# Behavior-Based Formation Control for Multirobot Teams

Tucker Balch, *Member, IEEE*, and Ronald C. Arkin, *Senior Member, IEEE*

***Abstract*—New reactive behaviors that implement formations in multirobot teams are presented and evaluated. The formation behaviors are integrated with other navigational behaviors to enable a robotic team to reach navigational goals, avoid hazards and simultaneously remain in formation. The behaviors are implemented in simulation, on robots in the laboratory and aboard DARPA's HMMWV-based Unmanned Ground Vehicles. The technique has been integrated with the Autonomous Robot Architecture (AuRA) and the UGV Demo II architecture. The results demonstrate the value of various types of formations in autonomous, human-led and communications-restricted applications, and their appropriateness in different types of task environments.**

***Index Terms*— Autonomous robots, behavior-based control, robot formation.**

## I. Introduction

THIS article presents a behavior-based approach to robot formation-keeping. Since behavior-based systems integrate several goal oriented behaviors simultaneously, systems using this technique are able to navigate to waypoints, avoid hazards and keep formation at the same time. The initial target for this work is a team of robotic vehicles intended to be fielded as a scout unit by the U.S. Army (Fig. 1). Formation is important in this and other military applications where sensor assets are limited. Formations allow individual team members to concentrate their sensors across a portion of the environment, while their partners cover the rest. Air Force fighter pilots for instance, direct their visual and radar search responsibilities depending on their position in a formation [9]. Robotic scouts also benefit by directing their sensors in different areas to ensure full coverage (Fig. 2, [7]). The approach is potentially applicable in many other domains such as search and rescue, agricultural coverage tasks and security patrols.

The robots in this research are mechanically similar, or in the case of simulation, identical. Nevertheless, they are considered heterogeneous since each robot's position in formation depends on a unique identification number (ID), i.e., heterogeneity arises from functional rather than physical differences. This is important in applications where one or more of the agents are dissimilar. In Army scout platoons for instance, the leader is not usually at the front of the formation, but in the middle, or to one side.

The formation behaviors were implemented as *motor schemas*, within the Autonomous Robot Architecture (AuRA) architecture, and as steering and speed behaviors within the Unmanned Ground Vehicle (UGV) Demo II architecture. In both cases, the individual behaviors run as concurrent asynchronous processes with each behavior representing a high-level behavioral intention of the agent. Perceptions are directly translated into a response vector in AuRA, or as turning or speed votes on the UGV. Readers are referred to [2] and [18] for more information on schema-based reactive control and the DAMN Arbiter used within the UGV Demo II architecture.

### A. Background

Formation behaviors in nature, like flocking and schooling, benefit the animals that use them in various ways. Each animal in a herd, for instance, benefits by minimizing its encounters with predators [20]. By grouping, animals also combine their sensors to maximize the chance of detecting predators or to more efficiently forage for food. Studies of flocking and schooling show that these behaviors emerge as a combination of a desire to stay in the group and yet simultaneously keep a separation distance from other members of the group [8]. Since groups of artificial agents could similarly benefit from formation tactics, robotics researchers and those in the artificial life community have drawn from these biological studies to develop formation behaviors for both simulated agents and robots. Approaches to formation generation in robots may be distinguished by their sensing requirements, their method of behavioral integration, and their commitment to preplanning. A brief review of a few of these efforts follows.

An early application of artificial formation behavior was the behavioral simulation of flocks of birds and schools of fish for computer graphics. Important results in this area originated in Craig Reynolds pioneering work [17]. He developed a simple egocentric behavioral model for flocking which is instantiated in each member of the simulated group of birds (or "boids"). The behavior consists of several separate components, including: inter-agent collision avoidance, velocity matching and flock centering. Each of the components is computed separately, then combined for movement. An important contribution of Reynold's work is the generation of successful overall group behavior while individual agents only sense their

Manuscript received July 3, 1997; revised April 15, 1998. This work was supported by the Mobile Robot Laboratory, Georgia Institute of Technology and ONR/DARPA Grant N00014-94-1-0215. This paper was recommended for publication by Associate Editor K. Kosuge and Editor V. Lumelsky upon evaluation of the reviewers' comments.

T. Balch is with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213-3891 USA.

R. C. Arkin is with the Mobile Robot Laboratory, College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280 USA.

Publisher Item Identifier S 1042-296X(98)09446-4.

1042–296X/98$10.00 © 1998 IEEE

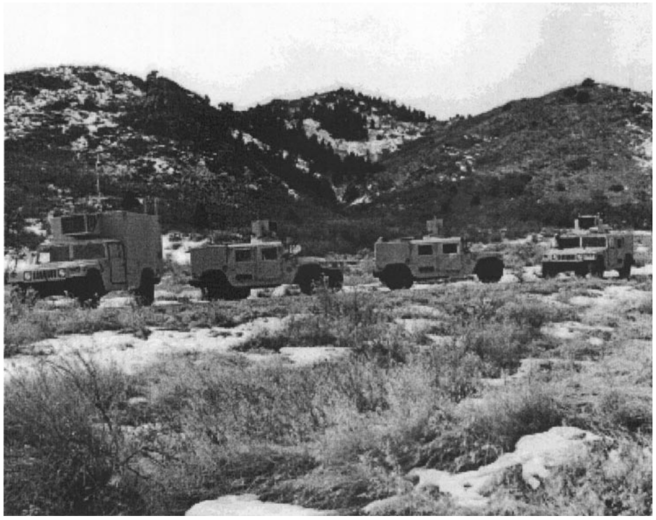


Fig. 1. A team of four robotic scout vehicles manufactured for DARPA's Demo II project. The formation techniques reported in this article were implemented on these robots. Photograph courtesy of Lockheed-Martin.

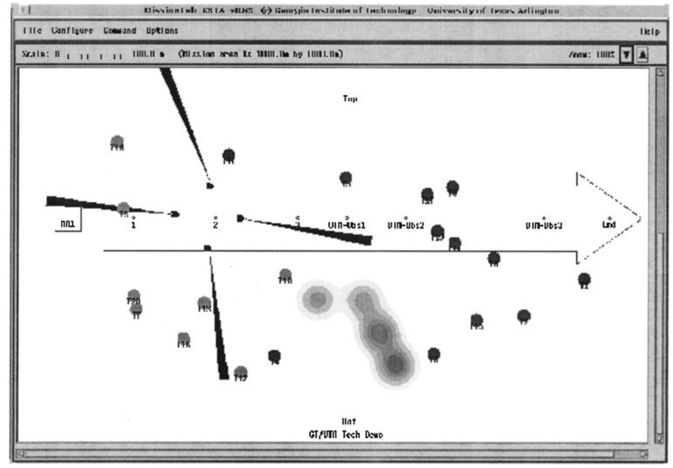


Fig. 2. An example of how scouts in formation focus their sensor assets so as to ensure complete coverage. Four robot scouts sweep from left to right in a *diamond* formation. The wedges indicate the sensor focus for each scout. Figure courtesy of Diane Cook of the University of Texas at Arlington [7].

local environment and close neighbors. Improvements to this approach have recently been made by Tu and Terzopoulos and separately by Brogan and Hodgins. Tu and Terzopoulos [17] developed more realistic simulated fish schooling by accurately modeling the animals' muscle and behavioral systems. Brogan and Hodgins [19] developed a system for realistically animating herds of one-legged agents using dynamical models of robot motion. Both results are more visually realistic than Reynolds' because they simulate the mechanics of motion; Reynolds' approach utilized particle models only.

The individual components of Reynolds' flocking and Brogan's herding behaviors are similar in philosophy to the motor schema paradigm used here, but their approaches are concerned with the generation of visually realistic flocks and herds for large numbers of simulated animals, a different problem domain than the one this article addresses. In contrast, our research studies behaviors for a small group (up to four) of mobile robots, striving to maintain a specific geometric formation.

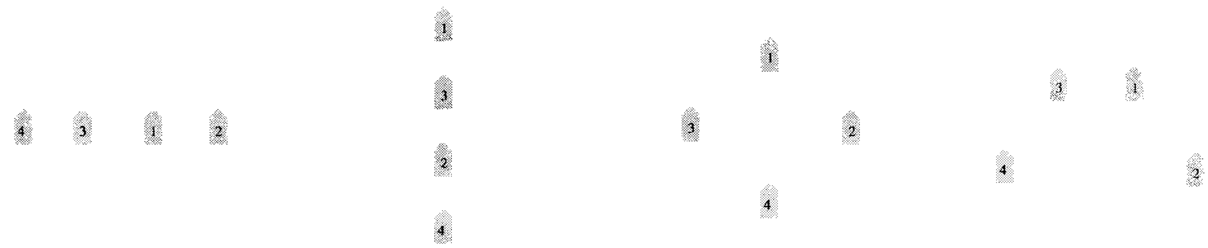
The dynamics and stability of multi-robot formations have drawn recent attention [21], [6]. Wang [21] developed a strategy for robot formations where individual robots are given specific positions to maintain relative to a leader or neighbor. Sensory requirements for these robots are reduced since they only need to know about a few other robots. Wang's analysis centered on feedback control for formation maintenance and stability of the resulting system. It did not include integrative strategies for obstacle avoidance and navigation. In work by Chen and Luh [6] formation generation by distributed control is demonstrated. Large groups of robots are shown to cooperatively move in various geometric formations. Chen's research also centered on the analysis of group dynamics and stability, and does not provide for obstacle avoidance. In the approach forwarded in this article, geometric formations are specified in a similar manner, but formation behaviors are fully integrated with obstacle avoidance and other navigation behaviors.

Mataric has also investigated emergent group behavior [13], [14]. Her work shows that simple behaviors like avoidance, aggregation and dispersion can be combined to create an emergent flocking behavior in groups of wheeled robots. Her research is in the vein of Reynolds' work in that a specific agent's geometric position is not designated. The behaviors described in this article differ in that positions for each individual robot relative to the group are specified and maintained.

Other recent related papers on formation control for robot teams include [10], [16], [23], [22]. Parker's thesis [16] concerns the coordination of multiple heterogeneous robots. Of particular interest is her work in implementing "bounding overwatch," a military movement technique for teams of agents; one group moves (bounds) a short distance, while the other group overwatches for danger. Yoshida [23], and separately, Yamaguchi [22], investigate how robots can use only local communication to generate a global grouping behavior. Similarly, Gage [10] examines how robots can use local sensing to achieve group objectives like coverage and formation maintenance.

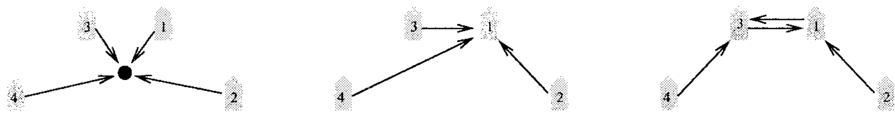
In the work most closely related to this research, Parker simulates robots in a line-abreast formation navigating past waypoints to a final destination [15]. The agents are programmed using the layered subsumption architecture [5]. Parker evaluates the benefits of varying degrees of global knowledge in terms of cumulative position error and time to complete the task. Using the terminology introduced in this article, Parker's agents utilize a *leader-referenced line* formation. The approach includes a provision for obstacle avoidance, but performance in the presence of obstacles is not reported. Parker's results suggest that performance is improved when agents combine local control with information about the leader's path and the team's goal.

The research reported in this article is similar to Parker's to the extent that it includes an approach for robotic *line* formation maintenance. The work serves to confirm Parker's results, but it goes significantly beyond that. In addition to *line* formations, this research evaluates three additional formation geometries and two new types of formation reference.




(a) (b) (c) (d)

Fig. 3. Formations for four robots: (a) *line*, (b) *column*, (c) *diamond*, and (d) *wedge*. Each robot has a specific position to maintain in the formation, as indicated by its identification number (ID).




(a) (b) (c)

Fig. 4. Formation position determined by various referencing techniques (from left to right: unit-center, leader, neighbor).

Quantitative evaluations indicate that one of the new reference techniques (*unit-center*) provides better performance than the leader-referenced approach utilized in Parker's work. The behavioral approach to formation maintenance is also different. In the subsumption architecture used in Parker's investigation, behaviors are selected competitively; the agent must either be avoiding hazards, moving into formation, but not both. The motor schema approach utilized here enables behaviors for moving to the destination, avoiding obstacles, and formation keeping to be simultaneously active and cooperatively combined. Additionally, as well as running in simulation, our approach is validated on two different types of mobile robot platform.

## II. Approach

Several formations for a team of four robots are considered (Fig. 3):

*line* robots travel line-abreast;
*column* robots travel one after the other;
*diamond* robots travel in a diamond;
*wedge* robots travel in a "V."

These formations are used by U.S. Army mechanized scout platoons on the battlefield [3]. For each formation, each robot has a specific position based on its identification number (ID). Fig. 3 shows the formations and robots' positions within them. Active behaviors for each of the four robots are identical, except in the case of Robot 1 in leader-referenced formations (see below). The task for each robot is to simultaneously move to a goal location, avoid obstacles, avoid colliding with other robots and maintain a formation position, typically in the context of a higher-level mission scenario.

Formation maintenance is accomplished in two steps: first, a perceptual process, **detect-formation-position**, determines the robot's proper position in formation based on current environmental data; second, the motor process **maintain-formation**, generates motor commands to direct the robot toward the correct location. In the case of AuRA's motor schema control, the command is a movement vector toward the desired location. For the UGV Demo II Architecture, separate votes are cast for steering and speed corrections toward the formation position. Motor commands for each architecture are covered in more detail below.

Each robot computes its proper position in the formation based on the locations of the other robots. Three techniques for formation position determination have been identified.

*1) Unit-center-referenced:* unit-center is computed independently by each robot by averaging the $x$ and $y$ positions of all the robots involved in the formation. Each robot determines its own formation position relative to that center.

*2) Leader-referenced:* each robot determines its formation position in relation to the lead robot (Robot 1). The leader does not attempt to maintain formation; the other robots are responsible for formation maintenance.

*3) Neighbor-referenced:* each robot maintains a position relative to one other predetermined robot.

The orientation of the formation is defined by a line from the unit center to the next navigational waypoint. Together, the unit-center and the formation orientation define a local coordinate system in which the formation positions are described. This local coordinate system is re-computed at each movement step. The formation relationships are depicted in Fig. 4. Arrows show how the formation positions are determined. Each arrow points *from* a robot *to* the associated reference. The perceptual schema **detect-formation-position** uses one of these references to determine the position for the robot. Spacing between robots is determined by the *desired spacing* parameter of **detect-formation-position**.

Each robot determines the positions of its peers by direct perception of the other robots, by transmission of world coordinates obtained from global positioning systems (GPS) or by dead reckoning. When inter-robot communication is required, the robots transmit their current position in world coordinates with updates as rapidly as required for the given formation speed and environmental conditions. Position errors and latency in the transmission of positional information can negatively impact performance. In simulation runs there was no position error or communication latency. In experimental laboratory runs Nomad 150's experienced less than 10 cm position error; communication latency was approximately one second. Position error for the current UGV implementation was less than one meter due to the use of DGPS; communication latencies were sometimes as great as seven seconds.

The remainder of this article describes the implementation of these formation behaviors in simulation and on two types of mobile robot. The next section covers a motor schema implementation. It includes a performance analysis of the motor schema-based system in turns and across obstacle fields. The behaviors are demonstrated on Nomadic Technologies Nomad 150 robots. Comparisons between mobile robot and simulation runs support the significance of the data gathered in simulation experiments.

Section IV covers the implementation of this approach on the UGV Demo II Architecture. The UGV platform requires a decoupling of motor control into separate steering and speed behaviors. In spite of this difference, the UGV implementation utilizes the same perceptual mechanisms as the motor schema approach for determining a robot's position in formation. Both implementations "push" a robot back into position with a variable strength depending on how far it is out of position. Implementation of the same approach on these two very different platforms illustrates its portability and effectiveness.

## III. Motor Schema-Based Formation Control

Several motor schemas, **move-to-goal**, **avoid-static-obstacle**, **avoid-robot** and **maintain-formation** implement the overall behavior for a robot to move to a goal location while avoiding obstacles, collisions with other robots and remaining in formation. An additional background schema, **noise**, serves as a form of reactive "grease," dealing with some of the problems endemic to purely reactive navigational methods such as local maxima, minima and cyclic behavior [1]. Each schema generates a vector representing the desired behavioral response (direction and magnitude of movement) given the current sensory stimuli provided by the environment. A gain value is used to indicate the relative importance of the individual behaviors. The high-level combined behavior is generated by multiplying the outputs of each primitive behavior by its gain, then summing and normalizing the results. The gains and other schema parameters used for the experimental simulations reported in this article are listed in Table I. The Appendix contains information on the specific computation of the individual schemas used in this research. See [1] for a complete discussion of the computational basis of motor schema-based navigation.

Once the desired formation position is known, the **maintain-formation** motor schema generates a movement vector toward it. The vector is always in the direction of the desired formation position, but the magnitude depends on how far the robot is away from it. Fig. 5 illustrates three zones, defined by distance from the desired position, used for magnitude computation. The radii of these zones are parameters of the **maintain-formation** schema. In the figure, Robot 3 attempts to maintain a position to the left of and abeam Robot 1. Robot 3 is in the controlled zone, so a moderate force toward the desired position (forward and right) is generated by **maintain-formation**. In general, the magnitude of the vector is computed as follows:

TABLE I
Motor Schema Parametes for Formation Navigation Experiments in Simulation

| Parameter | Value | Units |
|---|---|---|
| avoid-static-obstacle | | |
| gain | 1.5 | |
| sphere of influence | 50 | meters |
| minimum range | 5 | meters |
| avoid-robot | | |
| gain | 2.0 | |
| sphere of influence | 20 | meters |
| minimum range | 5 | meters |
| move-to-goal | | |
| gain | 0.8 | |
| noise | | |
| gain | 0.1 | |
| persistence | 6 | time steps |
| maintain-formation | | |
| gain | 1.0 | |
| desired spacing | 50 | meters |
| controlled zone radius | 25 | meters |
| dead zone radius | 0 | meters |

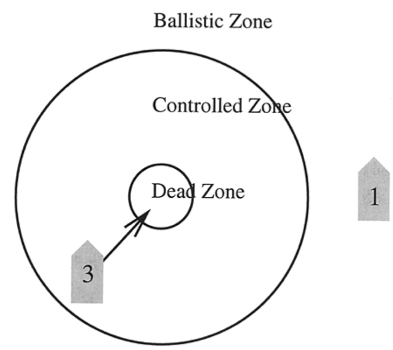



Fig. 5. Zones for the computation of **maintain-formation** magnitude.

**Ballistic zone** the magnitude is set at its maximum, which equates to the schema's gain value.

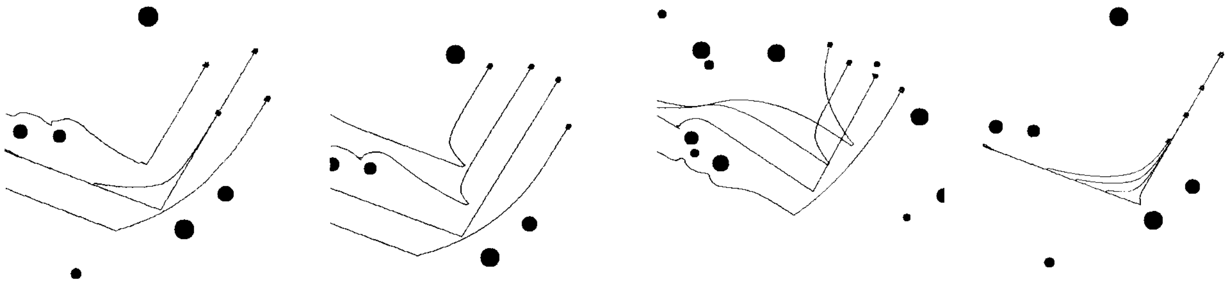
**Controlled zone** the magnitude varies linearly from a maximum at the farthest edge of the zone to zero at the inner edge.

**Dead zone** in the dead zone vector magnitude is always zero.

The role of the dead zone is to minimize the problems associated with position reporting errors and untimely communication. The dead zone provides a stable target *area* (as opposed to a point) that provides high tolerance to positional uncertainty. It is assumed that the dead zone is greater than or equal to the errors associated with these uncertainties.

In simulation, no dead zone was required for stable performance (dead zone radius is set to 0), but mobile robots require a small dead zone to avoid oscillations about the formation position due to latency in communication or errors in position determination. These factors are negligible in the simulation studies.

Recall that the orientation of the formation is defined by a line from the unit center to the next navigational waypoint.



(a) (b) (c) (d)

Fig. 6. Four robots in leader-referenced (a) *diamond*, (b) *wedge*, (c) *line*, and (d) *column* formations.

Together, the unit-center and the formation orientation define a local coordinate system in which the formation positions are described. This local coordinate system is re-computed at each movement step. The motion of the formation as a whole also arises from the impetus provided by the other active behaviors, primarily **move-to-goal**.

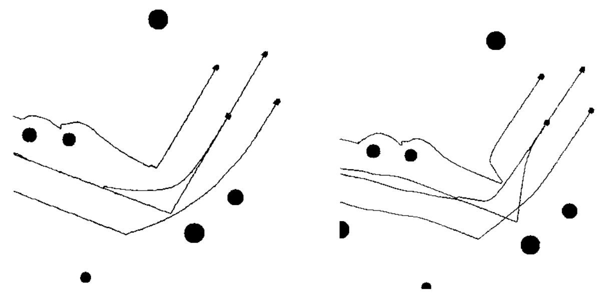
The formation behavior is only one component of the robots' overt actions. In extreme conditions, for example, if a barrier significantly larger than the entire formation is encountered, then the formation will either move as a unit around the barrier or will divide into subgroups with some proceeding around each side. The resultant action depends upon the relative strength of the formation behavior to the other goal-oriented behaviors (e.g., **move-to-goal**). If the goal attraction is very much stronger, the individual robot's needs will take precedence. On the other hand if the formation behavior has a high gain and is thus a dominant factor, the formation will act more or less like a single unit and not be allowed to divide. The level of "obedience" to remain in formation is controllable through the setting of the relative gain values of these behaviors during mission specification. This same discussion applies to when there are multiple corridors in front of the robots or other similar conditions.

### *A. Motor Schema Results in Simulation*

Results were generated using Georgia Tech's *MissionLab* robot simulation environment [12]. *MissionLab*[1] runs on Unix machines (SunOS and Linux) using the X11 graphical windowing system. The simulation environment is a 1000 by 1000 m two dimensional field upon which various sizes and distributions of circular obstacles can be scattered. Each simulated robot is a separately running C program that interacts with the simulation environment via a Unix socket. The simulation displays the environment graphically and maintains world state information which it transmits to the robots as they request it. Fig. 6 shows four typical simulation runs. The robots are displayed as five-sided polygons, while the obstacles are black circles. The robots' paths are depicted with solid lines.

Sensors allow a robot to distinguish between three perceptual classes: robots, obstacles and goals. When one of the robot's perceptual processes requires obstacle information a request for that data is sent via a socket to the simulation process. A list comprised of angle and range data for each obstacle in sensor range is returned. Robot and goal sensor information is similarly provided. A robot moves by transmitting its desired velocity to the simulation process which automatically maintains the position and heading of each robot.

[1] MissionLab software is available on the World Wide Web at http://www.cc.gatech.edu/aimosaic/robot-lab/research.



(a) (b)

Fig. 7. Comparison of (a) leader-referenced and (b) unit-center-referenced *diamond* formations.

The *line*, *column*, *wedge*, and *diamond* formations were implemented using both the unit-center-referenced and leader-referenced approaches. Fig. 6 illustrates robots moving in each of the basic formations with the leader-referenced approach. In each of these simulation runs the robots were first initialized on the left side of the simulation environment, then directed to proceed to the lower center of the frame. After the formation was established, a 90° turn to the left was initiated. Results were similarly obtained for the unit-center-referenced formations.

Qualitative differences between the two approaches can be seen as the formation of robots moves around obstacles and through turns (Fig. 7). For leader-referenced formations any turn by the leader causes the entire formation to shift accordingly, but when a "follower" robot turns, the others in formation are not affected. In unit-center-referenced formations any robot move or turn impacts the entire formation. In turns for leader-referenced formations, the leader simply heads in the new direction; the other robots must adjust to move into position. In unit-center-referenced turns, the entire formation initially appears to spin about a central point, as the robots align with a new heading.

To investigate quantitative differences in performance between the various formation types and references, two experiments were conducted in simulation: the first evaluates performance in turns, and the second evaluates performance across an obstacle field.

TABLE II
PERFORMANCE FOR A 90 DEGREE TURN FOR BOTH UNIT-CENTER AND LEADER-REFERENCED FORMATIONS, SMALLER NUMBERS ARE BETTER. THE STANDARD DEVIATION IS INDICATED WITHIN PARAMETERS

| Formation Type | Path Ratio | | Position Error | | Time out of Formation | |
|---|---|---|---|---|---|---|
| | Unit | Leader | Unit | Leader | Unit | Leader |
| *diamond* | 1.03 (0.08) | 1.06 (0.08) | 6.8 (0.2) m | 11.4 (5.9) m | 20.8 (0.3) % | 21.6 (10.8) % |
| *wedge* | 1.04 (0.09) | 1.06 (0.09) | 9.4 (4.5) m | 9.1 (6.2) m | 25.6 (6.0) % | 17.3 (9.6) % |
| *column* | 1.04 (0.06) | 1.16 (0.02) | 8.4 (5.6) m | 21.1 (17.3) m | 22.4 (8.1) % | 32.4 (22.8) % |
| *line* | 1.04 (0.10) | 1.05 (0.06) | 8.5 (5.5) m | 8.2 (5.1) m | 25.7 (7.4) % | 18.9 (10.8) % |

TABLE III
PERFORMANCE FOR NAVIGATION ACROSS AN OBSTACLE FIELD

| Formation Type | Path Ratio | | Position Error | | Time out of Formation | |
|---|---|---|---|---|---|---|
| | Unit | Leader | Unit | Leader | Unit | Leader |
| *diamond* | 1.05 (0.04) | 1.08 (0.05) | 5.2 (1.9) m | 7.1 (5.0) m | 38.9 (15.0) % | 34.8 (21.8) % |
| *wedge* | 1.04 (0.04) | 1.08 (0.05) | 5.2 (1.4) m | 9.5 (8.4) m | 37.9 (9.4) % | 37.2 (24.3) % |
| *column* | 1.05 (0.04) | 1.08 (0.04) | 3.4 (1.6) m | 6.4 (5.2) m | 23.2 (11.8) % | 28.5 (20.2) % |
| *Line* | 1.05 (0.05) | 1.05 (0.04) | 5.3 (1.5) m | 9.4 (8.5) m | 36.1 (10.5) % | 35.6 (23.8) % |

### B. Motor Schema Performance in Turns

To evaluate performance in turns, the robots are commanded to travel 250 m, turn right, then travel another 250 m. The robots attempt to maintain formation throughout the test. A turn of 90° was selected for this initial study, but performance likely varies for different angles. In this evaluation, no obstacles are present. For statistical significance, ten simulations were run for each formation type and reference. To ensure the robots are in correct formation at the start of the evaluation, they travel 100 m to align themselves before the evaluation starts. This initial 100 m is not included in the 500 m course evaluation. A run is complete when the unit-center of the formation is within 10 m of the goal location. Even though a unit-center computation is used to determine task completion, it is not required for leader-referenced formation maintenance.

Three performance metrics are employed: path length ratio, average position error, and percent of time out of formation. Path length ratio is the average distance traveled by the four robots divided by the straight-line distance of the course. A lower value for this ratio indicates better performance. A ratio of 1.02, for example, means the robots had to travel an average of 2% further because they were in formation. Position error is the average displacement from the correct formation position throughout the run. Robots occasionally fall out of position due to turns, etc.; this is reflected in the percent of time out of formation data. To be "in position" a robot must be within 5 m of its correct position. Five meters was selected arbitrarily, but amounts to 10% of the overall formation spacing. Results for the turn experiments are summarized in Table II; the standard deviation for each quantity is listed in parentheses.

For turns in a unit-center-referenced formation, *diamond* formations perform best. The *diamond* formation minimizes path ratio (1.03), position error (6.8 m) and time out of formation (20.1%). Unit-center-referenced formations appear to turn by rotating about their unit-center, so robots on the outside edge of the formation have to travel further in turns. The improved performance in *diamond* formations may reflect the smaller "moment of inertia" as compared to other formations. In the *diamond* formation, no robot is further than 50 m from the unit-center. In contrast, the flanking robots in *wedge*, *line*, and *column* formations are 75 m from the unit center.

For turns in a leader-referenced formation, *wedge* and *line* formations perform about equally. The *line* formation minimizes position error (8.2 m), while the *wedge* formation minimizes time out of formation (17.3%). Leader-referenced formations pivot about the leader in sharp turns. Robots significantly behind the leader will be pushed through a large arc during the turn. *line* and *wedge* formations work well because fore and aft differences between the lead robot and other robots (0 and 50 m, respectively) are less than *diamond* and *column* formations (100 and 150 m). Performance for *column* formations is significantly worse than that for *line*, *wedge*, and *diamond* formations because the trail robot is 150 m back.

### C. Motor Schema Performance in an Obstacle Field

Performance was also measured for four robots navigating across a field of obstacles in formation. In this evaluation, the robots are commanded to travel between two points 500 m apart. Obstacles are placed randomly so that 2% of the total area is covered with obstacles 10–15 m in diameter. As in the turn evaluation above, path length ratio, average position error, and percent out of formation is reported for each run. Data from runs on 10 random scenarios were averaged for each datapoint, the standard deviation of each factor is also recorded. Results for this experiment are summarized in Table III.

For travel across an obstacle field, the best performance is found using *column* formations. *column* formations minimize position error and percent time out of formation for unit-center- and leader-referenced formations. This result reflects the fact that *column* formations present the smallest cross-section as they traverse the field. Once the lead robot shifts laterally to avoid an obstacle, the others can follow in its "footsteps."

In most instances, unit-center-referenced formations fare better than leader-referenced formations. A possible explana-

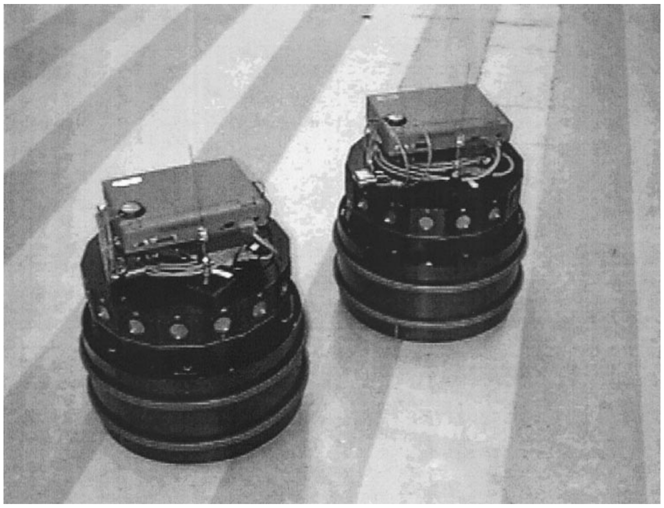


Fig. 8. Shannon and Sally, the two Nomad 150 robots used in formation experiments.

tion is an apparent emergent property of unit-center-referenced formations; the robots appear to work together to minimize formation error. For instance, if one robot gets stuck behind an obstacle the others "wait" for it. The unit-center is anchored by the stuck robot so the **maintain-formation** schema instantiated in the other robots holds them back until the stuck robot navigates around the obstacle. This does not occur in leader-referenced formations.

Overall path length for robots in a leader-referenced formation is generally longer than in unit-center-referenced formations. This may be because any turn or detour by the lead robot is followed by all four robots, even if their path is not obstructed by the obstacle the leader is avoiding. A detour by the lead robot in a unit-center-referenced formation affects the entire formation, but the impact is 75% less than that found in leader-referenced formations since in the unit-center case an individual robot must shift 4 m to move the formation's unit-center 1 m.

### *D. Motor Schema Results on Mobile Robots*

Experiments were conducted in the Mobile Robot Laboratory to demonstrate formation performance on mobile robots and to validate the quantitative results from simulation experiments. *MissionLab* is designed so that at runtime a researcher may choose between a simulated run, or a run on physical robots. The same behavioral control code is used both in simulation and to control the robots. Currently, the system can command Denning MRV-3, MRV-2 and DRV robots, as well as Nomadic Technologies Nomad 150 robots and a Hummer four-wheel drive vehicle instrumented for robotic use at Georgia Tech.

The experimental platform for the results reported here is a two-robot team of Nomad 150 robots: Shannon and Sally (Fig. 8). Nomad 150's are three-wheeled holonomic robots equipped with a separately steerable turret and 16 ultrasonic range sensors for hazard detection. The Nomad 150's are controlled using on-board laptop computers running Linux. They communicate over a wireless network supporting Unix sockets via TCP/IP.

TABLE IV
MOTOR SCHEMA PARAMETERS FOR FORMATION NAVIGATION ON NOMAD 150 ROBOTS

| Parameter | Value | Units |
|---|---|---|
| **avoid-static-obstacle** | | |
| gain | 1.5 | |
| sphere of influence | 2.0 | meters |
| minimum range | 0.5 | meters |
| **avoid-robot** | | |
| gain | 1.0 | |
| sphere of influence | 1.2 | meters |
| minimum range | 0.6 | meters |
| **move-to-goal** | | |
| gain | 1.0 | |
| **maintain-formation** | | |
| gain | 2.0 | |
| desired spacing | 1.5 | meters |
| controlled zone radius | 0.75 | meters |
| dead zone radius | 0.1 | meters |

Experimental runs were conducted in a test area measuring approximately 10 by 5 m. The robots were directed to navigate from West to East across the room (left to right in Figs. 9–11). Runs were conducted for line, *wedge*, and *column* unit-center referenced formations. Separate runs were conducted for each type of formation with and without obstacles. The robots estimate their position using shaft encoders. In order to calculate the formation's unit-center each robot communicates its position to the other over a wireless network.

The behavioral configuration of the robots was the same as that used in simulation runs, except that parameter values were adjusted to account for the use of smaller robots (Nomad 150's versus HMMWV's) and a smaller test area.

Table IV lists the motor schema parameter values used on the mobile robots. The **noise** motor schema was not activated in these experiments because sensor noise provides a sufficient random input to help robots around shallow local minima.

Fig. 9–11 show Shannon and Sally traversing the test area in *column*, *wedge*, and *line* formations with and without obstacles present. For comparison, the runs with and without obstacles for each formation type are reproduced on the top and middle of each page, while snapshots of the robots during the run with obstacles are shown at the bottom.

During the runs, the robots remained in their appropriate formation position, except for short periods while negotiating obstacles. In the case of obstacles, it was evident that one robot would "wait" for the other robot if it got delayed behind an obstacle.

To further validate the accuracy of the simulation data, an additional set of simulation runs matching the experimental setup were conducted. The simulations used the same parameter values and obstacle locations as in the mobile robot tests. Results for these tests are shown in Fig. 12. Differences between the simulation and real runs are primarily due to sensor noise and positional inaccuracies.

## IV. FORMATION CONTROL FOR THE UGV DEMO II ARCHITECTURE

UGV Demo II is a DARPA-funded project aimed at fielding a robotic scout platoon for the Army. Each Unmanned Ground Vehicle (UGV) is a High Mobility Multipurpose Wheeled

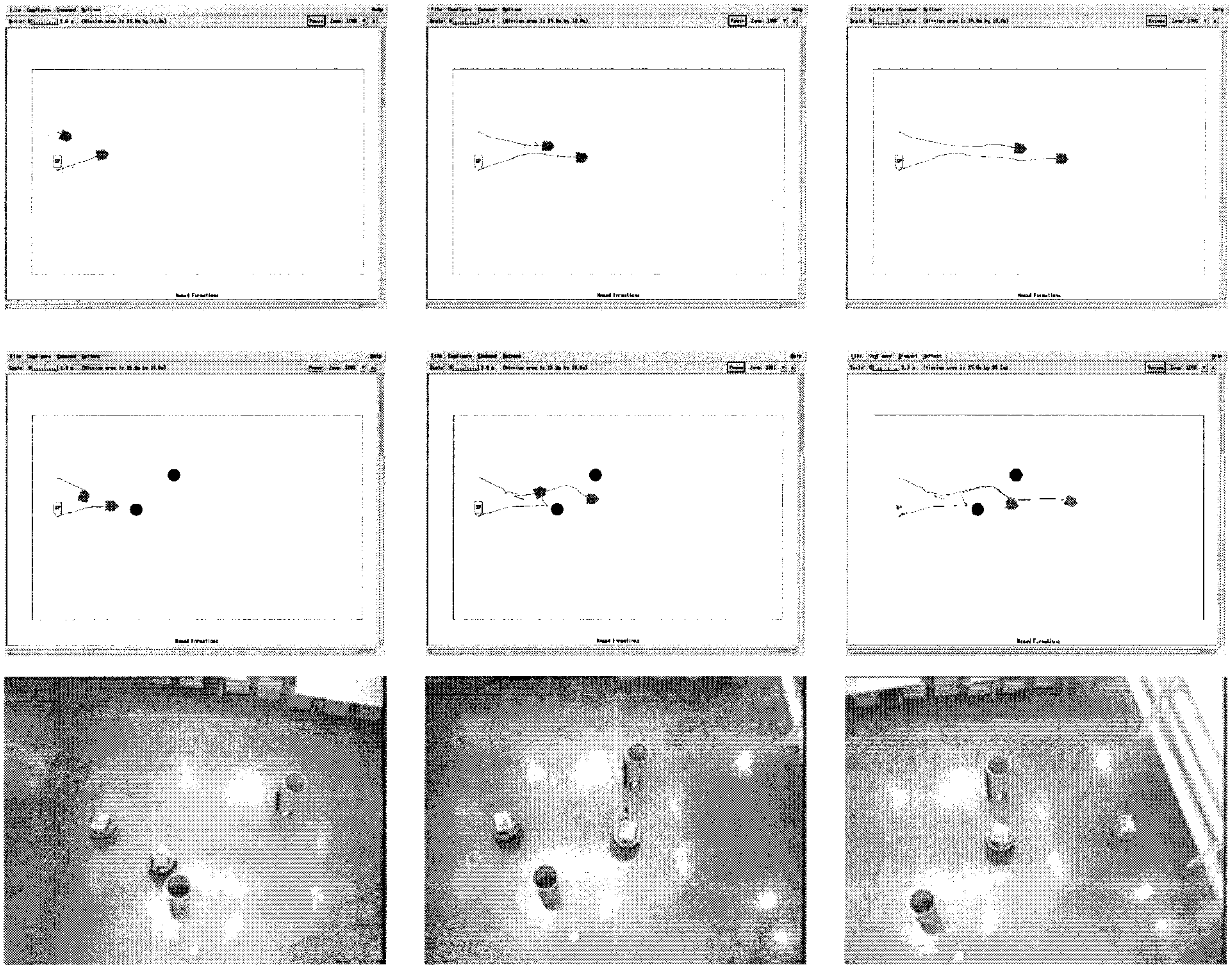


Fig. 9. Telemetry and photos of Shannon and Sally moving into and then traveling in *column* formation. Top row: *column* formation telemetry with no obstacles present. Middle row: *column* formation telemetry with obstacles present. Bottom row: photos of the robots in *column* formation with obstacles present. The photo sequence corresponds to telemetry in the middle row with obstacles (wastebaskets) present. This experiment was recorded in the foyer of the Georgia Tech Manufacturing Research Center, looking down on the robots from 20 feet above so that formation positions are more easily observed.

Vehicle (HMMWV) equipped with position, vision and hazard sensors, control computers and actuation devices for steering and speed control. Four UGV's were built by Lockheed Martin, and up to three have been operated simultaneously in formation (Fig. 1). This section shows how formation behaviors were adapted for use on these autonomous robots.

The UGV Demo II Architecture differs from the motor schema method where behaviors generate both a direction and magnitude. Instead, in the UGV Demo II Architecture, separate motor behaviors are developed for the speed and turning components of a behavior. The behaviors are coordinated by speed and turn arbiters. Each arbiter runs concurrently and accepts votes from the various active motor behaviors. For turning, behaviors vote for one of 30 discrete egocentric steering angles; the angle with the most votes wins. A behavior may actually cast several votes for separate headings at once, where the votes are spread about a central angle with a Gaussian distribution. In speed voting, the lowest speed vote always wins. Details on the mathematical formation of the arbitration process are available in [11]. One strength of the formation behaviors lies in their ability to be easily reformulated for this and other alternate behavior-based coordination methods.

As in the case of motor schema-based robots, the UGV's must simultaneously navigate to a goal position, avoid collisions with hazards and remain in formation. This is accomplished by concurrent activation of independent behaviors for each. Here we will deal only with the formation behaviors.

For the UGV, formations and formation positions were determined in the same way as described in Section II. The approach described here for maintaining a given formation position is equally applicable to unit-center, leader, and neighbor referenced formations, but only unit-center was implemented. We now focus on the control strategies for moving a robot into formation, given the desired position is known.

Car-like nonholonomic constraints on UGV movement call for a revision of the formation motor behavior. In the nonholonomic case the robot's heading during formation corrections significantly impacts its ability to remain in position. Not only should the vehicle be in the right location, but its heading should be aligned with the axis of the formation. If it is very far off heading, the robot will quickly fall out of position either laterally, fore-aft or both. A technique used by pilots for aircraft formation [9] is well suited for this

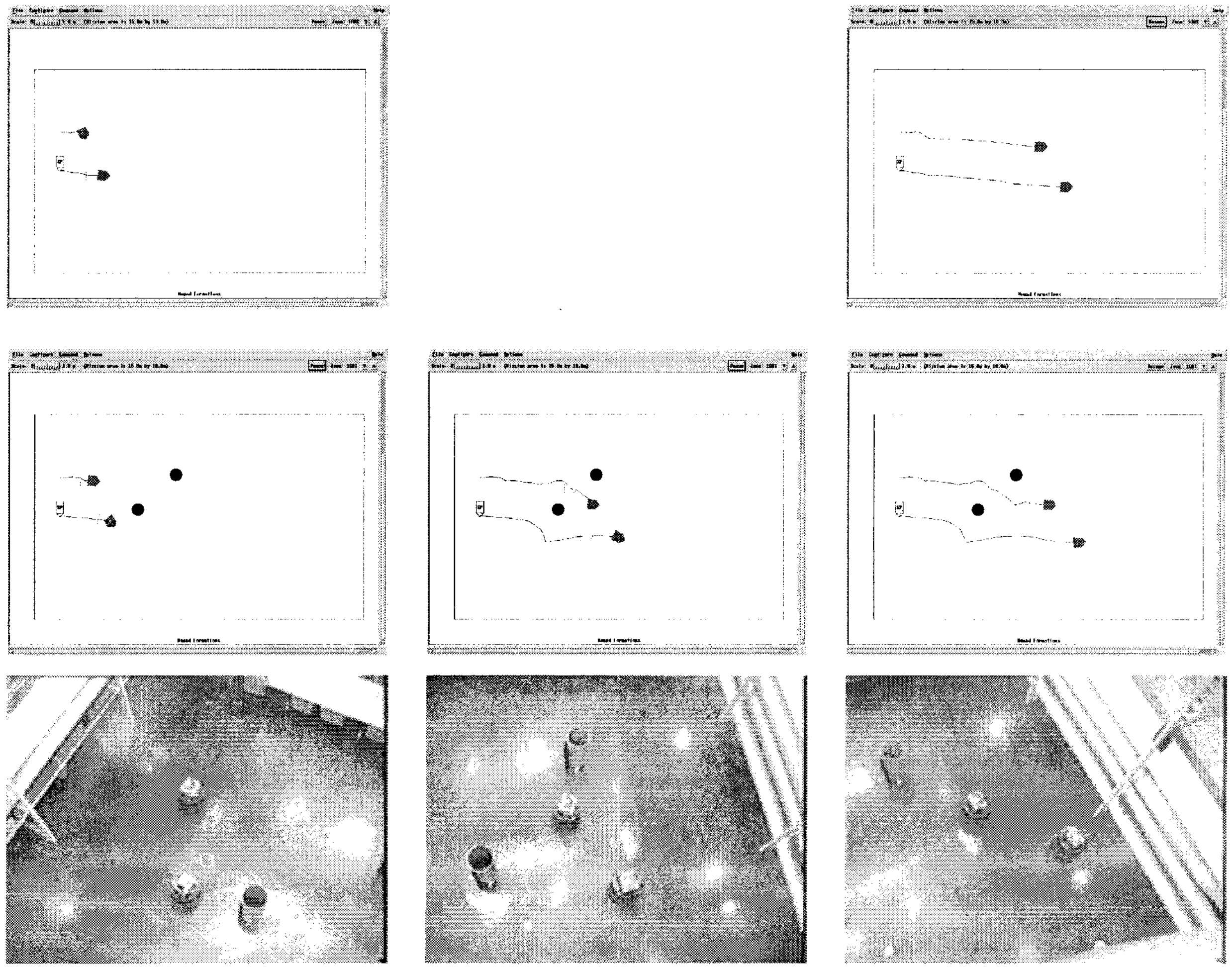


Fig. 10. Telemetry and photos of Shannon and Sally moving into and then traveling in *wedge* formation. Top row: *wedge* formation telemetry with no obstacles present. Middle row: *wedge* formation telemetry with obstacles present. Bottom row: photos of the robots in *wedge* formation with obstacles present. The photo sequence corresponds to the telemetry in the middle row with obstacles present.

task: positioning is decomposed into fore-aft and side-side corrections. Fore-aft corrections are made by adjusting speed only, while lateral corrections are made by adjusting heading only. Each correction is applied independently. A consequence of the approach is that when a robot is ahead of its position it will not attempt to turn around, but just slow down. The following observations summarize the approach.

For speed selection:

1) If the robot is in formation, the best speed for maintaining that formation is the current speed.
2) If the vehicle is behind its position, it should speed up.
3) If the vehicle is in front of its position, it should slow down.
4) The selected change in speed should depend on how far out of position the robot is.
5) Since the speed arbiter implemented in the Demo II Architecture selects the lowest speed vote of all the active behaviors for output to the vehicle, formation speed control is only possible by slowing down.

For steering:

1) If the robot is in formation, the best heading for position maintenance is the formation axis.
2) If the robot is out of position laterally and the formation is moving, it should turn toward the formation axis with an angle that depends on how far out of position it is.
3) If the robot is out of position and the formation has stopped moving, the robot should head directly toward its position.

### *A. UGV Behaviors for Formation*

While the motor schema approach combines the lateral and fore-aft components of position correction into one behavior, the Demo II Architecture requires a decomposition of control into separate steering and speed control components. Two behaviors, **maintain-formation-speed** and **maintain-formation-steer** run concurrently to keep the vehicle in position. The outputs of these two behaviors roughly correspond to the orthogonal components of the single-vector output motor schema. Each UGV behavior determines an appropriate value at each movement step and votes accordingly. The votes, along with those from other behaviors are tallied and acted upon by the speed and steering arbiters.

To facilitate the discussion that follows, the following formation terms are introduced (see Fig. 13):

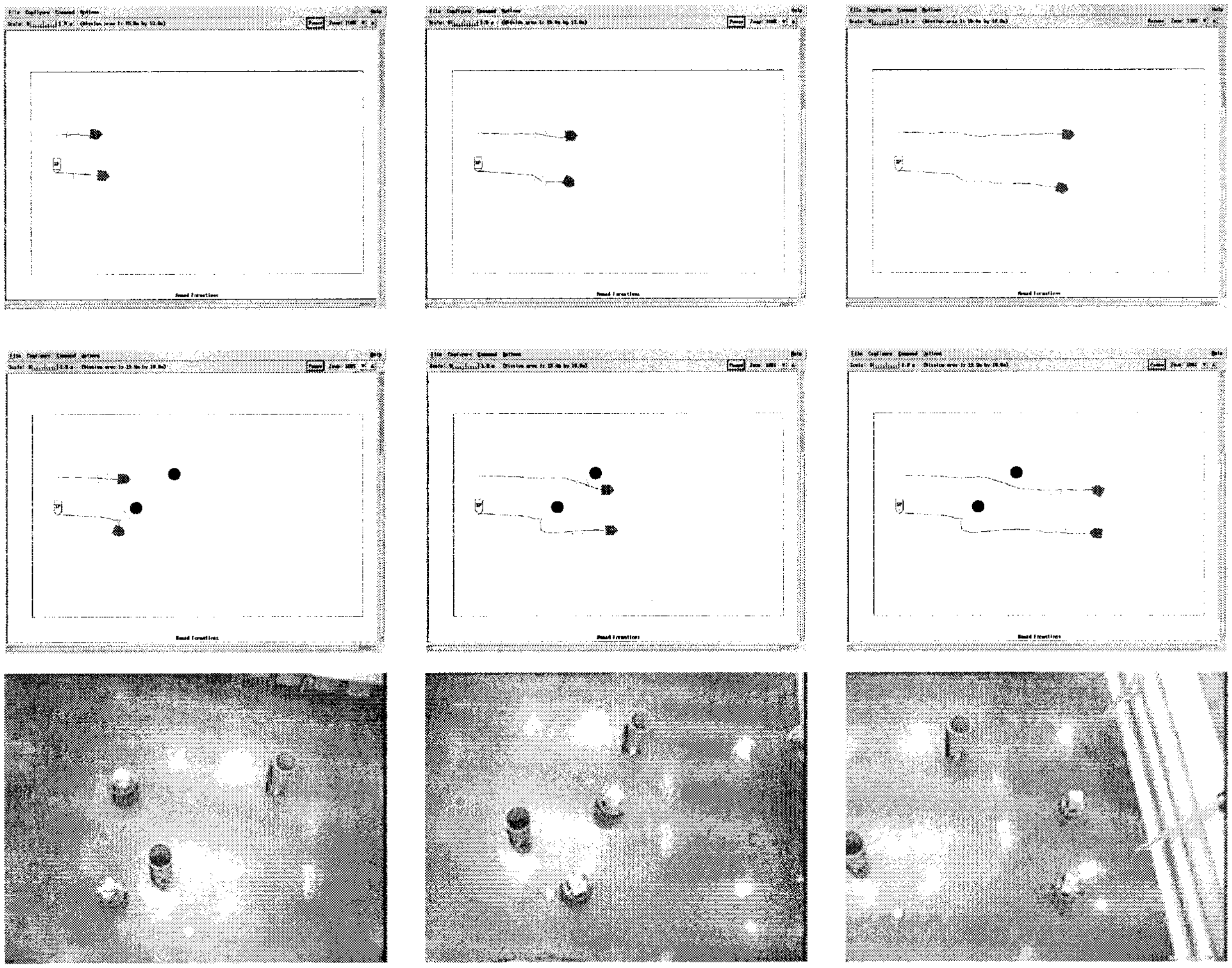


Fig. 11. Telemetry and photos of Shannon and Sally moving into and then traveling in *line* formation. Top row: *line* formation telemetry with no obstacles present. Middle row: *line* formation telemetry with obstacles present. Bottom row: photos of the robots in *line* formation with obstacles present. The photo sequence corresponds to the telemetry in the middle row with obstacles present.

| | |
|---|---|
| $R_{\text{pos}}, R_{\text{dir}}$ | robot's present position and heading; |
| $R_{\text{mag}}$ | robot's present speed; |
| $F_{\text{pos}}$ | robot's proper position in formation; |
| $F_{\text{dir}}$ | direction of the formation's movement; toward the next navigational waypoint; |
| $F_{\text{axis}}$ | formation's axis, a ray passing through $F_{\text{pos}}$ in the $F_{\text{dir}}$ direction; |
| $H_{\text{desired}}$ | desired heading, a computed heading that will move the robot into formation; |
| $\delta_{\text{heading}}$ | computed heading correction; |
| $\delta_{\text{speed}}$ | computed speed correction; |
| $V_{\text{steer}}$ | steer vote, representing the directional output of the motor behavior, sent to the steering arbiter; |
| $V_{rmspeed}$ | speed vote, the speed output of the motor behavior, sent to the speed arbiter. |

The **maintain-formation-speed** behavior first determines the magnitude of the required speed correction, then casts its vote by adding the correction to the current speed:

$$V_{\text{speed}} = R_{\text{mag}} + K \times \delta_{\text{speed}}$$

$K$ is a parameter set before runtime to adjust the rate of correction. $\delta_{\text{speed}}$ is the correction computed by the formation speed behavior. It varies from $-1.0$ (slow down) to 1.0 (speed up) depending on how far fore or aft the robot is of the desired position. Three zones, perpendicular to the formation axis and defined by distance fore or aft of $F_{\text{pos}}$ determine $\delta_{\text{speed}}$ (Fig. 14). The size of these zones are parameters of the formation behavior. $\delta_{\text{speed}}$ is set negative if the robot is in front of $F_{\text{pos}}$ and positive otherwise. In a manner similar to the motor schema-based approach the magnitude is computed as follows:

| | |
|---|---|
| **Ballistic zone** | 1.0; |
| **Controlled zone** | magnitude varies linearly from a maximum of 1.0 at the farthest edge of the zone to zero at the inner edge; |
| **Dead zone** | in the dead zone the magnitude is always zero. |

The **maintain-formation-steer** behavior follows a similar sequence of steps to determine an egocentric steering direction, (the angle for the front wheels with respect to the vehicle body. The behavior computes the magnitude of correction necessary, the desired heading for that correction, then finally, it votes for an appropriate steering angle. The magnitude of correction is determined based on how far laterally the robot is from its formation position. The maximum correction is

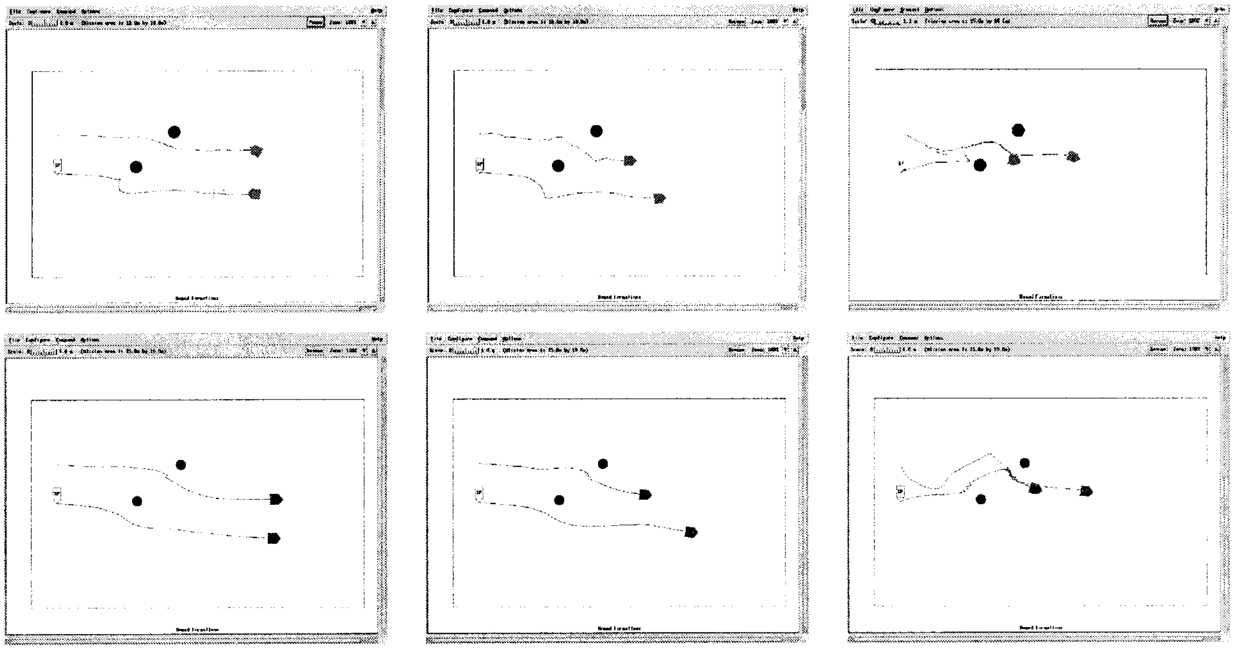


Fig. 12. A comparison of telemetry from actual robot formation runs (top row) and runs in in simulation (bottom row). From left to right: *line*, *wedge*, and *column* formations.

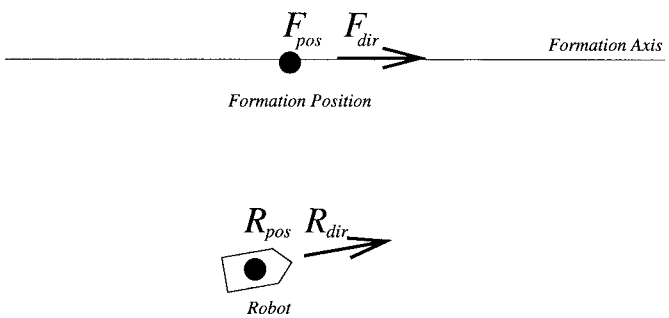



Fig. 13. Illustration of terms used in describing formation behaviors for UGV's. In this diagram the robot is behind and to the right of the desired position in formation. The robot's position and direction are indicated by $R_{\rm pos}$ and $R_{\rm dir}$. The desired formation position is $F_{\rm pos}$. The formation is moving in the direction $F_{\rm dir}$.

for the robot to head directly *toward* the formation axis, the minimum is for the robot to head directly along the formation axis. The magnitude of $\delta_{\rm heading}$ computed by the formation heading behavior is determined as follows (Fig. 14):

**Ballistic zone** 90°, i.e. head directly toward the axis.

**Controlled zone** the turn varies linearly from a maximum of 90° at the farthest edge of the zone to 0° at the inner edge.

**Dead zone** 0°, i.e. head parallel to the axis.

The sign of the correction is set according whether the robot is left or right of the formation axis. If the robot is left of the axis, calling for a right turn, the sign is positive, it is set negative otherwise. $H_{\rm desired}$ can now be determined with reference to the formation axis

$$H_{\rm desired} = F_{\rm dir} - \delta_{\rm heading.}$$

As the robot moves forward, this heading will simultaneously bring it to and properly align it with the formation axis. In the special case where the formation has stopped moving, $H_{\rm desired}$ is instead set to take the robot directly to its position

$$H_{\rm desired} = F_{\rm pos} - R_{\rm pos.}$$

Finally, $H_{\rm desired}$ is translated into an egocentric angle for the vehicle's front wheels

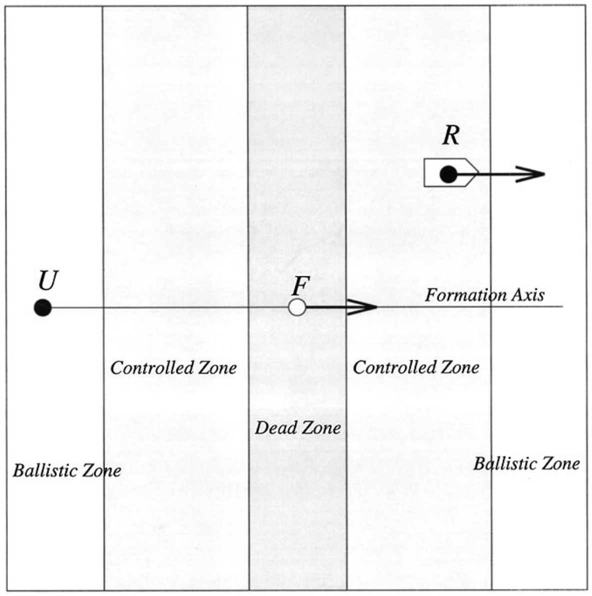
$$V_{\rm steer} = H_{\rm desired} - R_{\rm dir.}$$

Positive angles indicate a right turn and negative ones a left turn. If the result is either greater than 180° or less than −180, 360° is added or subtracted to ensure the result is within bounds. Finally the angle is clipped to the physical limits of the vehicle.

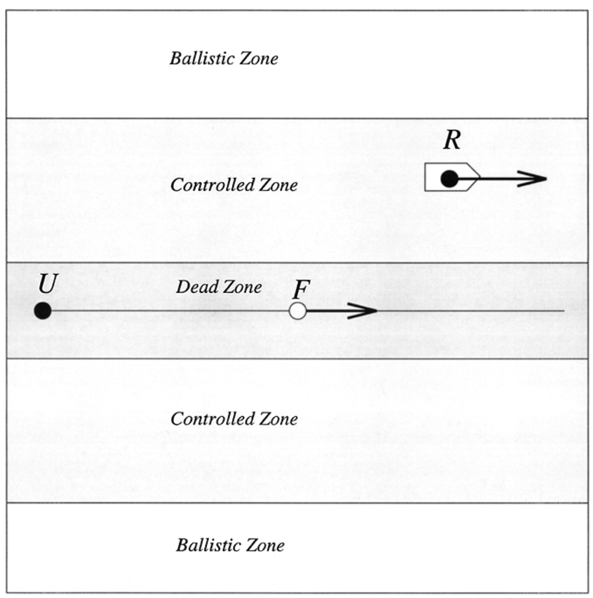
## V. Results for UGV Demo II Mobile Robots

The behaviors were initially implemented and evaluated at Georgia Tech using a single-robot simulator provided by Lockheed Martin. The behaviors were debugged by generating an artificial fixed trajectory for one vehicle, then observing a simulated robot's attempt to maintain position with the fixed trajectory. Final integration with HMMWV's was completed by Lockheed Martin in Denver, Colorado. Positional information on the HMMWV's was reported via differential global positioning system (DGPS) receivers.

Fig. 15 shows a sample run using this simulation. The notional robot follows a straight-line track from west to east (left to right), while the simulated robot attempts to maintain a line-abreast formation on the south. Initially the robot is pointed north, so it must turn to the south to get into position. Note that for the robot to get into position it must initially move away from the formation axis, until it is turned around.




(a)




(b)

Fig. 14. Zones centered on $F_{\mathrm{pos}}$, the desired formation position. The zones in (a) are used for computing speed, corrections, while those in (b) are for heading corrections.

The unit-center referenced approach was used on the HMMWV's because the UGV Demo II Architecture only provides the ability for a robot to slow down to keep formation. It was felt that since the leader would never slow down to keep formation and a trailer could never speed up if it fell behind due to architectural limitations, a leader-referenced approach would be unsuccessful.

Formation played a key role in the success of UGV Demo C in the Summer of 1995. At a technology demonstration two HMMWV's ran through a series of tests including a sequence of formations (Figs. 16 and 17). The HMMWV's followed a one-kilometer course across open undulating terrain while smoothly shifting from *column* to *wedge* to *line* then back to *column* formation.

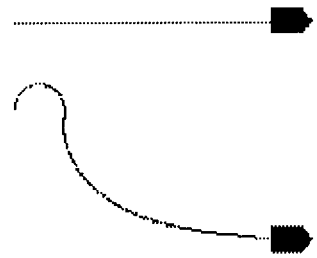


Fig. 15. Simulation of two DARPA UGV's in formation. The robots are moving from left to right in a *line* formation. The robot at the top of the figure follows a fixed path, while the other robot utilizes behaviors described in the text to maintain a unit-center-referenced *line* formation.

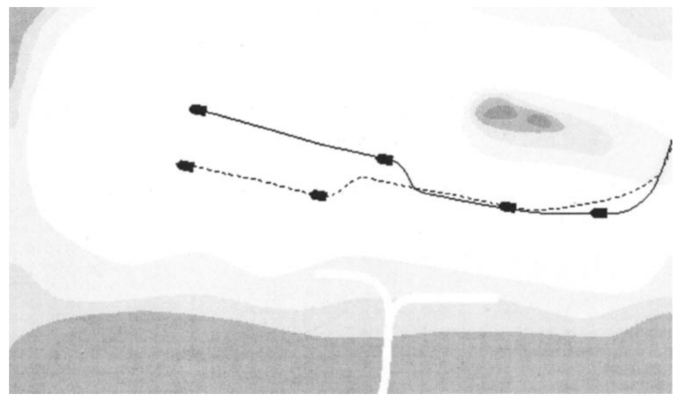


Fig. 16. Reconstruction of the ground track of DARPA UGV's depicted in Fig. 17. The pair of robots are shown at three points in time as they move from right to left. They transition from *column* (right) to *wedge* (center) to *line* formations as they traverse the field.

A formation expert software tool was developed and integrated into the UGV Demo II architecture which provides the operator a graphical user-interface for the selection of formation types and parameters. This rule-based system drew both on the recommendations of military personnel and doctrine as presented in U.S. Army manuals [3]. The operator uses this tool to determine what formations fit the task confronting him.

The three robot formations have run satisfactorily. Performance in these tests was limited by a communications system that induced up to 7 s of latency in robot to robot position reports. This problem points to the utility of using a passive approach for locating team members, versus the explicit exchange of location based on DGPS readings.

## VI. Conclusion

Reactive behaviors for four formations and three formation reference types were presented. The behaviors were demonstrated successfully in the laboratory on mobile robots, and outdoors on nonholonomic four-wheel-drive HMMWV's. In the course of these evaluations, the approach was implemented on two reactive robotic architectures, AuRA and the UGV Demo II Architecture. The AuRA implementation is conceptually simpler and applicable to holonomic robots, while the UGV implementation addresses the additional complexity of nonholonomic vehicle control.

Separate experiments in simulation evaluated the utility of the various formation types and references in turns and across

(a) (b) (c)

Fig. 17. Two DARPA UGV's in formation, from left to right: (a) *column*, (b) *wedge*, and (c) *line*.

obstacle fields. For 90° turns, the *diamond* formation performs best when the unit-center-reference for formation position is used, while *wedge* and *line* formations work best when the leader-reference is used. For travel across an obstacle field, the *column* formation works best for both unit-center- and leader-referenced formations. In most cases, unit-center-referenced formations perform better than leader-referenced formations. Even so, some applications probably rule out the use of unit-center-referenced formations:

*1) Human leader:* A human serving as team leader cannot be reasonably expected to compute a formation's unit-center on the fly, especially while simultaneously avoiding obstacles. A leader-referenced formation is most appropriate for this application.

*2) Communications restricted applications:* The unit-center approach requires a transmitter and receiver for each robot and a protocol for exchanging position information. Conversely, the leader-referenced approach only requires one transmitter for the leader, and one receiver for each following robot. Bandwidth requirements are cut by 75% in a four robot formation.

*3) Passive sensors for formation maintenance:* Unit-center-referenced formations place a great demand on passive sensor systems (e.g. vision). In a four robot visual formation for instance, each robot would have to track three other robots which may spread across a 180° field of view. Leader- and neighbor-referenced formations only call for tracking one other robot.

## Appendix
## Motor Schema Formulae

This appendix describes the methods by which each of the individual primitive schemas used in this research compute their component vectors. The results of all active schemas are summed and normalized prior to transmission to the robot for execution.

1) **Move-to-goal**: Attraction to goal with variable gain. Set high when heading for a goal.

$$V_{\text{magnitude}} = \text{adjustable gain value}$$
$$V_{\text{direction}} = \text{in direction toward perceived goal}$$

2) **Avoid-static-obstacle**: Repel from object with variable gain and sphere of influence. Used for collision avoidance

$$O_{\text{magnitude}} = \begin{cases} 0, & \text{for } d > S \\ \dfrac{S-d}{S-R} * G, & \text{for } R < d \leq S \\ \infty, & \text{for } d \leq R \end{cases}$$

where

| | |
|---|---|
| S | adjustable Sphere of Influence (radial extent of force from the center of the obstacle); |
| R | radius of obstacle; |
| G | adjustable Gain; |
| d | distance of robot to center of obstacle; |
| $O_{\text{direction}}$ | along a line from robot to center of obstacle moving away from obstacle |

3) **Avoid-robot** is a special case of **avoid-static-obstacle** where the robot to be avoided is treated as an obstacle using the formula above, but has a different parameter set (See Table IV).

4) **Noise**: Random wander with variable gain and persistence. Used to overcome local maxima, minima, cycles, and for exploration.

$$N_{\text{magnitude}} = \text{Adjustable gain value}$$
$$N_{\text{direction}} = \text{Random direction that persists for for } N_{\text{persistence}} \text{ steps } (N_{\text{persistence}} \text{ is adjustable})$$

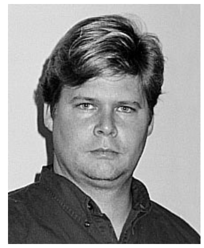
## Acknowledgment

The authors would like to thank D. MacKenzie and J. Cameron for writing the simulation software and helping debug the motor schema-based formation behaviors and developing the CNL language and compiler in which the formation behaviors are implemented in AuRA, K. Ali for assisting in porting the implementation to the UGV Demo II Architecture, B. Glass and M. Morgenthaler, Lockheed Martin, for completing the integration of the behaviors on DARPA's UGV's, and J. Pani and K. Ali for their help in gathering the experimental data on Nomad 150 robots.

## References

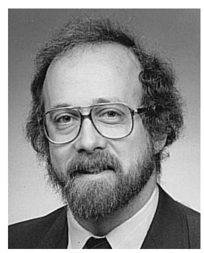
[1] R. C. Arkin, "Motor schema based mobile robot navigation," *Int. J. Robot. Res.*, vol. 8, no. 4, pp. 92–112, 1989.
[2] R. C. Arkin and T. R. Balch, "AuRa: Principles and practice in review," *J. Exper. Theor. Artif. Intell.*, vol. 9, no. 2, 1997.
[3] U.S. Army, *Field Manual No 7-7J*, Washington, DC, 1986.

[4] D. C. Brogan and J. K. Hodgins, "Group behaviors for systems with significant dynamics," *Auton. Robots*, vol. 4, no. 1, pp. 137–153, Mar. 1997.
[5] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Automat.*, vol. RA-2, p. 14, Feb. 1986.
[6] Q. Chen and J. Y. S. Luh, "Coordination and control of a group of small mobile robots," in *Proc. 1994 IEEE Int. Conf. Robot. Automat.*, San Diego, CA, 1994, pp. 2315–2320.
[7] D. J. Cook, P. Gmytrasiewicz, and L. B. Holder, "Decision-theoretic cooperative sensor planning," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1013–1023, Oct. 1996.
[8] J. M. Cullen, E. Shaw, and H. A. Baldwin, "Methods for measuring the three-dimensional structure of fish schools," *Animal Beh.*, vol. 13, pp. 534–543, 1965.
[9] U.S. Air Force, *Air Combat Command Manual 3-3*, Washington, DC, 1992.
[10] D. W. Gage, "Command control for many-robot systems," *Unmanned Syst. Mag.*, vol. 10, no. 4, pp. 28–34, 1992.
[11] D. Langer, J. Rosenblatt, and M. Hebert, "A behavior-based system for off-road navigation," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 776–783, Dec. 1994.
[12] D. MacKenzie, R. Arkin, and J. Cameron, "Multiagent mission specification and execution," *Auton. Robots*, vol. 4, no. 1, pp. 29–52, 1997.
[13] M. Mataric, "Designing emergent behaviors: From local interactions to collective intelligence," in *Proc. Int. Conf. Simulation of Adaptive Behavior: From Animals to Animats 2*, 1992, pp. 432–441.
[14] ——, "Minimizing complexity in controlling a mobile robot population," in *Proc. 1992 IEEE Int. Conf. Robot. Automat.*, Nice, France, May 1992, pp. 830–835.
[15] L. Parker, "Designing control laws for cooperative agent teams," in *Proc. 1993 IEEE Int. Conf. Robot. Automat.*, 1993, pp. 582–587.
[16] L. E. Parker, *Heterogeneous Multi-Robot Cooperation*, Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Mass. Inst. of Technol., Cambridge, MA, 1994.
[17] C. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *Comput. Graph.*, vol. 21, no. 4, pp. 25–34, 1987.
[18] J. Rosenblatt, "DAMN: A distributed architecture for mobile navigation," in *Working Notes AAAI 1995 Spring Symp. Lessons Learned for Implemented Software Architectures for Physical Agents*, Palo Alto, CA, Mar. 1995.
[19] X. Tu and D. Terzopoulos, "Artificial fishes: Physics, locomotion, perception, behavior," in *Proc. SIGGRAPH 94 Conf.*, Orlando, FL, July 1994, pp. 43–50.
[20] S. L. Veherencamp, "Individual, kin, and group selection," in *Handbook of Behavioral Neurobiology, Volume 3: Social Behavior and Communication*, P. Marler and J. G. Vandenbergh, Eds. New York: Plenum, 1987, pp. 354–382.
[21] P. K. C. Wang, "Navigation strategies for multiple autonomous robots moving in formation," *J. Robot. Syst.*, vol. 8, no. 2, pp. 177–195, 1991.
[22] H. Yamaguchi, "Adaptive formation control for distributed autonomous mobile robot groups," in *Proc. 1997 IEEE Conf. Robot. Automat.*, Albuquerque, NM, Apr. 1997.
[23] E. Yoshida, T. Arai, J. Ota, and T. Miki, "Effect of grouping in local communication system of multiple mobile robots," in *Proc. 1994 IEEE Int. Conf. Intell. Robots Syst.*, Munich, Germany, 1994, pp. 808–815.



**Tucker Balch** (M'98) received the B.S. degree from the Georgia Institute of Technology, Atlanta, the M.S. degree from the University of California, Davis, and the Ph.D. degree in computer science from the Georgia Institute of Technology, in 1998.

In 1984, he joined Lawrence Livermore National Laboratory, Livermore, CA, as a Computer Scientist. He left Livermore in 1988 to join the U.S. Air Force, where he flew F-15 Eagles until 1995. He holds the rank of Captain, Air Force Reserve. In 1996, he was a member of the Robotic Vehicles Group, Jet Propulsion Laboratory, Pasadena, CA. He is currently a Postdoctoral Fellow in the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. His recent work focuses on behavioral diversity and learning in multiagent societies. He is also interested in the integration of deliberative planning and reactive control, communication in multirobot societies, and parallel algorithms for robot navigation.



**Ronald C. Arkin** (M'90–SM'91) received the B.S. degree from the University of Michigan, Ann Arbor, the M.S. degree from the Stevens Institute of Technology, Hoboken, NJ, and the Ph.D. degree in computer science from the University of Massachusetts, Amherst, in 1987.

He then became an Assistant Professor in the College of Computing, Georgia Institute of Technology, Atlanta, where he is Professor and the Director of the Mobile Robot Laboratory. His research interests include reactive control and action-oriented perception for the navigation of mobile robots and unmanned aerial vehicles, robot survivability, multiagent robotic systems, and learning in autonomous systems. He has over 80 technical publications in these areas. He has recently completed a new textbook entitled *Behavior-Based Robotics* (Cambridge, MA: MIT Press, May 1998) and co-edited a book entitled *Robot Colonies* (Norwell, MA: Kluwer, 1997). He serves/served as an Associate Editor for IEEE EXPERT and the *Journal of Environmentally Conscious Manufacturing*, as a member of the Editorial Boards of *Autonomous Robots* and the *Journal of Applied Intelligence*, and is the Series Editor for the new MIT Press book series *Intelligent Robotics and Autonomous Agents*.

Dr. Arkin is a member of AAAI and ACM.