[8] M. A. Erdmann and M. T. Mason, “An exploration of sensorless manipulation,” *IEEE J. Robot. Automat.*, vol. 4, pp. 369–379, Aug. 1988.

[9] A. Rao, D. J. Kriegman, and K. Goldberg, “Complete algorithms for feeding polyhedral parts using pivot grasp,” *IEEE Trans. Robot. Automat.*, vol. 12, pp. 331–342, Apr. 1996.

[10] W. J. C. Heemskerk, “Method for Causing Sitting Poultry to Stand up and Apparatus for Carrying out this Method,” U.S. Patent 5 088 959, 1992.

[11] M. E. Keiter, “Method and Apparatus for Orienting Live Poultry,” U.S. Patent 5 129 857, 1992.

[12] K-M. Lee, “On the development of a compliant grasping mechanism for on-line handling of live objects, Part I: Analytical model,” in *IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (AIM’99)*, Atlanta, GA, Sept. 19–23, 1999, pp. 354–359.

[13] K-M. Lee, B. Webster, J. Joni, X. Yin, R. Carey, M. Lacy, and R. Gogate, “On the development of a compliant grasping mechanism for on-line handling of live objects, part II: Design and experimental investigation,” in *AIM’99*, Atlanta, GA, Sept. 19–23, 1999, pp. 360–365.

[14] K-M. Lee, J. Joni, and X. Yin, “Compliant grasping force modeling for handling of live objects,” in *Proc. 2001 IEEE ICRA*, Seoul, Korea, May 21–26, 2001, pp. 1059–1064.

Autonomous Vehicle Navigation Utilizing Electrostatic Potential Fields and Fuzzy Logic

Nikos C. Tsourveloudis, Kimon P. Valavanis, and Timothy Hebert

***Abstract*—An electrostatic potential field (EPF) path planner is combined with a two-layered fuzzy logic inference engine and implemented for real-time mobile robot navigation in a 2-D dynamic environment. The environment is first mapped into a resistor network; an electrostatic potential field is then created through current injection into the network. The path of maximum current through the network corresponds to the approximately optimum path in the environment. The first layer of the fuzzy logic inference engine performs sensor fusion from sensor readings into a fuzzy variable, *collision*, providing information about possible collisions in four directions, *front*, *back*, *left*, and *right*. The second layer guarantees collision avoidance with dynamic obstacles while following the trajectory generated by the electrostatic potential field. The proposed approach is experimentally tested using the *Nomad 200* mobile robot.**

***Index Terms*—Collision avoidance, dynamic environment, fuzzy logic, mobile robots, navigation, path planning, potential fields, sensor fusion.**

I. Introduction

This paper is the natural outgrowth of recently published research [44]. It presents a novel approach to solving the autonomous mobile robot (AMR) navigation problem in 2-D dynamic environments, by combining the electrostatic potential field (EPF) path planner (already presented in [44]) with a two-layered fuzzy logic (FL) inference engine, operating in tandem to plan, replan, and execute a collision free path in real-time. Tasks are performed by the object detection, localization, path planning, and collision avoidance modules, including on-line sonar sensor-based environment map generation and trajectory following [17].

The main idea and contribution for the proposed EPF/FL planner is to combine “planned and reactive behavior.” Given a 2-D environment (with initial information from potentially existing environment *a priori* maps and on-line sonar sensor data), the EPF plans the initial trajectory and starts executing it via the motion control module (see Fig. 1). Once the object detection module (working in parallel with the EPF) detects through sensor readings a “high collision possibility,” it forces the motion control module to “forget” the initial EPF path, take corrective actions in terms of robot steering and robot speed to avoid the collision, until new sensor readings dictate a “low” or “not-possible” collision possibility (FL-reactive). Then, the motion control module takes into account the initial trajectory as computed at this time instant by the EPF planner. The EPF planner is reinvoked every time the environment map is updated. An additional contribution of the proposed approach is that it considers a complete sensor-based model of the robot environment and makes no assumption regarding the location or trajectory of the obstacles. In further detail, the EPF/FL planner works as follows.

Using approximate cell decomposition, the environment is first mapped onto a resistor network allowing a current source and a current sink to be placed at the initial and goal positions, respectively. Each (square) cell is actually represented by a node with eight resistors connected to the neighboring cells, unless the cell is located in the boundary of the map in which case those resistors on the outer edges are left open circuited. The current flow through the network establishes a true potential field, whose negative gradient may be followed in a “quickest descent” method to generate in real-time, (approximately) optimal local minima free trajectories (static environments), which may be modified at each sampling instant to account for dynamic obstacles. Thus, a completely replanned path may be generated online.

The EPF-generated global path is combined with sonar sensor information in a two-layer FL inference engine. The first layer of the FL inference engine performs sensor fusion from sensor readings into the linguistic variable collision, providing information about potential *collisions* in four directions *front*, *back*, *left*, and *right*. The second layer guarantees collision avoidance with dynamic obstacles while following the trajectory generated by the potential field.

No assumptions are made on the amount of information contained in the environment *a priori* map (it may be completely empty) and on the shape of obstacles and their velocities. Environment map resolution depends on the “size” of the smallest possible grid cell. The EPF generated path complexity is linear with respect to the obstacle edges number within the environment [44]. Implementation on Nomadic Inc.’s *Cognos* software development system and the *Nomad 200* mobile robot platform [16] and experimental results demonstrate the effectiveness of the individual EPF, FL, and the combined EPF/FL approaches.

A comprehensive study of the problem and a survey of techniques used for navigational planning is given in [41], while a long list of additional references is given in [9] and [17]. Global planners may be classified into roadmaps (visibility graphs, Voronoi diagrams, freeway net, and silhouette) [21]–[27], cell decomposition approaches (exact and approximate) [28]–[31], and artificial potential field (APF) approaches [4], [6]–[10], [17]–[19], [33]. The actual electrostatic potential field [32], [36], [37] and the magnetic field [35] have been also used to solve specific navigation problems.

Manuscript received August 10, 1999; revised December 20, 2000. This paper was recommended for publication by Associate Editor E. Pagello and Editor A. De Luca upon evaluation of the reviewers’ comments. This work was supported in part by the Hellenic General Secretariat of Research and Technology under Grant PENED99 and Grant PAVE99-BE118.

N. C. Tsourveloudis and K. P. Valavanis are with the Department of Production Engineering and Management, Technical University of Crete, Chania 73100, Crete, Greece (e-mail: nikost@dpem.tuc.gr; kimonv@dpem.tuc.gr).

T. Hebert was with the University of Louisiana at Lafayette, CACS, Lafayette, LA 70504 USA.

Publisher Item Identifier S 1042-296X(01)08889-9.

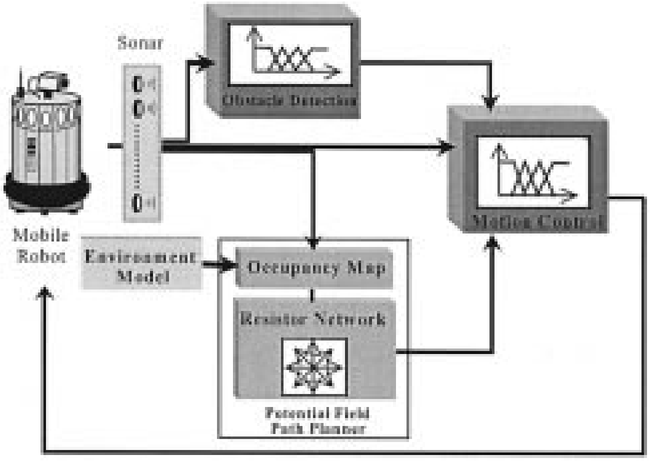

Fig. 1. The navigation architecture.

Approaches to fuzzy navigation in dynamic environments follow either a *classical paradigm* or a *behavior-based paradigm* [1]–[3], [11]–[15], [42]. Saffiotti provides in [20] a comprehensive study to the FL-based autonomous robot navigation problem.

The rest of the paper is organized as follows. Section II discusses the basic ideas related to the generation of the natural potential field and FL-controlled navigation, as well as the computational complexity of the proposed EPF path planner. Section III discusses experimental results, while Section IV offers a comparison of the computational complexities of several reviewed approaches, thus providing justifications for the proposed solution, at least in terms of computational complexity efficiency. Section V concludes the paper.

II. Proposed Method of Solution

The EPF and the two-layered FL inference engine operate in tandem (Fig. 1). An environment *occupancy map* is first created and then mapped onto a *resistor network* used to derive and interpret the EPF. The EPF goal-driven navigation generates in real-time minimum occupancy trajectories. The FL inference engine provides a comprehensive sensor fusion approach that interprets the potential field results in light of the current local environment situation, thus allowing both reactive and reflexive navigation. The *obstacle detection module* outputs the position and the degree of possibility with which any collision may occur. This information is combined in the *motion control module* with the *path planner* output. If an obstacle blocks the planned path, a collision avoidance decision that affects the robot heading and speed is taken.

A. The Natural Potential Field

The solution to the navigation problem may be compared to the flow of electric current within a sheet of conducting material; a set mapping is performed from the real environment into a discrete electric circuit (resistance). Obstacles are mapped into a discrete resistor network. *The path of minimum resistance within the circuit maps into a path of minimum occupancy within the environment.* The algorithm to create the natural potential field follows three steps: 1) create an *occupancy map* of the environment; 2) create the *resistor network*; and 3) solve the *resistor network* to obtain the *potential field*.

Each cell is mapped onto a resistor network (Fig. 2) by replacing each cell in the occupancy map with a set of eight resistors, each connected at a central point. The only exceptions are the cells on the outside edge(s) and corners with five resistors and three resistors respectively connected. The value of the resistors is determined by the value of the corresponding cell in the occupancy map. Since a minimum resistance path in the network is followed by maximum potential drop, reversing the mapping generates an optimal path in the environment corresponding to a minimum occupancy path. The path always begins at the highest potential (initial vehicle position), and ends at the lowest potential (final destination). Hereafter, references to an optimal path imply a path of minimum occupancy.

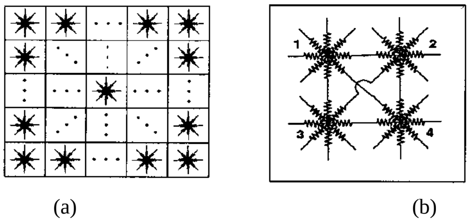

Fig. 2. (a) An n-by-n node resistor network and (b) a 2-by-2 node detail.

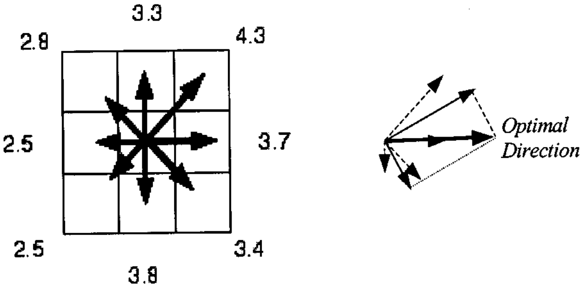

Fig. 3. Vector representation of an EPF solution.

Formally defined, consider an environment map, which contains obstacles of various shapes and sizes. The initial position of the robot is q_o and the destination point is q_f. Assume a square, (always) bounded region centered about q_o which includes q_f and can be divided into an $n \times n$ grid, X. The grid is discretely represented by the *occupancy* matrix O, where the value of each entry is the percentage of the area of the grid cell, occupied by obstacles of the environment map. Details on the construction of the occupancy matrix can be found in [44]. To determine a desired direction of travel from the EPF, a vector is associated with each cell connected to the cell containing q_0 with magnitude equal to the amount of current flowing through the specified branch. If the resistance between the central node and all of its neighbor nodes is equal, then the potential drop can be used in place of the current. The sum of these vectors is then reported to be the direction of travel along the minimum occupancy path. Fig. 3 shows the vectors created through a solution of the resistor network. The current magnitudes along each branch are shown at the end of the associated vector. The geometric sum of these forces results in a single vector pointing in the direction of the optimal path, as shown in Fig. 3. The overall path generated by the resistor network is, in reality, "an approximation" to the real optimum path. Increasing the connectivity of the cells or reducing their size (if possible) may increase the accuracy of the generated path. It has been observed that, if the size of the square cell is lower-bounded by the size of the real robot (56 cm bumper-to-bumper for the Nomad 200), the accuracy of the generated path is poor.

1) Interpretation of the EPF: The criterion used for determining the path optimality is the total occupancy of the path swept by the robot as it follows the trajectories, or the total *swept occupancy*. Each square unit of area in the environment is assigned a minimum occupancy. Highly cluttered areas are assigned a larger occupancy value. This criterion for optimality is superior to a simple distance criterion when the algorithm is to be implemented in a real environment. A planned path, which minimizes distance, tends to drive the robot arbitrarily close to any obstacles between the robot and the goal point. Minimizing swept

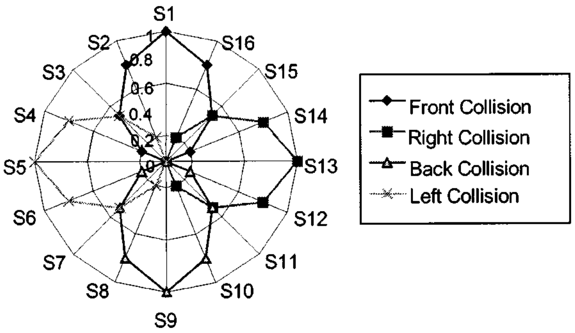

Fig. 4. The sonar arrangement and relative importance to collision detection.

occupancy, the EPF path planner avoids the areas close to the object, which increase the total swept occupancy.

2) Computational Complexity of the EPF: The complexity C of the total EPF-based solution is [44]:

$$C = mn_M + m(4n_M + 1)size^2 + 3size^2 + size. \qquad (1)$$

Removing the nonvariable terms, (1) reduces to $\text{O}((4size^2)mn_M)) = \text{O}(mn_M)$ which is linear with respect to the variables present, the number of polygons in the space, m, and the maximum number of sides of any polygon, n_M.

B. Fuzzy Logic Inference Engine

The navigation control for the AMR is generated through a two-layered FL inference engine. The first layer is primarily responsible for obstacle detection and performs sensor data fusion from sonar sensor readings. Each reading is represented as two different fuzzy variables: *sensor_direction* and *sensor_distance*. The variable *sensor_direction* has four different values that describe the sensor's membership in four cardinal relative directions: *front_collision*, *left_collision*, *back_collision*, and *right_collision*. Each of the collision values is represented as a fuzzy variable with values *not_possible*, *possible*, and *high*. The complete local environment of the vehicle is represented in the variable *collision*. Any obstacle that is close to the AMR in any direction is represented in the *collision* variable. The second layer receives the output of the first inference, representing the immediate collision possibilities, as well as the output of the artificial potential field and the speed of the robot as input and generates the control: *change of speed* and *steering*.

A mobile robot, the Nomad 200, with a ring of 16 sonar sensors arranged at 22.5° apart, covering a full 360° circle is used for experimental purposes. Every sonar sensor is fixed in its relative location to the front of the vehicle. Therefore, *sensor_direction* is a function only of the number of the sonar. Sonar S1 always belongs to *front_collision* with weight 1.0 and sonar S2 always belongs to *front_collision* with weight 0.8 and *left_collision* 0.2, as shown in Fig. 4. To simplify the implementation, *sensor_direction* has not been used but instead each sonar has been assigned to at least one but no more than two values of the variable *collision* (e.g., sensor 2 belongs only to left and front, sensor 5 belongs only to left, etc). The obstacle detection inference system is shown in Fig. 5.

The rule base of the obstacle detection controller contains rules of the following two types:

R1) IF d_i is $LD^{(k)}$ THEN c_j is $LC^{(k)}$;
R2) IF d_i is $LD^{(k)}$ AND d_{i-1} is $LD^{(k)}$ d_{i+1} is $LD^{(k)}$ THEN c_j is $LC^{(k)}$;

where k is the rule number, d_i represents the readings of sensor i, $LD^{(k)}$ is a linguistic value of the term set $D = \{$*Close, Near, Far*$\}$, c_j is the collision of type j ($j \in \{$*Front, Left, Right, Back*$\}$), and $LC^{(k)}$ is a linguistic value of the term set $C = \{$*Not-Possible, Possible, High*$\}$. The whole rule base is presented in [17]. Some of the rules for the collision of type $j =$ *front*, are

IF d_1 is *Close* THEN *front* collision is *High*.
IF d_1 is *Far* AND d_2 is *Far* AND d_{16} is Far THEN *front* collision is *Not-Possible*.
IF d_2 is *Near* THEN *front* collision is *Possible*.
IF d_1 is *Near* AND d_2 is *High* AND d_{16} is *Far* THEN *front* collision is *Possible*.
IF d_1 is *Close* THEN *front* collision is *High*.
IF d_1 is Far AND d_2 is *Far* AND d_{16} is *Far* THEN *front* collision is *Not-Possible*.
IF d_2 is *Near* THEN *front* collision is *Possible*.
IF d_1 is *Near* AND d_2 is *Far* AND d_{16} is *Far* THEN *front* collision is *Possible*.

The mathematical meaning of the kth single antecedent rule (type R1) is given as a fuzzy relation $R^{(k)}$ on $D \times C$, which in the membership functions domain is

$$\mu_{R^{(k)}}(d_i, c_j) = f_{\rightarrow}\left[\mu_{LD^{(k)}}(d_i), \mu_{LC^{(k)}}(c_j)\right] \qquad (2)$$

where $f_{\rightarrow} = \min$ for rules of Mamdani type. The membership function of the obstacle's position and distance from the robot is computed by the max–min composition [43] between the sensor readings, which represent the distance from the obstacles and the fuzzy relation described by (2). The second layer fuzzy controller, shown in Fig. 6, takes as input the variables: *collision*, *angle_error*, and *speed* and generates the control *change of speed* and *steering*. The potential field generates a heading directive based upon the sum of forces approach described in the previous section. The heading relative to the current heading of the AMR is given as the *angle_error* θ. In most surveyed approaches, this angle represents the angle to the goal point. In this case, it represents the angle in which the vehicle should point in order to follow the desired path to the goal point. Since the desired heading given by the potential field already considers some reactive navigation, the second-layer fuzzy inference should implement the *steering* necessary to reduce the *angle_error* to zero, only changing the desired heading if collision in the vicinity of the desired path is *possible*.

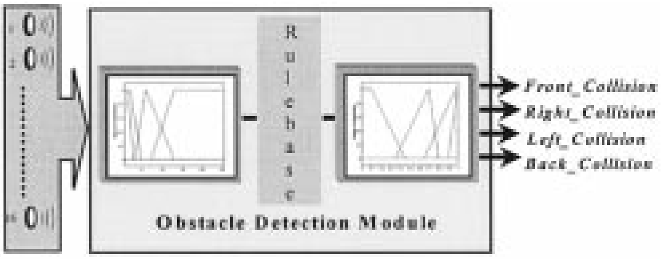

Fig. 5. The obstacle detection controller implementing sensor fusion.

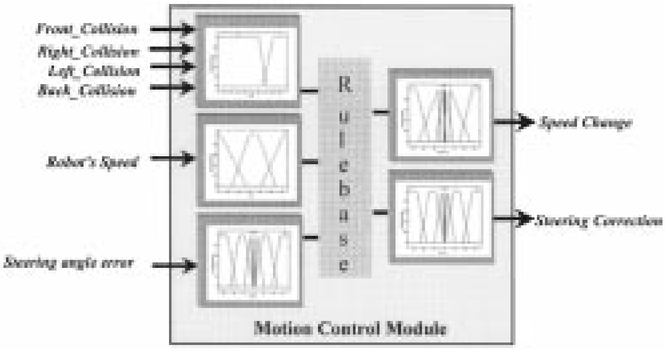

Fig. 6. The second-layer fuzzy controller.

TABLE I
PART OF THE RULE BASE OF THE MOTION CONTROL MODULE

INPUT Variables						**OUTPUT Variables**	
Collision				***Speed***	***Steering Angle Error***	***Change-of-Speed***	***Steering Angle Correction***
Front	**Left**	**Back**	**Right**				
N-P	N-P	N-P	N-P	Low	RB	NC	TLF
N-P	N-P	N-P	N-P	Normal	R	NC	TL
N-P	N-P	N-P	N-P	Normal	Zero	ACC-F	NC
N-P	High	N-P	N-P	High	LS	DEC-SL	TRS
N-P	N-P	N-P	High	Low	LB	NC	TRS
High	High	N-P	N-P	Low	Zero	NC	TRF
High	N-P	High	High	Normal	RS	DEC-F	TLF
N-P	High	High	N-P	Normal	R	DEC-SL	NC
N-P	High	N-P	High	High	RB	DEC-SL	NC
High	High	High	N-P	High	Zero	DEC-F	TRF
High	N-P	High	N-P	Low	LS	NC	TR

Goal-directed behaviors are controlled through the steering *angle_error* variable. The potential field generates the best path to achieve the goal position. Reaction-directed behaviors are controlled through analysis of the collision of the current situation. For example, IF *collision* is *front possible* AND *speed* is *normal* THEN *change of speed* is *decelerate slow*. A similar rule, which examines the *angle_error* as well as all values of *collision*, generates the control *steering*.

The rules of the second fuzzy module can be described compactly as follows.

IF c_j is $LC^{(k)}$ AND θ is $L\Theta^{(k)}$ AND v is $LV^{(k)}$ THEN s is $LS^{(k)}$ AND dv is $LDV^{(k)}$

where k is the rule number, c_j is collision of type j, i.e., the output of the obstacle detection module, θ is the steering angle error, v is the speed of the robot, s is the steering angle correction, dv is the change of speed, and $LC^{(k)}$, $L\Theta^{(k)}$, and $LV^{(k)}LS^{(k)}LDV^{(k)}$ are the linguistic values of c_j, θ, v, s, and dv, respectively. AND $= \min$ in all rules. The steering angle error θ is computed by continuously comparing the desired steering angle, θ_{desired}, i.e., the angle that guides the robot to the target point given, with the actual steering angle, θ_{actual}. The steering error is important when the robot detects no collision on the target path. In case of possible collision, the information about the error θ becomes of less importance and the rules that contain such information are firing with smaller strength than the rules which perform collision avoidance. The generic mathematical expression of the kth navigation rule is

$$\begin{aligned}&\mu_{R^{(k)}}(c_j, \theta, v, s, dv)\\ &\quad = \min\left[\mu_{LC^{(k)}}(c_j), \mu_{L\Theta^{(k)}}(\theta),\right.\\ &\qquad\left.\mu_{LV^{(k)}}(v), \mu_{LS^{(k)}}(s), \mu_{LdV^{(k)}}(dv)\right]. \qquad (3)\end{aligned}$$

The overall navigation output is given by the max–min composition [43] and in particular is

$$\begin{aligned}&\mu_N^*(s, dv)\\ &\quad = \max \min_{c_j, \theta, v} \left[\mu_{AND}^*(c_j, \theta, v), \mu_R(c_j, \theta, v, s, dv),\right] \qquad (4)\end{aligned}$$

where

$$\mu_R(c_j, \theta, v, s, dv) = \bigcup_{k=1}^{K} \mu_{R^{(k)}}(c_j, \theta, v, s, dv)$$

and $\mu_{AND}^*(c_j, \theta, v)$ is the minimum of the fuzzified sonar readings. The navigation action dictates change in robot speed and/or steering correction and it comes out from a defuzzification formula, which calculates the center of the area covered by the membership function computed from (4). Part of the rule base is shown in Table I.

III. RESULTS AND COMPARISONS

The proposed navigation system has been implemented on the Nomad 200 robot using the *Cognos* development software package that provides a communication link with the actual mobile robot [16]. Matlab's stand-alone fuzzy inference engine has been used to obtain the results of the fuzzy system bypassing the FL toolbox Graphical User Interface (GUI), thus greatly increasing the simulation processing speed. Simulation results resemble experimental results as explicitly shown in [17] for several case studies. The robot sensor readings are loaded as inputs into the first layer of the fuzzy inference engine. The results of the first inference are concatenated with the vehicle speed and angle error to form the input vector to the second-layer fuzzy inference engine. The results of this engine are utilized to issue commands to the mobile platform. The translational velocity and the change in heading of the mobile platform are controlled. In all graphs, distance is measured in meters and speed in cm/sec. Collision possibility 1 means that the robot will hit the obstacle, while 0 means that there are no obstacles within the sonar range.

A. Simulated Dynamic Environment

A case of a simulated dynamic environment with hidden obstacle is shown in Figs. 7–9 to provide the clearest justification for an online global path planner. The motion of the obstacle begins when the robot reaches point A, and it slides into its final location. It is assumed that the EPF path planner has complete knowledge of the dynamic object, including its velocity vector. This allows the robot to completely avoid the "blocked" area. Complete knowledge of the environment is unlikely in dynamic situations, making this particular behavior more of an ideal than a reality and not suitable in rapidly changing environments.

In the FL approach shown in Fig. 8, the robot has no prior knowledge of the environment. As the robot reaches point A and turns along its path to the final goal, the obstacle slides down blocking the path. The robot does not detect the blocking obstacle until it reaches point B at which time it turns to the right—the side with the lowest collision possibility. Since the goal is now to the left of the robot, it continues to track in a counterclockwise direction along the object until its path is blocked by the obstacle at point C. After turning around at point C the

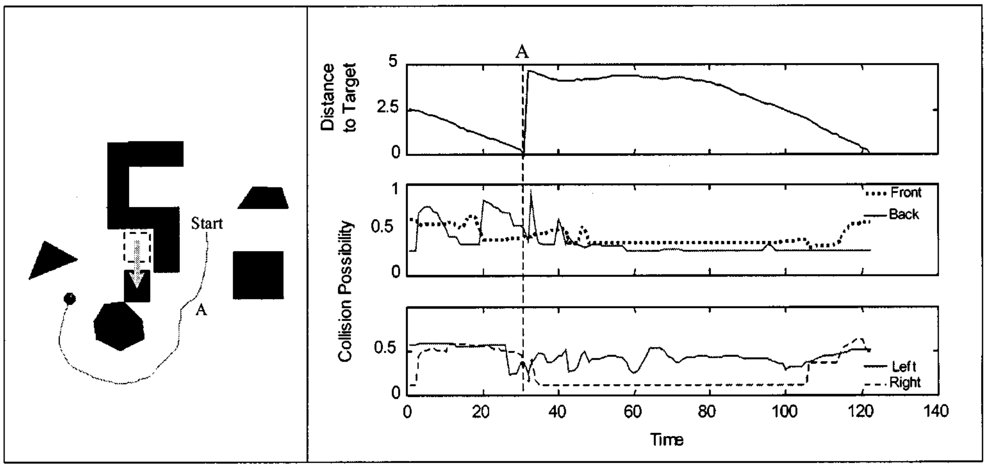

Fig. 7. Simulated dynamic environment test case: the EPF approach.

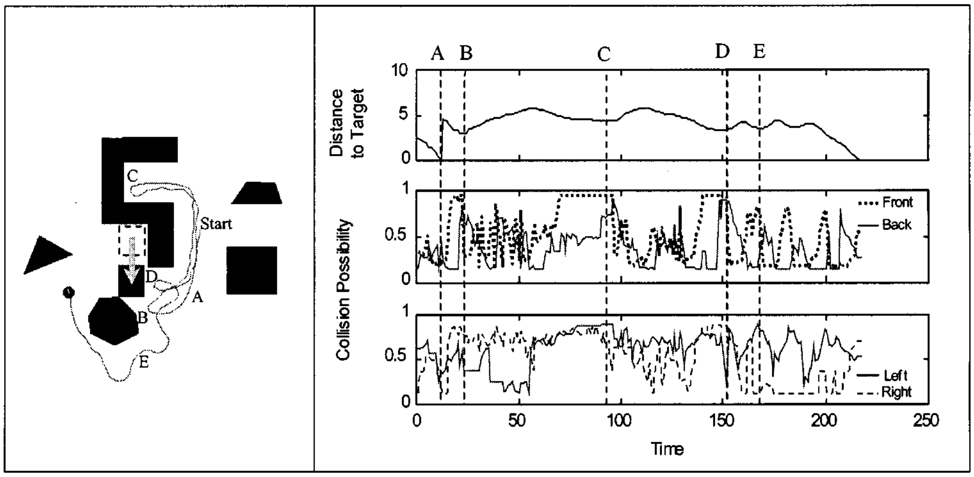

Fig. 8. Simulated dynamic environment test case: the EPF approach.

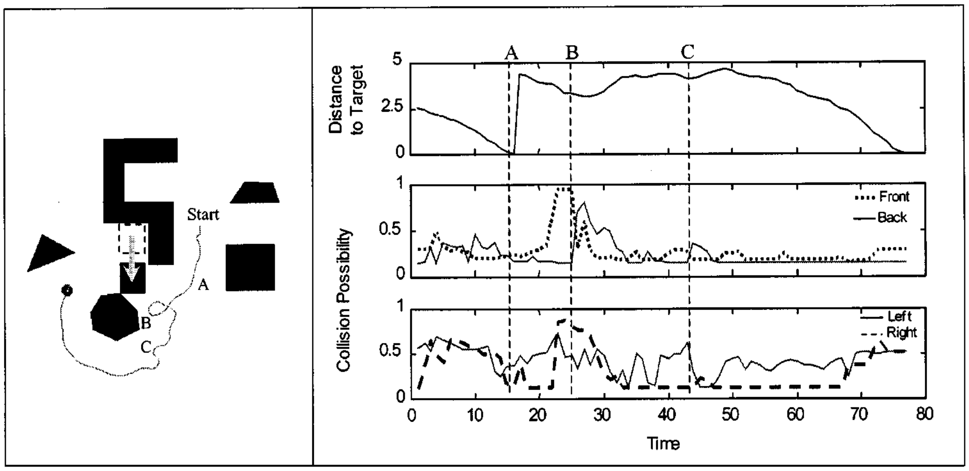

Fig. 9. Simulated dynamic environment test case: the combined EPF/FL approach.

goal is again to the right of the robot. The robot now moves clockwise around the object through points D and E to the final goal.

Fig. 9 shows the combined EPF/FL approaches and demonstrates its advantage. After the object blocks the robot's path, the robot turns to the right toward the side of smallest collision. After this, the EPF planner redirects the robot downwards instead of allowing the robot to continue to track around the obstacle. Following the EPF directives, the robot moves through point C to the final goal.

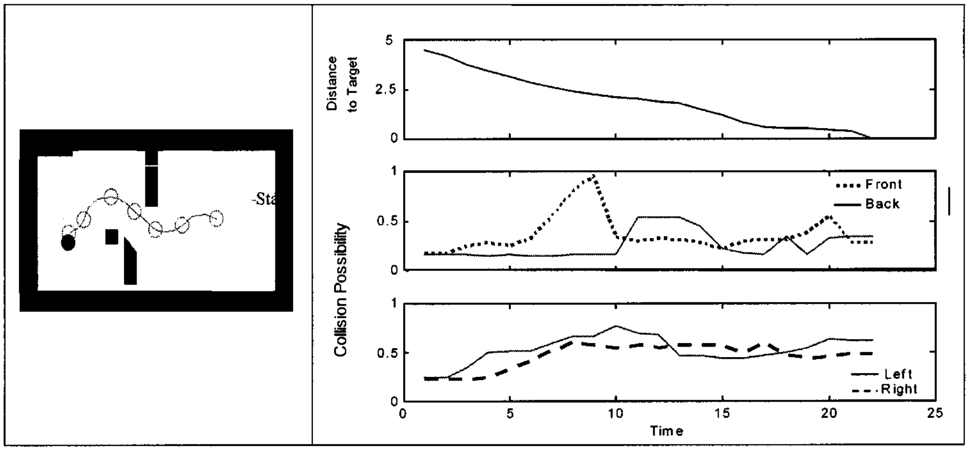

Fig. 10. Experimental test case for the EPF approach.

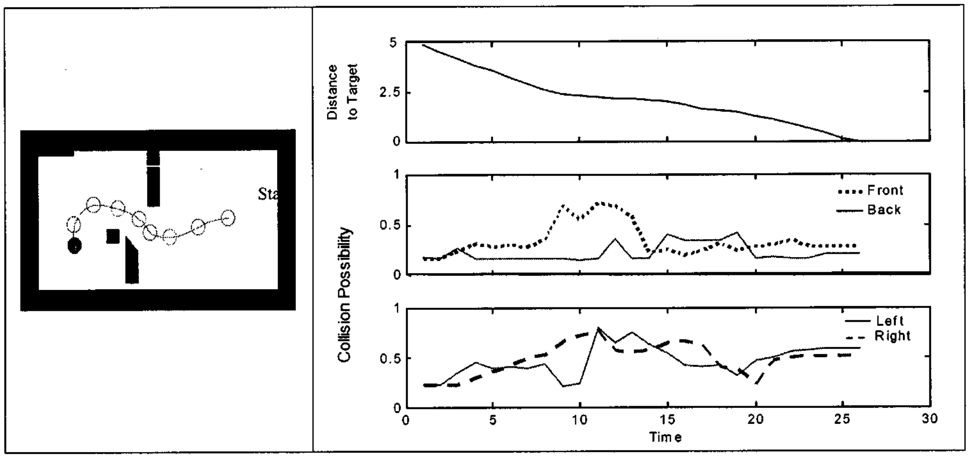

Fig. 11. Experimental test case for the FL approach.

B. Experimental Results

Real-time experimental results have been obtained using the Nomad 200 in a laboratory environment. The room is a clean environment with rectangular obstacles and measures approximately 7.6 m by 3.8 m. The grid size used to calculate the EPF based path was set to be 13 × 13 giving a sampling rate of approximately 1 s.

A grid size of 11 × 11 reduces the sampling rate to approximately 0.65 s. The objects were placed in the room, their position measured, and the result placed in a map, which can be viewed by the *Cognos* software package. The robot is localized within the room before each test run. The position of the robot is recorded at regular time intervals and the subsequent path is displayed through the *Cognos's* GUI. During the test executions, all programs were run, under the *Linux* operating system, directly on the main processor of the robot, a Pentium 133. A test case is presented in Figs. 10–12.

During experimentation we compared the navigation results using the *Cognos* simulation package and the Nomad 200 robot for a variety of test cases. For both simulated and real-time results, the combined EPF/FL navigation approach has been used. The initial and final positions were the same with a single intermediate goal point. The path generated in both simulations and real-time tests is seen to be the same; however, the performance of the simulation and the real robot is different. The simulation has a faster cycle time for the iterations of the EPF/FL approach. This allows the robot to follow the smoother path. The reactions of the robot in the simulation are also different from the real robot. For instance, once a speed command is given to the real robot, the set point is obtained over a period of time. In the simulation, the speed set point is almost immediately achieved.

C. Discussion

The simulation and experimental results show that the pure FL implementation provided very good reaction to the presence of static and dynamic obstacles within the robot's immediate environment. The EPF approach provided well-planned paths that (sometimes) brought the robot very close to edges of obstacles; however, the EPF path planner was sometimes very slow to react to the presence of unknown moving obstacles. The combined EPF/FL approach allowed the potential field to plan the paths and allowed the fuzzy inference to implement the paths while avoiding collision with all obstacles. The combined EPF/FL approach benefits from the advantages of both approaches.

The combined EPF/FL approach finished first in about 80% of the test situations. The EPF approach finished first in 10% of the test cases and second in 20% of the test cases, providing the shortest path distance in 60% of the test cases. In no test case did the approach that provided the shortest distance path finish first, as detailed in [17].

Considering a dynamic environment, the EPF solution usually finishes the path later than the other two approaches. Two different speed control laws are implemented among the three different approaches.

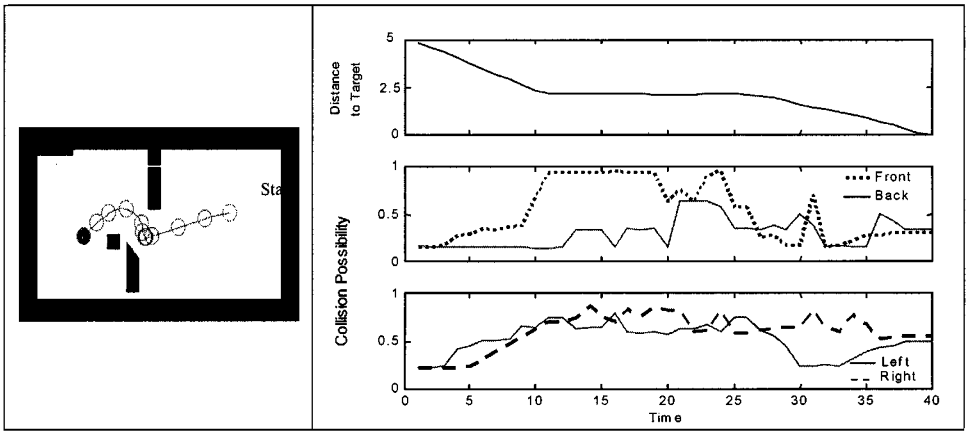

Fig. 12. Experimental test case for the EPF/FL approach.

The FL inference and the combined EPF/FL approach utilize the fuzzy rule base to generate the speed control. The potential field implementation does not have access to the fuzzy inference, and thus a different control must be utilized. It was realized early in the experimentation that the potential field reacts more slowly to moving objects that may collide with the robot. Thus, to help avoid the objects, the maximum speed and the average speed of the robot is much more restricted than in the pure fuzzy and combined EPF/FL cases.

The presented fuzzy motion control inference uses only the desired change to the heading, thus resembling a classical error proportional controller. In the test cases presented, a sinusoidal-like path is observed as the robot moves through otherwise straight path segments. This behavior resembles the behavior of a proportional-to-error P-type controller under the influence of a rapidly changing set point. The overall performance of the combined EFP/FL approach may be enhanced through the addition of one more input which is the derivative of the change of the setpoint. This makes the inference resemble a proportional-derivative PD-type controller and may allow the EPF setpoints to be attained faster with a resulting smoother path.

In all tests, we have used fuzzy inference systems of the so-called Mamdani-type. The membership functions are subjectively specified in an *ad hoc* manner from observation and experience and tuned by trial-and-error, since there is no analytical method (for the Mamdani-type inference) that guarantees optimal selection. If one uses model-based inference (such the Takagi–Sugeno [43] representation), then a variety of (gradient-based) methods for the determination of the memberships can be employed. Indeed, there are several approaches in the literature that use neural nets, genetic algorithms, neuro-fuzzy methods, etc., to fine-tune the membership functions. However, in our view, these methodologies have a significant drawback when used for the collision-free navigation problem. They do improve the performance of the inference, but only for known environments for which some kind of training data are available. In other words, they do not provide a robust motion behavior capable of navigating the vehicle in unknown/dynamic environments, where the FL is actually needed.

IV. Computational Complexity Comparisons

Robot motion planning computational complexity results strongly suggest that the complexity of path planning increases exponentially with the dimension of the configuration space [4], [34], [38]–[40]. Table II summarizes the computational complexities of several roadmap and cell decomposition approaches and of the potential panel-based path planning method introduced by Zhang and Valavanis [9], [18], [19].

TABLE II
Comparison of Computational Complexities of Different Approaches

Approach	***Computational Complexity***
Roadmap Path Planning	$O(n_c^2)$ [21] [22] $O(n_c \log n_c + n_e)$ [23] [24] $O(n_c(m + \log n_c))$ [25] $O(m^2 + n_c \log n_c)$ [26] [27]
Cell Decomposition	$O(n_t^5)$ [28] $O(n_t^2)$ [29] $O(n_t \log^2 n_t)$ [29] $O(n_t^2 \log n_t)$ [30] $O(n^6_{e1} n^6_{e2} \alpha(n_{e1} n_{e2}))$ [31]
Potential Panel-Based Path Planning Method	$O(m^3 + km)$ [9]
EPF Method	$O(mn_M)$ [17] [44]

The comparison of the different approaches is in terms of n_c, the total number of corners of all obstacles, n_e, the total number of edges of all obstacles, n_t, the total number of edges and vertices of all obstacles, n_{e1} the total number of edges of a polygonal object, n_{e2}, the total number of edges of all polygonal obstacles, m, the total number of obstacles in the workspace, and k, the total number of the step displacements on a generated path. A complete discussion may be found in [9] and [17], along with extensions covering 3-D environments.

V. Conclusion

Several advantages of the natural potential field approach are quickly observed over other potential field solutions.

1) An analysis of traditional electrostatic theory reveals that in a network composed solely of resistors with positive resistance values, the system of equations that describe the network and are based on Kirchhoff's Laws is linearly independent.
2) The collision-free paths generated from the electrostatic potential field, necessarily lead to the goal position. Further, local minima are not generated within the field, and stagnation points within the field do not exist.
3) The electrostatic potential field approach selected generates an optimal, minimum occupancy path.

4) The proposed solution accomplishes real-time path generation. The fuzzy logic controller interprets the potential path and performs sensor fusion in order to increase the ability of the vehicle to react to dynamic obstacles.

In general, the EPF path planner performs best in cluttered environments. If the potential field has several obstacles grouped in one location of the grid, the large open area of the network tends to pull the path further away from the obstacle(s) and into the open area. Relatively thin obstacles do not provide the EPF with enough information to consistently generate collision-free paths. The FL controller incorporates omni-directional sensing, which allows the vehicle to detect and react quickly to any obstacle in the environment, regardless of the placement of the obstacle relative to the robot.

REFERENCES

[1] J. Zhang and P. Bohner, "A fuzzy control approach for executing subgoal guided motion of a mobile robot in a partially-known environment," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1993, pp. 545–550.

[2] M. Zhang, S. Peng, and Q. Meng, "Neural network and fuzzy logic techniques based collision avoidance for a mobile robot," *Robotica*, pp. 627–632, 1997.

[3] M. Alwan, P. Cheung, A. Saleh, and N. C. Obeid, "Combining goal-directed, reactive and reflexive navigation in autonomous mobile robots," in *Proc. Australian–New Zealand Conf. on Intelligent Information Systems*, 1996, pp. 247–250.

[4] J. C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer, 1991.

[5] I. Connolly and J. B. Burns, "Path planning using Laplace's equation," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 3, 1990, pp. 2102–2106.

[6] Y. K. Hwand and N. Ahuja, "A potential field approach to path planning," *IEEE Trans. Robot. Automat.*, vol. 6, pp. 23–32, Feb. 1992.

[7] Y. Kitamura, T. Tanaka, F. Kishino, and M. Yachida, "3-D planning in a dynamic environment using an doctree and an artificial potential field," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, vol. 2, 1995, pp. 474–479.

[8] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 501–518, Oct. 1992.

[9] Y. Zhang, "Sensor-based potential panel method for robot motion planning," Ph.D. dissertation, University of Southwestern Louisiana, 1995.

[10] J. O. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 338–349, June 1992.

[11] J. Zhang, F. Wille, and A. Knoll, "Fuzzy logic rules for mapping sensor data to robot control," in *Proc. EUROBOT '96*, 1996, pp. 29–38.

[12] W. Li, "Perception-action behavior control of a mobile robot in uncertain environments using fuzzy logic," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 1, 1994, pp. 439–446.

[13] A. Ramirez-Serrano and M. Boumedine, "Real-time navigation in unknown environments using fuzzy logic and ultrasonic sensing," in *Proc. 1996 IEEE Int. Symp. on Intelligent Control*, 1996, pp. 26–30.

[14] F. G. Pin and Y. Watanabe, "Using fuzzy behaviors for the outside navigation of a car with low-resolution sensors," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1993, pp. 548–553.

[15] Y. Chee, S. Y. T. Lang, and P. W. T. Tse, "Fuzzy mobile robot navigation and sensor integration," in *Proc. Fifth IEEE Int. Conf. on Fuzzy Systems*, vol. 1, 1996, pp. 7–12.

[16] Nomadic Technologies, Inc., *Nomad 200: User's manual: Software version: 2.6*, 1996.

[17] T. Hebert, "Navigation of an autonomous vehicle using a combined electrostatic potential field/fuzzy inference approach," Ph.D. dissertation, University of Southwestern Louisiana, 1998.

[18] Y. Zhang and K. P. Valavanis, "Sensor-based 2-D potential panel method for robot motion planning," *Robotica*, pt. 1, vol. 14, pp. 81–89, 1996.

[19] ——, "A 3-D potential panel method for robot motion planning," *Robotica*, pt. 5, vol. 15, pp. 421–434, 1997.

[20] A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Computing*, vol. 1, no. 4, pp. 180–197, 1997.

[21] T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility of disjoint polygons," *Algorithmica*, vol. 1, pp. 46–63, 1986.

[22] E. Welzl, "Constructing the visibility graph for n line segments in $O(N^2)$ time," *Inform. Processing Lett.*, vol. 20, pp. 167–171, 1985.

[23] M. Fredman and R. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *J. ACM*, vol. 34, no. 3, pp. 596–615, 1987.

[24] S. Ghosh and D. Mount, "An output sensitive algorithm for computing visibility graphs," in *Proc. 28th IEEE Symp. on Foundations of Computer Science*, 1987, pp. 11–19.

[25] J. Reif and J. Storer, "Shortest paths in Euclidean space with polyhedral obstacles," Brandeis University, TR-05-85, 1985.

[26] J. Mitchell, "Planning shortest paths," Stanford University, Res. Rep. 561, 1986.

[27] H. Rohnert, "Shortest paths in the plane with convex polygonal obstacles," *Inform. Processing Lett.*, vol. 23, pp. 71–76, 1986.

[28] J. T. Schwartz and M. Sharir, "On the 'piano movers' problem I: The case of a 2-dimensional rigid polygonal body moving amidst polygonal barriers," *Commun. Pure Appl. Math.*, vol. 36, pp. 345–398, 1983.

[29] A. van der Stappen, D. Halperin, and M. Overmars, "Efficient algorithms for exact motion planning amidst fat obstacles," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1993, pp. 297–304.

[30] D. Leven and M. Sharir, "An efficient and simple motion planning algorithm for a ladder moving in 2-dimensional space amidst polygonal barriers," in *Proc. First ACM Symp. on Computational Geometry*, 1985, pp. 211–227.

[31] F. Avnaim, J. Boissonnat, and B. Faverjon, "A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1988, pp. 1656–1661.

[32] J. Hutchinson, C. Koch, J. Lea, and C. Mead, "Computing motion using analog and binary resistive networks," *IEEE Computer Mag.*, vol. 21, pp. 52–63, 1988.

[33] J. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1991, pp. 790–796.

[34] J. Reif, "Complexity of the mover's problem and generalizations," in *Proc. 20th Symp. on Foundations of Computer Science*, 1979, pp. 421–427.

[35] L. Singh, J. Wen, and H. Stephanou, "Motion planning and dynamic control of a linked manipulator using modified magnetic fields," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 1142–1147.

[36] L. Tarassenko and A. Blake, "Analogue computation of collision-free paths," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1991, pp. 540–545.

[37] L. Tarassenko, M. Brownlow, G. Marshall, and J. Tombs, "Real-time autonomous robot navigation using VLSI neural networks," *Advances in Neural Information Processing Systems*, pp. 422–428, 1991.

[38] J. F. Canny and J. H. Reif, "New lower bound techniques for robot motion planning problems," in *Proc. 28th IEEE Symp. on Foundations of Computer Science*, 1987, pp. 49–60.

[39] J. H. Reif and M. Sharir, "Motion planning in the presence of moving obstacles," in *Proc. 25th IEEE Symp. on Foundations of Computer Science*, 1985, pp. 144–154.

[40] J. T. Schwartz and M. Sharir, "On the 'piano movers' problem II: General techniques for computing topological properties of real algebraic manifolds," *Advances Appl. Math.*, vol. 4, pp. 298–351, 1983.

[41] R. C. Arkin, *Behavior Based Robotics*. Cambridge, MA: MIT Press, 1998.

[42] J. Yen and N. Pfluger, "A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, pp. 971–978, 1995.

[43] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*, 2nd ed. Berlin, Germany: Springer-Verlag, 1996.

[44] K. P. Valavanis, T. Hebert, R. Kolluru, and N. C. Tsourveloudis, "Mobile robot navigation in 2-D dynamic environments using electrostatic potential fields," *IEEE Trans. Syst., Man Cybern. A*, vol. 30, pp. 187–197, 2000.