Real-Time Obstacle Avoidance for Fast Mobile Robots

JOHANN BORENSTEIN, MEMBER, IEEE, AND YOREM KOREN, SENIOR MEMBER, IEEE

Abstract—A new real-time obstacle avoidance approach for mobile robots has been developed and implemented. This approach permits the detection of unknown obstacles simultaneously with the steering of the mobile robot to avoid collisions and advancing toward the target. The novelty of this approach, entitled the virtual force field, lies in the integration of two known concepts: Certainty grids for obstacle representation, and potential fields for navigation. This combination is especially suitable for the accommodation of inaccurate sensor data (such as produced by ultrasonic sensors) as well as for sensor fusion, and enables continuous motion of the robot without stopping in front of obstacles. This navigation algorithm also takes into account the dynamic behavior of a fast mobile robot and solves the local minimum trap problem. Experimental results from a mobile robot running at a maximum speed of 0.78 m/s demonstrate the power of the proposed algorithm.

I. Introduction

REAL-TIME obstacle avoidance is one of the key issues to successful applications of mobile robot systems. All mobile robots feature some kind of collision avoidance, ranging from primitive algorithms that detect an obstacle and stop the robot short of it in order to avoid a collision, through sophisticated algorithms, that enable the robot to detour obstacles. The latter algorithms are much more complex, since they involve not only the detection of an obstacle, but also some kind of quantitative measurements concerning the obstacle's dimensions. Once these have been determined, the obstacle avoidance algorithm needs to steer the robot around the obstacle and resume motion toward the original target.

Autonomous navigation represents a higher level of performance, since it applies obstacle avoidance simultaneously with the robot steering toward a given target. Autonomous navigation, in general, assumes an environment with known and unknown obstacles, and it includes global path planning algorithms [3] to plan the robot's path among the known obstacles, as well as local path planning for real-time obstacle avoidance. This paper however assumes motion in the presence of unknown obstacles, and therefore concentrates only on the local obstacle avoidance aspect.

One approach to autonomous navigation is the wall-following method [1], [17], [18]. Here the robot navigation is based on moving alongside walls at a predefined distance. If an obstacle is encountered, the robot regards the obstacle as just another wall, following the obstacle's contour until it may resume its original course. This kind of navigation is technologically less demanding, since one major problem of mobile robots—the determination of their own position—is largely facilitated. Naturally, robot navigation by the wall-following method is less versatile and is suitable only for very specific applications. One soon-to-be introduced commercial system uses this method on a floor cleaning robot for long hallways [16].

A more general and commonly employed method for obstacle avoidance is based on edge-detection. In this method the algorithm tries to determine the position of the vertical edges of the obstacle and consequently attempts to steer the robot around either edge. The line connecting the two edges is considered to represent one of the obstacle's boundaries. This method was used in our own previous research [5], [6], as well as in several other research projects, such as [9], [11], [28]. A disadvantage with obstacle avoidance based on edge-detecting is the need of the robot to stop in front of an obstacle in order to allow for a more accurate measurement.

A further drawback of edge-detection methods is their sensitivity to sensor accuracy. Unfortunately ultrasonic sensors that are mostly used in mobile robot applications offer many shortcomings in this respect.

1) Poor directionality that limits the accuracy in determination of the spatial position of an edge to 10–50 cm, depending on the distance to the obstacle and the angle between the obstacle surface and the acoustic beam.
2) Frequent misreadings that are caused by either ultrasonic noise from external sources or stray reflections from neighboring sensors (crosstalk). Misreadings cannot always be filtered out and they cause the algorithm to see nonexisting edges.
3) Specular reflections that occur when the angle between the wave front and the normal to a smooth surface is too large. In this case the surface reflects

Manuscript received July 29, 1988; revised April 21, 1989. This work was supported in part by the Department of Energy Grant DE-FG02-86NE37967. The material in this paper was partially presented at the Third International Symposium on Intelligent Control, Arlington, VA, August 24–26, 1988.

The authors are with Advanced Technology Laboratories, University of Michigan, 1101 Beal Ave., Ann Arbor, MI 48109-2110.

IEEE Log Number 8929220.

0018-9472/89/0900-1179$01.00 ©1989 IEEE

the incoming ultrasound waves away from the sensor, and the obstacle is either not detected at all, or (since only part of the surface is detected) seen much smaller than it is in reality.

To reduce the effects just listed we have decided to represent obstacles with the certainty grid method. This method of obstacle representation allows adding and retrieving data on the fly and enables easy integration of multiple sensors.

The representation of obstacles by certainty levels in a grid model has been suggested by Elfes [15], who used the certainty grid for off-line global path planning. Moravec and Elfes [23], and Moravec [24] also describe the use of certainty grids for map-building. Since our obstacle avoidance approach makes use of this method, we will briefly describe the basic idea of the certainty grid.

In order to create a certainty grid, the robot's work area is divided into many square elements (denoted as cells), which form a grid (in our implementation the cell size is 10 cm by 10 cm). Each cell (i, j) contains a certainty value $C(i, j)$ that indicates the measure of confidence that an obstacle exists within the cell area. The greater $C(i, j)$, the greater the level of confidence that the cell is occupied by an obstacle.

In our system the ultrasonic sensors are continuously sampled while the robot is moving. If an obstacle produces an echo (within the predefined maximum range limit of two meters), the corresponding cell contents $C(i, j)$ are incremented. A solid, motionless obstacle eventually causes a high count in the corresponding cells. Misreadings on the other hand, occur randomly, and do not cause high counts in any particular cell. This method yields a more reliable obstacle representation in spite of the ultrasonic sensors' inaccuracies.

The novelty of our approach lies in the combination of certainty grids for obstacle representation with the potential field method for local path planning. Section II explains our basic virtual force field (VFF) obstacle avoidance approach in which we apply the potential field method to a certainty grid. The VFF method is further enhanced by taking into account the dynamic behavior of a mobile robot at high speeds and by a comprehensive heuristic solution to the trap problem (which is associated with the potential field method). A discussion on these two algorithms is included in Sections III and IV. The described algorithms have been implemented and tested on our mobile robot, computer-aided robotics for maintenance, emergency, and life support (Carmel).

II. The Virtual Force Field (VFF) Method

The idea of having obstacles conceptually exerting forces onto a mobile robot has been suggested by Khatib [20]. Krogh [21] has enhanced this concept further by taking into consideration the robot's velocity in the vicinity of obstacles. Thorpe [26] has applied the potential fields method to off-line path planning. Krogh and Thorpe [22] suggested a combined method for global and local path planning, which uses Krogh's generalized potential field (GPF) approach. These methods however assume a known and prescribed world model of the obstacles. Furthermore, none of these methods has been implemented on a mobile robot that uses real sensory data. The closest project to ours is that of Brooks [7], [8], who uses a force field method in an experimental robot equipped with a ring of twelve ultrasonic sensors. Brooks' implementation treats each ultrasonic range reading as a repulsive force vector. If the magnitude of the sum of the repulsive forces exceeds a certain threshold, the robot stops, turns into the direction of the resultant force vector, and moves on.

A. The Basic VFF Method

This section explains the combination of the potential field method with a certainty grid. This combination produces a powerful and robust control scheme for mobile robots, denoted as the virtual force field (VFF) method.

As the robot moves around, range readings are taken and projected into the certainty grid, as explained. Simultaneously the algorithm scans a small square window of the grid. The size of the window is 33×33 cells (i.e., $3.30\text{m} \times 3.30\text{m}$) and its location is such that the robot is always at its center.

Each occupied cell inside the window applies a repulsive force to the robot, pushing the robot away from the cell. The magnitude of this force is proportional to the cell contents $C(i, j)$ and is inversely proportional to the square of the distance between the cell and the robot:

$$\boldsymbol{F}(i, j) = \frac{F_{cr} C(i, j)}{d^2(i, j)} \left[\frac{x_t - x_0}{d(i, j)} \hat{x} + \frac{y_t - y_0}{d(i, j)} \hat{y} \right] \quad (1)$$

where

F_{cr}	force constant (repelling),
$d(i, j)$	distance between cell (i, j) and the robot,
$C(i, j)$	certainty level of cell (i, j),
x_0, y_0	robot's present coordinates,
x_i, y_j	coordinates of cell (i, j).

Notice that in (1) the force constant is divided by d^2. By this method occupied cells exert strong repulsive forces when they are in the immediate vicinity of the robot, and weak forces when they are further away.

The resultant repulsive force $\boldsymbol{F}_r$ is the vectorial sum of the individual forces from all cells:

$$\boldsymbol{F}_r = \sum_{i, j} \boldsymbol{F}(i, j). \quad (2)$$

At any time during the motion, a constant-magnitude attracting force $\boldsymbol{F}_t$ pulls the robot toward the target. $\boldsymbol{F}_t$ is generated by the target point T, whose coordinates are known to the robot. The target-attracting force $\boldsymbol{F}_t$ is given by

$$\boldsymbol{F}_t = F_{ct} \left[\frac{x_t - x_0}{d(t)} \hat{x} + \frac{y_t - y_0}{d(t)} \hat{y} \right] \quad (3)$$

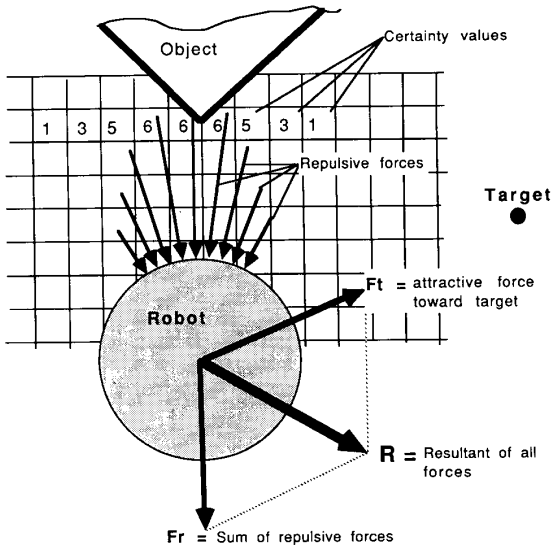

Fig. 1. Virtual force field concept: Occupied cells exert repulsive forces onto robot; magnitude is proportional to cell's certainty value $C(i, j)$ and inversely proportional to d^2.

where

F_{ct} force constant (attraction to the target),
$d(t)$ distance between the target and the robot,
x_t, y_t target coordinates.

Notice that $\boldsymbol{F}_t$ is independent of the absolute distance to the target.

As shown in Fig. 1, the vectorial sum of all forces, repulsive from occupied cells and attractive from the target position, produces a resultant force vector $\boldsymbol{R}$:

$$\boldsymbol{R} = \boldsymbol{F}_t + \boldsymbol{F}_r. \tag{4}$$

The direction of $\boldsymbol{R}$, $\delta = \boldsymbol{R}/|\boldsymbol{R}|$ (in degrees), is used as the reference for the robot's steering-rate command Ω:

$$\Omega = K_s[\theta(-)\delta] \tag{5}$$

where

K_s proportional constant for steering (in s^{-1}),
θ current direction of travel (in degrees).

$(-)$ is a specially defined operator for two operands, α and β (in degrees), and is used in the form $\gamma = \alpha(-)\beta$. The result γ (in degrees), is the shortest rotational difference between α and β. Therefore γ is always in the range $-180° < \gamma < 180°$.

A typical obstacle map in certainty grid representation shows obstacle-boundaries as clusters of cells with high certainty values. Misreadings on the other hand, occur at random and therefore produce mostly isolated cells with low certainty values. Summation of repulsive forces from occupied cells (2) makes the robot highly responsive to clusters of filled cells, while almost completely ignoring isolated cells.

B. Advantages Over Conventional Methods

The VFF algorithm has several advantages over edge detecting methods, which are presently used in many mobile robot applications:

1) In edge detection methods, misreadings or inaccurate range measurements may be misinterpreted as part of an obstacle contour, thereby gravely distorting the perceived shape of the obstacle. The sharply defined contour required by these methods cannot accommodate the blurry and inaccurate information provided by ultrasonic sensors. Edge detection methods require binary knowledge about the obstacle contour (exists or does not exist), and therefore cannot implement certainty methods, in which the data is weighted. The VFF method on the other hand, does not utilize sharp contours in the world model, but rather responds to clusters of high likelihood for the existence of an obstacle. This results in increased robustness of the algorithm in the presence of misreadings.

2) The VFF method does not require the robot to stop for data acquisition and evaluation, or for corner negotiation (as in the cases reported in [6], [8], [9], [11], [12], [19]). Except for the artificially introduced effect of damping (see discussion in Section III-A), the VFF method allows the robot to negotiate all obstacles while it travels at up to its maximum velocity.

3) Updating the grid-map sensor information and using the grid-map for navigation are two independent tasks that are performed asynchronously, each at its optimal pace. The edge detection method, by contrast, requires the following activities to be performed in sequence: detect an obstacle, stop the robot, measure the obstacle (find its edges), recalculate path, and resume motion.

4) The grid representation for obstacles lends itself easily to the integration of data from groups of similar, as well as from different types of sensors (such as vision, touch, and proximity), in addition to data from previous runs or from preprogrammed stationary obstacles (such as walls).

III. Dynamic Motion Enhancements for Robots Running at High Speeds

This section discusses the main enhancements that have been incorporated in the VFF method in order to account for the dynamic properties of a mobile robot [2], [4] moving at higher speeds (up to 0.78 m/s).

The mobile robot used in all experiments is a commercially available Cybermation K2A mobile platform [13]. The K2A has a maximum travel speed of $V_{max} = 0.78$ m/s, a maximum steering rate of $\Omega = 120$ deg/s, and weights (in its current configuration) about $W = 125$ kg. This platform has a unique three-wheel drive (synchro-drive) that permits omnidirectional steering. In order to control this mobile platform, two data items must be sent (through a serial link) to its onboard computer: a velocity command, V, and a steering-rate command, Ω (computed in (5)).

Our mobile platform has been equipped with a ring of 24 Polaroid ultrasonic sensors [25] and an additional com-

puter (a PC-compatible single-board computer, running at 7.16 MHz), to control the sensors. Similar sensor configurations have been designed for other mobile robots, such as [14] or [27].

Since the remaining part of this paper focuses on various aspects of motion performance, we have chosen to *simulate* obstacles in the certainty grid, rather than use real sensory data. The undeterministic nature of actual ultrasonic range information makes it difficult to reproduce test runs or compare them with each other. The VFF algorithm however, works equally well with real obstacles, as is demonstrated in the last experimental result in this paper (Fig. 8).

All of the following examples and plots have been obtained from actual runs of our robot (with its sensors disabled). Although the maximum speed of $V_{max} = 0.78$ m/s was used, the average speed in each run was lower, since the algorithm reduces the robot's speed under certain conditions (as will be explained below).

A. Low Pass Filter for Steering Control

For smooth operation of the VFF method, the following condition relating the grid resolution Δs and the sampling period T must be satisfied as

$$\Delta s > TV_{max}. \tag{6}$$

In our case $\Delta s = 0.1$ m and $TV_{max} = 0.1 * 0.78 = 0.0078$ m, and therefore this condition is satisfied.

Since the distance dependent resultant force vector $\boldsymbol{F}_r$ (see (2)) is quantized to the grid resolution (10 cm × 10 cm), rather drastic changes in the resultant force vector $\boldsymbol{R}$ may occur as the robot moves from one cell to another (even with condition (6) satisfied). This results in an overly vivacious steering control, as the robot tries to adjust its direction to the rapidly changing direction of $\boldsymbol{R}$. To avoid this problem, a digital low-pass filter, approximated in the algorithm by $\tau = 0.4$ s, has been added at the steering-rate command. The resulting steering-rate command is given by

$$\Omega_i = \frac{T\Omega'_{i-1} + (\tau - T)\Omega_{i-1}}{\tau} \tag{7}$$

where

- Ω_i steering-rate command to the robot (after low-pass filtering);
- Ω_{i-1} previous steering-rate command;
- Ω'_i steering-rate command (before low-pass filtering);
- T sampling time (here: $T = 0.1$ s);
- τ time constant of the low pass filter.

The filter smoothes the robot's motion when moving alongside obstacles. However it also introduces a time delay τ, which deteriorates the steering response of the robot by the displacement delay $\tau\Omega$.

B. Damping (for Speed Command)

Ideally when the robot encounters an obstacle, it would move smoothly alongside the obstacle until it can turn again toward the target. At higher speeds (e.g., $V > 0.5$ m/s), however, the root introduces a considerable relative delay in responding to changes in steering commands caused by the combined effects of its inertia and the low-pass filter mentioned earlier. Due to this delay the robot might approach an obstacle very closely, even if the algorithm produces very strong repulsive forces. When the robot finally turns around to face away from the obstacle, it will depart more than necessary for the same reason. The resulting path is highly oscillatory, as shown in Fig. 2(a).

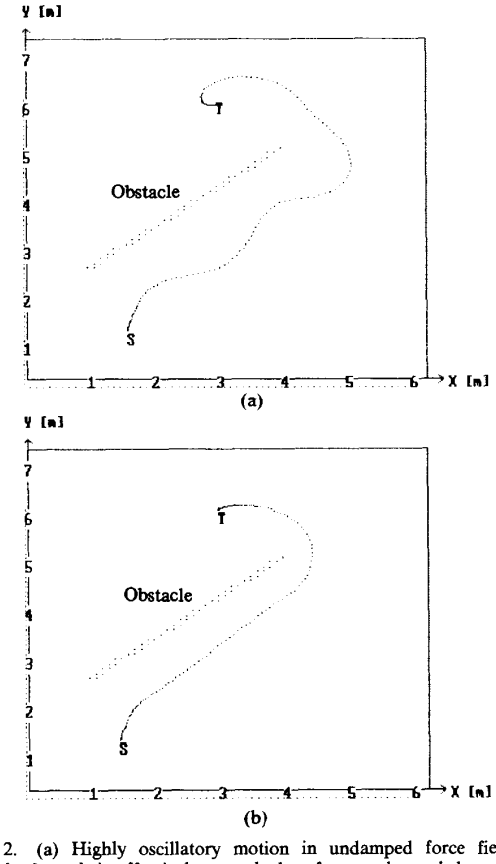

Fig. 2. (a) Highly oscillatory motion in undamped force field. (b) Robot's path is effectively smoothed as force and speed damping are applied.

One way to dampen this oscillatory motion is to increase the strength of the repulsive forces when the robot moves toward an obstacle, and reduce it when the robot moves alongside the obstacle. The general methodology calls for variations in the magnitude of the sum of the repulsive forces $\boldsymbol{F}_r$ as a function of the relative directions of $\boldsymbol{F}_r$ and the velocity vector $\boldsymbol{V}$. Mathematically this is achieved by multiplying the sum of the repulsive forces by the directional cosine ($\cos\theta$) of the two vectors, $\boldsymbol{F}_r$ and $\boldsymbol{V}$, and using the product as follows:

$$\boldsymbol{F}_r' = w\boldsymbol{F}_r + (1 - w)\boldsymbol{F}_r(-\cos\theta) \tag{8}$$

where $\boldsymbol{F}_r'$ is the adjusted sum of the repulsive forces and w is a weighting factor that was set to $w = 0.25$ in our system.

The directional cosine in (8) is computed by

$$\cos\theta = \frac{V_x F_{rx} + V_y F_{ry}}{|V||F_r|} \tag{9}$$

where

V_x, V_y x and y components of velocity vector V;
F_{rx}, F_{ry} x and y components of the sum of the repulsive forces, F_r.

The effect of this damping method is that the robot experiences the repulsive forces at their full magnitude, as it approaches the obstacle frontally (with $-\cos\theta = 1$). As the robot turns toward a direction alongside the obstacle's boundary, the repulsive forces are weakened by the factor $0.75 * \cos\theta$, and will be at their minimum value when the robot runs parallel to the boundary. Notice that the setting $w = 0$ is undesirable, since the robot will eventually run into an obstacle as it approaches it at a very small angle.

Careful examination of (8) reveals the fact that the damped sum of repulsive forces, F_r', may become negative (thereby actually attracting the robot), as the robot moves away from the obstacle (and $\cos\theta > 0$). We found the attraction-effect to improve damping and reduce oscillatory motion.

C. Speed Control

The intuitive way to control the speed of a mobile robot in the VFF environment is to set it proportional to the magnitude of the sum of all forces, $R = F_r + F_t$. Thus if the path was clear, the robot would be subjected only to the target force and would move toward the target, at its maximum speed. Repulsive forces from obstacles, naturally opposed to the direction of F_t (with disregard to the damping effect discussed previously), would reduce the magnitude of the resultant R, thereby effectively reducing the robot's speed in the presence of obstacles.

However we have found that the overall performance can be substantially improved by setting the speed command Ω proportional to $\cos\theta$ (see (9)). This function is given by

$$V = \begin{cases} V_{\max} & \text{for } |F_r| = 0 \text{ (i.e., in the absence of obstacles)} \\ V_{\max}(1 - |\cos\theta|), & \text{for } |F_r| > 0 \end{cases} \tag{10}$$

With this function the robot still runs at its maximum speed if no obstacles are present. However in the presence of obstacles, speed is reduced only if the robot is heading toward the obstacle (or away from it), thus creating an additional damping effect. If however the robot moved alongside an obstacle boundary, its speed is almost not reduced at all and it moves at its maximum speed, thereby greatly reducing the overall travel-time.

Fig. 2(b) shows the joint effect of both damping measures on the resulting path.

IV. Recovery from Local Minimum Traps

One problem inherent to the basic VFF method is the possibility for the robot to get trapped. This situation may occur when the robot runs into a dead end (e.g., inside a U-shaped obstacle). Traps can be created by a variety of different obstacle configurations, and different types of traps can be distinguished. This section presents a comprehensive set of heuristic rules to recover from different trap conditions. Chattergy [10] presented some heuristic local path planning solutions for various obstacle configuration (and trap conditions), based on distance measurements to the obstacle. While his approach to recovery from a single trap is similar to ours (through wall-following, see discussion forthcoming), his solution to the problem of multiple traps differs completely from ours. Also Chattergy offers no solution to the inside-wall problem (as discussed in Section IV-B).

A. Trap-State Detection

In an ideal, noninertial system, trap-states may be discovered by simply monitoring the speed of the robot. If caught in a trap, the robot's speed will become zero as the robot converges to the equilibrium position with $R = 0$. In a dynamic system, however, the robot overshoots the equilibrium position and will either oscillate or run in a closed loop, as shown in Fig. 3(a) for an actual run. Therefore it is impractical to monitor the magnitude of the result force $|R|$ for trap-state detection. Our method for trap-state detection compares the robot-to-target direction, θ_t, with the actual instantaneous direction of travel, θ_0. If the robot's direction of travel is more than 90° off-target, namely, if

$$|\theta_t - \theta_0| > 90° \tag{11}$$

the robot starts to move away from the target and is very likely about to get trapped. Therefore to avoid trap-situations, the controller monitors the condition in (11). If (11) is satisfied, the system switches to the recovery algorithm discussed next. Notice that (11) expresses an overconservative condition, and under certain circumstances this condition may be satisfied without the robot being subsequently trapped. However this test is computationally simple, and the evoked recovery algorithm does not reduce the overall performance (if evoked only temporarily), even if the robot would not be trapped.

B. Recovery Through Wall-Following

The recovery algorithm, referred to as the wall-following method (WFM), steers the robot such as to follow the current obstacle contour. The current obstacle contour is the boundary of the obstacle that pushed the robot away and made it assume a heading more than 90° off the target direction. With the WFM, the robot follows the contour until it starts heading back into a direction less than 90° off-target (i.e., $|\theta_t - \theta_0| < 90°$). Subsequently the robot resumes its (normal VFF) mode, and heads toward the

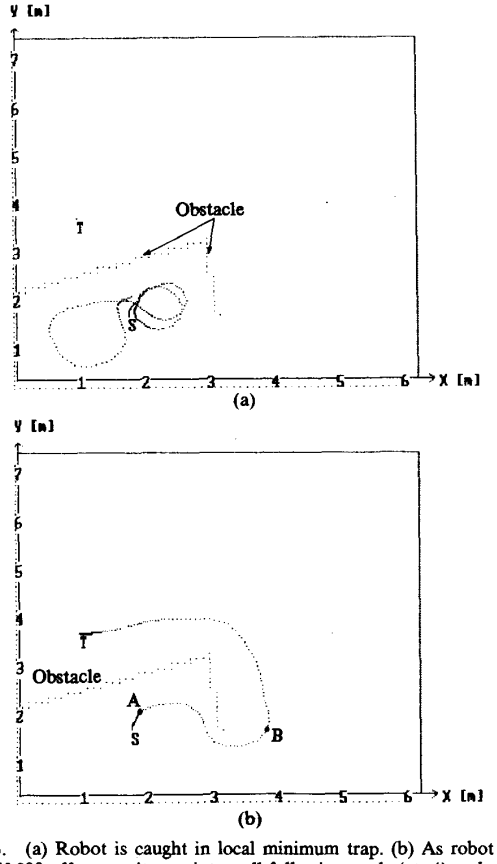

Fig. 3. (a) Robot is caught in local minimum trap. (b) As robot finds itself 90° off target, it goes into wall-following mode (at A) and reaims at target (at B).

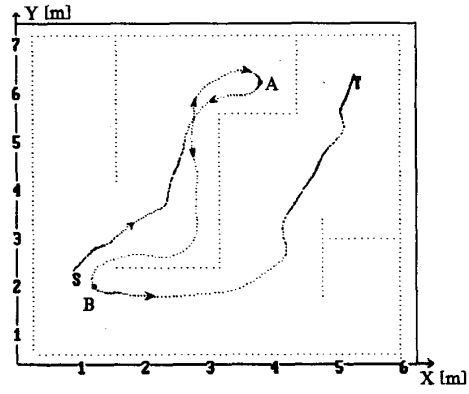

Fig. 4. Robot successfully recovers from labyrinth-like obstacle course.

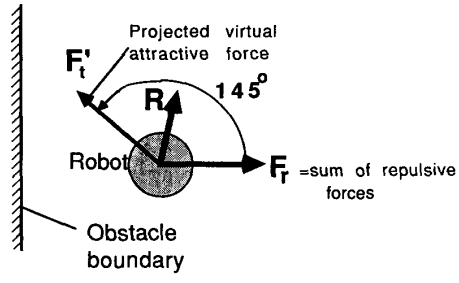

Fig. 5. Derivation of virtual attractive force F_t' for wall-following mode.

target. The joint operation of the WFM and the VFF mode is demonstrated in Fig. 3(b), in which the robot reaches the target in 16.0 s at an average speed of 0.44 m/s. At point A, the algorithm detects a trap situation, and switches back into WFM. At point B the off-angle becomes less than 90°, and the algorithm switches back into VFF mode. This recovery algorithm works even for extreme constellations of obstacles, such as the labyrinth-like obstacle course in Fig. 4. Again at point A the system activates the WFM algorithm, and at point B it switches back to VFF mode.

The WFM algorithm is implemented in the following manner. First the robot calculates the sum of all repulsive forces, F_r, in order to determine the direction of F_r, θ_r. Next as is depicted in Fig. 5 (for following a wall to the left of the robot), the algorithm adds an angle α to θ_r, where $90° < \alpha < 180°$ (or subtracts α from θ_r, if following a wall to the right of the robot). In our system $\alpha = 145°$. Then the algorithm projects a new virtual attractive force F_t' in the resulting direction, which temporarily replaces the attractive force from the target location, F_0. The new resultant R will now point (or eventually converge) into a direction parallel to the obstacle boundary. While wall-following could also be implemented through controlling a fixed distance to the wall, our method is preferable, since it is less sensitive to misreadings of the ultrasonic sensors. A similar implementation for wall-following has been suggested by Brooks and Connell [8], with $\alpha = 120°$.

There is however one catch to the WFM, which may occur if more than one trap is present. Fig. 6(a) shows one such case. Here the robot switches to WFM at point A and follows a wall to its left. Then at point B, it switches back to VFF mode, since its heading is now less than 90° off the target direction. At point C, a new trap-condition is detected, and the robot goes again into WFM (notice that this time the robot follows the obstacle at its right). This pattern repeats itself at points D, E, and F, inside the second and third trap. Subsequently the robot returns and oscillates between the traps. To solve this problem two possible wall-following modes must be distinguished: L-mode, where the robot follows a wall to its left, and R-mode, where the robot follows a wall to its right. This distinction is utilized in the implementation of the WFM in our system.

In a given run the robot might select either L- or R-mode at the first trap-situation, but then it must stick always to this mode within the given run. The result of running this algorithm is depicted in Fig. 6(b) (for the same obstacle configuration as in Fig. 6(a)). The robot encounters a trap at point A and chooses L-mode. At

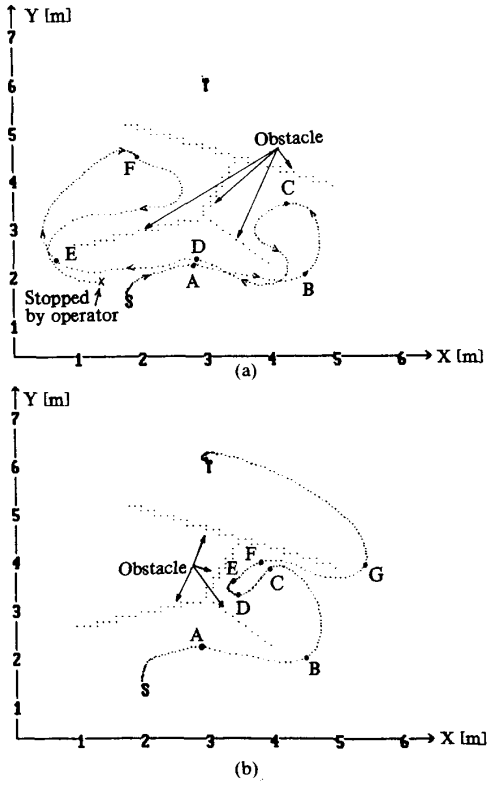

Fig. 6. (a) Robot oscillates between multiple traps. (b) Remedy of multiple trap oscillations by adherence to original wall-following mode.

point B the trap-situation is resolved and the robot returns to VFF mode. However a new obstacle is encountered, which pushes the robot into R-mode (at point C). Since the robot has to stick to the original WFM, it slows down and performs a U-turn (at point D). Subsequently the robot resumes motion in VFF mode (at point E). A new trap is detected at point F, but this trap can be resolved by running in (the original) L-mode. At point G the robot resumes motion in VFF mode and reaches the target. The average speed for this run was 0.5 m/s.

One last exception that needs to be addressed occurs when the robot is in WFM on an inside wall of a closed room (with the target in the same room). In this case the robot will follow that wall indefinitely, since the condition for exiting WFM ($|\theta_t - \theta_0| < 90°$) will not be satisfied (Note that θ_0 is updated absolutely and not by modulus 360°, so that e.g., $420° \neq 60°$).

However this situation may be detected by monitoring the sum Φ of the changes of the target direction, θ_t, between sampling intervals i as follows:

$$\Phi = \Sigma\left[\theta_t(i) - \theta_t(i-1)\right]. \tag{12}$$

$\Phi > 360°$ indicates that the robot has traveled at least one full loop around the target. This only happens when the robot follows an inside wall completely surrounding the target.

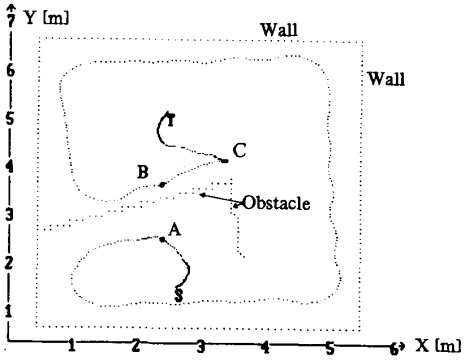

Fig. 7. Wall-following on inner wall of room may result in indefinite loop. However, full loop around target can be identified and used to remedy situation.

Once detected there are several ways to remedy the situation. In our algorithm the robot is simply forced out of the WFM and back into normal target pursuit. Fig. 7 shows an example. At point A the algorithm detects the trap condition and switches into WFM with R-mode. At point B the robot has completed one full revolution about the target (since A) and the loop condition ($\Phi > 360°$) is detected. The robot slows down and stops at point C. Theoretically there is no need to stop at C, but for the trivial purpose of untangling the umbilical cord (note that the robot has spun almost 720° by the time it reaches C) the robot does stop at C. It then rotates on the spot until its heading is directed toward the target again, and resumes motion in VFF mode.

Fig. 8 shows a run of the robot with *actual ultrasonic data*, obtained in real-time during the robot's motion. Partitions were set up in the lab such as to resemble the simulated obstacles in Fig. 3. The robot ran at a maximum speed of 0.78 m/s and achieved an average speed of 0.53 m/s. The maximal range for the sensors was set to 2.0 m, which is why only part of the rightmost wall is shown, whereas the rear wall and most of the leftmost wall remained undetected.

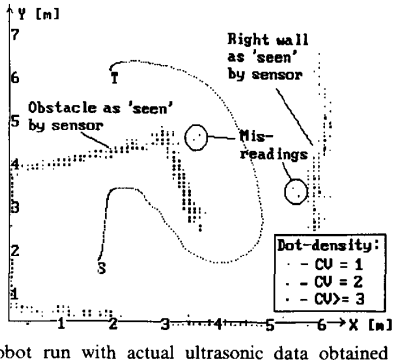

Fig. 8. Robot run with actual ultrasonic data obtained in real-time during robot's motion. Maximum speed = 0.78 m/s and average speed = 0.53 m/s.

Each dot in Fig. 8 represents one cell in the certainty grid. In our current implementation, certainty values (CV) range only from zero to three. The CV = 0 means no sensor reading has been projected into the cell during the run (no dot at all). CV = 1 (or CV = 2) means that one (or two) readings have been projected into the cell, and this is shown in Fig. 8 with dots comprising of one (or two) pixels. CV = 3 means that three or more readings have been projected into the same cell, and this is represented by a 4-pixel dot in Fig. 8.

At least two misreadings can be identified in Fig. 8, which have been encircled for emphasis.

V. Conclusion

A comprehensive obstacle avoidance approach for fast-running mobile robots, denoted as the VFF method, has been developed and tested on our experimental mobile robot Carmel. The VFF method is based on the following principles.

1) A certainty grid for representation of (inaccurate) sensory data about obstacles provides a robust real-time world model.
2) A field of virtual attractive and repulsive forces determines the direction and speed of the mobile robot.
3) The combination of results in 1) and 2) is the characteristic behavior of the robot: The robot responds to clusters of high-likelihood for the existence of an obstacle, while ignoring single (possibly erroneous) data points.
4) Trap states are automatically detected and recovery routines are activated. These routines distinguish among several different situations and take appropriate action for each situation.
5) Oscillatory robot motion is resolved by damping algorithms (which only marginally compromise the robot's average speed).

Based on the VFF method for autonomous obstacle avoidance, we are currently developing a new mode of operation for the remote guidance of mobile robots. Under this mode of operation, the mobile robot follows the general direction prescribed by the operator. If the robot encounters an obstacle, it autonomously avoids collision with that obstacle, trying to match the prescribed direction as good as possible. With this integrated self-protection mechanism, robots can be steered at high speeds and in cluttered environments, even by inexperienced operators.

References

[1] G. Bauzil, M. Briot, and P. Ribes, "A navigation sub system using ultrasonic sensors for the mobile robot Hilare," *1st Int. Conf. on Robot Vision and Sensory Controls*, Stratford-upon-Avon, UK, 1981, pp. 47–58 and pp. 681–698.

[2] J. Borenstein and Y. Koren, "A mobile platform for nursing robots," *IEEE Trans. on Ind. Electron.*, vol. 32, no. 2, pp. 158–165, 1985.

[3] ____, "Optimal path algorithms for autonomous vehicles," *Proc. 18th CIRP Manufacturing Systems Sem.*, Stuttgart, W. Germany, June 5–6.

[4] ____, "Motion control analysis of a mobile robot," *Trans. ASME, J. Dynam. Measurement Contr.*, vol. 109, no. 2, 1987, pp. 73–79.

[5] J. Borenstein, "The nursing robot system," Ph. D. Dissertation, Technion, Haifa, Israel, 1987.

[6] J. Borenstein and Y. Koren, "Obstacle avoidance with ultrasonic sensors," *IEEE J. Robotics and Automat.*, vol. RA-4, no. 2, pp. 213–218, 1988.

[7] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robotics and Automat.*, vol. RA-2, no. 1, 14–23, 1986.

[8] R. A. Brooks and J. H. Connell, "Asynchronous distributed control system for a mobile robot," *Proc. SPIE*, vol. 727, pp. 77–84, 1986.

[9] R. A. Cooke, "Microcomputer control of free-ranging robots," *Proc. 13th Int. Symp. Ind. Robots and Robots*, Chicago, IL, Apr. 1983, pp. 13.109–13.120.

[10] R. Chattergy, "Some heuristics for the navigation of a robot," *Int. J. Robotics Res.*, vol. 4, no. 1, 1985, pp. 59–66.

[11] J. L. Crowley, "Dynamic world modeling for an intelligent mobile robot," *Proc. IEEE Seventh Int. Conf. Pattern Recognition*, Montreal, Canada, 1984, July 30–August 2, pp. 207–210.

[12] ____, "Navigation for an intelligent mobile robot," Technical Report, The Robotics Inst., Carnegie-Mellon Univ., Aug. 1984.

[13] Cybermation, "K2A Mobile Platform," *Commercial offer*, 5457 JAE Valley Road, Roanoke, Virginia 24014, 1987.

[14] Denning Mobile Robotics, Inc., "Securing the future," *Commercial Offer*, 21 Cummings Park, Woburn, MA 01801, 1985.

[15] A. Elfes, "A sonar-based mapping and navigation system," Technical Report, The Robotics Inst., Carnegie-Mellon Univ., pp. 25–30, 1985.

[16] Engelberger, J., Transitions Research Corporation, private communication, 1986.

[17] G. Giralt, "Mobile robots," *NATO ASI Series, vol. F11, Robotics and Artificial Intelligence*. New York: Springer-Verlag, 1984, pp. 365–393.

[18] J. Iijima, S. Yuta, and Y. Kanayama, "Elementary functions of a self-contained robot 'YAMABICO 3.1'," *Proc. 11th Int. Symp. Ind. Robots*, Tokyo, 1983, pp. 211–218.

[19] C. Jorgensen, W. Hamel, and C. Weisbin, "Autonomous robot navigation," *BYTE*, pp. 223–235, Jan. 1986.

[20] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *1985 IEEE Int. Conf. Robotics and Automat.*, St. Louis, MO, March 25–28, 1985, pp. 500–505.

[21] B. H. Krogh, "A generalized potential field approach to obstacle avoidance control," *Int. Robotics Research Conf.*, Bethlehem, PA, Aug. 1984.

[22] B. H. Krogh and C. E. Thorpe, "Integrated path planning and dynamic steering control for autonomous vehicles," *Proc. 1986 IEEE Int. Conf. Robotics and Automation*, San Francisco, CA, Apr. 7–10, 1986, pp. 1664–1669.

[23] H. P. Moravec and A. Elfes, "High-resolution maps from wide angle sonar," *IEEE Conf. Robotics and Automation*, washington DC, 1985, pp. 116–121.

[24] H. P. Moravec, "Certainty grids for mobile robots," Technical Report Preprint, The Robotics Inst., Carnegie-Mellon Univ., 1986.

[25] Polaroid Corporation, "Ultrasonic ranging marketing," 1 Upland Road, Norwood, MA 02062, 1982.

[26] C. F. Thorpe, "Path relaxation: Path planning for a mobile robot," Carnegie-Mellon University: The Robotics Institute, Mobile Robots Laboratory, Autonomous Mobile Robots, Annual Report, 1985, pp. 39–42.

[27] S. A. Walter, "The sonar ring: Obstacle detection for a mobile robot," *Proc. IEEE Int. Conf. Robotics and Automat.*, Raleigh, NC, Mar. 31–Apr. 3, 1987, pp. 1547–1579.

[28] C. R. Weisbin, G. de Saussure, and D. Kammer, "Self-Controlled: A real-time expert system for an autonomous mobile robot," *Comput. Mech. Eng.*, pp. 12–19, Sept. 1986.

Johann Borenstein (M'88) received the B.Sc., M.Sc., and D.Sc. in mechanical engineering in 1981, 1983, and 1987, respectively, from the Technion —Israel Institute of Technology, Haifa, Israel.

Since 1987 he has been with the Robotics Systems Division at the University of Michigan, Ann Arbor, where he is currently an Assistant Research Scientist and Head of the MEAM Mobile Robotics Lab. His research interests include mobile robot navigation, obstacle avoidance, path planning, real-time control, sensors for robotic applications, multisensor integration, computer interfacing and integration.

Yoram Koren (M'76–SM'88) has 20 years of research, teaching, and consulting experience in the automated manufacturing field. At present he is the Chairman of the Mechanical Sciences Division in the Department of Mechanical Engineering and Applied Mechanics, at the University of Michigan, Ann Arbor. He is the author of *Computer Control of Manufacturing Systems* (McGraw-Hill, 1983) which is used as a textbook at major universities and received the 1984 Textbook Award from the Society of Manufacturing Engineering. His book, *Robotics for Engineers*, (McGraw-Hill, 1985) was translated into Japanese and French and is used by engineers throughout the world. Professor Koren is a member of ASME, an active member of CIRP, and a Fellow of SME/Robotics-International.