Proceedings of the 1998 IEEE
International Conference on Robotics & Automation
Leuven, Belgium • May 1998

VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots

Iwan Ulrich and Johann Borenstein

The University of Michigan, Advanced Technology Laboratories
1101 Beal Avenue, Ann Arbor, MI 48109
iwan@ri.cmu.edu, johannb@umich.edu

ABSTRACT

This paper presents further developments of the earlier Vector Field Histogram (VFH) method for real-time mobile robot obstacle avoidance. The enhanced method, called VFH+, offers several improvements that result in smoother robot trajectories and greater reliability. VFH+ reduces some of the parameter tuning of the original VFH method by explicitly compensating for the robot width. Also added in VFH+ is a better approximation of the mobile robot trajectory, which results in higher reliability.

1. INTRODUCTION

The *VFH+* method is an improved version of the *Vector Field Histogram* (VFH) method originally developed by Borenstein and Koren for real-time local obstacle avoidance with mobile robots [2]. VFH+ was developed for a special type of mobile robot called the *GuideCane*. The GuideCane, shown in Figure 1, is a novel guidance device for the blind. In operation, a blind user pushes the unpowered GuideCane ahead of himself. When the GuideCane encounters an obstacle it steers around it. The user feels the changing orientation of the GuideCane handle and can follow the device intuitively and without any conscious effort [3].

Because of the similarity in function between the GuideCane and conventional mobile robots, the VFH+ obstacle avoidance method is equally well applicable to other mobile robots. The VFH+ algorithm was extensively tested in simulation and with the real GuideCane in unstructured and unknown environments.

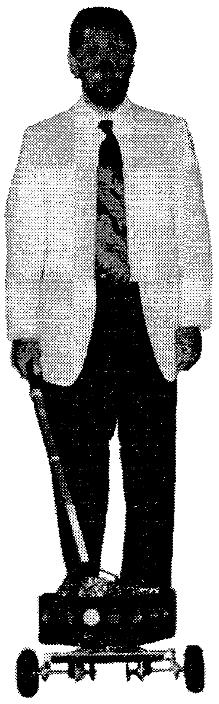
Figure 1: A blindfolded experimenter walks with the GuideCane.

2. THE VFH+ ALGORITHM

The concept of the VFH+ obstacle avoidance algorithm is similar to the original VFH algorithm. The input to this algorithm is a map grid of the local environment, called histogram grid [1], which is based on the earlier certainty grid [7] and occupancy grid [4] methods.

The VFH+ method employs a four-stage data reduction process in order to compute the new direction of motion. In the first three stages, the two-dimensional map grid is reduced to one-dimensional *polar histograms* that are constructed around the robot's momentary location. In the fourth stage, the algorithm selects the most suitable direction based on the masked polar histogram and a cost function. The following sections briefly summarize each stage.

2.1 First Stage - The Primary Polar Histogram

A more detailed description for the first part of this stage is given in [2]. The first data reduction stage maps the active region C_a of the map grid C onto the *primary polar histogram* H^p. The active region C_a is a circular window of diameter w_s that moves with the robot. The content of each active cell in the map grid is treated as an obstacle vector. Based on the reference system of Figure 4, the vector direction $\beta_{i,j}$ is determined by the direction from the active cell to the robot center point (*RCP*):

$$\beta_{i,j} = \arctan\left(\frac{y_j - y_o}{x_i - x_o}\right) \tag{1}$$

where:

x_0, y_0: Present coordinates of the *RCP*.
x_i, y_j: Coordinates of active cell $C_{i,j}$.

The vector magnitude $m_{i,j}$ for an active cell $C_{i,j}$ is given by:

$$m_{i,j} = c_{i,j}^2\left(a - b d_{i,j}^2\right) \tag{2}$$

where:

$c_{i,j}$: Certainty value of active cell $C_{i,j}$.
$d_{i,j}$: Distance from active cell $C_{i,j}$ to the *RCP*.

For $m_{i,j}$ to be equal to the square of $c_{i,j}$ at the boundary of the active region, the parameters a and b are chosen according to:

$$a - b\left(\frac{w_s - 1}{2}\right)^2 = 1 \tag{3}$$

Note that $c_{i,j}$ is squared in equation 2. This expresses our confidence that recurring range readings (high $c_{i,j}$ values) represent actual obstacles, as opposed to single occurrences of range readings (low $c_{i,j}$ values) which may be caused by noise. The vector magnitude is also a function of the squared distance $d_{i,j}$, so that occupied cells produce larger vector magnitudes when they are close to the robot.

A convenient property of this magnitude function is that it is rotationally symmetric with respect to the *RCP*. As a result, the robot's behavior is independent of the direction in which obstacles are encountered.

Based on the obstacle vectors, the primary polar histogram H^p is built. H^p has an arbitrary angular resolution α so that $s = 360°/\alpha$ is an integer. In our implementation, α is set to 5°, resulting in $s = 72$ angular sectors. Each angular sector k corresponds to a discrete angle $\varphi = k \cdot \alpha$.

The original VFH method does not explicitly take into account the width of the robot. Instead, it uses an empirically determined low-pass filter to compensate for the robot width and to smooth the polar histogram. The tuning of this filter is the main difficulty in implementing the original VFH algorithm [6]. However, even with a well-tuned filter, the robot has a tendency to cut corners.

The VFH+ method, by contrast, uses an analytically determined low-pass filter to compensate for the width of the robot. Obstacle cells in the map are enlarged by the robot radius r_r, which is defined as the distance from the robot center to its furthest perimeter point. For further safety, the obstacle cells are actually enlarged by a radius $r_{r+s} = r_r + d_s$ where d_s is the minimum distance between the robot and an obstacle.

With the obstacles enlarged by r_{r+s}, the robot can be treated as a point-like vehicle [8]. This simple C-space method works well for mobile robots whose shape can be approximated by a disk. If the robot's shape is very asymmetrical, then the obstacle cells have to be enlarged according to the dimensions and the momentary orientation of the robot.

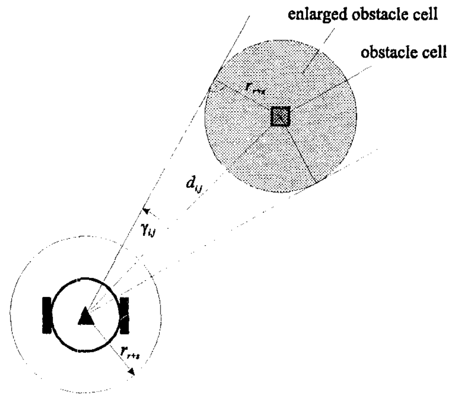

Figure 2: Enlargement angle $\gamma_{i,j}$.

The width compensation method is implemented very efficiently by enlarging the obstacles while building the primary polar histogram. Instead of updating only one histogram sector for each cell as done in the original VFH method, all histogram sectors that correspond to the enlarged cell are updated. An example for an obstacle cell is shown in Figure 2.

For each cell, the enlargement angle $\gamma_{i,j}$ is defined by:

$$\gamma_{i,j} = \arcsin \frac{r_{r+s}}{d_{i,j}} \tag{4}$$

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

$$H_k^p = \sum_{i,j \in C_a} m_{i,j} \cdot h'_{i,j} \tag{5}$$

with:

$$\begin{aligned} h'_{i,j} &= 1 \quad \text{if } k \cdot \alpha \in \left[\beta_{i,j} - \gamma_{i,j}, \beta_{i,j} + \gamma_{i,j}\right] \\ h'_{i,j} &= 0 \quad \text{otherwise} \end{aligned} \tag{6}$$

The result of this process is a polar histogram that takes into account the width of the robot. The h' function also serves as a low-pass filter and smoothes the polar histogram. Another important improvement is that this process eliminates the difficult tuning of the original method's low-pass filter.

Since the histogram is built around the current robot position, independent of its orientation, this first stage of the algorithm can be implemented very efficiently by the use of tables of the size $w_s \times w_s$. The $\beta_{i,j}$, $\gamma_{i,j}$, and $a - bd_{i,j}^2$ values for each cell in the active region C_a can be stored in tables for faster execution.

2.2 Second Stage – The Binary Polar Histogram

For most applications, a smooth trajectory is desired and oscillations in the steering command should be avoided. The original VFH method usually displays a very smooth trajectory. However, the fixed threshold τ can cause a problem in environments with several narrow openings, as the corresponding opening in the histogram can alternate several times between an open and a blocked state for several sampling times. In this situation, the robot's heading can also alternate several times between this narrow opening and another opening. The result is an indecisive behavior, during which the mobile robot can get very close to an obstacle.

This problem can easily be reduced by a hysteresis based on two thresholds, namely τ_{low} and τ_{high}. Based on the primary polar histogram H^p and the two thresholds, a *binary polar histogram* H^b is built. Instead of having polar density values, the sectors of H^b are either *free* (0) or *blocked* (1). The binary polar histogram indicates which directions are free for a robot that can instantaneously change its direction of motion. At time n, the binary polar histogram is updated by the following rules:

$$\begin{aligned} H^b_{k,n} &= 1 && \text{if } H^p_{k,n} > \tau_{high} \\ H^b_{k,n} &= 0 && \text{if } H^p_{k,n} < \tau_{low} \\ H^b_{k,n} &= H^b_{k,n-1} && \text{otherwise} \end{aligned} \qquad (7)$$

2.3 Third Stage – The Masked Polar Histogram

The original VFH method neglects the dynamics and the kinematics of the robot. It implicitly assumes that the robot is able to change its direction of travel instantly as shown in Figure 3a. Unless the robot stops at every sampling time, this assumption is clearly violated.

The VFH+ method uses a simple, but closer approximation of the trajectory of most mobile robots. It assumes that the robot's trajectory is based on circular arcs (constant curvature curves) and straight lines, as shown in Figure 3b. The curvature of a curve is defined by $\kappa = 1/r$.

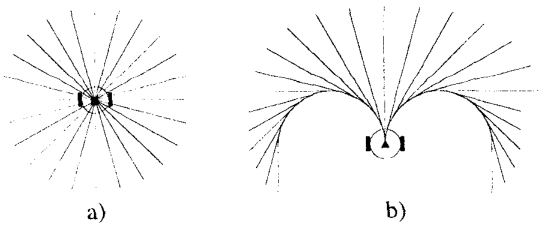

Figure 3: Approximation of trajectories: a) without dynamics, b) with dynamics.

The maximum trajectory curvature of a mobile robot is often a function of the robot velocity. The faster the robot travels, the smaller the maximum curvature. The minimum steering radius can be zero for a differential drive mobile robot if it has zero traversal speed. For mobile robots that are based on the *Ackerman steering* or the *tricycle* mechanism, the minimum steering radius never equals zero. In these cases, the minimum steering radius is approximately constant if the maximum velocity is not too high. In special cases, e.g., the GuideCane, the maximum curvature values are different for right and left turns.

The values of the minimum steering radii as a function of the robot velocity can easily be measured. We define these radii for both sides as $r_r = 1/\kappa_r$ and $r_l = 1/\kappa_l$.

With these parameters and the map grid, we can determine which additional sectors are blocked by obstacles because of the robot trajectory. An example with two small obstacles is shown in Figure 4. To take the width of the robot into account again, the obstacles are enlarged by r_{r+s}. If a trajectory circle and an enlarged obstacle cell overlap, all directions from the obstacle to the direction opposite the robot's heading are blocked. In our example, obstacle A blocks all directions to its left because of the robot dynamics. On the other hand, obstacle B does not block the directions to its right.

With the original VFH method, the directions to the left of obstacle A are considered to be suitable directions of motion. If the target direction were to the left, the original VFH algorithm would incorrectly guide the robot to the left and into obstacle A.

With the VFH+ method, the robot would correctly proceed between obstacles A and B and make a left turn after obstacle A was cleared.

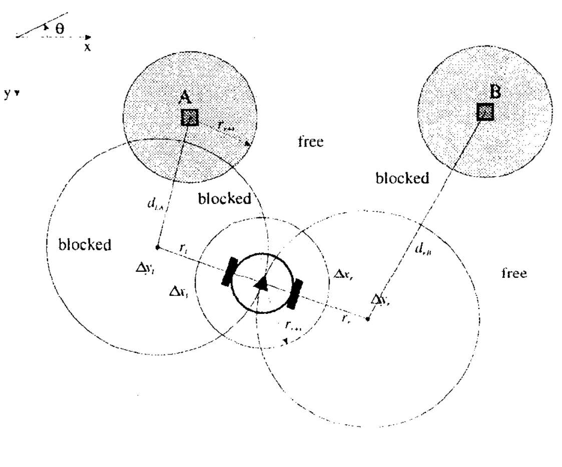

Figure 4: Example of blocked directions.

The positions of the right and left trajectory centers relative to the current robot position are defined by:

$$\Delta x_r = r_r \cdot \sin\theta \qquad \Delta y_r = r_r \cdot \cos\theta$$
$$\Delta x_l = -r_l \cdot \sin\theta \qquad \Delta y_l = -r_l \cdot \cos\theta \tag{8}$$

The distances from an active cell $C_{i,j}$ to the right and left trajectory centers are given by:

$$d_r^2 = (\Delta x_r - \Delta x(j))^2 + (\Delta y_r - \Delta y(i))^2$$
$$d_l^2 = (\Delta x_l - \Delta x(j))^2 + (\Delta y_l - \Delta y(i))^2 \tag{9}$$

An obstacle cell blocks the directions to its right if:

$$d_r^2 < (r_r + r_{r+s})^2 \qquad \text{[condition 1]} \tag{10}$$

And an obstacle cell blocks the directions to its left if:

$$d_l^2 < (r_l + r_{r+s})^2 \qquad \text{[condition 2]} \tag{11}$$

By checking every active cell with these two conditions, we obtain two limit angles: φ_r for right angles and φ_l for left angles. We also define $\varphi_b = \theta + \pi$ as the direction opposite to the current direction of motion.

This stage can be implemented very efficiently by an algorithm that only considers cells that have an influence on either φ_r or φ_l:

1) Determine φ_b. Set φ_r and φ_l equal to φ_b.
2) For every cell $C_{i,j}$ in the active window C_a with $c_{i,j} > \tau$:
   a) If $\beta_{i,j}$ is to the right of θ and to the left of φ_r, check condition 1. If condition is satisfied, set φ_r equal to $\beta_{i,j}$.
   b) If $\beta_{i,j}$ is to the left of θ and to the right of φ_l, check condition 2. If condition is satisfied, set φ_l equal to $\beta_{i,j}$.

If the robot's sensors are not very reliable, φ_r and φ_l could also be determined in a more stochastic way. Instead of comparing the cell certainty values to a threshold, one could build a polar histogram whose sector values indicate the certainty that a sector is blocked because of the robot dynamics. The values for φ_r and φ_l could then be determined by applying a threshold to this histogram. As the first method is more efficient, the second method should only be applied if really necessary.

With φ_r, φ_l, and the binary polar histogram, we can build the *masked polar histogram*:

$$H_k^m = 0 \quad \text{if} \quad H_k^b = 0 \ \textit{and} \ (k \cdot \alpha) \in \{[\varphi_r, \theta], [\theta, \varphi_l]\}$$
$$H_k^m = 1 \quad \text{otherwise} \tag{12}$$

The masked polar histogram shows which directions of motion are possible at the current speed. If all sectors were blocked, the robot could not proceed at the current speed, and it would have to determine a set of new values (φ_r, φ_l) based on a slower speed. If the masked polar histogram were still blocked in all directions, the robot would have to stop immediately. The masked polar histogram can therefore also be used to detect that the robot is trapped in a dead-end.

In Figure 5, the primary polar histogram, the binary polar histogram, and the masked polar histogram are shown for the simple situation of Figure 4. The binary polar histogram incorrectly indicates that the directions to the left of obstacle A are free. The masked polar histogram correctly blocks these directions.

Note that the vector magnitude for obstacle A is larger than for obstacle B, because obstacle A is closer to the robot. Also note that obstacle A occupies more sectors than obstacle B. As obstacle A is closer to the robot, its enlargement angle is wider.

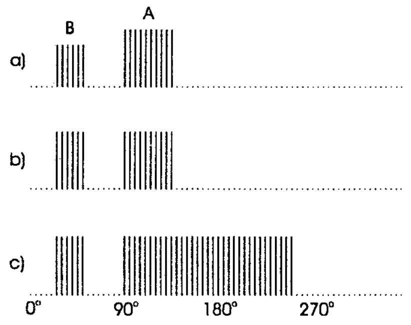

Figure 5: a) Primary polar histogram, b) binary polar histogram, c) masked polar histogram.

2.4 Fourth Stage - Selection of the Steering Direction

The masked polar histogram shows which directions are free of obstacles and which ones are blocked. However, some free directions are better candidates than others for the new direction of motion.

The original VFH method is very goal-oriented by selecting the valley that most closely matches the target direction k_t. It then selects the new direction of motion dependent on the size of the valley.

The VFH+ method first finds all openings in the masked polar histogram and then determines a set of possible candidate directions. A cost function that takes into account more than just the difference between the candidate and the target direction is then applied to these candidate directions. The candidate direction k_d with the lowest cost is then chosen to be the new direction of motion $\varphi_d = k_d \cdot \alpha$.

In the first step, the right and left borders k_r and k_l of all openings in the masked polar histogram are determined. Similar to the original VFH method, two types of openings are distinguished: *wide* and *narrow* ones. An opening is considered wide if the difference between its two borders is larger than s_{max} sectors (in our system s_{max} = 16). Otherwise, the opening is considered narrow.

For a narrow opening, there is only one candidate direction so that the robot steers through the center of the gap between the corresponding obstacles:

$$c_d = \frac{k_r + k_l}{2} \qquad \text{centered direction}^{+} \qquad (13)$$

For a wide opening, there are at least two candidate directions, c_r to the right and c_l to the left side of the opening. The target direction is also a candidate direction, if it lies between the two other candidate directions:

$$\begin{aligned} c_r &= k_r + \frac{s_{\max}}{2} && \text{towards the right side} \\ c_l &= k_l - \frac{s_{\max}}{2} && \text{towards the left side} \qquad (14) \\ c_t &= k_t && \text{if } k_t \in [c_r, c_l] \end{aligned}$$

The candidate directions c_r and c_l make the robot follow an obstacle contour at a safe distance, while c_t leads the robot towards the target direction.

For robots that are not goal-oriented, other candidate directions could be added. For a robot that should randomly explore its environment, one could add the candidate directions that are equal to the current direction of motion θ_n or equal to the previously selected direction of motion $k_{d,n-1}$:

$$\begin{aligned} c_\theta &= \frac{\theta_n}{\alpha} && \text{if } \frac{\theta_n}{\alpha} \in [c_r, c_l] \qquad (15) \\ c_\varphi &= k_{d,n-1} && \text{if } k_{d,n-1} \in [c_r, c_l] \end{aligned}$$

In the case of the goal-oriented robot, we get between one and three candidate directions for each opening in the masked polar histogram. Next, we need to define an appropriate cost function, so that the robot selects the most appropriate candidate direction as its new direction of motion φ_d. We propose the following cost function g as a function of a candidate direction c:

$$g(c) = \mu_1 \cdot \Delta(c, k_t) + \mu_2 \cdot \Delta\left(c, \frac{\theta_n}{\alpha}\right) + \mu_3 \cdot \Delta(c, k_{d,n-1}) \qquad (16)$$

where $\Delta(c_1,c_2)$ is a function that computes the absolute angle difference between two sectors c_1 and c_2 so that the result is $\leq s/2$. One possible implementation is:

$$\Delta(c_1, c_2) = \min\{|c_1 - c_2|, |c_1 - c_2 - s|, |c_1 - c_2 + s|\} \qquad (17)$$

The first term of our cost function $g(c)$ represents the cost associated with the difference of a candidate direction and the target direction. The larger this difference is, the more the candidate direction will guide the robot away from its target direction, and hence the larger the cost.

The second term represents the cost associated with the difference of a candidate direction and the robot's current wheel orientation. The larger this difference is, the larger the required change of the direction of motion.

The third term represents the cost associated with the difference of a candidate direction and the previously selected direction of motion. The larger this difference is, the larger the change of the new steering command.

In short, the first term is responsible for the goal-oriented behavior, while the second and third term make the mobile robot *commit* to a direction. These two terms provide the robot with a form of short-term memory. The second term is similar to a mechanical memory. With the help of the third term, the robot can commit to a direction even before its wheel orientation has changed.

The higher μ_1 is, the more goal-oriented the robot's behavior. The higher μ_2 is, the more the robot tries to execute a smooth path with a minimum change of direction of motion. The higher μ_3 is, the more the robot tries to head towards the previously selected direction and the smoother are the motor commands.

Only the relationship between the three parameters is important, not their magnitudes. To guarantee a goal-oriented behavior, the following condition must be satisfied:

$$\mu_1 > \mu_2 + \mu_3 \qquad \text{[condition 3]} \qquad (18)$$

If a smooth path is more important than variations in the steering commands, then μ_2 should be set higher than μ_3. If the smoothness of the steering commands is more important, then μ_3 should be set higher than μ_2.

Experiments have shown that a good set of parameters for a goal-oriented mobile robot is: μ_1 = 5, μ_2 = 2, and μ_3 = 2.

It is also possible to add other terms to the cost function. For example, one can make the mobile robot

+ Care must be taken when applying these equations because of the histogram boundaries.

avoid narrow openings by adding a term that takes into account the opening width: $\mu_4/\Delta(k_r, k_l)$.

On the other hand, the cost function could also be temporarily modified to make the mobile robot look for and go through narrow openings like doors by adding the term: $\mu_4 \cdot \Delta(k_r, k_l)$.

The cost function allows the user to implement a more subtle behavior than the coarse approach of the original VFH method. The commitment effect of the cost function is very important, especially when the robot approaches a single object that is right in its path. Without the commitment effect, the robot can actually bump into this obstacle by hesitating between avoiding it on the right or the left side. Another advantage is that the mobile robot behavior can easily be changed by modifying either the cost function parameters or the cost function itself.

3. EXPERIMENTAL RESULTS

The VFH+ method has been implemented and extensively tested on the real GuideCane in unstructured and unknown environments. All improvements were motivated by problems encountered during the testing of the real GuideCane's obstacle avoidance performance.

Our tests have shown that VFH+ allows safe travel at speeds of up to 1 m/s without noticeable oscillations. The maximum speed was limited by the number of sonars and their sampling rate, not by the VFH+ performance.

In our extensive testing we found the VFH+ method to be generally more reliable than its predecessor. We attribute this improvement mostly to the fact that VFH+ takes the robot trajectory into account. This is especially important for systems that work in a semi-autonomous mode (as is the case with the GuideCane application), in which the robot could be directed into an obstacle by entering a new desired direction of motion at the wrong time. For the same reason, the VFH+ performance does not degrade if the target direction differs more than 90° from the robot's orientation. The behavior of the robot when driving around corners is also improved due to the width compensation. We also found the VFH+ method to be rather insensitive to its parameter values; that is, it performed well as long as condition 3 was satisfied and the parameter values were reasonably selected.

On a PC 486 running at 66 MHz, the VFH+ algorithm takes at most 6 ms for each iterative cycle (i.e., sampling time). The speed of an obstacle avoidance algorithm is essential for several reasons. The higher the sampling rate of the obstacle avoidance method is, the faster the robot can travel without oscillations and without risk of bumping into an obstacle, and the more computational power can be used for generally time-consuming high-level behaviors.

4. CONCLUSION

The VFH+ method is the result of several improvements over the original VFH method. First of all, by using a threshold hysteresis, the robot trajectory becomes smoother and more reliable. Secondly, the VFH+ method explicitly takes into account the robot width, and therefore it can easily be implemented on robots of different sizes. This improvement also eliminates the time-consuming adjusting of the previously used low-pass filter. Thirdly, the VFH+ method takes into account the trajectory of the mobile robot by additionally masking sectors that are blocked by obstacles in other sectors. As a result, the robot can not be directed into an obstacle, as it was possible with the original VFH method. Finally, by applying a cost based direction selection, the performance of the obstacle avoidance algorithm becomes better and more reliable due to the commitment effect. The cost function also gives the possibility of switching between behaviors by simply changing the cost function or its parameters.

The remaining problem of the VFH+ method is its local nature, which sometimes leads the mobile robot into dead-ends that could be avoided. To overcome this problem, we are currently working on adding local planning to the obstacle avoidance algorithm.

ACKNOWLEDGMENT:

This research was funded by the Whitaker Foundation.

REFERENCES

[1] Borenstein, J., and Koren, Y., "Histogramic In-Motion Mapping for Mobile Robot Obstacle Avoidance", IEEE Transactions on Robotics and Automation, August 1991, pp. 535-539.

[2] Borenstein, J., and Koren, Y., "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots", IEEE Journal of Robotics and Automation, June 1991, Vol. 7, No. 3, pp. 278-288.

[3] Borenstein, J., and Ulrich, I., "The GuideCane - A Computerized Travel Aid for the Active Guidance of Blind Pedestrians", IEEE International Conference on Robotics and Automation, Albuquerque, April 1997, pp. 1283-1288.

[4] Elfes, A., "Using occupancy grids for mobile robot perception and navigation", Computer Magazine, June 1989, pp. 46-57.

[5] Koren, Y., and Borenstein, J., "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation", IEEE International Conference on Robotics and Automation, Sacramento, California, April 1991, pp. 1398-1404.

[6] Manz, A., Liscano, R., and Green, D., "A Comparison of Realtime Obstacle Avoidance Methods for Mobile Robots", Experimental Robotics, Toulouse, France, June 1991.

[7] Moravec, H.P., "Sensor fusion in certainty grids for mobile robots", AI Magazine, summer 1988, pp. 61-74.

[8] Udupa, S., "Collision Detection and Avoidance in Computer Controlled Manipulators", Ph.D. Dissertation, Department of Electrical Engineering, California Institute of Technology, 1977.