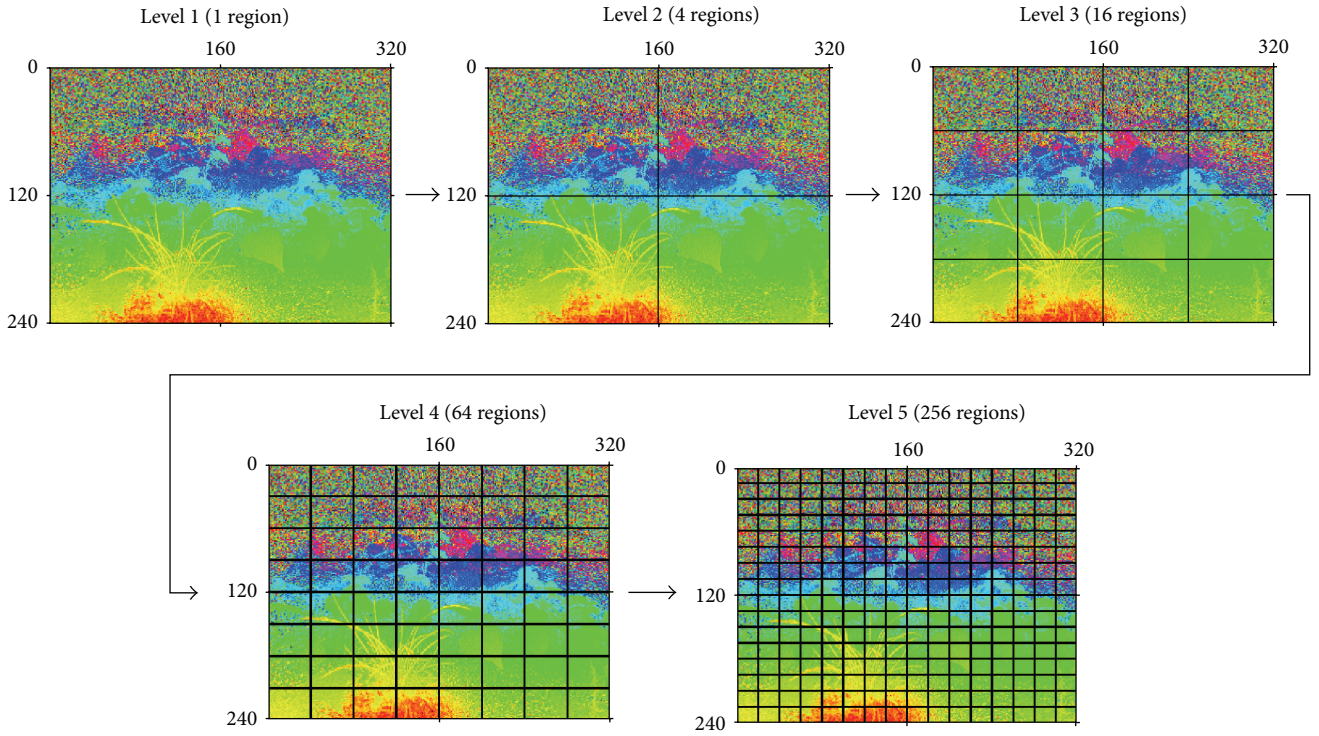


Figure 10: Recursive quadtree image regions used for feature extraction.

5.3. Feature Extraction. Division of depth images into uniformly sized rectangular regions in the quadtree manner, with distance and shape related features extracted from each region proved to be an effective and efficient way of traversability affordance learning and prediction [43]. Problem of terrain classification in our work does not require any knowledge of scene object shapes. Instead, we use similar grid structure but only extract statistical data about each region, minimum, maximum, mean, and standard deviation values. Filtered depth image is divided into regions in an iterative manner; first level is the full 320 × 240 pixels image. This region is then divided into four uniform regions of 160 × 120 pixels. Each of the four obtained regions is divided into another four uniform rectangles, which provides 16 regions and 16 + 4 + 1 = 21 total regions at level 3. This process continues and for levels 4 and 5 there are a total of 85 and 341 regions, respectively (see Figure 10). At every stage statistical data of each region is extracted and stored in a vector. With four features obtained from a region, iterations 2, 3, 4, and 5 generate feature vectors of 20, 84, 340, and 1364 lengths, respectively. The feature vectors are scaled to the range [0, 1] and the scaling coefficients are stored.

5.4. Terrain Classification. For the five-class terrain recognition problem we utilized Linear Discriminant Analysis (LDA) and Support Vector Machines (SVM) classifiers. LDA [44] is extensively used for dimensionality reduction and classification. LDA preserves the discriminant information of the original feature space; it moves the data to a subspace where the scatter between the classes is maximized, while the scatter within a class is minimized. Equations (5) and (6) are between and within scatter matrices, respectively [45]. Consider

$$S_b = \sum_{j=1}^{c} \left(\mu_j - \mu\right) \cdot \left(\mu_j - \mu\right)^T, \tag{5}$$

$$S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} \left(g_i^j - \mu_j\right) \cdot \left(g_i^j - \mu_j\right)^T, \tag{6}$$

where g_i^j, μ_j, μ, c, and N_j are sample i of class j, mean of class j, mean of all classes, number of classes, and number of samples in class j, respectively.

SVM [46] is a widely used kernel-based binary classification technique which builds an optimal hyperplane separating samples belonging to different classes. Consider the training sample $\{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in R^n$ and $y_i \in \{1, -1\}$ represent instances and classes, respectively. In the linear case, there are usually a number of nonunique hyperplanes separating two classes which are described by

$$\mathbf{w}^T \cdot \mathbf{x} + b = 0 \quad \mathbf{w} \in R^n, \; b \in R, \tag{7}$$

where $\mathbf{w}$ is the normal vector to the hyperplane and b is the scalar basis. SVM algorithm tries to find the hyperplane, which maximizes the margin between the closest samples from each of the classes by solving the quadratic optimization problem

$$\begin{aligned} &\min_{\mathbf{w},b} \quad \frac{1}{2} \|w\|^2 \\ &\text{subject to} \quad y_i \left(\mathbf{w}^T \cdot \mathbf{x} + b\right) \geq 1, \quad \forall i. \end{aligned} \tag{8}$$

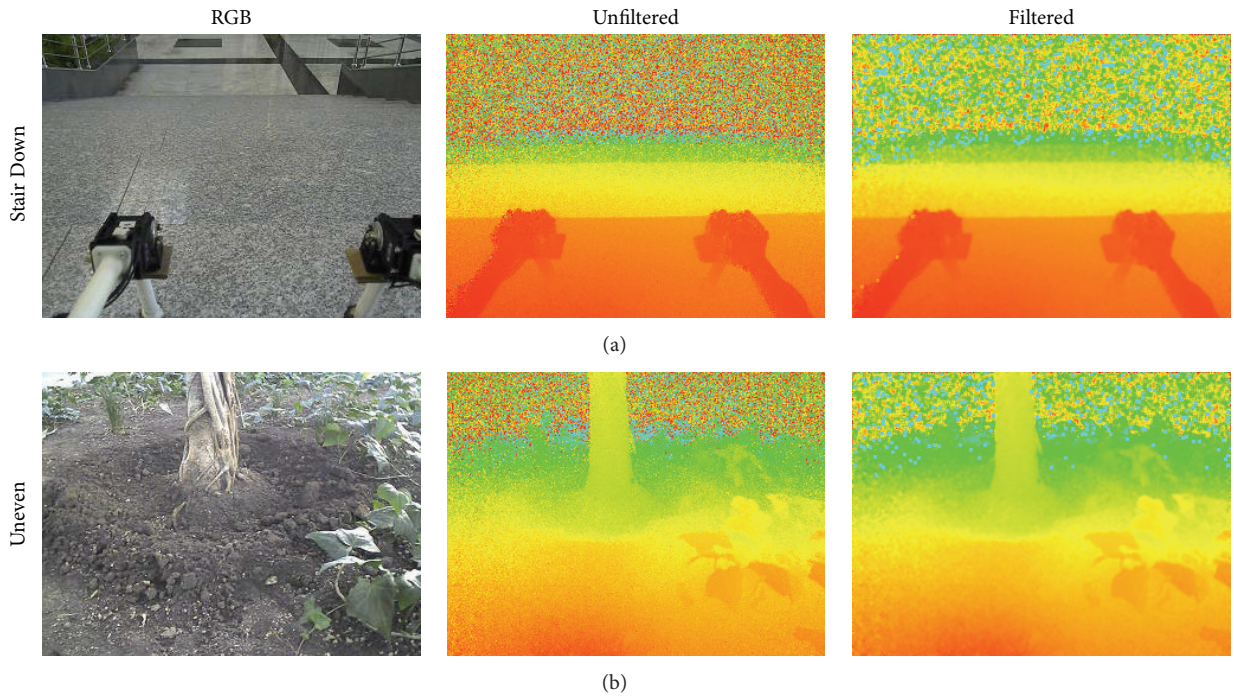


Figure 11: Filtered and unfiltered depth images and corresponding color image for *Stair Down* (a) and *Uneven* (b) Terrain samples.

Multiclass terrain recognition problem in this work is solved using one-versus-one technique by creating $c(c-1)/2$ SVM binary classifiers using linear kernels. We used LIBSVM library for SVM implementation [47].

Both LDA and SVM classification models are trained offline using training dataset feature vectors of varying length (20, 84, 340, and 1364) corresponding to the different feature extraction levels. The optimal combination of data dimensionality and classifier is then searched by evaluating classification models with prescaled testing dataset of corresponding length of feature vector. Entire set of information required for data scaling and classification (depending on the chosen classifier model and data dimensionality) is stored for use in online real-time terrain recognition.

5.5. Majority Voting Filter. Occasionally a single depth image, or a sequence of them, can be misclassified by the recognizer for various reasons, for example, an image captured while a robot is executing a highly dynamic motion. In case of hybrid mobile platforms, misclassified terrain can cause choosing wrong locomotion mode, which in its turn might lead to more serious failures. The robot might get stuck, stumble, or fall. Therefore, we implemented a majority voting scheme similar to the one applied in [48]. Specifically, the majority voting filter uses a sliding window of predefined size. This window contains consecutive terrain recognizer decisions which are processed to determine the environment which has the "votes" of 50% or more members. The apparent drawback of majority voting filter application is the increased decision-making delay and inferior reaction to the environment change. Depending on the number of classification results considered in the sliding window and sampling rate of the recognizer the latency might decrease the supervisory control performance. Thus there exists a trade-off between level of confidence and delay in the supervisory controller decisions. Simulation experiments were carried out to determine the length of the majority voting filter.

6. Results and Discussion

6.1. Terrain Recognizer Parameter Optimization. Initial set of experiments aimed to assess the effects of depth image filtering and search for an optimal combination of extracted feature vector length and classification model. For this we used both filtered and unfiltered training and testing datasets and applied various levels of feature extraction, specifically levels 2, 3, 4, and 5 corresponding to feature vector lengths of 20, 84, 340, and 1364, respectively. Level 1 was omitted as providing only four features for the entire image. Level 6 and the levels above were not considered due to the high number of features limiting real-time operation potential. These combinations were evaluated with both LDA and SVM classifiers resulting in 16 different classifier models. Generated SVM models did not undergo optimization and were used with default parameters of LIBSVM, except for the kernel type which was set to *linear*. Recorded accuracy and sampling time information is illustrated in Table 2.

Firstly, Table 2 shows that depth image filtering always improves the accuracy of terrain recognition. The qualitative effect of depth image filtering can be observed in Figure 11. It is also seen that SVM notably outperforms LDA in seven out of eight combinations. Additionally, SVM accuracy grows with more features used for classification. Accuracy improvements diminish with increasing feature vector lengths. On

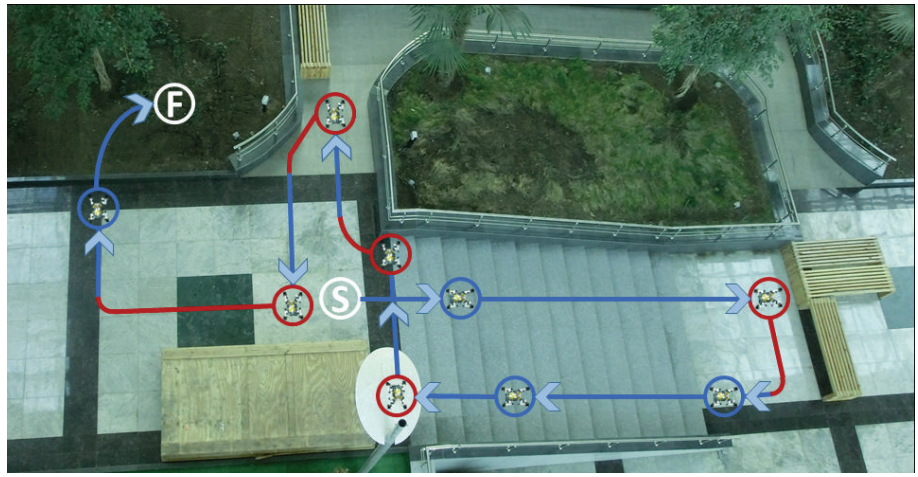


Figure 12: Top view of the experimental site with the outline of the route executed at the final trial. S and F letters in white circles denote route start and finish, respectively. Segments of the route with manual mode enabled are denoted by red lines and circles.

Table 2: Effects of different feature extraction levels and filtering to SVM and LDA classification accuracies.

Number of iterations	2	**3**	4	5
Number of features	20	**84**	340	1364
Extraction time (ms)	1.1	**2**	3.7	8.3
Unfiltered data				
LDA	70.16%	92.99%	94.57%	92.72%
SVM	77.65%	90.86%	95.27%	96.32%
Filtered data				
LDA	83.46%	94.78%	96.71%	95.12%
SVM	92.78%	**97.02%**	98.13%	98.46%

Table 3: Terrain classification confusion matrix of SVM classifier trained using 84-dimension data samples (filtered).

Terrain	Predicted class				
	1	2	3	4	5
Actual class					
Level Ground (1)	1760	0	6	21	13
Nontraversable (2)	1	1774	0	24	1
Stair Down (3)	2	1	1791	3	3
Stair Up (4)	8	14	0	1777	1
Uneven (5)	111	11	2	46	1630

the other hand, computation time required for feature extraction increases from 1.2 ms required to extract 20 features to 8.3 ms to extract 1364 features.

The length of the feature vector affects not only the feature extraction time, but also the computation time for classification. Considering the real-time nature of the quadruped supervisory control problem, we decided to use the level 3 feature extraction, which takes around 2 ms to compute. The confusion matrix for depth image filtered level 3 features with SVM classification is given in Table 3. Online real-time classification is accomplished using the classifier model generated during offline classifier model formation.

Subsequently, the length of the optimal majority voting filter was chosen. Presumably, a longer filter size would increase the confidence of the terrain classification. However, it would also introduce latency in the system response. Therefore, a trade-off exists between the terrain classification reliability and delay. Considering that the worst case computation time for the terrain recognition is around 30 ms during Stair Ascent and Descent, the sampling rate of the high level control was set to 30 Hz. An exhaustive simulation experiment was conducted using the third dataset, in which the effects of the majority voting filter length were evaluated. Table 4 shows the terrain recognition accuracy and corresponding terrain change detection delay for different voting filter sizes. It is seen that the terrain recognition accuracy increases with increasing filter size. However, it plateaus around 96.4% accuracy when the voting filter length reaches 13. Based on this, the filter length was set to 13 frames for real-time operation.

6.2. Real-Time Locomotion Experiments. A route consisting of *Level Ground*, *Stair Down*, *Stair Up*, *Uneven Terrain,* and *Nontraversable* regions at the Nazarbayev University atrium was selected as the test site for the system (see Figure 12). Experiment starts with the *Stair Down*, which is followed by the *Level Ground* and *Nontraversable* regions with the latter causing transfer of control to the human operator. Consequently, the robot is directed away from the *Nontraversable* terrain type and directed to the *Stair Up*. The section then follows by the combination of *Nontraversable* region and *Level Ground* types repeated twice one after another. The final segment of the route is represented by *Uneven Terrain*, the soil surface planted with various vegetation and trees. The ground is elevated above even terrain and has a step transition. For this configuration we added the step to *Uneven Terrain* class and programmed the single step ascent procedure for the robot locomotion. After transition into this environment, the robot implements legged locomotion and moves forward for some period of time after which the experiment is finished.

Table 4: Terrain recognition accuracies for different majority voting vector lengths.

Size	1	3	5	7	9	11	13	15	17	19	21	23	25
Level Ground, %	94,5	94,8	95,0	95,3	95,5	95,6	95,6	95,7	95,9	96,0	96,0	96,2	96,2
Nontraversable, %	95,8	97,3	98,0	98,9	98,9	99,0	99,6	99,6	99,8	100	100	100	100
Stairs Down, %	99,7	100	100	100	100	100	100	100	100	100	100	100	100
Stairs Up, %	99,6	99,9	100	100	100	100	100	100	100	100	100	100	100
Uneven, %	85,9	86,2	86,0	86,3	86,2	86,4	86,9	86,7	86,5	86,3	86,4	86,4	86,4
Total, %	95,1	95,6	95,8	96,1	96,1	96,2	96,4	96,4	96,4	96,5	96,5	96,5	96,5
Approximate delay (ms)	0	30	60	90	120	150	180	210	240	270	300	330	360

Real-time locomotion experiment was completed in approximately 8 minutes. Directions such as forward, backward, turn left, and turn right were given by the user for steering the platform away from *Nontraversable* regions when the presence of such resulted in the manual control invocation. All other aspects of the robot control were executed autonomously. The video of this experiment is provided as multimedia material (see multimedia material in Supplementary Material available online at http://dx.doi.org/10.1155/2015/425732).

Terrain recognizer results (both majority voting filtered and unfiltered), angular velocities of both active wheels, and joint trajectories of the front right leg were recorded during the experiment and a 150-second long segment of the trial is shown in Figure 13. The set of events which occurred in this time interval correspond to those starting with the robot Stair Ascent and ending with navigation in the *Uneven Terrain*. Four gray shaded regions on figure indicate the manual mode which is seen to be activated by recognition of *Nontraversable* terrain. Terrain recognizer results are still recorded during these intervals, and the manual mode is only disabled by the human operator. The fourth gray shaded region with manual control extended over *Level Ground* area represents the final segments of the route, where the user drives the robot away from *Nontraversable* terrain and directs it to the *Uneven Terrain* region.

The hybrid quadruped successfully completed the real-time experiment route. No false locomotion was implemented by the robot. Three Nontraversable objects (three wooden benches and one rectangular wooden box) were placed on the route of the robot (see Figure 12). Figure 13(a) shows the presence of terrain recognizer misclassification and "undefined" states (e.g., misclassification of *Uneven*, 12th second, and *Stair Up*, 16th second). Presumably, these instances are the result of the highly dynamic motion of the robot (e.g., climbing a Stair Up) combined with transitional stages which are not covered in the database (e.g., a wooden box appearing on the *Level Ground* exactly after the end of *Stair Up* segment).

Single case of misclassification of *Stair Up* and *Uneven* Terrain did not result in platform failure due to the reason that, upon entering into corresponding states, the robot starts searching for acclivity or declivity, that is, step down or up. As long as a step is not found, the robot continues moving and in case of actual terrain type being *Nontraversable* it transits into this mode smoothly and invokes the manual mode control. "Undefined" decisions do not affect system execution because the same locomotion controller is kept active until the majority voting filter chooses a different terrain type.

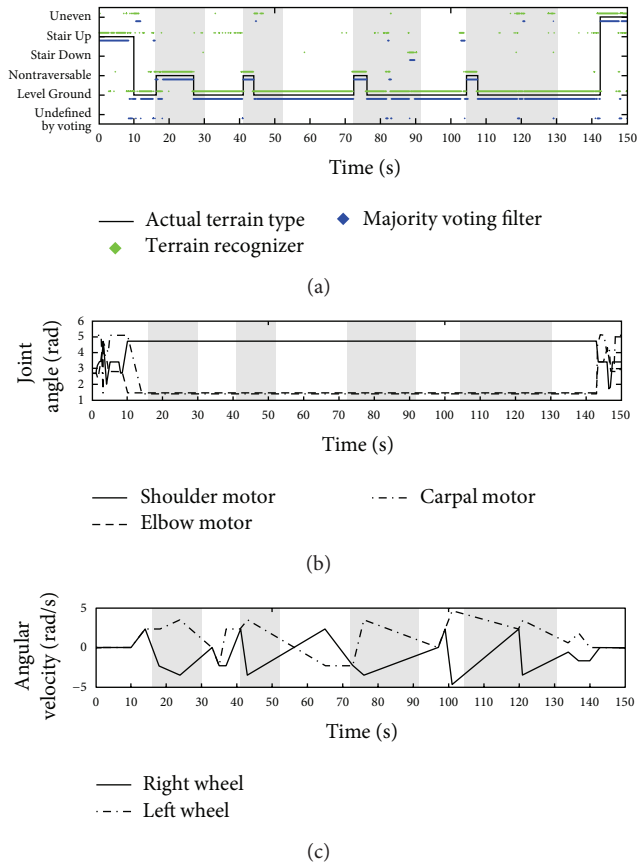


Figure 13: 150-second segment from the real-time locomotion experiment showing the terrain classification results (a), corresponding front right leg joint angles (b), and the wheel angular velocities (c). The gray shaded areas denote the manual control by the human operator activated either by the human user or by the detection of Nontraversable terrain.

7. Conclusion

Primary contribution of this work is showing the effectiveness of using depth sensor as the source of high fidelity perception for machine learning based terrain recognition. Real-time locomotion experiments in mixed terrain environments showed that the simple consumer depth sensor for finger and hand tracking might serve as an effective supervisory controller instrument for locomotion strategy selection for a hybrid robot with multiple locomotion modes. Additionally, we demonstrate the improvement of depth

image classification when simple confidence map based filters are used for depth image preprocessing.

As future work, we intend to extend the terrain recognition problem to higher number of classes and employ the algorithm in fully autonomous robots. Additionally, infrared daylight interference, one of the potential shortcomings of using the depth sensor, should be neutralized. Another area of future work is employing depth camera based terrain classification for the supervisory control of assistive devices such as exoskeletons for paraplegic patients.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] C. A. Brooks and K. D. Iagnemma, "Self-supervised classification for planetary rover terrain sensing," in *Proceedings of the IEEE Aerospace Conference*, pp. 1–9, Big Sky, Mont, USA, March 2007.

[2] Z. R. Luo, J. Z. Shang, and Z. X. Zhang, "Innovative design of six wheeled space exploration robot using module combination," in *Proceedings of the 19th International Conference on Mechatronics and Machine Vision in Practice (M2VIP '12)*, pp. 460–465, Auckland, New Zealand, November 2012.

[3] O. Matsumoto, S. Kajita, K. Tani, and M. Oooto, "A four-wheeled robot to pass over steps by changing running control modes," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '95)*, vol. 2, pp. 1700–1706, IEEE, Nagoya, Japan, May 1995.

[4] N. Shiroma, Y. H. Chiu, Z. Min, I. Kawabuchi, and F. Matsuno, "Development and control of a high maneuverability wheeled robot with variable-structure functionality," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, pp. 4000–4005, Beijing, China, October 2006.

[5] H. E. Yap and S. Hashimoto, "Development of a stair traversing two wheeled robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '12)*, pp. 3125–3131, Vilamoura, Portugal, 2012.

[6] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, The MIT Press, 2nd edition, 2011.

[7] S. Guccione and G. Muscato, "The wheeleg robot," *IEEE Robotics & Automation Magazine*, vol. 10, no. 4, pp. 33–43, 2003.

[8] H. Adachi, N. Koyachi, T. Arai, A. Shimiza, and Y. Nogami, "Mechanism and control of a leg-wheel hybrid mobile robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '99)*, vol. 3, pp. 1792–1797, Kyongju, Republic of Korea, 1999.

[9] G. Endo and S. Hirose, "Study on Roller-Walker (multi-mode steering control and self-contained locomotion)," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '00)*, pp. 2808–2814, San Francisco, Calif, USA, April 2000.

[10] J. A. Smith, I. Sharf, and M. Trentini, "PAW: a hybrid wheeled-leg robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '06)*, pp. 4043–4048, Orlando, Fla, USA, May 2006.

[11] J. Muller, M. Schneider, and M. Hiller, "Modeling, simulation, and model-based control of the walking machine ALDURO," *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 2, pp. 142–152, 2000.

[12] M. Bajracharya, T. Benyang, A. Howard, M. Turmon, and L. Matthies, "Learning long-range terrain classification for autonomous navigation," in *Proceedings of the International Conference on Robotics and Automation (ICRA '08)*, pp. 4018–4024, Pasadena, Calif, USA, May 2008.

[13] I. Halatci, C. A. Brooks, and K. Iagnemma, "A study of visual and tactile terrain classification and classifier fusion for planetary exploration rovers," *Robotica*, vol. 26, no. 6, pp. 767–779, 2008.

[14] Y. N. Khan, P. Komma, and A. Zell, "High resolution visual terrain classification for outdoor robots," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV '11)*, pp. 1014–1021, IEEE, Barcelona, Spain, November 2011.

[15] D. F. Wolf, G. S. Sukhatme, D. Fox, and W. Burgard, "Autonomous terrain mapping and classification using hidden Markov models," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '05)*, pp. 2026–2031, Barcelona, Spain, April 2005.

[16] A. Chilian and H. Hirschmüller, "Stereo camera based navigation of mobile robots on rough terrain," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, pp. 4571–4576, St. Louis, Mo, USA, October 2009.

[17] D. M. Helmick, A. Angelova, M. Livianu, and L. H. Matthies, "Terrain adaptive navigation for Mars rovers," in *Proceedings of the IEEE Aerospace Conference*, pp. 1–11, Big Sky, Mont, USA, March 2007.

[18] L. F. Posada, K. K. Narayanan, F. Hoffmann, and T. Bertram, "Ensemble of experts for robust floor-obstacle segmentation of omnidirectional images for mobile robot visual navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '11)*, pp. 439–444, Shanghai, China, May 2011.

[19] I. Rekleitis, J.-L. Bedwani, and E. Dupuis, "Over-the-horizon, autonomous navigation for planetary exploration," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 2248–2255, IEEE, San Diego, Calif, USA, November 2007.

[20] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Fast terrain classification using variable-length representation for autonomous navigation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07)*, Minneapolis, Minn, USA, June 2007.

[21] S. Zenker, E. E. Aksoy, D. Goldschmidt, F. Worgotter, and P. Manoonpong, "Visual terrain classification for selecting energy efficient gaits of a hexapod robot," in *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM '13)*, pp. 577–584, Wollongong, Australia, July 2013.

[22] M. Bellone, G. Reina, N. I. Giannoccaro, and L. Spedicato, "3D traversability awareness for rough terrain mobile robots," *Sensor Review*, vol. 34, no. 2, pp. 220–232, 2014.

[23] S. Foix, G. Alenyà, and C. Torras, "Lock-in time-of-flight (ToF) cameras: a survey," *IEEE Sensors Journal*, vol. 11, no. 9, pp. 1917–1926, 2011.

[24] S. Hsu, S. Acharya, A. Rafii, and R. New, "Performance of a time-of-flight range camera for intelligent vehicle safety applications,"

in *Advanced Microsystems for Automotive Applications 2006*, VDI-Buch, pp. 205–219, Springer, Berlin, Germany, 2006.

[25] S. May, B. Werner, H. Surmann, and K. Pervölz, "3D time-of-flight cameras for mobile robotics," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, pp. 790–795, October 2006.

[26] P. Breuer, C. Eckes, and S. Muller, "Hand gesture recognition with a novel IR time-of-flight range camera—a pilot study," in *Computer Vision/Computer Graphics Collaboration Techniques*, vol. 4418 of *Lecture Notes in Computer Science*, pp. 247–260, Springer, Berlin, Germany, 2007.

[27] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-flight cameras in computer graphics," *Computer Graphics Forum*, vol. 29, no. 1, pp. 141–159, 2010.

[28] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: using Kinect-style depth cameras for dense 3D modeling of indoor environments," *International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[29] G. Alenyà, S. Foix, and C. Torras, "ToF cameras for active vision in robotics," *Sensors and Actuators A: Physical*, vol. 218, pp. 10–22, 2014.

[30] Fotonic, "Fotonic E-Series," http://www.fotonic.com.

[31] MESA Imaging, "SR4000," http://www.mesa-imaging.ch/.

[32] BrainVision, BV-TOF, http://www.brainvision.co.jp/.

[33] BLUETECHNIX, ARGOS 3D—P100, http://www.bluetechnix.com/.

[34] Microsoft, Kinect for Windows, http://www.microsoft.com/en-us/kinectforwindows/.

[35] SoftKinetic, "DepthSenseCameras," http://www.softkinetic.com/.

[36] ROBOTIS, Dynamixel, http://www.robotis.com/.

[37] S. Hirose, N. Ootsukasa, T. Shirasu, H. Kuwahara, and K. Yoneda, "Fundamental considerations for the design of a planetary rover," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '95)*, pp. 1939–1944, Nagoya, Japan, May 1995.

[38] S. May, D. Droeschel, D. Holz, and C. Wiesen, "3D pose estimation and mapping with time-of-flight cameras," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS '08)*, Nice, France, 2008.

[39] M. Reynolds, J. Dobos, L. Peel, T. Weyrich, and G. J. Brostow, "Capturing time-of-flight data with confidence," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 945–952, Providence, RI, USA, June 2011.

[40] M. Frank, M. Plaue, and F. A. Hamprecht, "Denoising of continuous-wave time-of-flight depth images using confidence measures," *Optical Engineering*, vol. 48, no. 7, Article ID 077003, 2009.

[41] S. Y. Kim, M. Kim, and Y. S. Ho, "Depth image filter for mixed and noisy pixel removal in RGB-D camera systems," *IEEE Transactions on Consumer Electronics*, vol. 59, no. 3, pp. 681–689, 2013.

[42] F. Carreira, J. M. F. Calado, C. Cardeira et al., "Enhanced PCA-based localization using depth maps with missing data," *Journal of Intelligent & Robotic Systems*, vol. 77, pp. 341–360, 2015.

[43] E. Ugur, M. R. Dogar, M. Çakmak, and E. Sahin, "The learning and use of traversability affordance using range images on a mobile robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 1721–1726, Roma, Italy, April 2007.

[44] R. A. Fisher, "The statistical utilization of multiple measurements," *Annals of Eugenics*, vol. 8, no. 4, pp. 376–386, 1938.

[45] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 4th edition, 2008.

[46] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[47] C. C. Chang and C. J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, article 27, 2011.

[48] H. A. Varol, F. Sup, and M. Goldfarb, "Multiclass real-time intent recognition of a powered lower limb prosthesis," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 3, pp. 542–551, 2010.