MOTOR SCHEMA BASED NAVIGATION FOR A MOBILE ROBOT: An Approach to Programming by Behavior

Ronald C. Arkin

VISIONS / Laboratory for Perceptual Robotics
Computer and Information Science Department, University of Massachusetts,
Graduate Research Center, Amherst, Massachusetts, 01003

Abstract

Motor schemas are proposed as a basic unit of behavior specification for the navigation of a mobile robot. These are multiple concurrent processes which 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. An implementation strategy based on available tools at UMASS is described. Simulation results show 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 programming – using an inflexible, rigid, hard-coded approach – does 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 which a vehicle (or person) takes in moving from start to goal.

A potential solution can be drawn from models that have been developed in the domains of brain theory and robotics. Schemas, a model 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 paper is to provide insights into the design of a motor-schema-based control system for mobile robots. Section 2 will describe the motivations for the use of schema theory in this domain – drawing from work in both brain theory and robotics. Section 3 will discuss the tack being taken for a motor-schema-based control system in the UMASS autonomous robot architecture (AuRA), utilizing a mobile robot equipped with ultrasonic and video sensors; specifically the role of the pilot and the motor schema manager. Section 4 will present the results of simulations using schemas that specify different behaviors and draw on simulated sensor input. A summary and projection of future work will conclude this report.

This research was supported in part by the NSF under grant DCR-8318776 and the U.S. Army under ETL grant DACA76-85-C-008.

2. Motivation

The concept of schemas originated in psychology [1,2,3] and neurology [4,5]. Webster [6] defines a schema as "a mental codification of experience that includes a particular organized way of perceiving cognitively and responding to a complex situation or set of stimuli". The model used for this paper draws on more recent sources: the applications of schema theory to brain modeling and robotics. As brain theory 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, not always in the form envisioned by brain theoreticians. The previous work in robotics that relates to the schema-based approach to navigation will be described in the final part of this section.

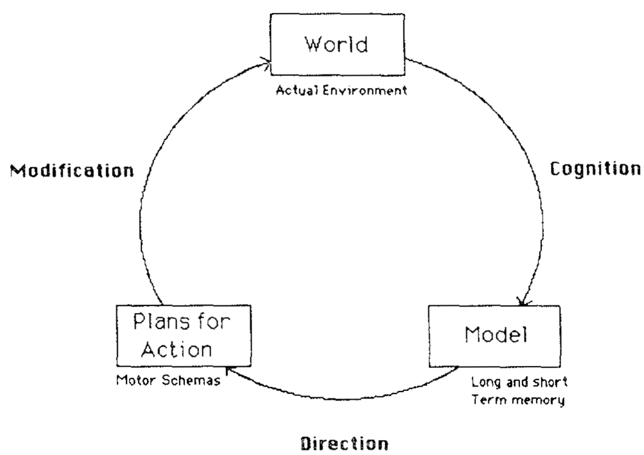


Figure 1. Action – perception cycle

2.1 Brain Theory and Psychology

The action-perception cycle (fig. 1) provides a principal motivation for the application of schema theory [7]. 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 which is pertinent to the task at hand. The question for the roboticist would be: how do we select from the wealth of sensor data available 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 [8,9] 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 [10] have developed a model for worm acquisition by the frog in an obstacle-cluttered environment (a spaced fence - fig. 2). 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 (fig. 3): a prey-attractant field, a barrier-repellent field, and a field for the animal itself. These fields, when combined, yield a model of behavior (fig. 4) that is consistent with experimental observations of the frog.

In the mobile robot system described below, analogs of these fields will be used (prey-attractant ⇒ **move-to-goal**, barrier-repellent ⇒ **avoid-static-obstacle**). Additionally, new fields will be added to describe additional 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 will be 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.

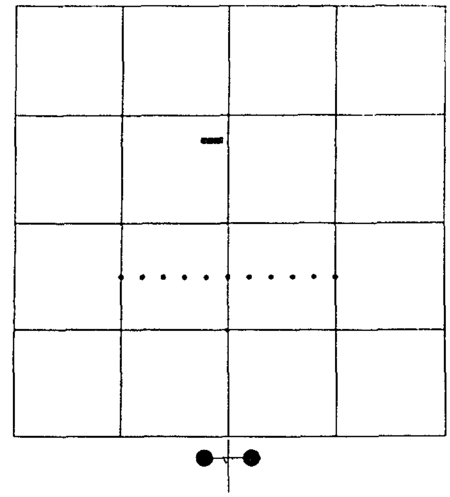


Figure 2. A depiction of a frog prey-selection scenario. The two large blackened circles at the bottom of the figure denote the frog's eyes, the smaller circles are fence-posts, and the darkened rectangle a supply of worms. (fig. 2,3,4 reprinted from [10] with permission).

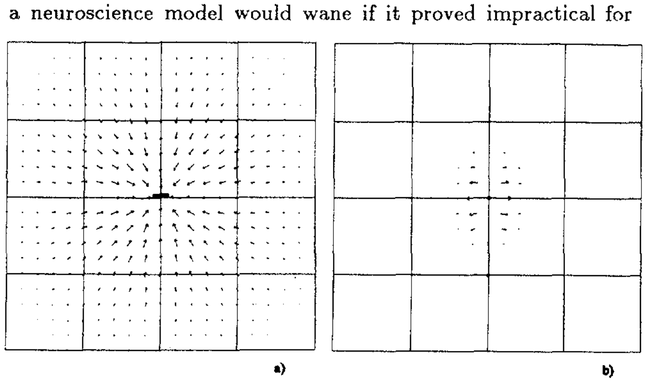


Figure 3. Primitive vector fields associated with figure 2.
a) Prey-attractant field
b) Barrier repellent field

2.2 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 [14] and Geschke's servo processes [12], (a schema analog), are used as approaches to task level control. Overton [15] has described the use of motor schemas in the assembly domain. The UMASS VISIONS group, guided by Hanson and Riseman, has applied perceptual schemas to the interpretation of natural scenes; Weymouth's thesis is the prime example of this work [13]. Although AuRA will, in the future, include perceptual schemas running in the context of the VISIONS system, perceptual schemas as they appear in the VISIONS system are not a principal concern of this paper.

One of the simplest and most straightforward definitions for a schema is "a generic specification of a computing agent" [14]. This definition fits well with the concept of a behavior (an individual's response to its environment) – each schema represents a generic behavior. Schema-based control systems are signifi-

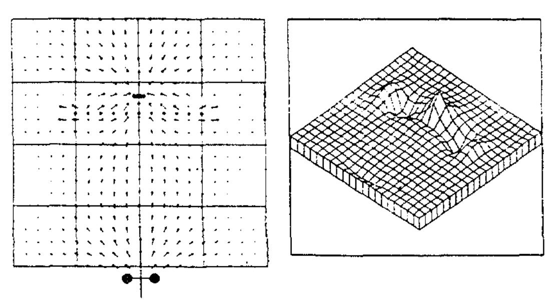


Figure 4. Resultant frog-prey selection field.

cantly more than a collection of frames or templates for behavior, however. The way in which they are set into action and interact immediately distinguish them from simpler representational forms. The instantiations of these generic schemas (SI – schema instantiation) provide the potential actions for the control of the robot. A schema instantiation is created when a copy of a generic schema is parameterized and activated as a computing agent.

Lyons further defines a motor schema as a control system or motor program which describes a task. Overton [15] describes a motor schema as "a control system which continually monitors feedback from the system it controls to determine the appropriate pattern of action for achieving the motor schema's goals, (these will, in general, be subgoals within some higher-level coordinated control program)". This more constrained definition is also in accord with the system described below. Sensory perception provides the feedback to affect individual instantiations of motor schemas, each SI thus providing an appropriate behavior which collectively determine the overall system's behavior. Some other definitions for motor schema include an "interaction plan" [25] or "unit of motor behavior" [16].

Other work in the path planning domain, although not schema based, bears a resemblance to the schema control system. Brooks [17] uses a planning system with a "horizontal decomposition" which effectively emulates multiple behaviors. Although related, there is still a rigid layering present which distinguishes it from a schema-based approach. Payton [23] 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. An arbitrary choice of behavior, however, based on a priority system, is made during execution, without provision for a mechanism to combine the results of concurrent behaviors. Kadonoff et al [18] 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 [19] generalized potential fields approach and to a lesser degree by Lyons' [11] tagged potential fields. It bears a superficial resemblance to the integrated path planning and dynamic steering control system described by Krogh and Thorpe [20]. 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 the 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.

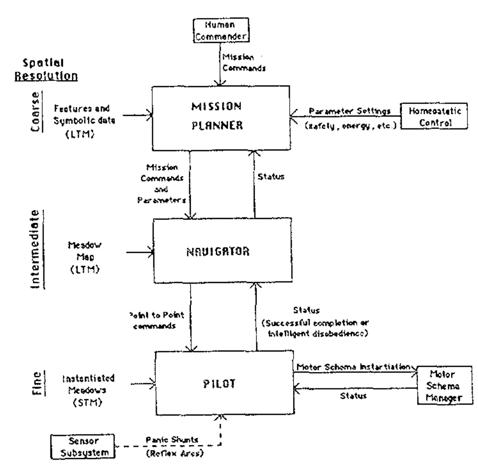


Figure 5. Hierarchical planner for AuRA.

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.

This section will first describe the overall UMASS autonomous robot architecture's planning subsystem; particularly the roles of the pilot and motor schema manager. Implementation strategies will then be described.

3.1 Path Planning and Navigation System

The AuRA high-level path planner (fig. 5) is hierarchical in design; consisting of a mission planner, navigator and pilot. The mission planner is delegated the responsibility for interpreting high level commands, determining the nature of the mission, setting criteria for mission, navigator and pilot failure, and setting appropriate navigator and pilot parameters. The mission planner, although part of the overall design, is not yet fully implemented, and has a relatively low priority.

The navigator accepts a start and goal point from the mission planner and using a "meadow map", a hybrid vertex-graph free-space representation, determines a path to achieve that goal. The navigator produces a piecewise linear path that avoids all modeled obstacles present in the *a priori* map constructed by the map-builder component of the cartographer. See [21] for a description of the navigator and the representations it uses.

The pilot is charged with implementing leg-by-leg this piecewise linear path. 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, the 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 due to 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 etc.)
- **Avoid-moving-obstacles** (people 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, 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 how certain we are that it exists. A more concrete example follows.

The robot is moving across a field in a particular direction (**move-ahead** schema). The **find-obstacle** schema is constantly on the look-out 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.

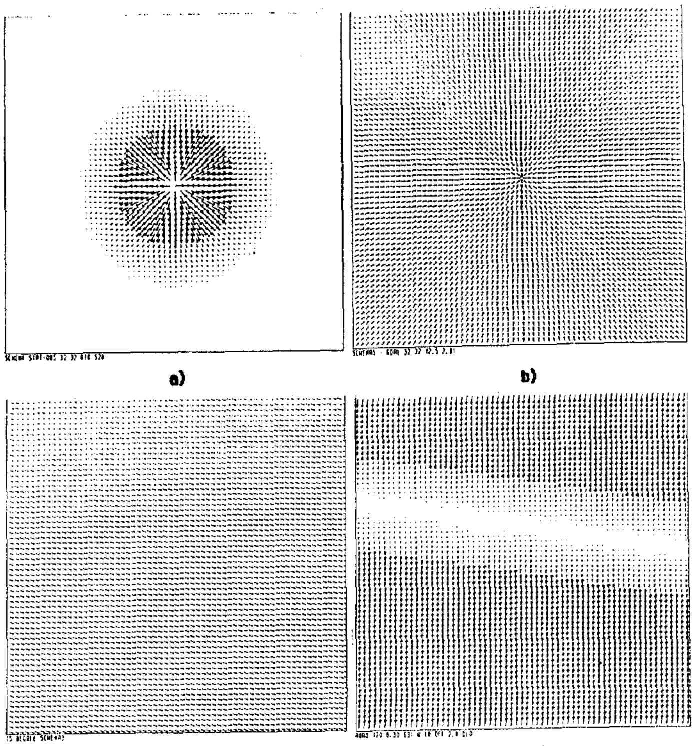


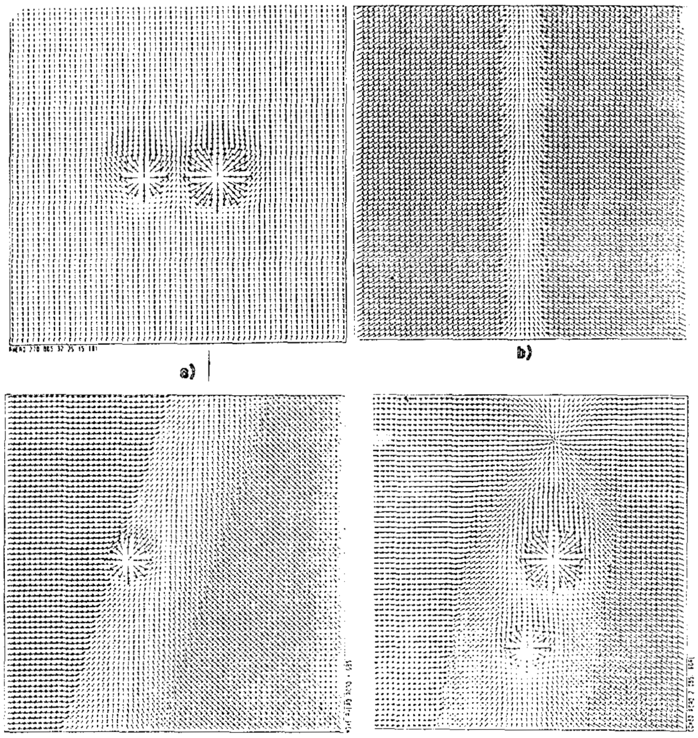
Figure 6. Isolated motor schema SI vector fields.
a) **Avoid-static-obstacle** b) **Move-to-goal**
c) **Move-ahead** d) **Stay-on-path**

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 range for perception 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 moving away from the object. Fig. 6a 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 fig. 6b-d. Various combinations of motor schemas are illustrated in fig. 7.

If each schema functions independently of each other, 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 vector fields. 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 field, 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 the 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 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 [14] 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 Schema Shell system (discussed below). In either case, communication mechanisms can solve problems that might otherwise prove intractable. An example to illustrate this point follows.

[1] The obstacle is first grown in a configuration space manner [27] to enable the robot to be treated henceforth as a point for path planning purposes.



a) b) c) d)

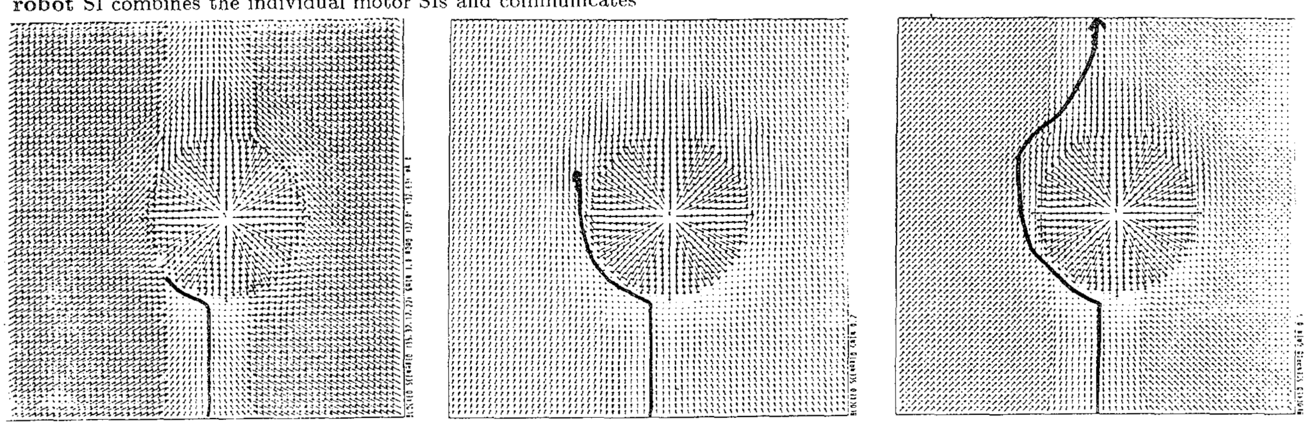
Figure 7. Several combined motor schemas.
a) **Move-ahead** SI + 2 **Avoid-static-obstacle** SIs.
b) **Move-ahead** SI + **Stay-on-path** SI.
c) **Move-ahead** SI + **Stay-on-path** SI + 1 **Avoid-static-obstacle** SI.
d) **Move-to-goal** SI + **Stay-on-path** SI + 2 **Avoid-static-obstacle** SIs.

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 stop (fig. 8a). The stoppage of the robot 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 control system). The **stay-on-path** SI, when created, was instructed to yield if an obstacle blocks the path. The **stay-on-path** motor schema reduces its field (fig. 8b) 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. 8c).

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.

Another approach to be explored is the addition of a background stochastic **noise** schema. This SI would produce a low-magnitude random direction velocity vector that would change at random time intervals, but persist sufficiently long to produce a change in the robot's position if the robot's velocity was otherwise zero. The behavior produced by this schema would correspond to the "wander" layer in Brook's horizontally layered architecture [17]. 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. 9). Other traps common to potential field approaches (e.g. box canyons) can be handled by establishing hard time deadlines for goal attainment. If these deadlines are violated, the pilot would be reinvoked 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, only concerned with avoiding a collision with it. This "divide and conquer" approach based on action-oriented perception simplifies programming and overall system design. A more complex scenario appropriate for a mobile robot appears in fig. 10.



a) b) c)

Figure 8. 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.

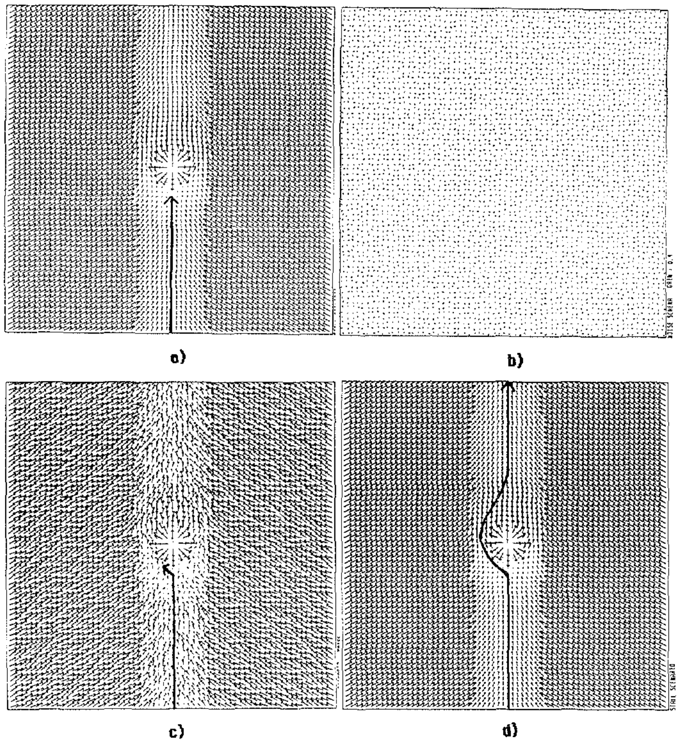


Figure 9. Stall scenario.

a) If the robot approachs an obstacle exactly head-on, it is possible for it to become 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) The robot can now succssfully bypass the obstacle. The **noise** SI is then deinstantiated.

4. Implementation Strategy

The implementation tool chosen for the motor schema system is the Schema Shell [22]: a system developed by the VISIONS group at UMASS for use in the perceptual schema analysis of natural scenes. It currently runs on a Texas Instruments Explorer workstation and is tied to the Computer Science Department's VAXen over Chaosnet. The schema communication mechanism is blackboard-based. The Schema Shell system is expected to be moved to the department's newly acquired Sequent parallel processor. Although the Explorer only simulates distributed processing, everything points towards the availability of a truly distributed environment in the not too distant future.

The schemas themselves (in the Schema Shell) consist of a *schema template* and *multiple strategies*. Associated with each instantiated schema is an object hypothesis maintenance (OHM) strategy. This part of the SI monitors the blackboard for new events (e.g. sensory data) that would produce changes in the SI's posted output. Other strategy components for each SI handle conflict resolution, cooperative enhancement, initialization and other relevant factors. Not all strategies are necessary or desirable for all schemas.

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. The parameters set at instantiation may depend on the sensory events that triggered the instantiation. Fig. 11 shows a typical generic motor schema cast in the Schema Shell format.

Pilot issues instructions to follow sidewalk while avoiding obstacles. Continue approximately 200 ft on sidewalk then turn right at lamppost onto intersection (first encountered). Watch for landmarks (mailbox on left, building edge on right) for localization

Motor Schemas instantiated by pilot:

→ **Complete_pilot_maneuver**(schema_list,schema_base_priority_vector) creates:

- Stay_on(identify_terrain(ahead,60%)) (assumes sidewalk is ahead to start)
- Move_ahead(200,10,start_heading) (nominal distance, then add some slop)
- Avoid_static_obstacles(15,identify_obstacle(robot_heading,nil,70%))
  Start maneuvering around when within 10 feet
  nil denotes static obstacle
- Avoid_dynamic_obstacles(20,identify_obstacle(robot_heading,~ robot_heading,40%))
  Start evasive action when head on approach within 20 feet
- Follow_dynamic_obstacle(8,start_heading,identify_obstacle(True,start_heading95%))
  when an obstacle is moving in the right direction within 8 feet of the robot , follow it (regardless of robot's current heading)
- Avoid_dynamic_obstacles(5,identify_obstacle(robot_heading,True,40%))
  Start evasive action wher. within 5 feet for any dynamic obstacle
  (includes crossing dynamic obstacles)
- Turn_when(recogniz :_landmark(lamppost_1+5ft,90,90%)) - right 90 degrees
- Turn_when(recogniz:_landmark(intersection_3a,90,90%)) - right 90 degrees
- Localize(recognize_landmark(mailbox_7,90%))
- Localize(recognize_landmark(building_2a.edge3, 85%))
  prune spatial error map on landmark recognition
- Stop_when(not (sidewalk_1 = identify_terrain(ahead,90%))) (missed turn)

Perceptual Schemas instantiated by pilot:

- Identify_obstacle(robot_heading,obstacle_heading,certainty)
  Only detects obstacles in the way of the robot (distinct from landmarks)
  robot_heading and obstacle_heading are filters
  certainty is threshold for identification
  returns obstacle position and type
  1 identify_obstacle spawned for each strategy type above
  * obstacle - generic - many spawned for each identify_obstacle
  returns certainty
  tracks motion over time
  types:
  static_obstacle - predicate
  dynamic_obstacle - predicate
  obstacle_heading (nil if static)
  speed (0 if static)
- Recognize_landmark(LTM_model,certainty) - 1 spawned per landmark
  Assumes robot's current position for observation is available in global coordinates (spatial error map)
  certainty is threshold for recognition
  returns landmark location
  (not necessarily in direct track of robot, could be anywhere)
  * landmark(LTM_model) - many/landmark spawned off
  returns certainty
- Identify_terrain(position,certainty)
  returns terrain type

At end of maneuver, deinstantiate all obstacle schemas.

Figure 10. Example mobile robot schema scenario.

```
;----------------------------------------------------------------
;                      MOVE-AHEAD Motor Schema
;        ->
;        ->        x-axis is 0; frame of reference is robot's initial heading afterturn;
;        ->
;        ->

(make-motor-schema
       :name "MOVE-AHEAD"
       :default-argument-list (list '("heading" 0))
       :body
       (progn
             (de-schema  move-ahead (original-heading current-heading move-ahead-force-table))

             (de-strategy move-ahead ohm ()
               (progn
                 (call-strategy 'move-ahead 'init)
        (loop
                 (setf #!current-heading
                     (read-or-wait '(current-orientation-messagep robot-position)))

                  (write-to-blackboard
                           (list "move-ahead" (look-up-move-ahead-force #!current-heading) (time-stamp))
                         'vector-section)  ; write resultant force to vector section
                  )
        ))

             strategies for move-ahead

             (de-strategy move-ahead init()
               (build-move-ahead-force-table)                        ; initialize lookup table
               (setf #!original-heading (read-or-wait '(orientation-messagep robot-position)))
               (write-to-window "move-ahead init done")
             )

       ; conflict in case of reverse direction- send message-to-mover-to-terminate
       ; contains list of contradictory motions or evidence

       ; support in case of confirmed direction
       ; support contains list of related schemas use to confirm hypothesis
       ; e.g. to stop-when's
            )  ); End make-motor-schema

;----------------------------------------------------------------
```

Figure 11.
Prototype **move-ahead** schema as implemented in the Schema Shell.

5. Simulation

Simulations were run on a VAX 750 using the following motor schemas: **stay-on-path**, **move-ahead**, **move-to-goal**, **avoid-static-obstacle**. Each simulation run (fig. 12–13) shows the sequence of resultant overall force fields based on perceived entities. These entities include path borders and obstacles. The grid size is 64 units by 64 units and sensory sampling update time (once per second) is based on a nominal velocity of 1 unit/second. 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 (a discrete approximation). All obstacles are currently modeled as circles (as in Moravec's tangent space [24]).

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 below:

$$O_{magnitude} = \begin{array}{l} 0 \ \ for \ \ d > S \\ \frac{S-d}{S-R} \ for \ \ R < d \leq S \\ \infty \ \ for \ \ d \leq R \end{array}$$

where:

S = Sphere of Influence (radial extent of force from the center of the obstacle)
R = Radius of obstacle
d = Distance of robot to center of obstacle
$O_{direction}$ = along a line from robot to center of obstacle moving away from obstacle

More complex equations could be used (e.g. cubic as in [20]) but were deemed unnecessary in these early stages of the research.

Figure 12 illustrates the robot's course on a sidewalk moving towards a goal. The course is studded with 8 obstacles, only 7 of which are perceptible to the robot during its journey (fig. 12a). Note how the vector fields change as the robot encounters more obstacles along the way (fig. 12b-d). When it 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 (fig. 12d). The robot stays on the path for the complete course successfully achieving its goal while avoiding each obstacle. An expanded version could update long-term memory as a result of experience, thus incorporating learning.

Figure 13 shows the robot's path to a specified goal through a field of 9 obstacles. This simulation prevents perceived objects that have too great an uncertainty from producing a repulsive field. In this case, the uncertainty increases with the distance from the obstacle. The simulation in figure 13 uses a **move-to-goal** SI. Actually the robot would operate under the control of a **move-ahead** SI until the goal is perceived (assuming dead-reckoning or inertial guidance is not used). At the moment of goal perception, the **move-ahead** SI would be deinstantiated and a **move-to-goal** SI created in its stead.

6. Summary and Future Work

Motor schemas are proposed as a means for navigation of a mobile robot. This schema-based methodology affords many advantages. These include 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.

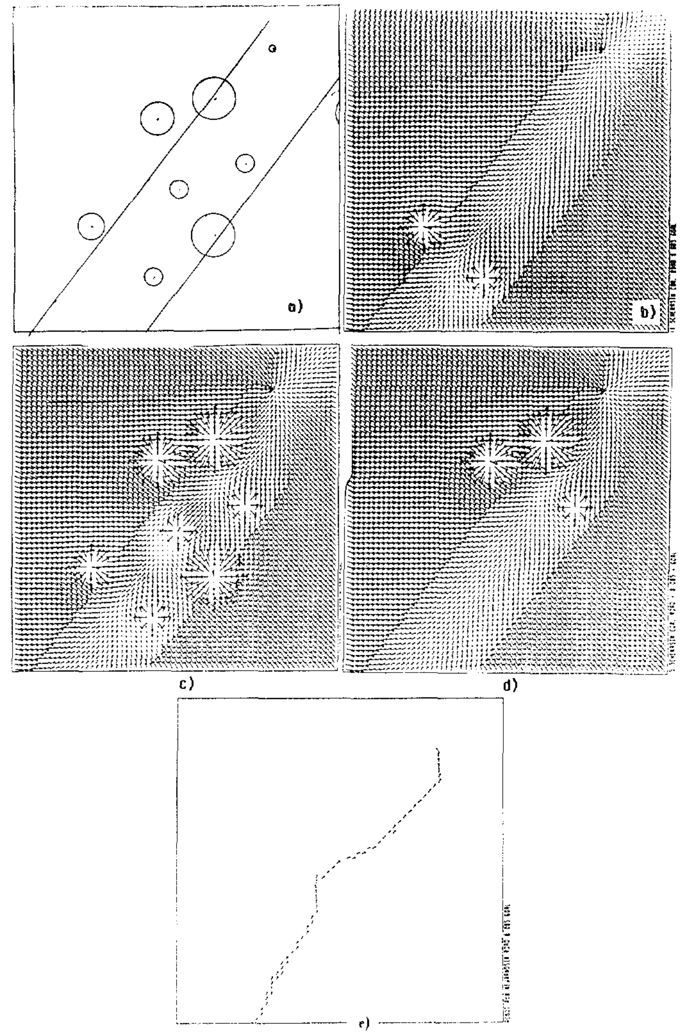


Figure 12. Simulation run.

This simulation shows 7 **avoid-static-obstacle** SI and a **stay-on-road** SI.
a) Shows the layout of the obstacle ridden course.
b-d) With the robot starting at the lower left, the robot's progress through the course can be observed. Note that the obstacles are added as they are perceived by the sensory system. No *a priori* knowledge of their whereabouts is assumed.
e) The robot's path through the course.

The next logical step is to complete the implementation of the system on the Explorer within the framework of the Schema Shell and to interface it with the high-level planning component of AuRA.

Work is underway for the acquisition of road edges using a new fast line-finding algorithm that can serve as the perceptual schema for the **stay-on-path** motor schema. Obstacle location using a multiple frame depth-from-motion algorithm [26,28] is being explored as a perceptual schema for the associated **avoid-static-obstacle** SI. Additionally, the use of ultrasonic data as input for the **avoid-obstacle** SIs is anticipated.

Motor schemas for following a moving object (tracking) and avoiding moving obstacles are being developed. These will enable the vehicle to emulate both follow-the-leader and dodging behaviors.

Long term goals include tying in the VISIONS system as the means for providing sensor-independent object-based input to the motor schemas.

Acknowledgments

The author would like to thank Prof. Ed Riseman, Prof. Michael Arbib, Dr. Damian Lyons and the members of the Laboratory for Perceptual Robotics and VISIONS groups for their assistance in the preparation of this paper.

References

1. Bartlett, F.C., *Remembering: A Study in Experimental and Social Psychology*, London, Cambridge at the Univ. Press, 1932.
2. Oldfield, R.C. and O.L. Zangwill, "Head's Concept of the Schema and its Application in Contemporary British Psychology - Parts I and II", *Br.J. Psychol.* 32:267-286 1942, 33: 58-64,113-129,143-149, 1943.
3. Piaget, J., *Biology and Knowledge*, Univ. of Chicago Press, 1971.
4. Head, H. and Holmes, G., "Sensory disturbances from cerebral lesions", *Brain* 34:102-254, 1911.
5. Frederies, J.A.M., "Disorders of the Body Schema", *Handbook of Clinical Neurology, Disorders of Speech Perception and Symbolic behavior*, Ed. Vinken and Bruyn, North Holland, 1969, Vol. 4, pp. 207-240.
6. Webster's *Ninth New Collegiate Dictionary*, Merriam-Webster, 1984.
7. Neisser, U., *Cognition and Reality: Principles and Implications of Cognitive Psychology*, Freeman, 1976.
8. Arbib, M., "Perceptual Structures and Distributed Motor Control", *Handbook of Physiology - The Nervous System II*, ed. Brooks, pp. 1449-1465, 1981.
9. Arbib, M., *The Metaphorical Brain*, Wiley, 1972.
10. Arbib, M. and House, D., "Depth and Detours: An Essay on Visually Guided Behavior", *COINS TR* 85-20, Univ. of Mass, 1985.
11. Lyons D., "Tagged Potential Fields: An Approach to Specification of Complex Manipulator Configurations", *IEEE Conf. on Robotics and Auto.*, pp. 1749-1754, 1986.
12. Geschke, C., "A System for Programming and Controlling Sensor-based Robot Manipulators", *IEEE Trans. on PAMI*, PAMI-5, No. 1, 1983, pp. 1-7.
13. Weymouth, T., "Using Object Descriptions in a Schema Network for Machine Vision", *Ph.D. Dissertation, COINS Tech. Rep.* 86-24, Univ. of Massachusetts, Amherst, May 1986.
14. Lyons, D., "RS: A Formal Model of Distributed Computation for Sensory-Based Robot Control", *Ph.D. Dissertation, COINS Tech. Rep.* 86-43, , Univ. of Massachusetts, Amherst, 1986.
15. Overton, K., "The Acquisition, Processing, and Use of Tactile Sensor Data in Robot Control", *Ph.D. Dissertation, COINS Tech. Rep.* 84-08, Chapter 6, Univ. of Massachusetts, Amherst, 1984.
16. Lyons, D. and Arbib, M., "A Task-Level Model of Distributed Computation for Sensory-Based Control of Complex Robot Systems", *COINS Tech. Rep.* 85-30 (also *IFAC Symposium on Robot Control*, Barcelona), 1985.
17. Brooks, R., "A Robust Layered Control System for a Mobile Robot", *IEEE Jour. of Robotics and Auto.*, Vol. RA-2, No. 1, 1986 pp. 14-23.
18. Kadonoff, M., Benayad-Cherif, F., Franklin, A., Maddox, J., Muller, L., Sert, B. and Moravec, H., "Arbitration of Multiple Control Strategies for Mobile Robots", to appear in MOBILE ROBOTS - *SPIE proc. Vol. 727*, 1986.
19. Krogh, B., "A Generalized Potential Field Approach to Obstacle Avoidance Control", *SME - RI Technical Paper* MS84-484, 1984.
20. Krogh, B. and Thorpe, C., "Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles", *IEEE Conf. on Robotics and Auto.*, pp. 1664-1669, 1986.
21. Arkin, R., "Path Planning for a Vision-based Mobile Robot", to appear in MOBILE ROBOTS - *SPIE Proc. Vol. 727*, 1986.
22. Draper, B., "Introduction to the Schema Shell", and "Programmer's Reference for using the Schema Shell", *VISIONS working papers*, Univ. of Massachusetts, Amherst, 1986.
23. Payton, D., "An Architecture for Reflexive Autonomous Vehicle Control", *IEEE Conf. on Robotics and Auto.*, pp. 1838-1845, 1986.
24. Moravec, H., *Robot Rover Visual Navigation*, UMI Press, 1981.

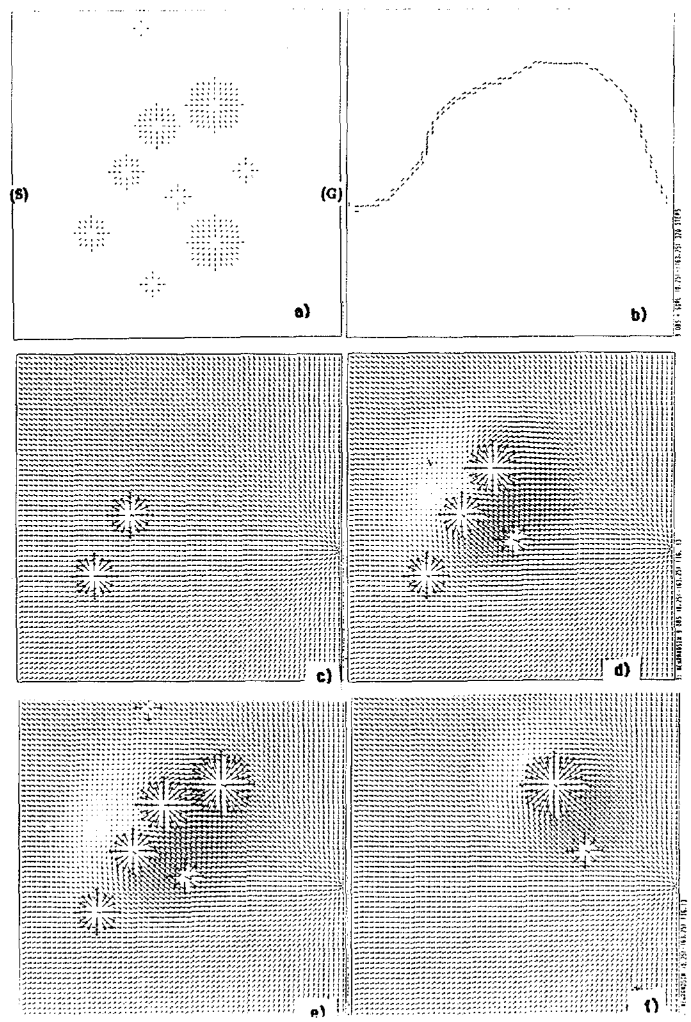


Figure 13. Simulation run.

This simulation include 9 avoid-static-obstacle SIs and 1 **move-to-goal** SI.

a) Location of the 9 obstacles.

b) Path of robot as it crosses from left to right around obstacles to the goal.

c-f) Velocity fields based on robot's perceptions as it moves from left to right as in b).

This simulation includes an uncertainty measure for obstacles which increases with the distance of the obstacle from the robot. If the obstacle is relatively uncertain, its position is shown but it produces no field (e.g. the two rightmost obstacles in fig. c). As the robot approaches, it becomes more certain of the obstacle and starts to produce a repulsive field surrounding the obstacle.

25. Arbib, M., Iberall, T. and Lyons, D., "Coordinated Control Programs for Movements of the Hand", *COINS TR* 83-25, Univ. of Mass., 1983 (also *Exp. Brain Res. Suppl.*, 10, pp. 111-129, 1985.
26. Bharwani, S., Riseman, E. and Hanson, A., "Refinement of Environmental Depth Maps over Multiple Frames", *Proc. IEEE Workshop on Motion Representation and Analysis*, pp. 73-80, May 1986.
27. Lozano-Perez, T., "Automatic Planning of Manipulator Transfer Movements", in *Robot Motion: Planning and Control*, ed. Brady et al, pp. 499-535,1982.
28. Arkin, R., Riseman, E. and Hanson, A., "Visual Strategies for Mobile Robot Navigation", *COINS Tech. Rep.*, in preparation, Univ. of Massachusetts, Amherst, 1987.