Motor Schema — Based Mobile Robot Navigation

Ronald C. Arkin

The International Journal of Robotics Research 1989; 8; 92
DOI: 10.1177/027836498900800406

Ronald C. Arkin
School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia

Motor Schema–Based Mobile Robot Navigation

Abstract

Motor schemas serve as the basic unit of behavior specification for the navigation of a mobile robot. They are multiple concurrent processes that operate in conjunction with associated perceptual schemas and contribute independently to the overall concerted action of the vehicle. The motivation behind the use of schemas for this domain is drawn from neuroscientific, psychological, and robotic sources. A variant of the potential field method is used to produce the appropriate velocity and steering commands for the robot. Simulation results and actual mobile robot experiments demonstrate the feasibility of this approach.

1. Introduction

Path planning and navigation, at the execution level, can most easily be described as a collection of behaviors. *"Don't run into things!" "Go to the end of the sidewalk, then turn right!" "Stay to the right side of the sidewalk except when passing!" "Watch out for the library—the turn is just beyond it!" "Follow that man!"* This collection of commands constitutes some of the possible behaviors for an entity trying to move from one location to another. Traditional control structures—those that use an inflexible and rigid approach to navigation—do not provide the essential adaptability necessary for coping with unexpected events. These events might include unanticipated obstacles, moving objects, or the recognition of a landmark in a seemingly inappropriate location. These unexpected occurrences should influence, in an appropriate manner, the course that a vehicle (or person) takes in moving from start to goal.

A solution to the problem of dynamic replanning in this context can be drawn from models that have been developed in the domains of brain theory and robotics. Schemas, a methodology used to describe the interaction between perception and action, can be adapted to yield a mobile robot system that is highly sensitive to the currently perceived world. Motor schemas operating in a concurrent and independent, yet communicating, manner can produce paths that reflect the uncertainties in the detection of objects. Additionally they can cope with conflicting data arising from diverse sensor modalities and strategies.

The purpose of this article is to provide insights into the design of a control system based on motor schemas for mobile robots. Section 2 describes the motivations for the use of schema theory in this domain, drawing from work in both brain theory and robotics. Section 3 discusses the tack taken for a motor schema-based control system in the Autonomous Robot Architecture (AuRA) (Arkin 1987b), utilizing a mobile robot equipped with ultrasonic and video sensors; it is concerned specifically with the role of the pilot and the motor schema manager. Section 4 presents the results of simulations using schemas that specify different behaviors and draw on simulated sensor input. Section 5 presents some schema-based navigational experiments using the mobile robot HARV. A summary and evaluation conclude this article.

2. Motivation

The concept of schemas originated in psychology (Bartlett 1932; Oldfield and Zangwill 1942; Piaget 1971) and neurology (Head and Holmes 1911; Frederiks 1969). The model used for this paper draws on more recent sources: the applications of schema theory to brain modeling and robotics. As brain theory

Portions of this paper originally appeared in the Proceedings of the IEEE International Conference on Robotics and Automation, *April 1987 pp. 264–271,* ©1987 IEEE. Reprinted by permission.

can unequivocally be called a sound basis for the study of intelligent behavior, the first part of this section will present the contributions of brain science that influenced the design of the schema control system described below. Roboticists for some time have drawn on schema theory, but not always in the form envisioned by brain theoreticians. The previous work in robotics that relates to the schema-based approach to navigation is described in the final part of this section.

2.1. What is a Schema?

Reactive/reflexive navigation is conducted through the instantiation of motor schemas. Unfortunately, the definitions of schema are highly variable, and the term is probably overused. In order to provide context for the discussion that follows, several definitions that are closely allied with the philosophy of this paper are cited below.

A schema is:

1. A pattern *of* action as well as a pattern *for* action (Neisser 1976).
2. A mental codification of experience that includes a particular organized way of perceiving cognitively and responding to a complex situation or set of stimuli (Merriam-Webster 1984).
3. A generic specification of a computing agent (Lyons 1986a).
4. A control system that continually monitors feedback from the system it controls to determine the appropriate pattern of action for achieving the motor schema's goals (Overton 1984).
5. An adaptive controller that uses an identification procedure to update its representation of the object being controlled (Arbib 1981).

In our case, each individual motor schema corresponds to a primitive behavior that, when combined with other motor schemas, can yield more complex behaviors. The pilot-selected schemas react to their world via sensing and do not draw on an a priori world model for navigation. Each motor schema has an embedded perceptual schema (an *identification procedure*) to provide the necessary sensor information for the robot to relate to its world. The motivation for motor schema usage draws on cognitive science, psychological, neuroscientific, and robotic sources.

2.2. Brain Theory and Psychology

The action-perception cycle provides a principal motivation for the application of schema theory (Neisser 1976). Sensor-driven expectations provide the plans (schemas) for appropriate motor action, which when undertaken provide new sensory data that is fed back into the system to provide new expectations. This cycle of cognition (the altering of the internal world model), direction (selection of appropriate motor behaviors), and action (the production of environmental changes and resultant availability of new sensory data) is central to the way in which schemas must interact with the world.

Most significantly, perception should be viewed as action-oriented. There is no need to process all available sensor data, only that data that is pertinent to the task at hand. The question for the roboticist is how to select from the wealth of available sensor data only that which is relevant. By specifying schemas, each individual component of the overall task can make its demands known to the sensory subsystem, and thus guide the focus of attention mechanisms and limited sensory processing that is available.

Guided by Arbib's work (Arbib 1972; 1981) in the study of the frog and its machine analog *Rana Computatrix,* the frog prey selection mechanism serves as a basis for analysis. In particular, Arbib and House (1987) have developed a model for worm acquisition by the frog in an obstacle-cluttered environment (a spaced fence). Although Arbib and House describe two models to account for the behavior of the frog, the second is the most readily applicable to the mobile robot's domain (the first model is based on visual orientation). In their work, they describe primitive vector fields: a prey-attractant field, a barrier-repellent field, and a field for the animal itself. These fields, when combined, yield a model of behavior that is consistent with experimental observations of the frog.

In the mobile robot system described below, analogs of these fields are used (prey-attractant ⇒ **move-to-goal,** barrier-repellent ⇒ **avoid-static-obstacle).** Additionally, new fields are added to describe other motor tasks (**stay-on-path, avoid-moving-obstacle,** etc.)

This model, in conjunction with expectation-driven sensing, provides a basic correlate with the functioning of the brain (albeit the frog brain). Although the brain has been handling visually guided detours since time immemorial, the benefits of using a neuroscience model would wane if it proved impractical for a mobile robot. In the sections following, the practicality of this approach is demonstrated, especially regarding the decomposition of the task to a form which is readily adaptable to distributed processing. This is essential if the real-time demands of mobile robot environmental interaction are to be met.

2.3. Robotics

Schema theory as applied to robotics has almost as many different definitions as there are developers. In the realm of robotic manipulators, Lyons' schemas (Lyons 1986a) and Geschke's servo processes (a schema analog) (Geschke 1983) are used as approaches to task level control. Overton (Overton 1984) has described the use of motor schemas in the assembly domain. The UMass VISIONS group, guided by Riseman and Hanson (1987), has applied perceptual schemas to the interpretation of natural scenes; Weymouth's thesis (Weymouth 1986) and Draper's paper (Draper et al. 1987) are prime examples of this work. However, perceptual schemas as they appear in the VISIONS system are not a principal concern of this article.

One of the simplest and most straightforward definitions for a schema is "a generic specification of a computing agent" (Lyons 1986a). This definition fits well with the concept of a behavior (an entity's response to its environment)—each schema represents a generic behavior. Schema-based control systems are significantly more than a collection of frames or templates for behavior, however. The way in which they are set into action and interact immediately distinguishes them from simpler representational forms. The instantiations of these generic schemas provide the potential actions for the control of the robot. A schema instantiation (SI) is created when a copy of a generic schema is parameterized and activated as a computing agent.

Other work in the path-planning domain, although not schema-based, bears a resemblance to the schema control system. Brooks (1986) uses a planning system with a "horizontal decomposition" that effectively emulates multiple behaviors. Payton (1986) describes a multi-behavior approach for reflexive control of an autonomous vehicle. The association of virtual sensors with a selected set of reflexive behaviors bears a similarity to the schema-based approach. However, an arbitrary choice of behavior, based on a priority system, is made during execution without provision for a mechanism to combine the results of concurrent behaviors. Kadonoff et al. (1986) also incorporate multiple behaviors for the control of a mobile robot and similarly arbitrate between these behaviors, proposing a production system for arbitrating competitive strategies and the use of an optimal filter for the treatment of complementary strategies.

The schema system described below is strongly influenced by Krogh's (1984) generalized potential fields approach and to a lesser degree by Lyons' (1986b) tagged potential fields. It bears a superficial resemblance to the integrated path-planning and dynamic steering-control system described by Krogh and Thorpe (1986). Potential fields are used, in each case, to produce the steering commands for a mobile robot. A major distinction between their system and our schema model lies in the tracking of the individual obstacles (individual SIs for each obstacle, important for the treatment of uncertainty) and the incorporation of additional behaviors such as road following and treatment of moving obstacles. The state of each obstacle's SI is dynamically altered by newly acquired sensory information. The potential functions for each SI reflect the measured uncertainty associated with the perception of each object. The schema approach is not limited to obstacle avoidance, but is versatile enough for road following, object tracking, and other behavioral patterns.

3. Approach

Motor schemas, when instantiated, must drive the robot to interact with its environment. On the highest level, this will be to satisfy a goal developed within the planning system; on the lowest level, to produce specific translations and rotations of the robot vehicle. The schema system enables the software designer to deal with conceptual structures that are easy to comprehend and handle. The task of robot programming is fundamentally simplified through the use of a "divide and conquer" strategy.

3.1. Schema-Based Navigation

AuRA's pilot is charged with implementing leg-by-leg the piecewise linear path developed by the navigator. To do so, the pilot chooses from a repertoire of available sensing strategies and motor behaviors (schemas) and passes them to the motor schema manager for instantiation. Distributed control and low-level planning occur within the confines of the motor schema manager during its attempt to satisfy the navigational requirements. As the robot proceeds, AuRA's cartographer, using sensor data, builds up a model of the perceived world in short-term memory. If the actual path deviates too greatly from the path initially specified by the navigator as a result of the presence of unmodeled obstacles or positional errors, the navigator will be reinvoked and a new global path computed. If the deviations are within acceptable limits (as determined by higher levels in the planning hierarchy), the pilot and motor schema manager will, in a coordinated effort, attempt to bypass the obstacle, follow the path, or cope with other problems as they arise. Additionally, the problem of robot localization is constantly addressed through the monitoring of short-term memory and appropriate **find-landmark** schemas. Multiple concurrent behaviors (schemas) may be present during any leg, for example:

Stay-on-path (a sidewalk or a hall)
Avoid-static-obstacles (parked cars, trees, etc.)
Avoid-moving-obstacles (people, moving vehicles, etc.)
Find-intersection (to determine end of path)
Find-landmark (building for localization)

The first three are examples of motor schemas; the last two, perceptual schemas. To provide the correct behavior, a subset of perceptual schemas must be associated with each motor schema. For example, in order to stay on the sidewalk, a **find-terrain** (sidewalk) perceptual schema must be instantiated to provide the necessary data for the **stay-on-path** motor schema to operate. If the uncertainty in the actual location of the sidewalk can be determined, the SI's associated velocity field, applying pressure to remain on the sidewalk, will reflect this uncertainty measure. The same holds for obstacle avoidance: if a perceptual schema for obstacle detection returns the position of a suspected obstacle and the relative certainty of its existence, the actual avoidance maneuvering will depend not only on whether an obstacle is detected but also on the certainty of its existence. Differing strategies within each SI will determine how to manage the perceptual uncertainty. If an event is potentially fatal, even large amounts of perceptual uncertainty will produce motor behavior, but erring in the direction of safety.

An example illustrating the relationship between motor schemas and perceptual certainty follows. The robot is moving across a field in a particular direction (**move-ahead** schema). The **find-obstacle** schema is constantly on the lookout for possible obstacles within a subwindow of the video image (windowed by the direction and velocity of the robot). When an event occurs (e.g., a region segmentation algorithm detects an area that is distinct from the surrounding backdrop, or an interest operator locates a high-interest point in the direction of the robot's motion), the **find-obstacle** schema spawns off an associated perceptual schema (**static-obstacle** SI) for that portion of the image. It is now the **static-obstacle** SI's responsibility to continuously monitor that region. Any other events that occur elsewhere in the image spawn off separate **static-obstacle** SIs. Additionally an **avoid-static-obstacle** SI motor schema is created for each detected potential obstacle.

The motor schema SI hibernates, waiting for notification that the perceptual schema is sufficiently confident in the obstacle's existence to warrant motor action. If the perceptual schema proves to be a phantom (e.g., shadow) and not an obstacle at all, both the perceptual and related motor SIs are deinstantiated before producing any motor action. On the other hand, if the perceptual SI's confidence (activation level) exceeds the motor SI's threshold for action, the motor schema starts producing a repulsive field surrounding the obstacle.[1] The sphere of influence (spatial extent of repulsive forces) and the intensity of repulsion of the obstacle are affected by the distance from the robot and the obstacle's perceptual certainty. Eventually, when the robot moves beyond the perceptual range of the obstacle, both the motor and perceptual SIs are deinstantiated. In summary, when obstacles are detected with sufficient certainty, the motor schema associated with a particular obstacle (its SI) starts to produce a force tending to move the robot away from the object. Figure 1A shows a typical repulsive field for an **avoid-static-obstacle** SI. The control of the priorities of the behaviors (e.g., when is it more important to follow the sidewalk than to avoid uncertain but possible obstacles) is partially dependent on the uncertainty associated with the obstacle's representation. Other isolated motor schema velocity fields are shown in Figure 1B–D. A typical combination of motor schemas is illustrated in Figure 2. Although the entire field is expensive to compute, each active motor SI need only determine the velocity vector at the robot's current location relative to the environmental object, making the computation very rapid. Further, as the SIs are activated in parallel, even better performance is attainable.

Multiple instantiations of a single schema are frequently the case. Each generic "skeleton" is parameterized when instantiated. Consequently, it is entirely possible that two different instantiations of the same generic schema produce significantly different fields under similar sensory conditions (as in the case of path following for a sidewalk or hall discussed above). The parameters set at instantiation may depend on the sensory events that triggered the instantiation or from information retrieved by the pilot from LTM.

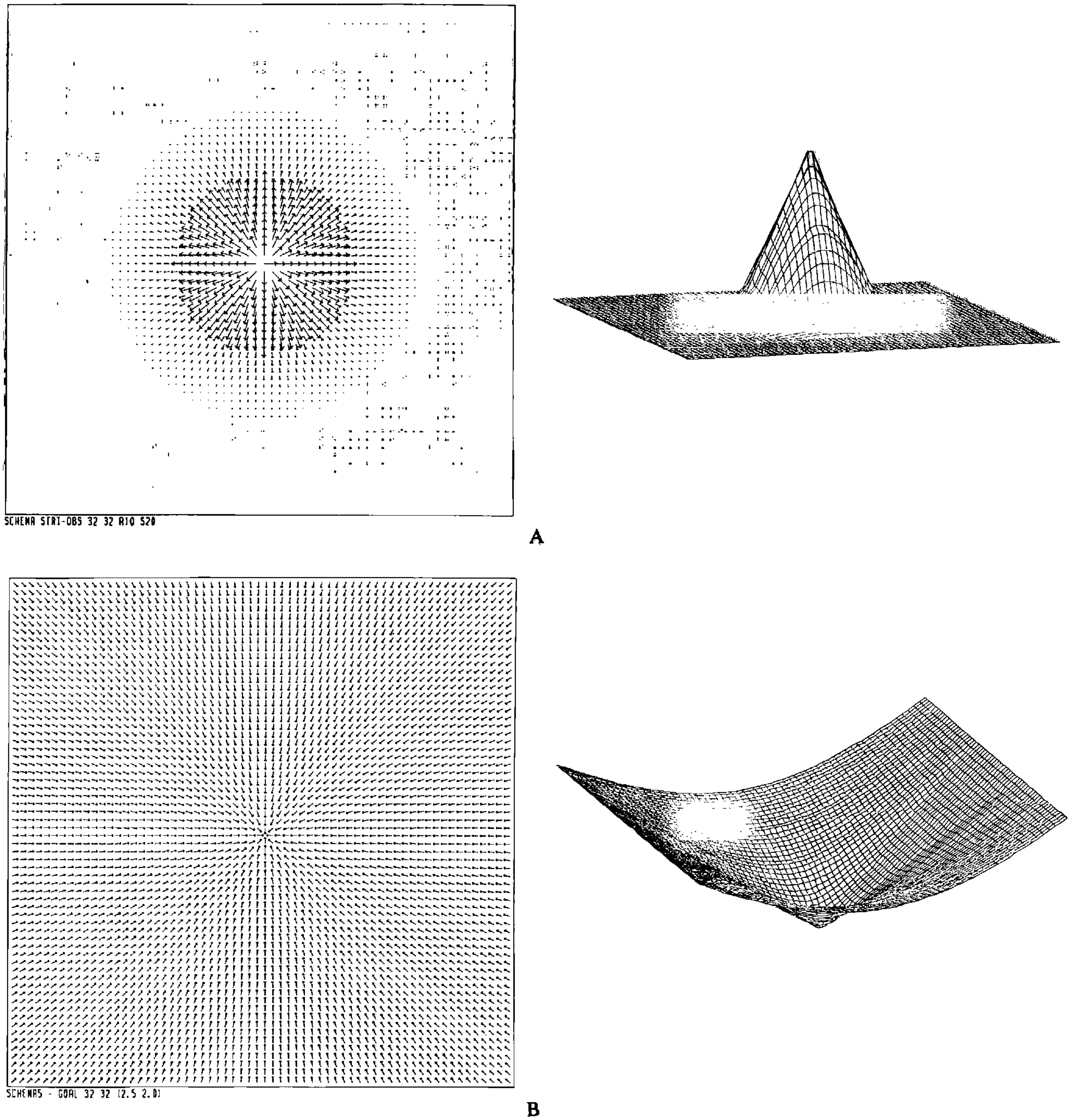
If each schema functions independently of the others, how can any semblance of realistic and consistent behavior be achieved? Two components are required to satisfactorily answer this question. First, a combination mechanism must be applied to all the SI-produced vectors. The result is then used to provide the necessary velocity changes to the robot. The simplest approach is vector addition. By having each motor SI create a normalized velocity vector, a single **move-robot** schema monitors the posted data for each SI, adds them together, makes certain it is within acceptable bounds, and then transmits it to the low-level robot control system. In essence, the specific velocity and direction for the robot can be determined at any point in time by summing the output vectors of all the active individual SIs. As each motor SI is a distributed computing agent (preferably operating on separate processors on a parallel machine) and needs only to compute the velocity at the point the robot is currently located and a few points in its projected track (and not the entire velocity field), real-time operation is within reach.

The second component of the response to the question posed in the previous paragraph is communication. Potential fields can have problems with dead spots or plateaus where the robot can become stranded. By allowing communication mechanisms between the SIs, the forces of conflicting actions can be reconciled. Lyons (1986a) proposes message passing between ports on one SI and connected ports on another SI as a schema communication mechanism. Alternatively, a blackboard mechanism is used in the VISIONS Schema Shell (Draper et al. 1987). In either case, communication mechanisms can solve problems that might otherwise prove intractable. An example to illustrate this point follows.

The robot is instructed to move in a particular direction, stay on the sidewalk, and avoid static obstacles. Suppose that the sidewalk is completely blocked by an obstacle; eventually the velocity would drop to 0 and the robot would stop (Fig. 3A). The fact that the vehicle has stopped is detected by the **stay-on-path** SI through interschema communication with the **move-robot** SI (the **move-robot** SI combines the individual motor SIs and communicates the results to the low-level motor control system). The **stay-on-path** SI, when created for this particular instance, was instructed to

[1] The obstacle is first grown in a configuration space manner (Lozano-Perez 1982) to enable the robot to be treated henceforth as a point for path planning purposes.

Fig. 1. Isolated motor schema SI vector fields. (*A*), **avoid-static-obstacle.** (*B*) **move-to-goal.** (*C*), **move-ahead.** (*D*) **stay-on-path.**



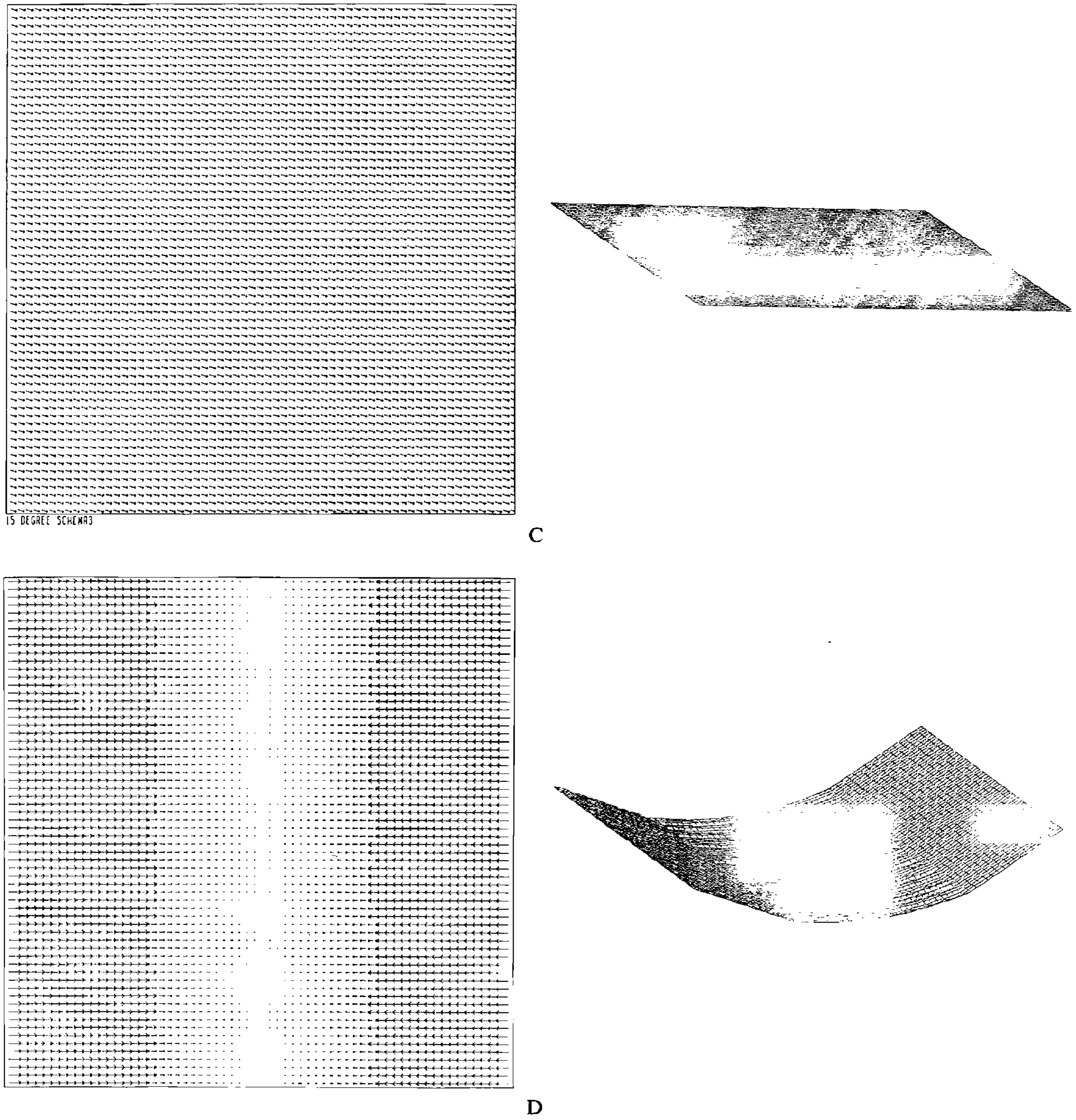
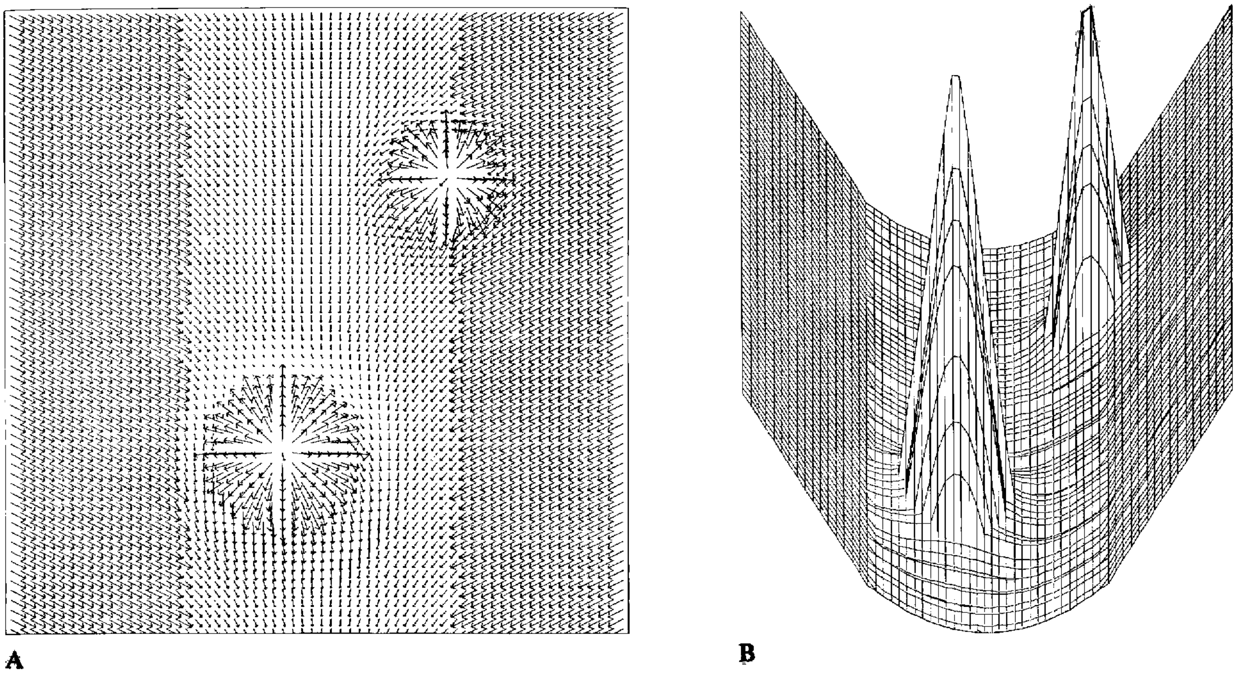

15 DEGREE SCHEMA3
C
D

Fig. 2. Combined motor schemas; two **avoid-static-obstacle** *SIs + one* **stay-on-path** *SI + one* **move-to-goal** *SI. (A) 2-D representation. (B) 3-D analog.*



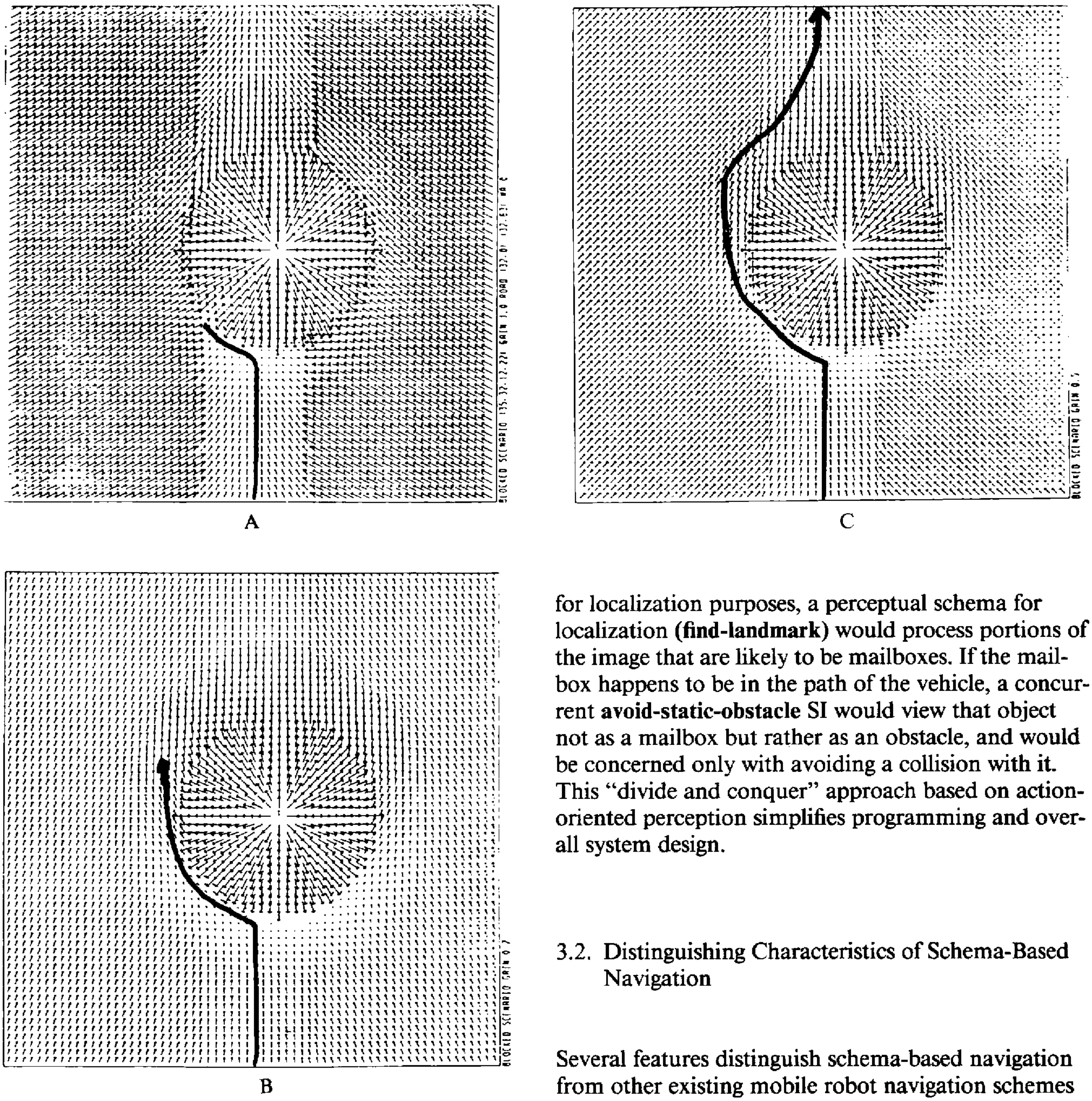
yield if an obstacle blocks the path. The **stay-on-path** motor schema reduces its field (Fig. 3B) and allows the robot to wander off the sidewalk, thus circumnavigating the obstacle. As soon as the direction of the force produced by the offending obstacle indicates it has been successfully passed, the **stay-on-path** field returns to its original state, forcing the robot back on the path (Fig. 3C).

Suppose, however, the **stay-on-path** SI was instantiated for a hall. Then, under no circumstances would the force field associated with the **stay-on-path** SI be reduced, or else the robot would crash into the wall. The robot would instead stop and signal for the navigator (higher level component of the planner) to be reinvoked and produce an alternate global path that avoids the newly discovered blocked passageway. These communication pathways are specified within the schema structures themselves.

It is entirely possible that the trajectory of the robot can be computed for a small distance rather than just its instantaneous velocity at the immediate location. Each motor schema would have to interact with the **move-robot** SI, using the vector summation output to determine the position of the robot relative to its perceptions for the next time step. This is of particular significance if the sensor sampling rates are low. Trajectories can be determined that reflect the robot's perceptions at a given point in time, rather than just reacting to current sensing. This is of value in determining when to activate other schemas in anticipation of special problems or needs. Care must be taken in highly dynamic environments (e.g., moving objects) so that the plans developed do not ignore changes in the world that are evidenced only through perception.

Another approach explored is the addition of a background stochastic **noise** schema. This SI produces a low-magnitude random direction velocity vector that changes at random time intervals but persists sufficiently long to produce a change in the robot's position if the robot's velocity was otherwise zero. Its role is to perturb the velocity of the robot slightly, removing the robot from undesirable equilibrium points that arise when the active motor SIs counterbalance each other. This schema would serve to remove the robot from any potential field plateaus or ridges upon which the robot becomes perched (e.g., from a direct approach to an obstacle; Fig. 4). Other traps common to potential field approaches (e.g., box canyons) can be handled by establishing hard real-time deadlines for goal attainment. If these deadlines are violated, the pilot is rein-

Fig. 3. Blocked sidewalk scenario. (A) Robot stops in dead spot due to pressure to both remain on sidewalk and avoid the obstacle. (B) Gain lowered on **stay-on-path** *SI allows robot to bypass obstacle. (C) Once obstacle is passed* **stay-on-path** *SI returns to normal, forcing robot back onto the sidewalk.*



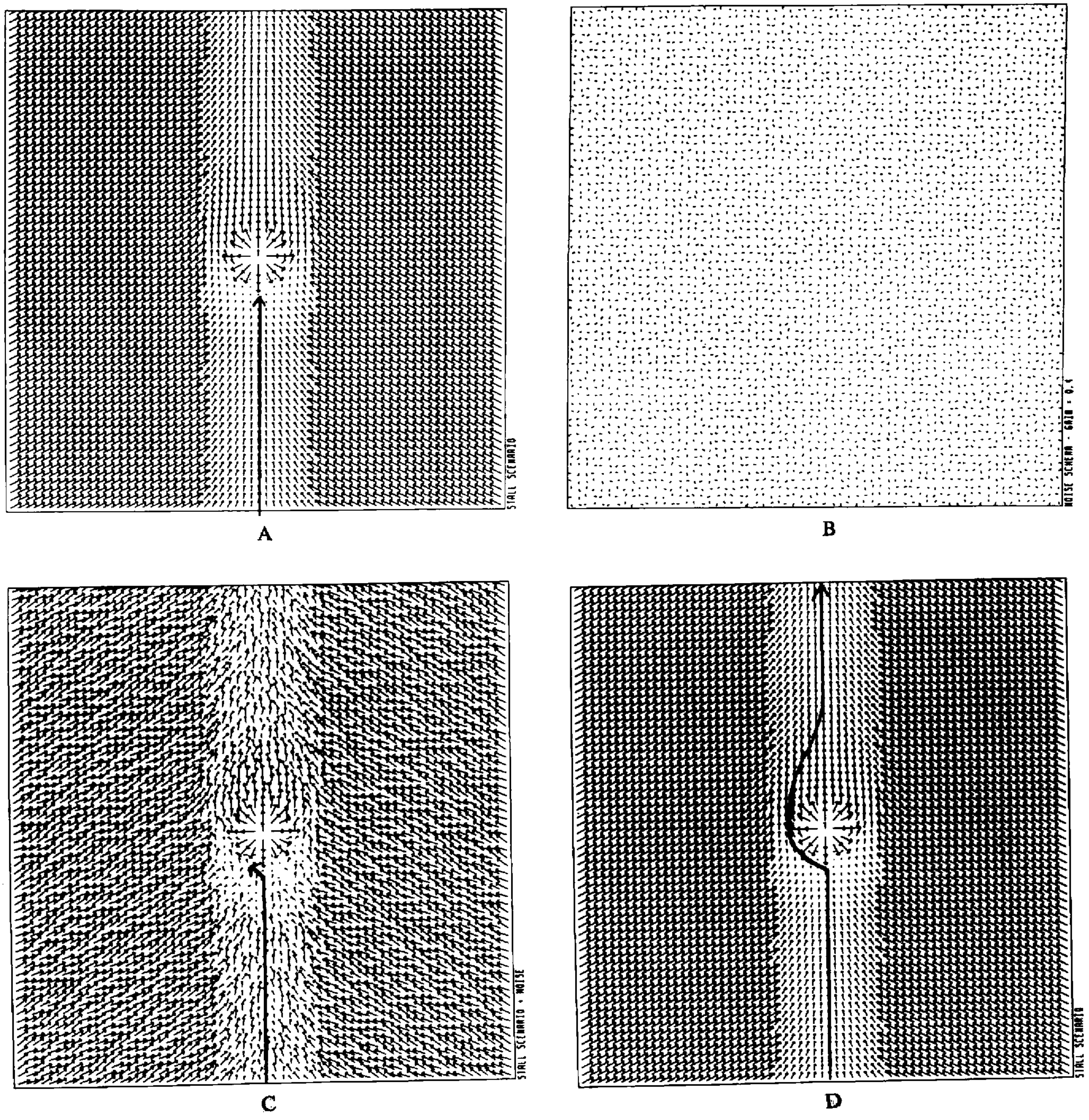
voked to establish an alternate route using STM data gathered by the cartographer during the route traversal.

It is worth noting that a single sensory event may have two or more SIs associated with it. For example, if the robot is looking for a mailbox to get its bearings for localization purposes, a perceptual schema for localization **(find-landmark)** would process portions of the image that are likely to be mailboxes. If the mailbox happens to be in the path of the vehicle, a concurrent **avoid-static-obstacle** SI would view that object not as a mailbox but rather as an obstacle, and would be concerned only with avoiding a collision with it. This "divide and conquer" approach based on action-oriented perception simplifies programming and overall system design.

3.2. Distinguishing Characteristics of Schema-Based Navigation

Several features distinguish schema-based navigation from other existing mobile robot navigation schemes (Brooks 1986; Krogh and Thorpe 1986). Superficially it may appear that a schema is a potential field; it is not. The *output* of a schema is a single velocity vector derived from a potential field formulation of the forces exerted upon the robot at any particular point in space. The entire potential field is never computed

Fig. 4. Stall scenario. (A) Robot approaches an obstacle exactly head-on, and becomes stalled. (B) **Noise** *SI provides small magnitude random direction vector to push robot off of the tiny plateau. (C)* **Noise** *schema added to (A). (D) Robot can now successfully bypass the obstacle; the* **noise** *SI is then deinstantiated.*



by the robot, and only a single vector based on the robot's current perception of the world is used to compute its trajectory through space (the full-field figures presented in this paper serve only to enhance the reader's comprehension). Thus the computational demand of any single schema is quite small.

The schema itself is an active individual computing agent. This methodology differs from Brooks' subsumption architecture (Brooks 1986) by avoiding layering entirely and instead setting up a dynamic network of schemas based on the current goals of the robot. There is no layering at all; it is more of a souplike collection of networked autonomous agents whose configuration changes dynamically as the needs and perceptions of the vehicle change. Lyons' port automata formulation for schemas (Lyons 1986a) presents a formal model for setting up the network of schemas. Schemas are *not* merely frames.

A principal research issue for mapping this navigational methodology onto distributed hardware involves dynamic load balancing, as there exists no a priori method for determining the total number of schemas to be invoked before navigation is undertaken. Since each obstacle detected will have an associated **avoid-obstacle** SI, and there is no means of foretelling how many individual obstacles will be encountered at any one time during the robot's path traversal, the total number of SIs active cannot be predicted ahead of time.

One other distinguishing characteristic of our use of schemas lies in their ability to allow perceptual uncertainty to affect the potential field output of the active motor schemas. As each motor SI has an embedded perceptual schema, if the perceptual process can provide a measure of belief in its perception, the motor schema can reflect that belief by acting lethargically in the presence of questionable evidence or by discounting it entirely.

In summary, schema-based navigation is unique in many aspects: it is a dynamic network of active computing agents as opposed to a layered architecture; the configuration of schemas is based on the robot's current perceptual needs and desired motor behaviors of the vehicle realized from a priori knowledge of the robot's world and goals; its ability to reflect uncertainty in perception readily; its roots that are motivated by psychological and neuroscientific studies (Arkin 1988); and the existence of formal models useful for its implementation (Lyons 1986a).

4. Simulation

Simulations were run on a VAX 750 using the following motor schemas: **stay-on-path, move-ahead, move-to-goal, avoid-static-obstacle.** An example simulation run (Fig. 5) shows the sequence of resultant overall force fields based on perceived entities. These entities include path borders, goals, and obstacles. The grid size is 64 units by 64 units and the sensory sampling update time (once per second) is based on a nominal velocity of 1 unit/s. The maximum vector length for display purposes has been set to 2.0 normal velocity units. The actual vector magnitude within the obstacles is set to infinity. All obstacles are currently modeled as circles (as in Moravec's tangent space [Moravec 1981]). The field equations for several of the motor schemas appear below.

The field equations for both the **avoid-static-obstacle** and **stay-on-path** schemas are linear. An example showing the velocity produced by an obstacle (O) is given below:

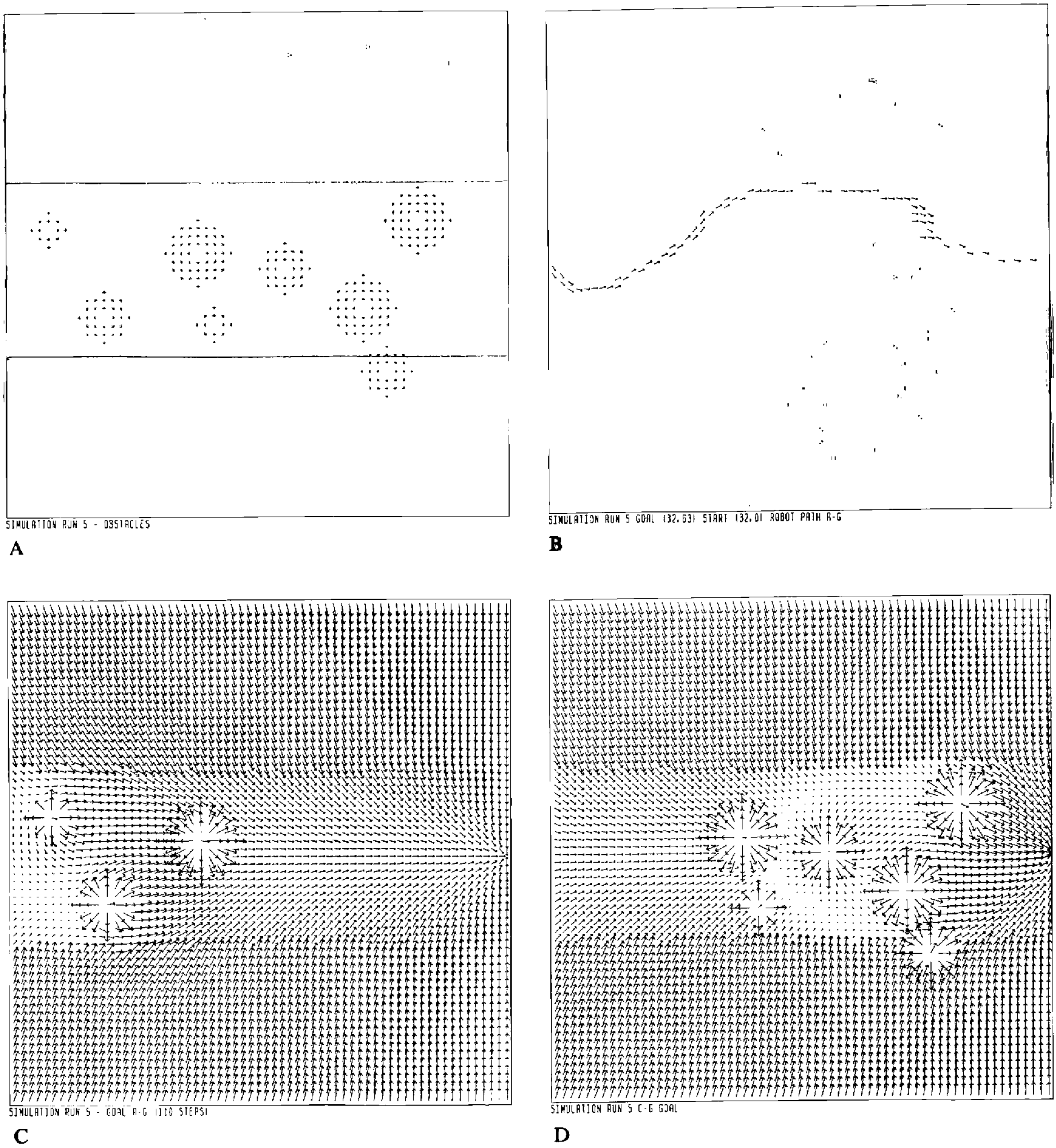
Avoid-obstacle

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

where:

S = sphere of influence (radial extent of force from the center of the obstacle)
R = radius of obstacle
G = gain
d = distance of robot to center of obstacle
$O_{direction}$ = along a line from robot to center of obstacle, moving away from obstacle

Fig. 5. 2D simulation run. (A) Obstacles and path the robot is to traverse. (B) Robot's path through the course. (C,D) Velocity fields produced as the robot approaches the goal.



More complex equations could be used (e.g., cubic as in Krogh and Thorpe [1986]) but were deemed unnecessary for the low velocities of our vehicle.

Stay-on-path

$$V_{magnitude} = \mathrm{P} \quad \text{for } d > (W/2)$$
$$\frac{d}{(W/2)} * G \quad \text{for } d \leq \frac{W}{2}$$

where:

W = width of path
P = off path gain
G = on path gain
d = distance of robot to center of path
$V_{direction}$ = along a line from robot to center of path heading toward centerline

Move-ahead

$V_{magnitude}$ = fixed gain value
$V_{direction}$ = in specified compass direction

Move-to-goal

$V_{magnitude}$ = fixed gain value
$V_{direction}$ = in direction towards perceived goal

In this simulation, the uncertainty in perception was allowed to decrease the sphere of influence of an obstacle. When a threshold was exceeded (50% certain), the sphere of influence of the obstacle started increasing linearly as the certainty increased, up to its maximum allowable value. Another alternative is to increase the gain on the obstacle proportionately with the increase in certainty (up to its maximum).

Figure 5A illustrates the robot's course on a sidewalk moving toward a goal. The course is studded with eight obstacles. Note how the vector fields change as the robot encounters more obstacles along the way (Fig. 5C–D). When the robot has successfully navigated obstacles and they have moved out of range, their representation is dropped from short-term memory, and the associated motor schema is deinstantiated. The robot stays on the path for the complete course (Fig. 5B), successfully achieving its goal while avoiding each obstacle.

5. Experiments in Motor Schema–Based Navigation

A real-time schema experimentation/demonstration system has been developed using the mobile robot HARV (a Denning Research Vehicle), based on sensing using ultrasonic and encoder data. The hooks for tying in visual sensing are in place but are currently not implemented as a result of the slow processing speeds for vision. Vision experiments using HARV are described in Arkin, Riseman, and Hanson (1987).

Five different motor schemas have been implemented: **move-ahead** (encoder-based), **move-to-goal** (encoder-based), **avoid-static-obstacle** (ultrasonic-based), **noise** (sensor-independent), and **follow-the-leader** (ultrasonic-based). The user is able to select the collection of motor schemas to use and associate a perceptual schema with each. After schema selection is complete, robot motion is initiated. The vehicle then behaves in an intelligent manner in response to its environmental stimuli. Several of the more interesting behaviors are described below. It should be noted that the schemas for this system emulate distributed processing but actually are evaluated sequentially.

Related results for experimental robot performance have been previously reported by this author (Arkin 1987a,b) and other researchers (Brooks and Connell 1986; Triendl and Kriegman 1987) using different navigational methodologies. In particular, Brooks' robot at MIT (Brooks and Connell 1986) exhibits similar motor performance for the avoidance, wandering, and wall-following behaviors in the context of his subsumption architecture.

5.1. Avoidance

By instantiating the **avoid-static-obstacle** schema with ultrasonic perception, the robot manifests an interesting behavior. The schema instantiation can be controlled by altering either the gain, which affects the velocity of the vehicle, or the sphere of influence of detected obstacles, which increases its sensitivity to the environment. When activated, the robot seeks out a potential field minimum and remains there unmoving (or slightly oscillating in cluttered environments as a result of the limited sampling rates and noise in the sensor data).

If a change in the environment occurs (e.g., a person approaches), the robot is repulsed and seeks out a new potential minimum. The robot can be "herded" by following behind it, forcing it to move to a desired location. It avoids obstacles during its journey and then settles into its new location when the environment stabilizes. Figure 6 illustrates this process.

5.2. Exploration

By combining the **noise** schema with the **avoid-static-obstacle** schema, exploration behavior can be observed. The **noise** schema's gain (strength) and persistence (how frequently the direction changes) can be set at startup by the experimenter. The robot meanders about the lab, exploring different regions while avoiding collision with the obstacles.

The robot responds quite well to changes in its environment, as when people surround it during a demonstration. The biggest problem the robot faces is the slow sensor sampling, which makes its reflexes quite slow. The vehicle can also be herded when running in this behavioral mode, but it is not quite as obedient, moving in the general direction forced upon it, but occasionally making some slight sidesteps as a result of the presence of noise. Actually this form of herding is more reminiscent of an animal's behavior, and HARV has been likened to a sheep by some observers when it is running in this mode.

5.3. Hall Following

By instantiating the **move-ahead** schema with an **avoid-static-obstacle** schema, the robot is able to safely

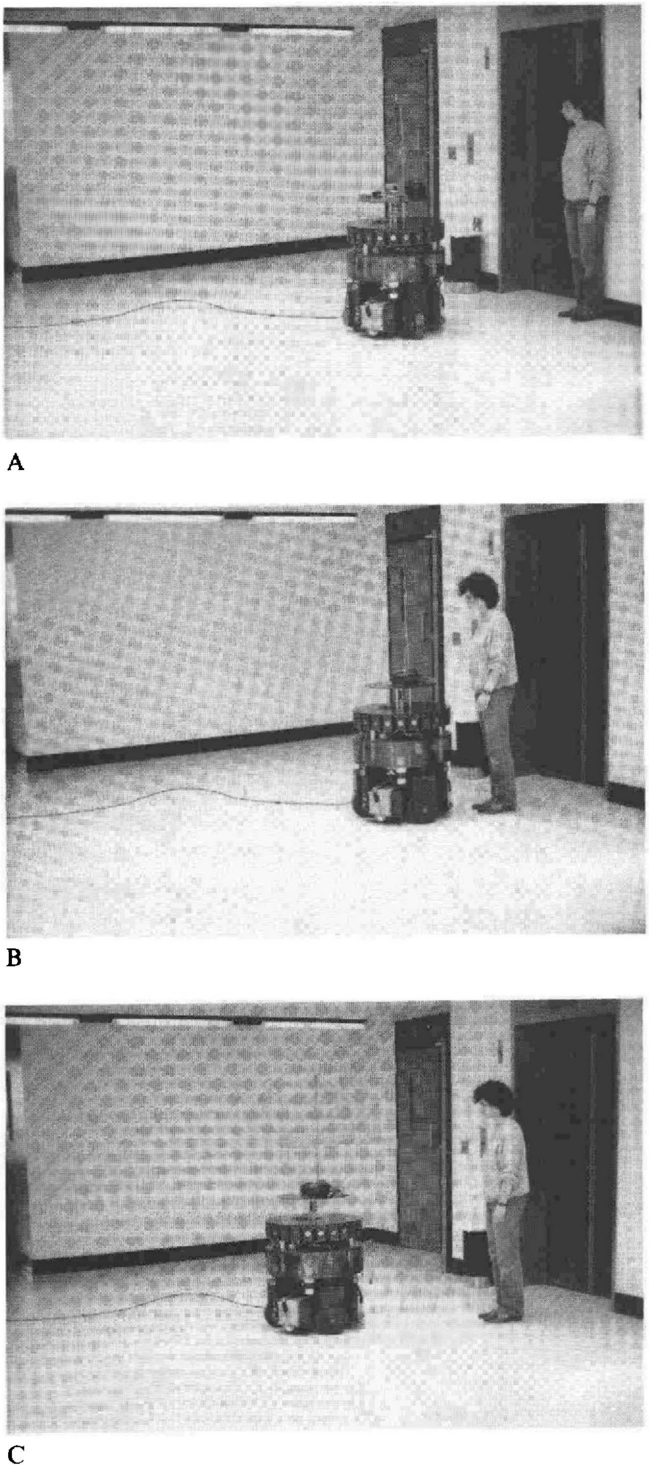


Fig. 6. Avoidance behavior. Robot is initially stationary (A). Robot is approached (B), is repulsed, and moves to its new position and stops (C). This behavior is produced by the instantiation of an **avoid-static-obstacle** *schema.*

travel down a hall. Although there is no model of the hall at the schema level, the robot senses the two sides as obstacles and positions itself near the center. The **move-ahead** schema drives it forward.

HARV responds to changes in the hall structure itself such as pillars, seeking out the middle of the hall. Small obstacles placed near the wall are interpreted as part of the wall by the ultrasonic sensors. Open doorways pose no problem as long as the **move-ahead** direction is roughly parallel to the hall. If this direction points steeply into the wall, the robot might pass through an open door (see door entry, described in section 5.5), but since the angle into the wall typically must be at least 45° or more for normal doorway entry, this allows a latitude of about 90° (±45°) of error in the **move-ahead** direction for successful navigation of halls using this method.

5.4. Navigation in the Presence of Obstacles

The same combination of schemas allows for navigation in cluttered hallways or outdoor situations. The **move-to-goal** schema can be substituted freely for the **move-ahead** schema if desired. Fig. 7 shows the robot's course through a series of cluttered obstacles outdoors. The **move-to-goal** schema relies on the shaft encoders to move the robot to a particular point. This is accomplished by specifying the distance to and direction toward the goal and then monitoring the encoder data to provide the input to the **move-to-goal** SI.

The robot's minimum detectable ultrasonic sensor reading is 0.9 ft. For this reason, whenever the ultrasonic sensor returns a value of 0.9, the robot must consider a collision to be imminent. This effectively increases the robot's diameter by almost 2 ft, making it more difficult to squeeze through tight spaces. This is particularly evident in the indoor hallway examples. A consequence of this fact, when coupled with the slow sampling rates for ultrasonic data (2–3 s for a complete scan, transmission, and interpretation by the VAX), is the production of motion oscillations when the robot is operating under continuous motion in constricted areas. The vehicle moves from side to side in a dance-like motion as it squeezes through the congested spot. The robot still attains its navigational goal and does not crash into obstacles, but it takes more time than would otherwise be necessary. This problem will vanish when the processing and communication speeds are improved. On the other hand, if the velocity of the robot is increased over the 0.3–0.4 ft/s used for these examples, the problem would be exacerbated. To attain higher velocities, faster processing is necessitated.

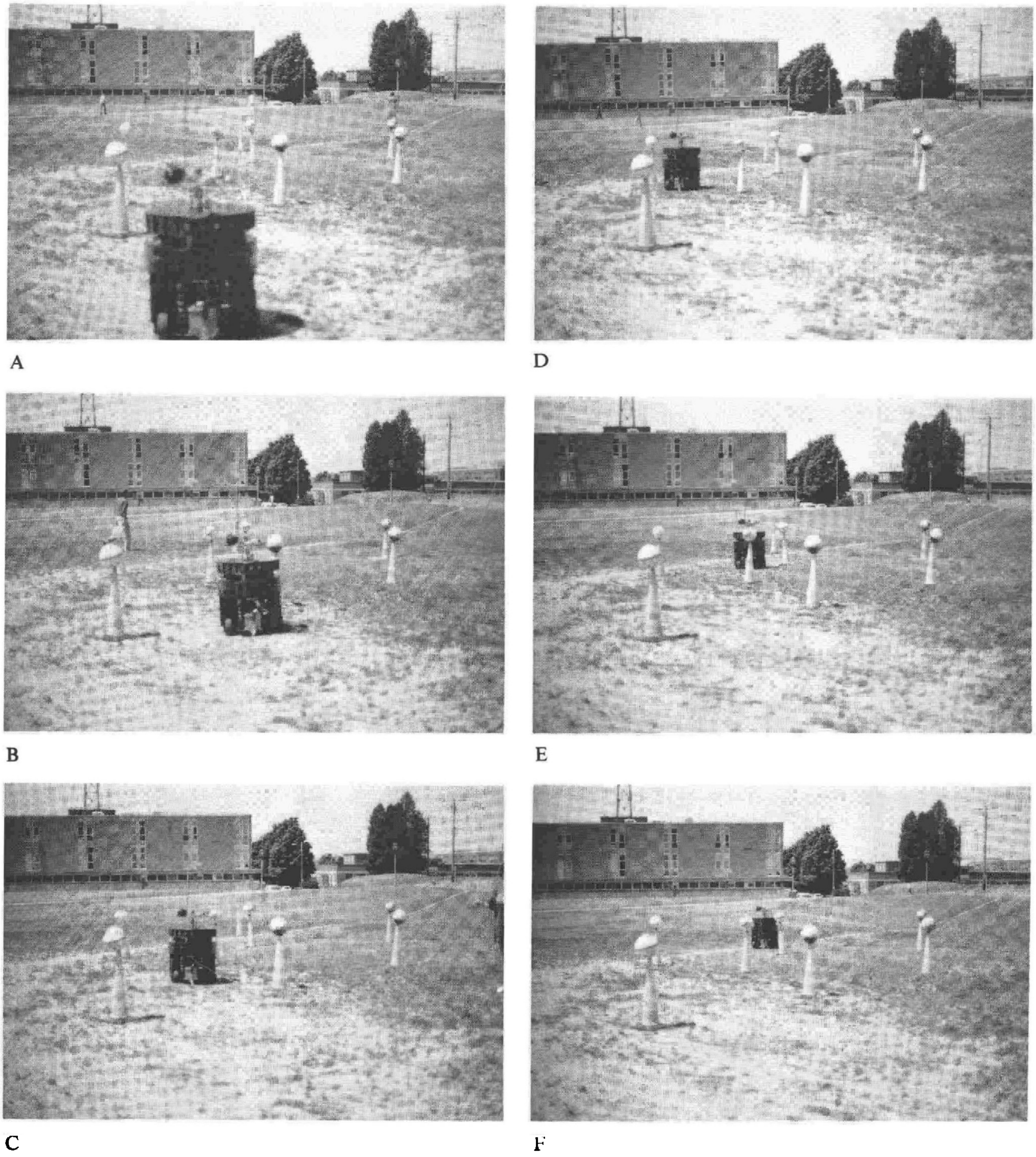
The experimental schema system offers three types of motion: step-by-step mode, where each motor step must be approved by the operator (used chiefly for debugging new schemas); lurch mode, where the robot waits approximately 2 s per step while sensor processing is completed between moves; and continuous motion, where the robot acquires sensor data while moving. The problem with continuous motion is that the vehicle changes its position in the 2 or 3 s it takes to process the sensor data. Generally this is not a problem, but it can give rise, especially in tight quarters, to the oscillatory situations described above. All of the behaviors shown in this section work well in continuous motion, with the possible exception of door entry (described below), again as a result of tight quarters.

5.5. Single Wall Following

A "drunken sailor" walk can be produced by directing a **move-ahead** schema into a wall with the obstacle avoidance behavior active (Fig. 8). The robot slides along a repulsive field a specified distance from the wall, reacting to obstacles as it moves. This allows the robot to enter doorways if the vector pointing into the wall is sufficiently large in magnitude and its angle is sufficiently steep.

It is difficult to get the robot to enter normal doorways when it is operating in continuous mode. As stated above, this is a consequence of the data being old relative to the robot's current position. It also requires some finesse in proper selection of gains and angle of attack to produce smooth door entry behavior.

Fig. 7. Navigation (outdoor). A **move-to-goal** *SI (shaft-encoder-based) was used instead of a* **move-ahead** *SI, the goal being a location near the distant cone that does not have a ball on it. The robot's course winds through the obstacles while making its way toward the goal.*



A

D

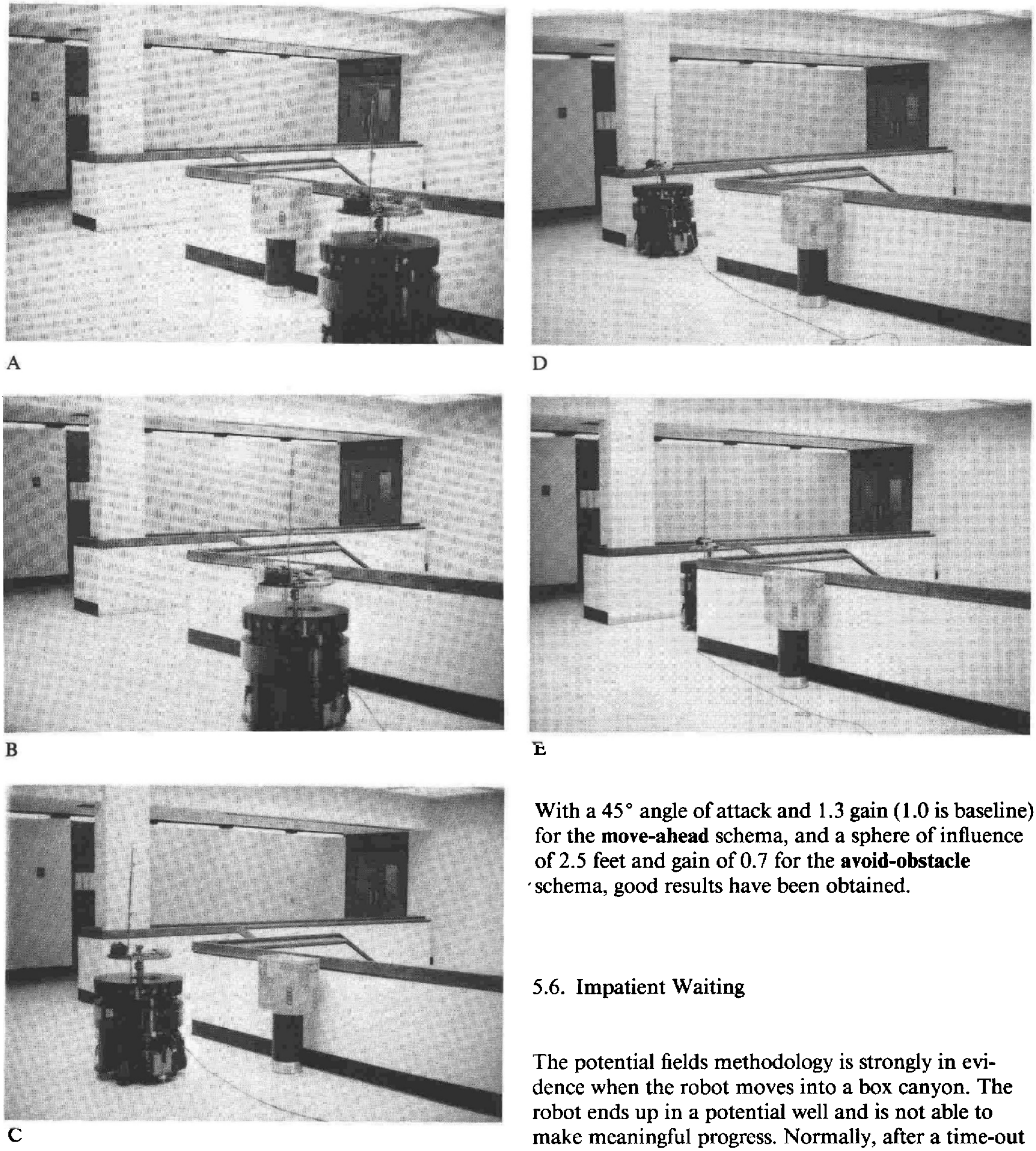
B

E

C

F

Fig. 8. Wall following ("drunken sailor") behavior produced by the combination of a **move-ahead** *SI directed obliquely into the wall while keeping the obstacle avoidance behavior active. In the sequence shown, the robot follows the single wall, moving around the obstacle, and then staggers toward the staircase.*



With a 45° angle of attack and 1.3 gain (1.0 is baseline) for the **move-ahead** schema, and a sphere of influence of 2.5 feet and gain of 0.7 for the **avoid-obstacle** schema, good results have been obtained.

5.6. Impatient Waiting

The potential fields methodology is strongly in evidence when the robot moves into a box canyon. The robot ends up in a potential well and is not able to make meaningful progress. Normally, after a time-out occurs based on a hard real-time deadline, the pilot and/or navigator would be reinvoked to compute an alternate path. Suppose, however, the blockage is temporary, as a result of closed doors at the end of a hall or an elevator. The robot can wait, rocking around in its potential well, until the obstruction is removed. This behavior is similar to what might be evidenced by a fly at a window. When the path becomes unblocked, the robot continues ahead, trying to satisfy its initial goals. Fig. 9 depicts this process.

5.7. Follow the Leader

HARV can track a moving object (with additional obstacle avoidance behavior to aid maneuvering in tight situations if desired). The appearance is similar to walking the robot on an invisible leash. It turns and tracks the nearest object within a given ultrasonic sensor spread, moving more rapidly as the distance between robot and object increases, but avoiding contact with the tracked object. The potential well (the ideal distance separating the tracked object and the robot) is user-specifiable, as is the separation distance at which the robot abandons following. Vision algorithms, well suited for this particular type of tracking, will be exploited for this purpose when suitable real-time image processing hardware arrives.

6. Summary and Conclusions

Motor schemas serve as a means for reactive/reflexive navigation of a mobile robot. This schema-based methodology affords many advantages, including the use of distributed processing, which facilitates real-time performance, and the modular construction of schemas for ease in the development, testing, and debugging of new behavioral and navigational patterns. Complex behavioral patterns can be emulated by the concurrent execution of individual primitive SIs.

The use of velocity fields to reflect the uncertainty associated with a perceptual process is another important advance. By allowing the force produced by a perceived environmental object to vary in relationship to the certainty of the object's identity (whether it be an obstacle, goal path, or whatever), dynamic replanning is trivialized. Since the *sensed* environment produces the forces influencing the trajectory of the robot, when the perception of the environment changes, so do the forces acting on the robot, and consequently so does the robot's path. This is all accomplished at a level beneath the a priori knowledge representations.

It is interesting to note that what might appear to be a naive approach, the summing of the individual vector outputs of the SIs, works quite well, both in simulations and the experimental results described in sections 4 and 5. Certainly as the velocity increases, so does the need to account for the velocity of the robot itself in the generation of its trajectory. More complex formulations have been forwarded by both Khatib (1985) and Krogh (1984) for obstacle avoidance using potential fields. These and other approaches for both potential field formulation and combination mechanisms surely merit additional investigation. Work is currently underway in extending the two-dimensional schema system to three dimensions (Arkin 1987a), ultimately providing navigational capabilities in both the aerospace and undersea domains.

There are times when this methodology of low-level reactive planning will fail, as it suffers from the pitfalls common to potential fields. Failure is detected when the robot's velocity drops to unacceptably low levels (in the case of potential field minima) or by exceeding a hard real-time deadline (in the case of cyclic behavior). At those times, the pilot is reinvoked to conduct a "local-global" form of planning. The pilot draws on information present in short-term memory, including instantiated meadows that are relevant to this particular leg and a sensor-based world model built by the cartographer. This form of replanning should be needed only rarely, as navigational planning helps to ensure avoidance of modeled obstacles. Generally only unmodeled obstacles can lead to the breakdown of schema-based navigation. Higher level knowledge must then be invoked to maneuver the robot out of its dilemma. Most of the time, however, schema-based navigation is more than adequate for the task.

Fig. 9. Impatient waiting ("fly at a window"). When the robot enters a potential well, a **move-ahead** *SI directs the robot to the end of the hall, while the* **avoid-static-obstacle** *behavior keeps it from colliding with the door. The robot roves impatiently at the end of the corridor until the door is opened. At that point, it moves through the door and continues on its way.*