*American Nuclear Society 10th International Conference on Robotics and Remote Systems for Hazardous Environments*, Gainesville, Fl, March 28-31, 2004

# Obstacle Avoidance for the Segway Robotic Mobility Platform

Cang Ye†and Johann Borenstein*

*The University of Michigan, Advanced Technologies Lab, 1101 Beal Avenue, Ann Arbor, MI 48109-2110*
*† yecang@eecs.umich.edu * johannb@umich.edu*

**Abstract**—*This paper presents an obstacle avoidance system for the Segway Robotic Mobility Platform (RMP). The system consists of four main modules: terrain mapping, terrain traversability analysis, path planning, and motion control. The main sensor in our system is a forward/downward-looking 2-D Sick laser rangefinder. The terrain mapping module registers real-time laser range data into a grid-type elevation map. The traversal property of the elevation map is then analyzed by the traversability analysis module, which transforms the elevation map into a traversability map.The paper introduces a new concept called "traversability field histogram," which is used to transform the traversability map into a one-dimensional polar histogram. Finally, the path planning module determines the steering and velocity commands and sends them to the motion control module.*

## I. INTRODUCTION

Autonomous navigation on non-flat terrain requires the ability to decide whether a terrain feature can be traversed or whether it must be circumnavigated. This ability is usually termed Obstacle Negotiation (ON). A special case of ON is navigation on flat terrain, which we will refer to as "Obstacle Avoidance" in this paper. For the Segway Robotic Mobility Platform (RMP), the operating environment is usually flat ground. However, we want to enhance the navigation capability of the platform by allowing debris, small steps, or other irregularities to be traversable. This is in contrast to conventional OA systems, which treat any sensor discernable elevation on the ground as an obstacle and attempt to circumnavigate it. For simplicity we will still refer to the system described in this paper as an OA-type system, although it has the capabilities of an ON system.

In our thus-enhanced OA system there are two closely related issues—terrain mapping and path planning.

Most existing terrain mapping algorithms employ stereovision [1, 2, 3, 4], which has to cope with several challenges: (1) sensitivity to environmental conditions, (2) low range resolution and accuracy, and (3) range measurement errors that increase proportionally with the true range. To avoid these challenges altogether, robotics researchers have used 3-D Laser Rangefinders (LRFs) since the early nineties [5, 6, 7]. However, 3-D LRFs are costly, bulky, and heavy, and they are thus not suitable for small and/or expendable robots. Furthermore, the slow frame rate produced by 3-D LRFs can affect the robot's mobility if range data are acquired frame by frame.

To overcome these disadvantages, we propose a 2-D LRF-based terrain mapping method for the Segway RMP, similar to our earlier work [8, 9]. Our proposed method fully utilizes the accuracy of the laser range data for terrain mapping but avoids the slow frame rate associated with 3-D LRFs.

The specific focus of this paper is on real-time path planning, that is, determining the robot's motion based on real-time sensor data. Path planning for non-flat terrain must address two problems: Terrain Traversability Analysis (TTA) and path generation. In our system we use a grid-type representation of the terrain, and the task of our TTA module is to assign so-called Traversability Indices (TIs) to all relevant cells in a local terrain map. TIs represent the overall traversal property of the cell with reference to the surrounding terrain. Similar approaches are described in the literature [4, 10, 11]: in [10] TIs are represented in binary form, [11] uses continuous numeric values, and in [4] TIs are described by fuzzy sets.

The other task of path planning is the generation of steering and velocity command for the robot. This task is sometimes achieved by what the scientific literature calls "behavior control." A behavior-based OA system [4, 10, 12] usually comprises a number of behavior modules with different functionality, e.g., an obstacle avoiding behavior or a goal seeking behavior [12]. Each of them performs its own task without considering the functionality of the others. As the actions determined by different modules may be conflicting with each other, a behavior arbitration mechanism is used to resolve the conflicts and to determine the action for the robot. This approach usually produces good real-time performance. However, the convergence of such algorithms is hard to prove.

Some researchers employed the so-called "arcs approach" [3, 10, 13, 14]. In the arcs approach the algorithm generates a number of candidate arcs and then either votes for the arc with the largest clearance [10] from obstacles or calculates the costs along each arc and selects the one with the lowest cost [3, 13, 14]. The robot is then steered along the winning arc. In [14] the cost function is the sum of traversability values along each arc. Here the D* algorithm is used to minimize the cost. Gennery's path planner [11] computes the cost of driving through a grid. The cost function comprised two components: the distance traveled and the probability that

the slope or roughness may be too large to traverse. A path planner is then employed to find a path that minimizes the total cost. The probability approach takes into account the inaccuracy of stereovision data.

There are some attempts to apply potential field methods to terrain navigation [15]. However, the application of potential field methods to OA may result in oscillatory motion [16] or cause the robot to get trapped in a local minimum. The Vector Field Histogram (VFH) method [17], developed by one of the co-authors of this paper, overcomes the limitations of potential field methods. The VFH algorithm has been demonstrated to produce non-oscillatory, fast obstacle avoidance for a variety of mobile robots at our lab and at other research labs.

In this paper, we modify and extend the capabilities of the VFH algorithm and apply the modified algorithm to terrain navigation. The paper is organized as follows: In Section II, we provide a brief overview of our OA system. In Sections III we detail our Terrain Mapping method. Then in Section IV we describe the Traversability Analysis algorithm and our Path Planning algorithm. In Section V, we present experimental results of our terrain mapping method and simulation results of the overall OA algorithm. Section VI concludes the paper.

## II. Overview of the Obstacle Avoidance System

Figure 1 shows a block diagram of our OA system. It consists of four main modules: Terrain Mapping, Terrain Traversability Analysis (TTA), Path Planning, and Motion Control.

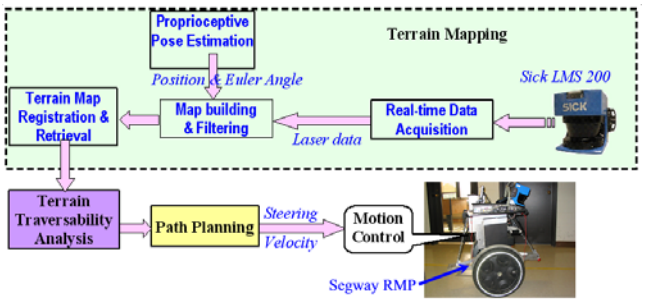



Fig. 1. Diagram of the obstacle avoidance system: the module within the dashed lines represents the Terrain Mapping module.

The main sensor in our system is the SICK 2-D laser rangefinder (LRF), and the configuration described here is with the LRF mounted on the front end of the Segway RMP. The LRF looks forward and downward at the terrain at an angle of -10° from the horizon. With this configuration, the LRF has a look-ahead distance of 5 meters. While the robot is in motion, the fanning laser beams sweep the terrain ahead of the robot and produce continuous range data of the terrain. The terrain data is then transformed into coordinate values in the world coordinate system using the robot pose information from the Proprioceptive Pose[1] Estimation (PPE) system, and registered in a map, which is represented by a 2-dimensional (2-D) array or "grid." Each element or "cell" in the map holds a value representing the height of the terrain/obstacle at that cell. Such a grid-type map is also called a "2½-D map," "elevation map" or simply a "terrain map" [8].

The task of the TTA module is to analyze the terrain map cell by cell and to generate a new 2-D grid-type map, called "Traversability Map." Each cell in that map holds a value that expresses the degree of difficulty for the robot to move across that cell. This value is called "Traversability Index" (TI).

The Path Planning module is a *local* path planner. It analyzes the traversability map and generates steering and velocity commands to avoid cells with high TIs.

## III. Terrain Mapping

### 1. The Segway RMP

As depicted in Figure 2, the main components for terrain mapping are the PPE system and the LRF. The Segway RMP is a dynamically self-balancing platform. In order to move forward or backward, the robot leans its body forward or backward. This behavior, adopted for the RMP platform from the popular Segway Human Transport (HT) platform, causes the robot accelerate forward or backward in order to maintain balance. Since much of the drive control involves leaning forward or backward, the platform and thus the sensor have non-zero and varying pitch angles even on flat ground. This unique behavior has certain advantages for our statically mounted 2-D LRF, namely it provides the LRF a vertically wider field of view. On the other hand, the pitching motion requires a very accurate pose estimation system. In our case this function is provided by in-house developed FLEXnav system [18]. Another helpful property of the Segway is that its footprint is roughly circular and it can turn on the spot. This feature simplifies path planning and obstacle avoidance.

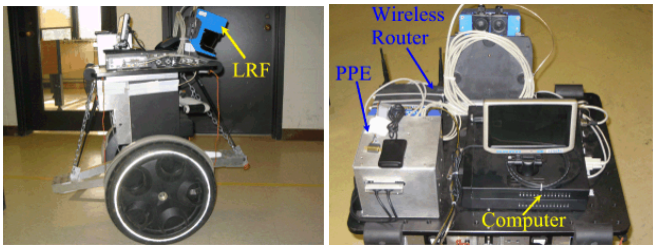



Fig. 2. The Segway RMP: The main components for terrain mapping are the PPE and the LRF, which are connected to the on-board computer. The wireless router is used for the access of the off-board computer to the mapping system.

### 2. Terrain mapping method

Terrain mapping is the process of transforming range measurements in the LRF's coordinate system to 3-D points in the world coordinate system (the latter is also called "navigational frame"). In our work we use homogeneous coordinate transformation for this purpose. Figure 3 depicts

[1] "Proprioceptive" in the context here means based on dead-reckoning only, i.e., without external references. "Pose" is the set of three Cartesian coordinates and the three attitude angles.

the coordinate systems used for terrain mapping. $x_by_bz_b$ is the coordinate system fixed to the robot's body with $y_b$ pointing forward, $x_b$ pointing right and $z_b$ pointing upwards. This coordinate system is called "robot body frame." The navigational frame $x_ny_nz_n$ is aligned with the robot body frame when the PPE is initialized. This frame is constant and fixed to the world coordinate system after the initialization. All other frames are attached to the components installed on the robot and they are moving with the robot body. The three Euler angles are defined as follows: roll ($\phi$), pitch ($\theta$) and yaw ($\psi$) are the rotation angles around $y_b$, $x_b$, and $z_b$ axes, respectively.

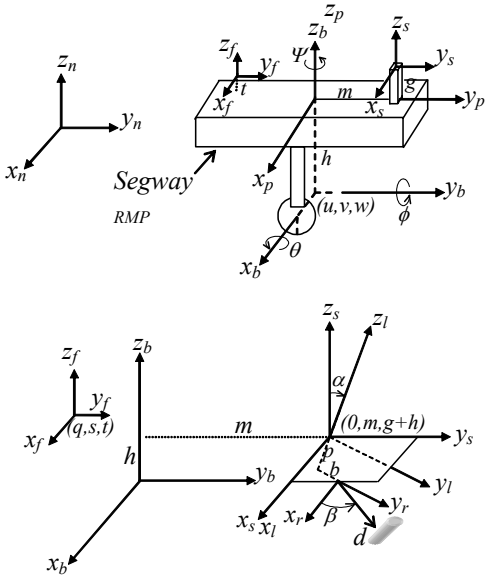
The transformation matrix $T_r^n$ of the homogeneous transformation can be derived through successive rotations and translations from $x_ry_rz_r$ to $x_ny_nz_n$. For conciseness, we omit this derivation here. $T_r^n$ is then used to transform a range datum in the laser measurement frame $x_ry_rz_r$ to a 3-D point in the navigational frame $x_ny_nz_n$ as follows:

$$d^n = T_r^n d^r = \begin{bmatrix} x_n & y_n & z_n & 1 \end{bmatrix}^T, \quad (1)$$

where $d^n$ and $d^r$ are the range measurement in the navigational frame and the laser measurement frame, respectively, and $x_n$, $y_n$, and $z_n$ are the coordinate values in the navigational frame. Once $x_n$, $y_n$, and $z_n$ are determined, the elevation value $h(i, j)$ of the corresponding cell *(i, j)* in the elevation map is updated by

$$h(i,j) = \begin{cases} z_n & \text{if } z_n > h(i,j) \\ h(i,j) & \text{otherwise} \end{cases}. \quad (2)$$

In our system, we also build a certainty map which represents the same space as the elevation map except that each cell in the certainty map represents the number of times the terrain represented by that cell was illuminated by LRF's light beam. The certainty map is used to filter corrupted range data. Details of the filter and the use of the certainty map are explained in [8]. The real-time data from the PPE and the LRF are acquired frame-by-frame by the onboard computer and registered in the terrain map while the robot is in motion.




$x_ny_nz_n$: navigational frame
$x_by_bz_b$: robot body frame
$x_fy_fz_f$: sensor base frame
$x_sy_sz_s$: LRF mounting set frame
$x_ly_lz_l$: LRF frame
$r_ny_nz_r$: laser measurement frame

Fig. 3. The coordinate systems for the Segway RMP terrain mapping

## IV. Obstacle Avoidance

As mentioned earlier, obstacle avoidance is implemented through the use of two functional modules: terrain traversability analysis and path planning. We discuss these modules in detail in this Section.

### *1. Terrain traversability analysis*

The terrain traversability analysis (TTA) module assigns a TI value to each cell in the terrain map, thereby transforming the terrain map into a traversability map. Let us assume we have a terrain map $\mathbf{E} = \{z_{i,j}\}$, where $i$ and $j$ are the row and column cell indices, respectively, and the Robot Center Point[2] (RCP) is located at $(x_i, y_j, z_{i,j})$ in the terrain map, where $x_i$ and $y_j$ are the coordinates corresponding to cells (*i, j*). We then define a square terrain patch $P = \{z_{k,l} \mid_{k=i-L,\cdots,i+L;\, l=j-L,\cdots,j+L}\}$. $P$ has a side length of $2L+1$ (in terms of number of cells) and it is centered at cell (*i, j*). $L$ is chosen in such a way that the terrain patch completely envelops the robot regardless of the robot's orientation. Each cell in the grid-type map has a size[3] of 250 mm × 250 mm. In order to account for the footprint of the Segway RMP, we selected $L$=2. A plane is then fitted to the terrain patch using the Least Square Error method and the normal **n** to the fitted plane is computed. The slope of the terrain patch is then estimated by

$$\alpha = \cos^{-1}(\mathbf{n} \cdot \mathbf{b}), \quad (3)$$

where $\mathbf{b} = (0,0,1)^T$; and the roughness of the terrain patch is approximated by

$$\sigma = |\mathbf{d}| = \sqrt{\sum_{i=1}^{i=N} d_i^2}, \quad (4)$$

where $|\mathbf{d}|$ is the Euclidean norm of the residual of the fit **d** and $d_i$ is the distance between the $i^{th}$ data and the fitted plane. Finally, cell (*i, j*) is assigned a TI value by

$$\tau_{i,j} = |F_1\alpha| + |F_2\sigma / N|. \quad (5)$$

$F_1$ and $F_1$ are empirically chosen and their value represents the contribution of the terrain slope and roughness to the TI value. In this study, $F_1$=*300* and $F_2$=*6* were used, which results in the slope having a slightly larger contribution than the roughness. The traversability analysis is performed cell-by-cell in the local terrain map. Once each cell is assigned a TI value, the terrain map **E** has been effectively transformed

[2] For simplicity we placed the RCP at the geometric center of the robot.
[3] In case a finer resolution is required for map rendering, a smaller grid size may be used terrain mapping. Under this condition, cell may be merged to form the 250 mm × 250 mm grid for obstacle avoidance

into a traversability map $\mathbf{T} = \{\tau_{i,j}\}$.

## 2. Path Planning Using the Traversability Field Histogram

The VFH method was proposed in [17] for fast obstacle avoidance. In this paper, we employ the VFH concept and propose a new local path planning method, the Traversability Field Histogram (TFH) algorithm for general OA. The TFH method extends the VFH's capability from OA on flat ground to ON on non-flat terrain. It should be noted that conventional 2-D obstacle avoidance is a special case under the TFH framework.

The MGTFH algorithm first forms a square-shaped local terrain map $S^*$, which overlays the global terrain map **E** like a window. There are $w_s \times w_s$ cells in $S$*, and their size is the same as that of cells in **E**. $S$* centers at the momentary position of the RCP, and we chose $w_s$=37 in our implementation. As a result of this choice, $S^*$ covers a physical region of 9.3 × 9.3 m (37×250 mm = 9.3 m).

Next, the MGTFH algorithm performs the terrain traversability analysis (described in the preceding section) over the cells in $S^*$ and transforms the local terrain map into a traversability map.

After the above preprocessing, the TFH algorithm computes the motion command based on the traversability map inside $S$* using two stages: (1) creation of a one-dimensional polar histogram, and (2) generation of the motion command. These two stages are explained next.

## 3. Creation of the Polar Histogram

Figure 4 is the 2-D representation of a traversability map where the TI value is rendered in color (or in gray level in a grayscale illustration). The cells with nonzero TI values in the traversability map generate an imaginary vector field, which exerts virtual repulsive forces on the robot and repels it away from an untraversable region. We call this field "traversability field" and we call the polar histogram **H** "traversability field histogram." Since we are extending the VFH method [17] to handle the general OA problem, we adopt the same terms, "obstacle vector" and "polar obstacle density" as in [17]. They are defined in this section.

The contents of each cell in the traversability map is treated as an obstacle vector [17] whose direction is calculated by

$$\beta_{i,j} = \tan^{-1}\frac{y_j - y_o}{x_i - x_o} \tag{6}$$

and the magnitude is given by

$$m_{i,j} = \tau_{i,j}^2 (a - b d_{i,j}), \tag{7}$$

where

$\beta_{i,j}$ – direction from cell $(i, j)$ in $S^*$ to the RCP,

$x_i, y_j$ – coordinates of cell $(i, j)$,

$x_o, y_o$ – present coordinates of the RCP,

$m_{i,j}$ – magnitude of the obstacle vector at cell $(i, j)$.

$\tau_{i,j}$ – TI value of cell $(i, j)$,

$a, b$ – positive constants (see below)

$d_{i,j}$ – distance between cell $(i, j)$ and the RCP,

In equation (7), $m_{i,j}$ is proportional to $-d$. Therefore, cells with non-zero TI values result in larger vector magnitudes when they are closer to the robot and smaller ones when they are farther away. It should be noted that $a$ and $b$ are chosen such that $a - bd_{\max} = 0$, where $d_{\max} = \sqrt{2}(w_s - 1)/2$ is the distance between the farthest cell of $S$* and the RCP. This arrangement guarantees $m_{i,j} = 0$ for the four farthest cells (the four vertices) of $S^*$.

As shown in Fig. 4, $S^*$ is divided into $n$ sectors, each of which has an angular resolution of $\alpha$ ($\alpha$ is chosen such that $360/\alpha$ is an integer). Each sector $k$, for $k = 0, \cdots, n-1$, corresponds to a discrete angle $\rho = k\alpha$ in **H**. The cells $(i, j)$ in $S$* are assigned to the $k^{th}$ sector according to

$$k = \text{int}(\beta_{i,j}/\alpha). \tag{8}$$

For each sector $k$, the polar obstacle density $h_k$ is then calculated by

$$h_k = \sum_{i,j} m_{i,j}. \tag{9}$$

In our implementation $\alpha$ = 5°, therefore, there are 72 sectors. Polar obstacle densities produced by this equation are discrete, hence **H** is discontinuous. This may lead to drastic change in motion since **H** is used to determine the robot's steering and velocity. In order to alleviate this problem, we use the following function to smooth the polar obstacle density:

$$h_k' = \frac{\sum_{l=-p}^{p} h_{k-l}}{2p+1}. \tag{10}$$

where $h_k'$ is the Smoothed Polar Obstacle Density (POD). The parameter $p$ determines how much the polar histogram

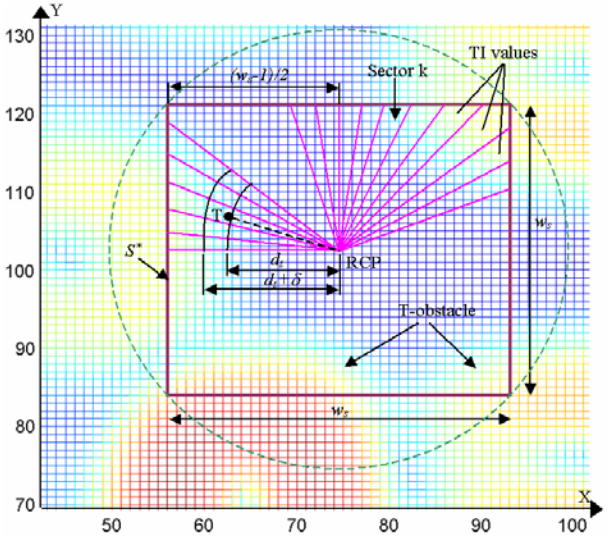



Fig. 4. Transformation of a traversability map into a polar histogram: X and Y axes represent the coordinates in the navigational frame.

is smoothed, and we found that $p$=3 produces the best results.

The POD of each sector represents the level of difficulty of moving in the corresponding direction. Figure 5 shows the polar histogram **H** for the momentary situation where the RCP is located at cell 'q'. The parallelepiped object produced large PODs in sectors 1-14 and sectors 64-72 (i.e., -40°–70°) because it is high in elevation. Cone 1 generated small PODs in sectors 19-30 (i.e., 95°–150°) since its low-profile tip faced the robot, whereas cone 2 produced relatively larger PODs in sector 47-60 (i.e., 235°–300°) because its base (with its higher elevation) faced the robot. This exemplifies how **H** correctly reflects the overall traversability of each sector in region S*.

### 4. *Steering Control*

When the robot travels on terrain, the polar histogram **H** changes from instance to instance. The objective of the OA algorithm is to steer the robot toward a direction with lower PODs, while still maintaining a direction that is close to the target direction. As can be seen from Fig. 5, a polar histogram typically has "hills" (sectors with high PODs) and "valleys" (sectors with low PODs). A "candidate valley" is a cluster of consecutive sectors with PODs below a certain threshold (e.g., Fig. 5 has two candidate valleys). The candidate valley, which is eventually used to determine the robot's next heading direction, is called the "*winning valley*."

Assuming that the robot's target is $(x_t, y_t)$, the sector number $k_t$ of the target vector is calculated by Eq. (6) and (8) ($x_i$ and $y_i$ are replaced by $x_t$ and $y_t$ in this case). We denote the $i^{th}$ candidate valley by $v_i = [k_i^R, k_i^L]$ where $k_i^R$ and $k_i^L$ represent the sector number of the right and left border of candidate valley $v_i$, respectively. The width of the valley, $s_i$, is $s_i = k_i^L - k_i^R + 1$ sectors. A valley is called a "wide valley" if $s \geq s_{\max}$ (in our system $s_{\max} = 12$), otherwise, we call it a "narrow valley." We denote the winning valley by $v_w$. The robot's next heading direction is then defined as

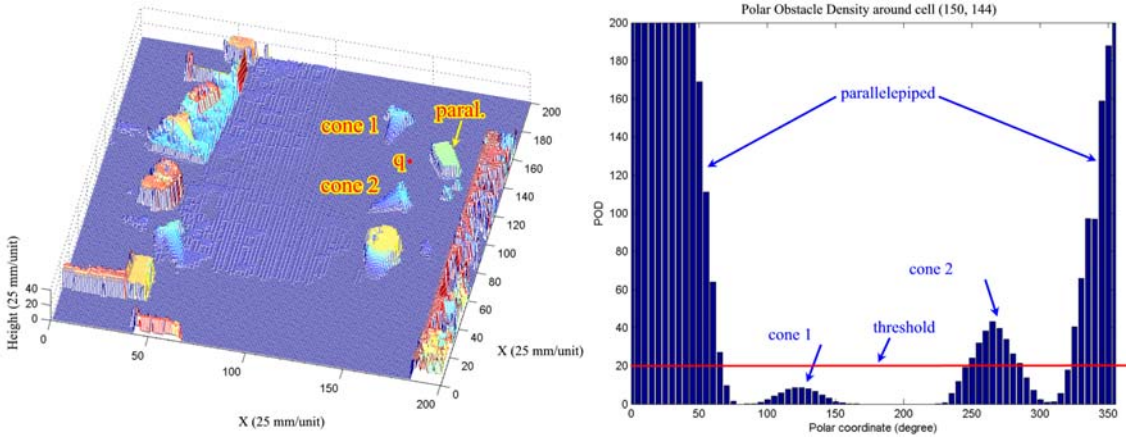
$$k_h = \begin{cases} k_w^R + \min(s_w, s_{\max})/2 & \text{if } |k_t - k_w^R| \leq |k_t - k_w^L| \\ k_w^L - \min(s_w, s_{\max})/2 & \text{otherwise} \end{cases}. \quad (11)$$

where $k_w^R$ and $k_w^L$ represents the sector number of the right and the left border of $v_w$, respectively. The border determining the next heading direction is called the winning border, and $s_w$ is the width of $v_w$. If the winning valley is narrow, the robot is directed to steer into the direction of the center of the valley. If the winning valley is a wide one, then a sector $s_{max}/2$ sectors from the winning border is selected as the new steering direction. Treating the final selection of a steering direction as a spatial problem within the polar histogram eliminates the oscillatory behavior inherent in potential field methods [16] and represents the major distinction between those methods and the VFH method. However, the steering command determined by Eq. (11) may cause the robot to circle the obstacle and thus miss the target. Therefore, we need an exit condition. Once the exit condition is met, the robot will be forced to move straight toward the target direction. In order to do this, we define the target direction to be "free" if each sector of the cluster of sectors $[k_t - s_{\max}/2, k_t + s_{\max}/2]$ lies in a candidate valley, i.e., the POD of each sector is smaller than the threshold value. The steering control algorithm first checks whether the target direction is free. If so, then the robot's next heading is pointed at the target direction. Otherwise, the steering scheme in Eq. (11) is applied.

### 5. *Speed Control*

The robot runs initially at its maximum speed $v_{max}$ (1 m/s in our simulator). The robot will try to maintain this speed unless forced by the MGTFH algorithm to a lower value, which is determined at each time step as follows:

The POD in the current direction of travel is denoted as $h_c^{'}$. $h_c^{'} > 0$ indicates that rugged terrain lies ahead of the robot. A large value of $h_c^{'}$ suggests that an obstacle lies ahead. In either case a reduction in speed is in order. In our system speed is reduced in inverse proportion to the POD




(a) Terrain map (b) Polar histogram at 'q'

Fig. 5. Terrain map and the corresponding polar histogram when the robot was at position 'q'.

value in the momentary direction of travel:

$$v' = v_{\max}(1 - \frac{\min(h_c', h_m)}{h_m}) . \quad (13)$$

$h_m$ is a constant, which is empirically determined to produce a sufficient reduction in speed. A reduction in speed is also required when the robot approaches the target. In our current simulator, we further reduce the speed by

$$v'' = \frac{v' \min(d_t, d_m)}{d_m} , \quad (14)$$

where $d_t$ is the distance between the target and the RCP, while $d_m$ is a constant (we used 1.5 m in our simulator). Note that $v''$ is actually the projection of the robot speed $V$ on the X-Y plane. It produces a movement $l_{xy}$ in the X-Y plane along the robot's next heading, and moves the RCP to $(x_o', y_o')$. We then calculate the robot's speed by

$$V = \frac{v'' l_{xyz}}{l_{xy}} , \quad (15)$$

where $l_{xyz}$ is the 3-D distance between the current RCP position at $(x_o, y_o, z_o)$ and the next RCP position at $(x_o', y_o', z_o')$.

## V. Experimental and Simulation Results

At the time of submission of this paper, we had implemented the map-building component of the described system on the real Segway RMP. However, the real-time path planning component was not yet sufficiently debugged to allow inclusion of real-world results from that component in this paper. Therefore, we present in this section real-world experimental results obtained with our terrain mapping method and the simulation results of our OA algorithm.

### *1. Experimental results from the map-building component*

Map-building experiment was carried out in the basement of the Advanced Technologies Lab at the University of Michigan. The terrain mapping was running with the robot moving in the building. The data from the PPE system and the LRF are acquired and registered into terrain maps in real-time. We ran numerous tests and all maps built by our system are accurate. This reveals the accuracy of the PPE data and the correctness of the terrain mapping method. Figure 6 depicts the bird view of one of the map. The robot move along the trajectory from 's' to 'g'.

Figures 7, 8 & 9 show the correspondence between the scenes and the maps when the robot was moving toward the door way, moving in the hallway, and moving towards the stair, respectively. The accurate correspondence relationships demonstrate once again the accuracy of our mapping system.

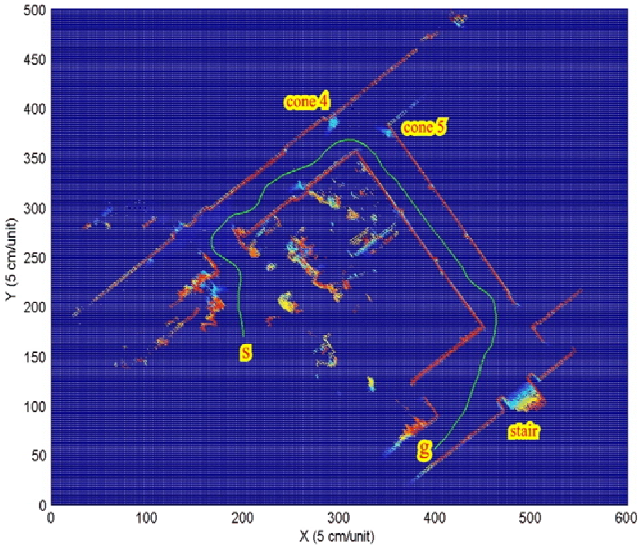
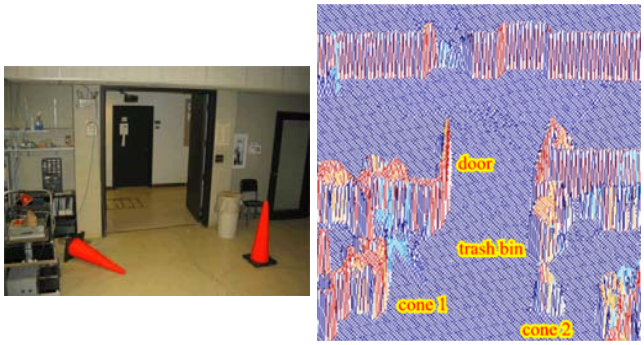
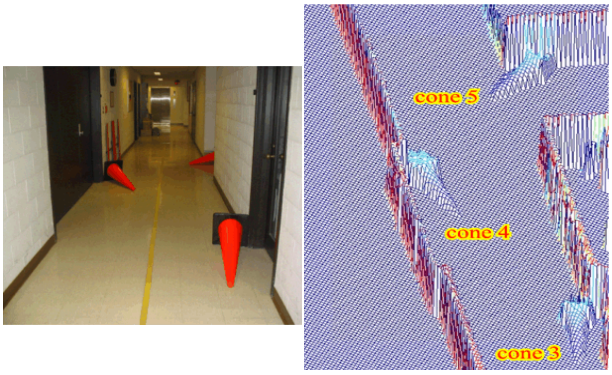



Fig. 6. Bird view of the map built by the terrain mapping method



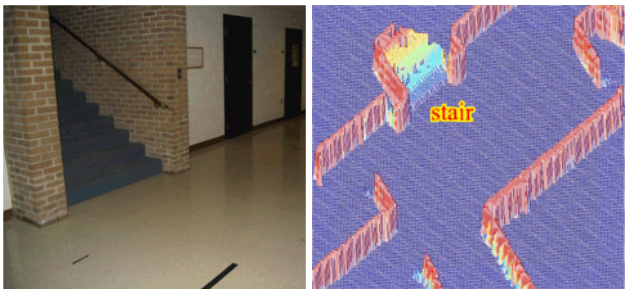

(a) Scene (b) Map

Fig. 7. The robot was moving towards the doorway




(a) Scene (b) Map

Fig. 8. The robot was moving in the hall way



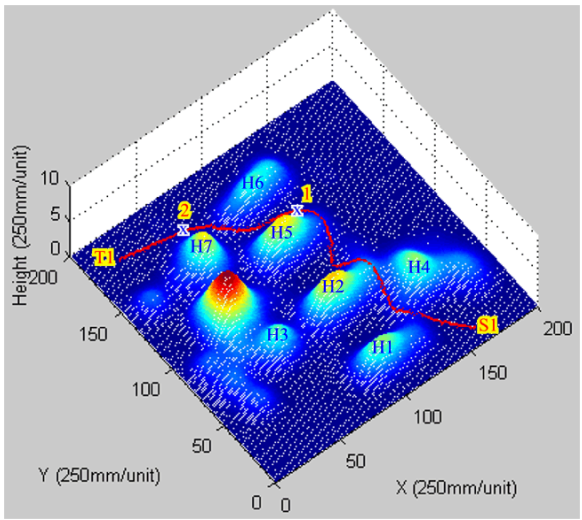

(a) Scene (b) Map

Fig. 9. The robot was moving toward the stair

*2. Simulation of the OA algorithm*

The Segway RMP is at greater risk of being damaged as a result of a collision than conventional mobile robots. This is because of its potential to tip over if an obstacle interferes with the robot's ability to accelerate/decelerate as is needed for maintaining balance. For this reason we developed a particularly detailed simulator, which allows us to test a large variety of environments and driving conditions.

As shown in Fig. 10, the simulator can visualize the OA process in two windows with one showing the robot maneuvering on the terrain (Fig. 10a) and the other one rendering the polar histogram (e.g. Fig. 10b & c) at each robot position. At any given step in time the polar histogram may be used to analyze whether the OA algorithm correctly determined the steering command. Fig. 10a depicts the terrain and the path determined by the OA algorithm for the navigation task from 'S1' to 'T1'. Fig. 10b displays the polar histogram



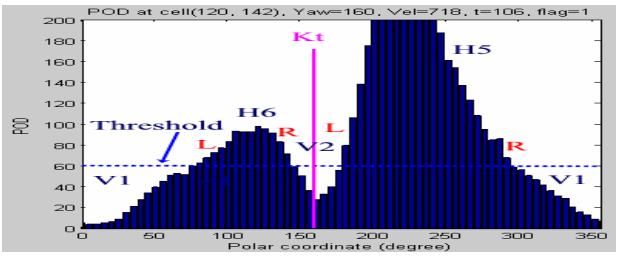

(a) Terrain map and the real-time path generation

Fig. 10 A navigation task from 'S1' to 'T1': Kt – target sector; L/R – left / right bo – yaw (heading) angle.

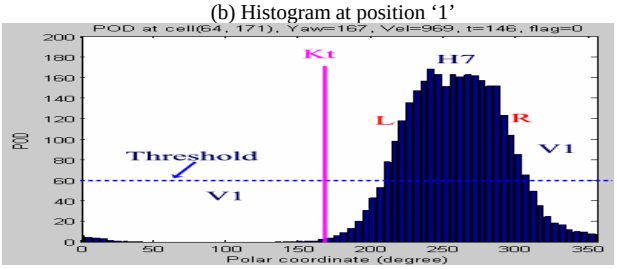
when the robot was at position '1'. Hills 'H5' and 'H6' in the terrain map created two peaks in the polar histogram (Fig. 10b) which are also labeled as 'H5' and 'H6', respectively. As the target sector fell into the narrow valley 'V2', the robot moved in the middle of the passage between hills 'H5' and 'H6' with a heading angle of 160°. When the robot arrived at position '2', hill 'H7' in the terrain map created peak 'H7' in the polar histogram (Fig. 10c). Valley 'V1' in this case is a wide one. Since the target direction is free, the robot moves directly toward the target and finally came to a full stop at 'T1'.

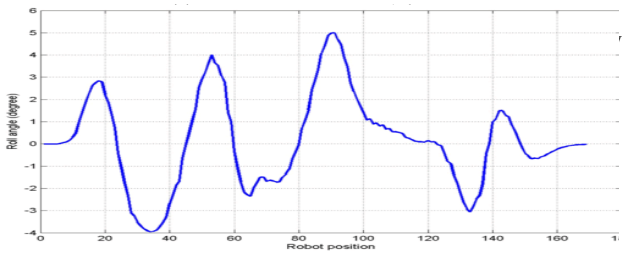
Since the proposed OA algorithm steers the robot away from untraversable regions, the roll and pitch angles of the terrain patch where the robot is located can be used to evaluate the performance of the algorithm. Fig. 11 plots the roll and pitch angles of the terrain patch where the robot was located in navigation task from 'S1' to 'T1'. As Fig. 11 shows, the roll and pitch angles are contained within [-4°, 5°] and [-6.5°, 6°], respectively. This suggests that our OA algorithm maintained a path of gently rolling terrain, while avoiding steep slopes. We performed numerous test runs with different terrains and with different starting/stopping locations. In all cases, the OA algorithm successfully navigated the robot.

To verify the claim that the OA algorithm is also able to handle conventional OA problems on 2-D flat ground, we created a number of obstacle courses in our simulator to test the OA algorithm. In all cases, our OA algorithm guided the robot along collision-free paths. Figure 12 is an example where the robot avoided a concave obstacle on flat ground. Besides confirming the OA algorithm's ability to handle conventional obstacle avoidance, the example also reveals that our algorithm is able to deal with symmetrical concave obstacle where the potentially field method may fail due to the problem of local minima.

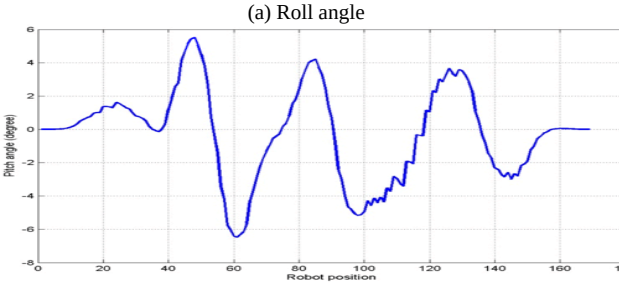



(b) Histogram at position '1'




(a) Roll angle




(a) Pitch angle

Fig. 11. Roll and pitch angles of the robot in the navigation task from S1 to T1 in Fig. 10a

## VI. CONCLUSIONS

In this paper we presented a novel obstacle avoidance

method for mobile robot navigation on 2-D and on moderately rugged 3-D terrain. The proposed method introduces a

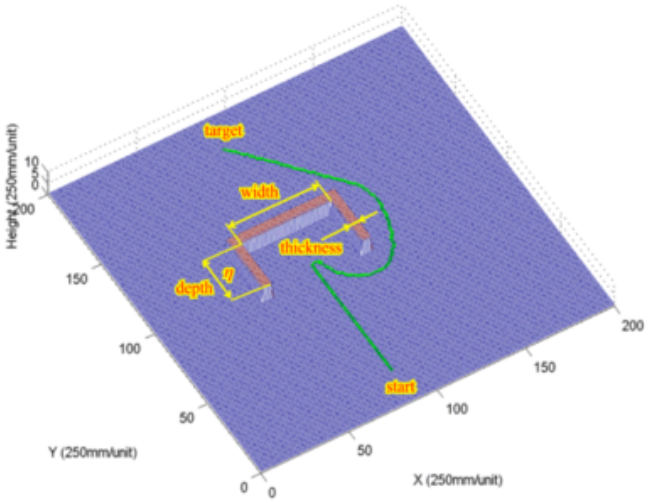



Fig. 12 A concave obstacle on the flat ground: depth=29 cells, width=51 cells, thickness=5 cells.

mapping system capable of building 3-D terrain maps with a 2-D LRF. The method utilizes the robot's motion to sweep the 2-D LRF's fanning laser beam to produce 3D data of the terrain surface.

The map building component is complemented by an obstacle avoidance algorithm based on the new concept of the Traversability Field. The algorithm guides the robot over moderate terrain along a collision-free path. The obstacle avoidance algorithm is capable to handle navigation problems on both flat ground surface and rough terrain. Currently, the algorithm has been tested on a large number of simulated 3-D terrains and on flat ground with simulated obstacles. We are currently implementing the algorithm on our Segway RMP and will begin real-world experimentation in early 2004.

## ACKNOWLEDGMENTS

This work was funded by the U.S. Department of Energy under Award No. DE-FG04-86NE3796, by DARPA under Grant #F007571, and by the University of Michigan's Automotive Research Center (ARC), which is supported by TACOM.

## REFERENCES

1 S. Betgé-Brezetz, et al, "Uncertain map making in natural environments," *Proc. IEEE International Conference on Robotics and Automation*, 1996, pp. 1048-1053.

2 D. S. Apostolopoulos, et al, "Technology and field demonstration of robotic search for Antarctic meteorites," *International Journal of Robotics Research*, vol. 19, no. 11, pp. 1015-1032, 2000.

3 R. Simmons, et al, "Experience with rover navigation for lunar-like terrains," *Proc. IEEE/RSJ International Conference on Intelligent Robots and System*, 1995, pp. 441-446.

4 H. Seraji and A. Howard, "Behavior-based robot navigation on challenging terrain: a fuzzy logic approach," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 3, 2002.

5 I. S. Kweon and Takeo Kanade, "High-resolution terrain map from multiple sensor data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 278-292, 1992.

6 E. Krotkov and R. Hoffman, "Terrain mapping for a walking planetary rover," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 728-738, 1994.

7 C. M. Shoemaker and J. A. Bornstein, "The demo III UGV program: a testbed for autonomous navigation research," *Proc. IEEE ISIS/CIRA/ISAS Joint Conference*, 1998, pp. 644-651.

8 C. Ye and J. Borenstein, "A new terrain mapping method for mobile robots obstacle negotiation," *Proc. the UGV Technology Conference at the 2003 SPIE AeroSense Symposium*, Orlando, FL, April 21-25, 2003.

9 C. Ye and J. Borenstein, "A novel filter for terrain mapping with laser rangefinders," conditionally accepted for publication in the *IEEE Transactions on Robotics and Automation*, 2003.

10 D. Langer, et al, "A Behavior-based system for off-road navigation," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 776-783, 1994

11 D. B. Gennery, "Traversability analysis and path planning for a planetary rover," *Autonomous Robots*, vol. 6, no. 2, pp. 131-146, 1999.

12 C. Ye and D. W. Wang, "A novel navigation method for autonomous mobile robots," *Journal of Intelligent and Robotic Systems*, vol. 32, no. 4, pp. 361-388, 2001.

13 S. Singh, et al, "Recent progress in local and global traversability for planetary rovers," *Proc. IEEE International Conference on Robotics and Automation*, 2000, pp. 1194-1200.

14 S. Lacroix, et al, "Autonomous Rover Navigation on Unknown Terrains: Functions and Integration," *International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 917-942, 2002.

15 H. Haddad, et al, "Reactive navigation in outdoor environments using potential fields," *Proc. IEEE International Conference on Robotics and Automation*, 1998, pp. 1232-1237.

16 Y. Koren and J. Borenstein, "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation," *Proc. IEEE International Conference on Robotics and Automation*, 1991, pp. 1398-1404.

17 J. Borenstein and Y. Koren, "The Vector Field Histogram–Fast Obstacle-Avoidance for Mobile Robots," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278-288, 1991.

18 L. Ojeda, M. Raju and J. Borenstein, "FLEXnav: A Fuzzy Logic Expert Dead-reckoning System for the Segway RMP." *Proc. of the Defense and Security Symposium (was Aerosense) Unmanned Ground Vehicle Technology VI (OR54)*, Orlando FL, April 12-16 , 2004.