Proceedings of the ASME 2017 Dynamic Systems and Control Conference
DSCC2017
October 11-13, 2017, Tysons, Virginia, USA

DSCC2017-5324

OBSTACLE IDENTIFICATION FOR VISION ASSISTED CONTROL ARCHITECTURE OF A HYBRID MECHANISM MOBILE ROBOT

Anil Kumar*
Robotics and Mechatronics Lab
Dept. of Mechanical Engineering
Virginia Tech,
Blacksburg, VA, USA

Hailin Ren*
Robotics and Mechatronics Lab
Dept. of Mechanical Engineering
Virginia Tech,
Blacksburg, VA, USA

Pinhas Ben-Tzvi**
Robotics and Mechatronics Lab
Dept. of Mechanical Engineering
Virginia Tech,
Blacksburg, VA, USA

ABSTRACT

This paper presents a monocular vision-based, unsupervised floor detection algorithm for semi-autonomous control of a Hybrid Mechanism Mobile Robot (HMMR). The paper primarily focuses on combining monocular vision cues with inertial sensing and ultrasonic ranging for on-line obstacle identification and path planning in the event of limited wireless connectivity. A novel, unsupervised vision algorithm was developed for floor detection and identifying traversable areas, in order to avoid obstacles in semi-autonomous control architecture. The floor detection algorithms were validated and experimentally tested in an indoor environment under various lighting conditions.

INTRODUCTION

Field mobile robots are extremely useful for operating in hazardous environments and performing urban search and rescue missions. To enhance mobility in unseen environments, most mobile robots are wirelessly operated. Although mobile robots have much potential in hazardous and hard-to-access environments, wireless connectivity has always been a limiting factor in the full-scale utilization of robots in these scenarios [1]. As fully autonomous robots are reliable only in controlled environments, semi-autonomous control based on onboard sensing and computing remains the only viable means to reduce dependence on wireless connectivity. In the absence of connectivity with a human operator, the robot's most critical computational task is obstacle identification. Since the robot may encounter many different types of obstacles, an alternative to identifying the obstacles themselves is to identify traversable areas in their field of sensing and then treat everything else as an obstacle.

Identification of the ground plane in a video feed for visual navigation of robots has attracted many researchers all over the world [2–6]. Although detecting the ground plane is easier with depth sensors, it is computationally more expensive to process 3D point cloud data than a 2D image. Researchers have often resorted to monocular vision processing to overcome such constraints.

Identification of traversable areas by floor/road detection is a key technique in semi-autonomous robotic navigation. Sensors such as RGB cameras, depth cameras, radar rangers, and LIDAR are often used to map surrounding environments, detect objects, and extract traversable regions from the current view field. Algorithms for detecting traversable regions can be classified into two categories: supervised algorithms and unsupervised algorithms. Supervised algorithms build machine learning classifiers that use training data such as color [7], texture [8], and shape [9] to model image appearance. These classifiers are then used to identify segments in test images [10,11]. However, such methods are limited by the quality of the training dataset. The performance of classifiers in identifying new scenes (i.e. scenes not included in training data) is very poor.

Unsupervised algorithms are based on the weak assumption that the segment that the robot maneuvers on is the floor, while any other objects visible in the image are treated as obstacles. In road region detection methods, texture and color features with homogeneous road appearance constraints are used to segment the view field [12]. Such methods also use depth cues in stereo vision [13] or image homographic cues in monocular [14] camera vision. Road boundary detection algorithms often use special filters or detailed road models to estimate the edge of the road, combining texture and color cues [15]. Detecting the vanishing point is a key technique in these methods. Chang et al. [16] combined region segmentation with road boundary detection by using voting mechanism methods to improve the segmentation performance in both indoor and

* Authors contributed equally; **Corresponding author – bentzvi@vt.edu

outdoor scenes. This paper presents a new unsupervised algorithm for combining robotic sensing with monocular machine vision to detect the floor/road in any environment and suggest a semiautonomous control strategy for the Hybrid Mechanism Mobile Robot (HMMR). Compared to other existing methods [11,17], the proposed system uses low cost sensing modalities like ultrasonic sensors, monocular vision and inertial sensing, which can be implemented in space constrained problems. In addition to HMMR, the presented algorithm can also be used in assistive robots for visually impaired people. The performance of the floor identification algorithm is presented on prerecorded video.

HMMR SYSTEM ARCHITECTUSRE

The HMMR is an articulated three-linkage, tracked mobile robot capable of using its manipulation links for locomotion and to traverse large obstacles [19–21]. Figure 1 shows a CAD rendering of the HMMR. As demonstrated in [19,22], the HMMR can traverse highly uneven terrain and climb over large obstacles. In a retracted form, the overall dimensions of the HMMR are 530mm(W)×630 mm(L)×140mm(H). The HHMR achieves locomotion using two tracks (left track and right track) which are connected through a shaft, together forming the Base Link (Link 1). Each track is driven by 250 W brushless DC motors through a planetary gearbox, a bevel gear, and a pulley assembly. In addition to the tracks, the two remaining revolute joints are also actuated from the base link through harmonic drives. Link 3 hosts a 3-DOF two-finger gripper. All the links and modules in the HMMR have individual power supplies and communicate wirelessly, allowing endless rotation in all revolute joints in the HMMR. The end effector can also rotate endlessly inside Link 3 to achieve full pitch/roll rotation.

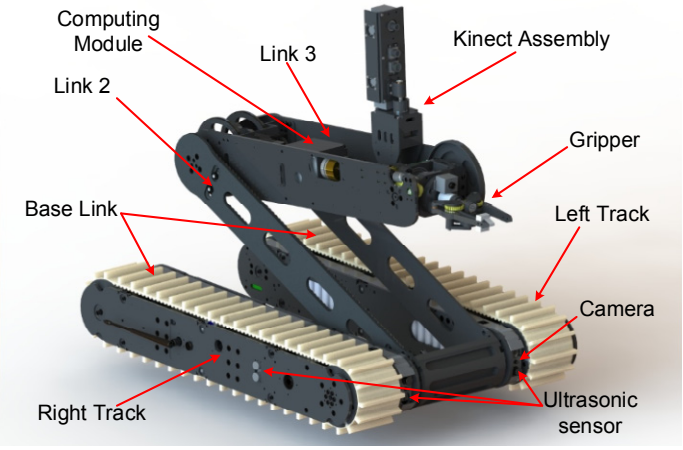

Figure 1. CAD rendering of the HMMR with onboard sensors labelled.

In each track, there are two lamps (one on the front side and one on the rear side), four ultrasonic sensors (two on the lateral side, one on the front side and one on the rear side), GPS (right track)/IMU (left track), and one camera (rear side of the right track or front side of the left track). In Link 3, there is one 2-DOF mounting platform for a Kinect. Driven by servo motors, the mounting platform can achieve pitch and yaw rotation, which helps the Kinect scan the surrounding environment. Figure 2 shows the mechatronic architecture of the HMMR.

A hierarchical system is proposed to fuse sensor data, send actuator commands, and achieve high-level tasks such as obstacle detection and motion planning. The computing architecture is divided into three computing levels: microcontroller (MCU) level, single board computer (SBC) level, and Operator's Computer (OC) level.

At the MCU level, a Teensy 3.2 is used as the microcontroller (MCU) in each link. It fetches raw sensor data and sends commands to the actuators. All the subsystems in the HMMR are controlled by dedicated 'slave' MCUs. The sensory data and commands are shared amongst the subsystems over a Wi-Fi network. The MCU mounted in the master board in Link 3 updates status data from the slave MCUs in the HMMR and the Operator's Control Unit (OCU).

At the SBC level, single board computers (Raspberry Pi Zero), mounted in both the right and left tracks, fetch video data from the rear and front cameras, respectively, preprocess data (filtering noise, undistorting frames), and stream processed data to a local network via Wi-Fi. Another single board computer (Odroid XU4) is mounted in Link 3, working as the central computing unit (CCU), and is connected to the Wi-Fi router via Ethernet, and to the master MCU via USB cable. The CCU fetches HMMR updated data from the Teensy MCUs and video data from the network, and then publishes them via Robot Operating System (ROS). Some high-level tasks are processed in CCU such as floor detection, motion planning, and environment mapping.

The HMMR is remotely controlled by a touchscreen enabled OCU powered by a high performance laptop computer. At the OC level, a ROS based GUI processes 2D/3D vision data received from the HMMR to visualize the robot's status and the sensed environment (in 2D/3D). The OCU also combines commands received from joysticks with other user interactions through the GUI and sends command packets over a long-range WiFi network.

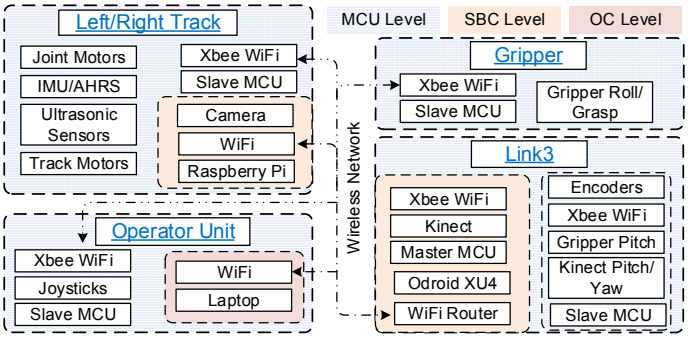

Figure 2. Mechatronics architecture of the HMMR.

IMAGE SEGMENTATION FOR FLOOR DETECTION

The most intuitive means of identifying area that is traversable by the robot is to identify the ground in the visual feed. The limited on-board computational capability of the HMMR requires simplicity in the segmentation process. The sole objective of the proposed method is to segment a given image $I(x,y)$ by assigning labels $L(x,y)$ to the pixels on the basis of regions/objects visible in the image and then classify the labels into 'floor' and 'not-floor' classes. The presented algorithm autonomously identifies the ground plane in a monocular video feed on the basis of multi-sensor cues without requiring any prior knowledge of the environment. The proposed algorithm can be divided into four major operations:

1. Image Over-segmentation,
2. Floor Subset Identification,
3. Feature Extraction,
4. Clustering and Floor Identification.

The proposed method first over-segments the input image to identify similar looking regions (superpixels) in the image in order to reduce computation time later in the process of the algorithm. Then, the algorithm combines ultrasonic sensor readings with the HMMR's attitude information (from IMU) to identify the floor region subset. The method then uses the floor subset region along with results from the previous iterations to extract four-dimensional features, and assigns them to the corresponding super pixels. In this final stage, the proposed method clusters the super-pixels on the basis of feature and identifies clusters representing the floor. Figure 3 shows the flow diagram of the proposed method.

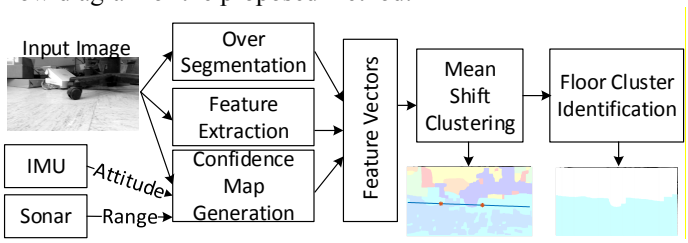

Figure 3. Flow diagram of the proposed algorithm.

A. *Image over-segmentation:*

Image segmentation is essentially assigning labels to each pixel in an input image. To assign labels, each feature corresponding to each pixel needs to be individually evaluated and compared against all other remaining pixels. Due to computational limitations, this process is unfeasible for real-time applications. In robotic vision, spatial continuity of the visible objects in an image can be exploited by treating similar looking adjacent pixels as a single entity, which drastically reduces computational requirements.

Many researchers have used over-segmentation as a tool for dimensional reduction in image segmentation [6,23–26]. The proposed system uses a watershed transform [27], which needs an edge map to operate, to over-segment the input image. To keep the number of superpixels low, the edge map was obtained on a lower resolution image through a 'Laplacian of Gaussian' (LoG) filter. Reduction of image size not only reduced the computational load, but also removed false images arising from texture content in an image. Figure 4 shows the result of the watershed over segmentation on an input image from the camera feed. The edge map resulted in 400-500 segments in an input image of size 324×576 pixels.

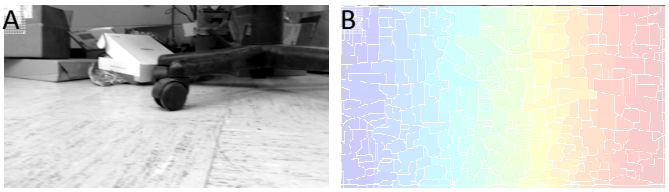

Figure 4. Image Over-segmentation (A) Input Image (grayscale channel), (B) Colored watershed transform segments.

B. *Floor subset Identification:*

Separating the floor from the background in images requires prior knowledge of the floor in the image domain. The HMMR uses IMU/AHRS data along with ultrasonic sonar sensor measurement to detect the floor subset in an image. In the proposed method, it is assumed that the ground is flat (no-inclination) in general with possible minor irregularities. Two sonar sensors, mounted on the same plane as the camera, give two distance estimates for any possible obstacles in front of the HMMR. With the attitude angles $\{\theta, \varphi, \psi\}$ available from the IMU sensor, and the location of the camera and sonar sensors known, the base points of the obstacles can be obtained by using basic geometry (Figure 5A).

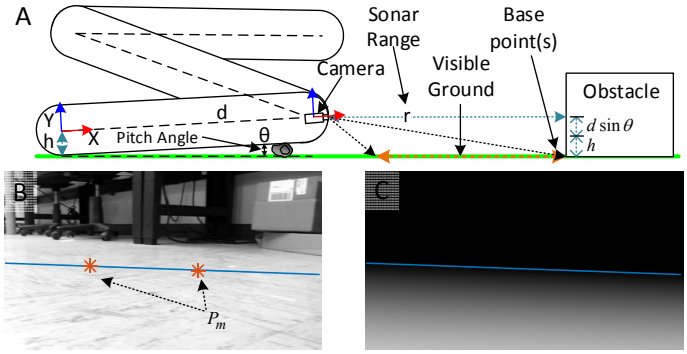

Figure 5. Floor subset Identification: (A) Perception schematics, (B) Base point markers, (C) Floor confidence map.

In addition, as the camera's intrinsic and extrinsic parameters are already known through camera calibration, the same obstacle base points can also be determined in the camera image.

$$P_m = \mathrm{CR}(\theta,\varphi,\psi)\begin{bmatrix} r \\ -d\sin\theta - h \\ 0 \end{bmatrix} \quad (1)$$

Here, P_m is the marker point coordinates in the image, r is the

SONAR range reading, C is the camera projection matrix composed on the Camera's intrinsic and extrinsic parameters, and R is the rotation matrix converting world frame of reference to the camera frame of reference (depending on the attitude angle).

In the absence of any obstacles, the sonar sensors return the maximum range as their output (450 cm in case of the HMMR). Thus, readings from two sonar sensors on the HMMR can be translated into two 'base points' in the capture image. These base points can be used to make a fair assumption that the region lying in the lower half of the line joining the two base points is the floor. This line is then translated into the floor 'confidence map' $C(x,y)$ as follows:

$$C(x,y) = \max\left(0, \frac{1-e^{-2t}}{1+e^{-2t}}\right) \quad \Big| \quad \mathrm{t} = \frac{y - mx - c}{\sqrt{m^2+1}} \tag{2}$$

Here, m and c represent the line parameters obtained from the two limit points (1). Figure 5 (B, C) shows base point markers and the corresponding floor confidence map for an input image. The value of the floor confidence map estimate increases with distance from the line joining the base point markers.

C. Feature extraction

The proposed method uses 4-dimensional (4-D) features to separate the floor from the rest of the background in the image. Each superpixel obtained in the 'over-segmentation' process was assigned a 4-D feature vector. Intensity and color, being the most intuitive descriptors of an object in an image, constitute two of the four features used for image segmentation. To obtain these two channels, the input color image was converted to *CIE L*a*b** color space and the median values of the L^* and b^* channels corresponding to each superpixel were computed. For color features, the b^* channel was specifically used because of its intensity-invariant capability to describe blue/green color shades. Figure 6 show intensity and color features assigned to the computed superpixels.

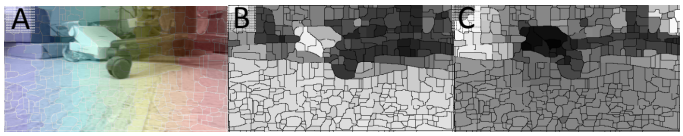

Figure 6. Feature extraction: (A) Superpixels, (B) Intensity channel, and (C) b^* color channel.

As most real-world objects possess a textured appearance, robust segmentation solely based on intensity and color information is not possible. Texture is a critical descriptor in image analysis. The literature classifies texture feature techniques into four basic categories, viz. statistical methods, geometric methods, model based methods, and signal processing methods [28]. As proposed by Haralick *et al.* [28], the Gray Level Co-occurrence Matrix (GLCM) is an excellent tool for extracting statistical texture features in an image. GLCM is a 2D histogram of joint occurrences of intensity levels i and j on pixels displaced by an offset d in direction θ.

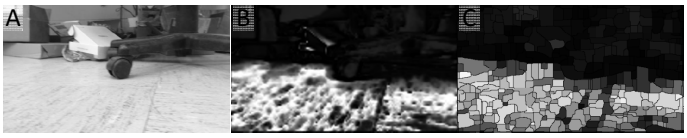

Figure 7. Texture feature: (A) Input image (grayscale channel), (B) GLCM texture descriptor, and (C) Pixelated texture descriptor on superpixels.

Texture, defined as repeated occurrences of intensity patterns, can be described by the probability of occurrence of intensity values in a neighborhood. For statistically modelling the 8-connectivity neighborhood in an 8-bit grayscale image $I(x,y)$, the proposed method computed four GLCMs $G(i,j,d,\theta)$ for four different directions $\theta=\{0°,45°,90°135°\}$, at offset $d=1$. Objects of different sizes feature different numbers of intensity pairs in an image and hence result in significantly different GLCM values for the concerned intensity pairs. Similar to Sachdeva et al. [29], intensity co-occurrence statistics are derived from GLCMs and used as texture descriptors in the proposed algorithm. The normalized GLCMs are used as look-up tables to transform the image into a statistical texture domain. The normalized texture descriptor $T(x,y)$ of image I is obtained by averaging G over the parameter θ as shown in (3). Figure 7 shows a normalized texture descriptor generated for an input image.

$$T(x,y) = \frac{1}{4}\sum_{\theta} G\left(I(x,y), I(x + d\cos\theta, y + d\sin\theta), d, \theta\right) \tag{3}$$

In addition to appearance-based characteristics, the floor boundary edge separating the ground plane from the rest of the content in the image is predominantly visible. This horizon edge splits the image into floor and not-floor regions and hence is a very useful cue in floor identification. To obtain the floor boundary cues, the horizontal edges in an input image were extracted using a Sobel filter. The obtained edges were integrated vertically upward to obtain a cumulated edge map. The mean value of the cumulated edge map in a superpixel was used as the fourth feature in the process of floor identification. Figure 8 shows floor boundary cues extracted for a grayscale input image.

Figure 8. Floor boundary cue: (A) Input image (grayscale channel), (B) Horizontal edges, and (C) Cumulated edge map.

D. Clustering and Floor Identification:

To identify the floor superpixels from non-floor superpixels, a prior knowledge of the floor in the image is necessary. The current expected value of the feature vector is estimated from the floor confidence map (2) as follows:

$$\overrightarrow{F_c} = \frac{\sum C(s_i)\vec{F}(s_i)}{\sum C(s_i)} \tag{4}$$

Here C represents the confidence map as computed in (2), s_i represents the centroid coordinates of the i^{th} superpixel, $\vec{F}$ is the corresponding 4-dimensional feature vector, and $\vec{F}_c$ is the expected value of the feature vector from the current readings. As the floor/road appearance normally does not change abruptly in image sequences, the floor segmentation in the previous iteration can be extremely useful. In addition to the current expected feature vector, the mean feature vector from the previous iteration's final floor segment is represented by $\vec{F}_p$. Both $\vec{F}_c$ and $\vec{F}_p$ are combined using a weighted sum to obtain the net expected feature vector

$$\vec{F}_n = \alpha\vec{F}_c + (1-\alpha)\vec{F}_p; \quad \alpha \in [0,1] \tag{5}$$

Here, α is an 'exploration factor' that balances the reliance of the algorithm on current readings (exploration) and previous results (recall). The proposed system used α as 0.75, to rely more heavily on current sensor readings than previous results. Before segmenting the image, the superpixel feature vectors were adjusted to separate the floor superpixels from the non-floor pixels as follows:

$$\vec{F}' = abs(\vec{F} - \vec{F}_n) \tag{6}$$

The processed superpixels were then congregated into image segments by using mean shift clustering [30] with a bandwidth of 0.25 as shown in Figure 9(B). The image segments falling 50% or more below the line connecting base markers were labelled as floor, shown in blue (Figure 9).

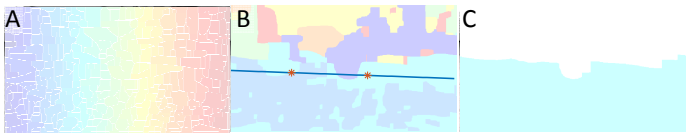

Figure 9. Floor Detection: (A) Superpixels, (B) Image segments, and (C) Final floor segment.

SEGMENTATION RESULTS

To test the performance of this algorithm, a 13-second video filmed at 30 FPS was recorded in the lab, resulting in 408 frames, and used as the test dataset. The video consists of navigation (moving forward, turning, tilting up, tilting down, etc.) on the floor. Obstacle base point markers were manually marked by the user because of the absence of ultrasonic sensors during the test. A custom made MATLAB GUI was developed to mark the base point marked in the video frames. The markers were placed up to 2ft distance away from the camera in image frames such that obstacles were always above the line joining base points. The video was then processed using this algorithm to detect the traversable area. Some samples of the results are shown in Figure 10, with the floor shown in blue. A MATLAB implementation of the algorithm took 0.27 s to process each frame of size 324×576 pixels on a quad core 3.3GHz Intel® Xeon™ computer. Python implementation of this algorithm takes approximately 0.22 seconds per frame on an 8-core 2.20GHz Intel®Core™i7-2670QM with GPU optimization. The video is also processed manually to cluster the floor and obstacles using the definitions stated before and used as a true value in the dataset.

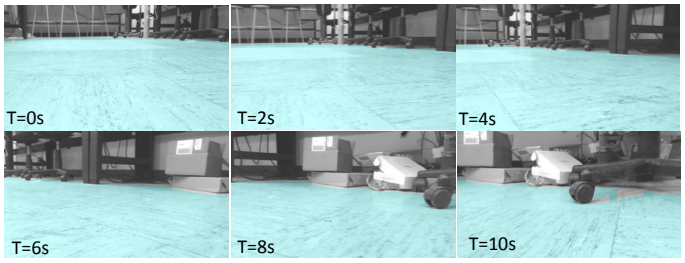

Figure 10. Floor detection results in a video image sequence.

The Tanimoto Index (*TI*), used to measure algorithm performance quantitatively, can be calculated as follows

$$TI = A \cap B \,/\, A \cup B \tag{7}$$

where, A is the floor set segmented by the algorithm and B is the floor set manually segmented by a human. As a result, TI can measure the similarity between the two sets/regions. A value of TI approaching 1 indicates that two regions are exactly the same and totally overlap with each other. TI approaching 0 indicates two dissimilar sets with nothing in common. Performance analysis shows that 89.95% of the frames were over 0.9, as shown in Figure 11. As the proposed algorithm primarily relies on visual cues, it is susceptible to false positive errors if the floor/ground visually appears similar to the surrounding obstacles. Outliers in figure 11 (instances with TI<0.8) correspond to similar situations where obstacles (like door/wall) resembled the floor.

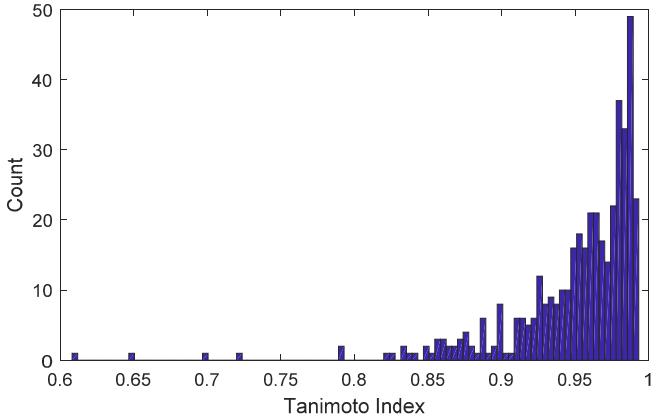

Figure 11. Performance analysis of floor segmentation.

The segmentation performance accuracy greatly depends on the sensing modalities used in the algorithm. The presented algorithm was capable of delivering equivalent or better segmentation accuracy than the existing methods [11,17,18] without using laser based depth sensing. Compared to the accuracy of 0.828, 0.865 and 0.918 of [11], [17] and [18], respectively, the presented algorithm delivered an accuracy of 0.899. Also, unlike other algorithms, the proposed algorithm adapts automatically according to the background and adapts to operate in different types of floor with a variety of lighting conditions.

HMMR'S SEMI-AUTONOMOUS CONTROL STRATEGY

Semi-autonomous control for the HMMR is essentially autonomous motion planning on the basis of frequent inputs received by the operator. The operator provides the HMMR with commands in the form of direction and intensity of motion with the help of a control stick. These commands can be easily translated into 3D goal positions in the world frame of reference with respect to the robot's current position by using some predefined scaling factor (s). If $\{j_x j_y\}$ is the 2-axis joystick input from the OCU, the 3D goal position can be estimated as follows

$$ {}^{I}P_G = R_z(\psi)\, s \begin{bmatrix} j_x & j_y & 0 \end{bmatrix}^T + {}^{I}P_R \qquad (8) $$

Here, ${}^{I}P_G$ and ${}^{I}P_R$ are the position vectors of the goal position and robot position in inertial frame of reference and R_z is the rotation matrix of the robot frame about the z-axis (yaw), with respect to the inertial frame of reference. The Monocular vision based robot control essentially requires mapping from the 3D world coordinate system to the 2D image coordinate system. The projection from a 3D point A on the ground plane to a 2D point on the image (Figure 12) can be obtained as follows:

$$ u_A = u_0 + f_u \frac{{}^{C}y_A}{s_u\, {}^{C}z_A}, v_A = v_0 + f_v \frac{{}^{C}x_A}{s_v\, {}^{C}z_A} \qquad (9) $$

where $\{u_A, v_A\}$ and $\{u_0, v_0\}$ represent the pixel coordinates of point A and the principal center (intersection point between the optical axis of the camera lens and the image plane), respectively, in an image of size $\{s_u, s_v\}$. The 3D coordinates of point A in the camera frame can be obtained by,

$$ \begin{bmatrix} {}^{C}x_A \\ {}^{C}y_A \\ {}^{C}z_A \end{bmatrix} = T_I^C \begin{bmatrix} {}^{I}x_A \\ {}^{I}y_A \\ {}^{I}z_A \end{bmatrix} = T_R^C T_I^R \begin{bmatrix} {}^{I}x_A \\ {}^{I}y_A \\ {}^{I}z_A \end{bmatrix}. \qquad (10) $$

Here, $\{{}^{I}x_A, {}^{I}y_A\ {}^{I}z_A\}$ are the coordinates of point A in the inertial frame, and ${}^{C}T_R$, ${}^{R}T_I$ are homogenous transformation matrices from the camera frame to the robot frame and from the robot frame to the inertial frame, respectively.

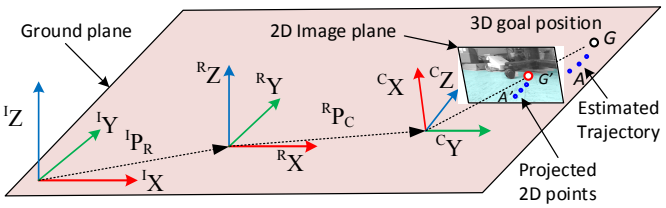

Figure 12. 3D Motion planning through 2D trajectory.

The projection from a 2D point on the image to a 3D point on the ground plane (${}^{I}z_A=0$) can be obtained using the inverse function of (9). Now, any one of the existing path planning algorithms can be used to determine the optimum trajectory ${}^{I}P_A$ as a function of time for the robot to reach the goal position ${}^{I}P_G$ from its initial position in the inertial frame of reference. The 3D trajectory points, which can be projected into the image frame using (10), can be used with floor identification segments to check the feasibility of the trajectory.

During the interval of the low frequency operator's command, path planning to the goal point would be done without any collisions with obstacles. The goal position will be updated when the robot receives the operator's command. Trajectory following could then be achieved by using motion control,

$$ \begin{bmatrix} {}^{I}\dot{X} \\ {}^{I}\dot{Y} \\ {}^{I}\dot{\Psi} \end{bmatrix} = R_z(\psi) \frac{1}{2b} \begin{bmatrix} bV_r(1-i_r) + bV_l(1-i_l) \\ 0 \\ V_r(1-i_r) - V_l(1-i_l) \end{bmatrix}. \qquad (11) $$

Here, $\{V_r, V_l\}$ and $\{i_r, i_l\}$ represent the theoretical speed and slip coefficient for the right and left tracks, respectively. The slip coefficients will be initialized using experimental data and updated by comparing motor driver commands and IMU data in real-time.

CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new method to detect traversable regions using monocular image sequences. Combined with inertial sensing and ultrasonic ranging, this method further shrinks the previous popular assumption that the region in front of the robot is traversable and improves performance of floor detection. In the future, we will further improve the algorithm performance and implement it on the Odroid XU4. A new hierarchical architecture will be developed, which allocates the light data preprocessing task to low level processors and heavy computation to powerful processors, to achieve faster and reliable performance. To achieve semi-autonomous control, the floor detection algorithm will be implemented on the HMMR to detect the traversable area and determine the driving path from the floor contour. A motion controller will be developed to keep the HMMR following the desired path and moving to the point assigned by high-level navigation commands. Besides HMMR, the algorithm can be used in any industrial/research scenario involving dynamic ground/floor detection. It can also be implemented on industrial robots for the detection of

obstacles/humans in the workspace to achieve enhanced safety in industrial environments.

REFERENCES

[1] Casper, J. L., Micire, M., and Murphy, R. R., 2000, "Issues in Intelligent Robots for Search and Rescue," Soc. Opt. Eng., **4024**, pp. 292–302.

[2] Scaramuzza, D., and Siegwart, R., 2008, "Appearance-Guided Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles," IEEE Trans. Robot., **24**(5), pp. 1015–1026.

[3] Kim, Y. K. Y., and Kim, H. K. H., 2004, "Layered Ground Floor Detection for Vision-Based Mobile Robot Navigation," Int. Conf. Robot. Autom., (ii), pp. 13–18.

[4] Pears, N., and Bojian Liang, 2001, "Ground Plane Segmentation for Mobile Robot Visual Navigation," Proc. 2001 IEEE/RSJ Int. Conf. Intell. Robot. Syst. Expand. Soc. Role Robot. Next Millenn. (Cat. No.01CH37180), **3**, pp. 1513–1518.

[5] Wu, Z., Fu, W., Xue, R., and Wang, W., 2016, "A Novel Line Space Voting Method for Vanishing-Point Detection of General Road Images," Sensors (Switzerland), **16**(7).

[6] Wang, H., and Gong, Y., 2014, "Road Detection via Superpixels and Interactive Image Segmentation," 4th Annu. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst., pp. 152–155.

[7] Vailaya, a., Jain, A., and Hong Jiang Zhang, 1998, "On Image Classification: City vs. Landscape," Proceedings. IEEE Work. Content-Based Access Image Video Libr. (Cat. No.98EX173), **31**(12), pp. 3–8.

[8] Chang, E., Goh, K., Sychay, G., and Wu, G., 2003, "CBSA: Content-Based Soft Annotation for Multimodal Image Retrieval Using Bayes Point Machines," IEEE Trans. Circuits Syst. Video Technol., **13**(1), pp. 26–38.

[9] Ulrich, I., and Nourbakhsh, I., 2000, "Appearance-Based Place Recognition for Topological Localization," Robot. Autom. 2000. Proceedings. ICRA '00. IEEE Int. Conf., **2**(April), pp. 1023–1029 vol.2.

[10] Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S., and Bradski, G., 2006, "Self-Supervised Monocular Road Detection in Desert Terrain," Proc Robot. Sci. Syst. RSS.

[11] Maier, D., Bennewitz, M., and Stachniss, C., 2011, "Self-Supervised Obstacle Detection for Humanoid Navigation Using Monocular Vision and Sparse Laser Data," Proc. - IEEE Int. Conf. Robot. Autom., pp. 1263–1269.

[12] Hoffmann, J., Jüngel, M., and Lötzsch, M., 2004, "A Vision Based System for Goal-Directed Obstacle Avoidance Used in the RC'03 Obstacle Avoidance Challenge," Proc. Int. Work. Rob. 2004 (Robot World Cup Soccer Games Conf., pp. 418–425.

[13] Sabe, K., Fukuchi, M., Gutmann, J.-S., Ohashi, T., Kawamoto, K., and Yoshigahara, T., 2004, "Obstacle Avoidance and Path Planning for Humanoid Robots Using Stereo Vision," IEEE Int. Conf. Robot. Autom., **1**(April 2017), pp. 592–597.

[14] Zhou, J., and Li, B., 2006, "Robust Ground Plane Detection with Normalized Homography in Monocular Sequences from a Robot Platform," Proc. - Int. Conf. Image Process. ICIP, (1), pp. 3017–3020.

[15] Alvarez, J. M., Gevers, T., and Lopez, A., 2010, "3D Scene Priors for Road Detection," IEEE Comput. Vis. Pattern Recognit., pp. 57–64.

[16] Chang, C. K., Siagian, C., and Itti, L., 2012, "Mobile Robot Monocular Vision Navigation Based on Road Region and Boundary Estimation," IEEE Int. Conf. Intell. Robot. Syst., pp. 1043–1050.

[17] Rasmussen, C., Lu, Y., and Kocamaz, M., 2009, "Appearance Contrast for Fast, Robust Trail-Following," 2009 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS 2009, pp. 3505–3512.

[18] Qingquan Zhang, Yong Liu, Yiyi Liao, and Yue Wang, 2015, "Traversable Region Detection with a Learning Framework," 2015 IEEE Int. Conf. Robot. Autom., pp. 1678–1683.

[19] Ben-Tzvi, P., Goldenberg, A. a., and Zu, J. W., 2008, "Design and Analysis of a Hybrid Mobile Robot Mechanism With Compounded Locomotion and Manipulation Capability," J. Mech. Des., **130**(7), p. 72302.

[20] Ben-Tzvi, P., Goldenberg, A. A., and Zu, J. W., 2010, "Articulated Hybrid Mobile Robot Mechanism with Compounded Mobility and Manipulation and on-Board Wireless Sensor/actuator Control Interfaces," Mechatronics, **20**(6), pp. 627–639.

[21] Ben-Tzvi, P., Ito, S., and Goldenberg, A. a., 2008, "A Mobile Robot with Autonomous Climbing and Descending of Stairs," Robotica, **27**(2), p. 171.

[22] Ben-Tzvi, P., Goldenberg, A. A., and Zu, J. W., 2008, "Design, Simulations and Optimization of a Tracked Mobile Robot Manipulator with Hybrid Locomotion and Manipulation Capabilities," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2307–2312.

[23] Kawatsu, C., Li, J., and Chung, C. J., 2012, "Automatic Image Segmentation Using Saliency Detection and Superpixel Graph Cuts," Robot Intell. Technol. Appl., **208**(96), pp. 1023–1034.

[24] Borges, P. V. K., and Moghadam, P., 2014, "Combining Motion and Appearance for Scene Segmentation," Proc. - IEEE Int. Conf. Robot. Autom., pp. 1028–1035.

[25] Hoiem, D., Efros, A. A., and Hebert, M., 2011, "Recovering Occlusion Boundaries from an Image," Int. J. Comput. Vis., **91**(3), pp. 328–346.

[26] Xu, C., 2012, "Bag of Textons for Image Segmentation

via Soft Clustering and Convex Shift," Proc. 2012 IEEE Conf. Comput. Vis. Pattern Recognit., pp. 781–788.

[27] Roerdink, J., and Meijster, a, 2000, "The Watershed Transform: Definitions, Algorithms and Parallelization Strategies," Fundam. Informaticae, **41**(1–2), pp. 187–228.

[28] Haralick, R., Shanmugan, K., and Dinstein, I., 1973, "Textural Features for Image Classification," IEEE Trans. Syst. Man Cybern., **3**, pp. 610–621.

[29] Sachdeva, J., Kumar, V., Gupta, I., Khandelwal, N., and Ahuja, C. K., 2012, "A Novel Content-Based Active Contour Model for Brain Tumor Segmentation," Magn. Reson. Imaging, **30**(5), pp. 694–715.

[30] Comaniciu, D., and Meer, P., 2002, "Mean Shift: A Robust Approach toward Feature Space Analysis," IEEE Trans. Pattern Anal. Mach. Intell., **24**(5), pp. 603–619.