In Proceedings of the 1997 IEEE International
Conference on Robotics and Automation,
Albuquerque, NM, April, 1997.

Tropism-Based Cognition for the Interpretation of Context-Dependent Gestures

Richard M. Voyles[†] Arvin Agah* Pradeep K. Khosla[‡] George A. Bekey[※]

[†]Robotics Ph.D. Program
[‡]Dept. of Elect. and Comp. Eng.
Carnegie Mellon University
Pittsburgh, PA

*Bio-Robotics Division
Mechanical Engineering Laboratory
Tsukuba, Japan

[※]Dept. of Computer Science
University of Southern California
Los Angeles, CA

Abstract

The Tropism System Cognitive Architecture provides an intuitive formalism for colonies of agents, either hardware or software. We present a fine-grained implementation of the architecture on a colony of software agents for the interpretation of human tactile gestures for robotic trajectory specification and modification. The fine-grained nature of the architecture and the use of the port-based object framework for agent instantiation allows the manual construction of a capable agent set that is reconfigurable and reusable across different gesture-based interaction tasks.

1 Introduction

Hierarchical cognitive architectures have been the mainstay in robotics since its inception. Shakey [6] is a classic example of a hierarchical, monolithic, sense-think-act architecture from the early 80's. In the past decade, there has been a strong push toward multi-agent systems as opposed to monolithic ones. These tend to incorporate reactive, sense-act behaviors that avoid the latencies of monolithic world modelers, but they still rely on hierarchical structures for goal management. For example, the subsumption architecture [3][4] is a classic behavior-based architecture that relies on the ability of higher-level layers to subsume the functions of lower-level layers.

While hierarchical architectures are appropriate for many types of goal-directed tasks, they are not a good model for many biological systems, such as insect colonies, which have demonstrated themselves to be successful and robust in complex environments (e.g. the natural world). The Tropism System Cognitive Architecture [1] was proposed as a basis for the study of the emergent and collaborative behavior of robot colonies with limited inter-agent communication. Tropisms are the positive and negatives responses of an agent to specific stimuli -- essentially the agent's likes and dislikes [10]. Each agent's behavior is determined by a set of evolving tropisms. A key benefit of this like/dislike representation is that it is easily understood by system designers.

In this paper a modified version of the tropism architecture is applied to the application domain of tactile gesture programming of robotic manipulators. It is shown that a multi-agent system equipped with fine-grained tropism architecture can be utilized to recognize and interpret human operators' gestures as part of the robot programming. Tropism-based cognition not only allows for simple construction of the control agents, but also yields a system that can function well both in reactive and proactive modes.

As information is conveyed to the agents through the operators' delivery of nudges (tactile gestures) to the robot, appropriate action is selected by the agents utilizing tropism cognition. The hand contacts covey the intentions of the robot programmer, much in the same way that the mouse movements, and dragging and dropping covey the intentions of a computer user of a graphical user interface (GUI). By selecting the corners, or the sides of a screen, and clicking and moving the mouse, the user's intentions of changing the shape and size of the screen are communicated to the user interface. The GUI will then make the changes happen according to the state of the system. Analogously, the robot operator, delivers tactile gestures to a rectangular model of a trajectory, intending to change the shape or the size of the rectangle. Tropism-based cognitive architecture is employed to make sense of these gestures in the context of the state of the system.

2 Gesture Interpretation

2.1 The manipulator task

The subject of this paper is the re-implementation of a gesture-based user interface for a robot manipulator that provides an intuitive mechanism for modifying its periodic trajectory [8]. The trajectory shape is not free-form, but one of a small number of polygonal trajectory families. For this implementation, each family of trajectories is constrained to the same vertical plane and the set of trajectory shapes consists of: a cross, a rectangle, a right triangle, and a pick-and-place path (Figure 1). There is also a "null" trajectory

that allows the operator to pause the robot. Note that the rectangle and pick-and-place trajectories were deliberately chosen to be very similar.

All shapes have variable width (w) and height (h) and the cross can also vary in thickness (t). These parameters all vary independently. Nothing else but these parameters is allowed to vary, including the orientation and center point (although appropriate agents could be developed). No provision has been made at this time for non-polygonal trajectories. Tactile gestures, in the form of "nudges" on the end-effector, are the only means of selecting a trajectory and varying its parameters.

2.2 What are gestures?

Tactile gestures and motion gestures, as defined by Voyles and Khosla, are imprecise events, involving hand contact or motion, that convey the intentions of the gesturer. (See [9] for a more complete description and other relevant gestural modes.) Because these gestures can be imprecise events (unlike sign language, for example), they are context dependent; in isolation they may convey little information. Yet, when combined with the state of the system and the environment at the time the gesture occurred, perhaps augmented with some past history of gestures, the intention becomes interpretable.

In effect, gestures form an "alphabet" from which "words" are formed when combined with state information. These gestural words are strung together by the gesturer into "sentences" which form the basis for interpretation. (See [9] for an example.) The key point is the interpretation of non-symbolic gestures develops over a series of gestures rather than instantaneously.

2.3 Gesture recognition and interpretation

Interpreting gestures is a two-step process. First, gestures must be recognized and extracted from the raw sensory data. Next, the sequence of gestures must be examined to build a coherent interpretation. Because these operations are distinct, it is easy to decompose them into separate agents for recognition and interpretation. The gesture recognition agents are virtual sensors that report gestural words (and may be noisy).

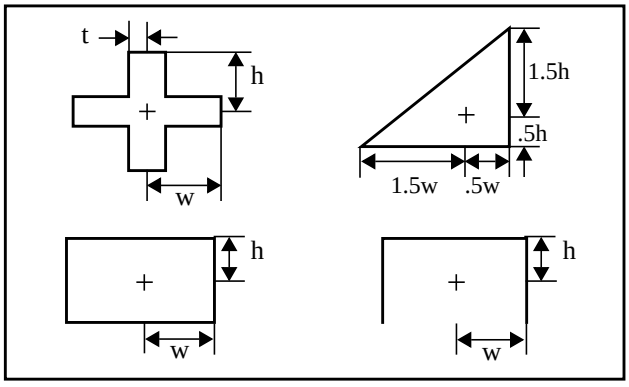

Figure 1: The shapes of the trajectory families.

3 Tropism System Cognitive Architecture

The Tropism-based cognitive architecture system proposed by Agah and Bekey [1] is depicted diagrammatically in Figure 2. The architecture is based on

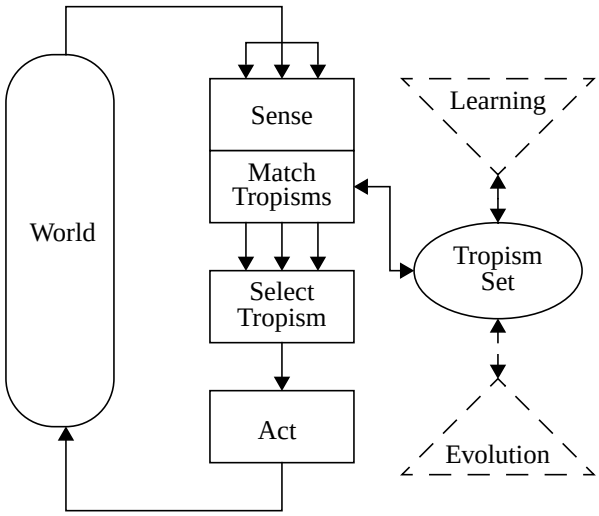

Figure 2: Standard Tropism System Cognitive Architecture

the idea that the behavior of an agent is dependent on the likes and dislikes of the agent. Biological systems tend to do things they like and avoid those which they dislike. When an agent encounters a situation, the likelihood of a response toward the more likable entity is higher. The tropism-based cognition enables an agent to behave in such a manner. It should be noted that the agent does not necessarily take the action with the highest tropism, but it is more likely to take an action that has a higher tropism. The agent is embedded in a world that is populated by entities, ε_i, each in state σ_i, which can be sensed. Based on this sensed information, an agent has preference τ_j to take action α_j. Thus, a *tropism elemen*t can be represented as a 4-tuple:

$$(\varepsilon, \sigma, \alpha, \tau)$$

The behavior of an agent is thus determined by a set of such tropism elements, Ξ. As this set can change dynamically, it is a function of time:

$$\Xi(t) = \{(\varepsilon, \sigma, \alpha, \tau)\}$$

After sensing the surrounding world, each agent matches sensed entity/state pairs with tropism elements in its tropism set. From this subset of matched tropism elements the agent probabilistically selects actions based on its own preferences. (See[1] for more detail.)

Previous applications of this architecture have focused primarily on colonies of individual robotic agents, each equipped with the tropism architecture (e.g. [1][2]). These colonies consisted of colonies (teams) of robotic agents with discrete sets of states and actions, embodied in worlds with a known set of entities. In this paper, we are interested in a heterogeneous colony of software agents with a high degree of agent speciality. In fact, each agent will be responsible for only one "action" in the user's world but the agent's world will consist of continuous actions and states (although we discretize the states using fuzzy sets).

In this paper we will only examine static tropism systems and will not address the issues of dynamic learning of individuals and group evolution (ontogenetic and phylogenetic learning), which are explained in [2], for this application.

4 Fine-Grained Tropism Architecture

As set forth in previous sections, we have implemented collections of finely-decomposed software agents capable of interpreting sequences of human gestures for the control of a robot manipulator. An underlying motivator for the fine-grained, multi-agent architecture is our interest in the development of rapidly-deployable systems. Our approach to rapidly-deployable systems engineering involves primitive software and hardware agents that are modular, reconfigurable, simple, and reusable [5]. In this light, we restrict each agent to a single interpretation or hypothesis, resulting in a fine-grained agent decomposition, based on the tropism cognition architecture.

From the world's (user's) perspective, each agent can take only one action corresponding to the hypothesis it is trying to prove or disprove. From the agent's perspective, however, it has a number of actions to take that either increase or decrease its confidence in the sole hypothesis.

4.1 Fine-grained implementation

Because the fine-grained architecture limits each agent to the single goal of confirming or refuting its hypothesis over a series of inputs (a "gestural sentence," in this case), the actions, α, involve the increase or decrease of the agent's confidence in its hypothesis. Hence, actions become scalar functions that are integrated over time.

Furthermore, in this context of gesture interpretation, the entities, ε, become instances of gestures while their state, σ, is the context in which they occurred (e.g. position/velocity of the manipulator, contact magnitude, etc.) The tuple (ε, σ) is a "gestural word," but because σ consists of continuous variables, it must be classified into fuzzy sets to create discrete tropism elements. These elements can be represented as:

$$(\varepsilon, \sigma_F, \alpha(\sigma), \tau)$$

where σ_F is the fuzzy set to which σ belongs and $\alpha(\sigma)$ is a scalar function proportional to membership value, hence, including the notion of preference. In operation, there is a high degree of variability in the gestural words generated by the operator. As a result, $\alpha(\sigma)$ represents a probabilistic distribution when considered over several samples (a gestural sentence) so we set τ to 1 without loss of generality:

$$(\varepsilon, \sigma_F, \alpha(\sigma), 1)$$

Adapting Figure 2 to this specific implementation results in the tropism system cognition for gesture recognition, as depicted in Figure 3 where "GRA" stands for Gesture Recognition Agent and "GIA" stands for Gesture Interpretation Agent.

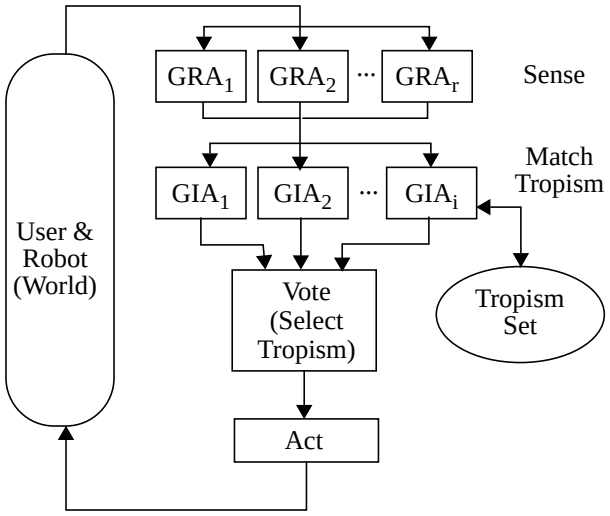

Figure 3: Static Fine-Grained Tropism System Cognitive Architecture as implemented for gesture recognition.

5 The Multi-Agent Network

5.1 What is an agent?

The term "agent" is used in many contexts, often without definition. Although we do not have a concise, quantitative definition of an agent, we view agents as hardware or software *information processors* and feel they should embody five qualitative properties:

- Input/output relationship
- Autonomy
- Persistence
- Non-parental influence
- Cognitive behavior

Input/output relationship. Agents must do something within their world, whether that world is real or simulated. They must also be responsive to that world or, at least parametrizable. Our model of port-based objects [7] allows agents to possess both "input/output ports" and "resource ports." Input/output ports are considered variables during run-time, while resource ports are considered variables during initialization. In either case the ports are inputs and outputs to which the agent responds or effects its world. The key difference is the dynamism during run-time.

Autonomy. Agents must be able to function on their own. It is not necessary that they be able to fully achieve their goals on their own nor must they be able to survive on their own. Instead, teams of agents may be needed for certain tasks, but each agent must have some level of autonomy in processing its inputs and outputs.

Persistence. There is a need to distinguish a subroutine from a child process. To this point, we can not rule out a subroutine because it can have resource ports (arguments and return value) and input/output ports and is autonomous for the duration of the call. Of course, a subroutine is wholly dependent on the main execution thread so, at best, it can be considered a *piecewise autonomous agent*. Yet, persistence is the key idea that differentiates a child process from a subroutine.

Non-parental influence. To be truly independent, an agent must be able to influence agents other than its parent. This helps distinguish levels of decomposition but does not exclude hierarchies in which a collection of agents can be considered an agent in its own right. This property also requires that the environment be considered an agent.

Cognitive behavior. Cognitive behavior, or, perhaps more appropriately, *nonlinear behavior*, is the most controversial property because it seems to be the most arbitrary. Nonetheless, we feel a need to exclude such trivial things as parametrizable constants. In essence, agents should possess some non-trivial behavior, but it is difficult to quantify or define "non-trivial."

5.2 Agent topology

The agent network for interpreting tactile gestures is illustrated graphically in Figure 4 (additional controller and utility agents are not shown but are described in [8]). The gray boxes represent gesture recognition agents and the open boxes represent gesture interpretation agents. *Confusion* is a virtual sensor agent, like the recognition agents, but it provides proprioceptive sense of the network's inability to reach consensus rather than sensing gestures from the operator. The CyberGlove GRA is for detecting hand motion gestures and symbolic gestures and is used in conjunction with another application.

The GRA's detect elemental gestures and attach application-specific state information, creating "gestural words." The GIA's (height, width, etc.) interpret the gestures, collectively arbitrate the most probable interpretation, and modify the execution parameters of the robot agent and its cartesian controller.

5.3 Tropism-based interpretation agents

The GIA's consist of a two-stage combination of characteristics from both fuzzy and neural systems to represent the tropism set. State information associated with each gesture is represented by fuzzy variables but the final confidence value is computed by a biological neuron-like accumulator of tropism "firings."

All interpretation agents are independent from one another and are responsible for evaluating the user's intentions with respect to one particular parameter. (i.e. *rect* is responsible for determining if the user is trying to switch to the rectangle trajectory.) These agents utilize the application-specific context appended to the gestural word by the GRA's which includes:

- average force components

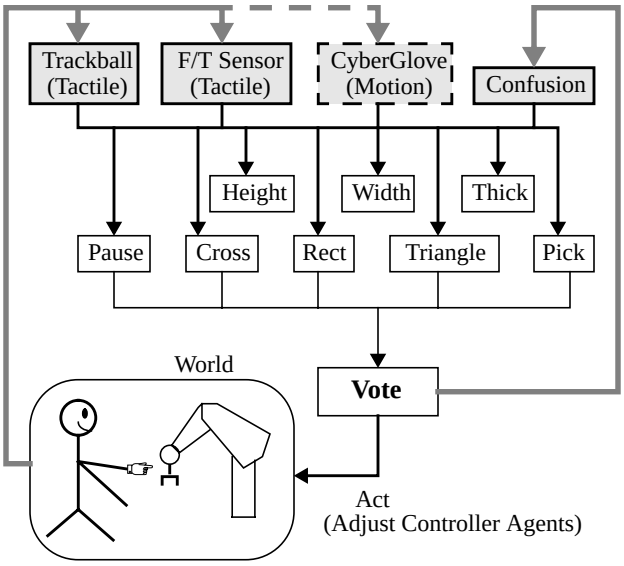

Figure 4: The tropism system instantiation.

- force magnitude
- perpendicularity of force and vertical (*perp-vert*)
- perpendicularity of force and robot velocity (*perp-vel*)
- parallelism of force and radial vector (*points-out*)
- quadrant location of gesture with respect to center
- closeness to a corner (*corner*)

For the development of the rules that make up the tropism set for representative agents, see [8].

The agents use fuzzy logic to classify gestures and match them against tropism elements and then algebraic equations to calculate actions to take that raise or lower the confidence in the agent's hypothesis.

For example, *triangle* maintains the sole hypothesis that the user intends to select the triangle trajectory. Its *likes* include nudges (tactile gestures) that tug the trajectory radially outward at the right angle corner and along the hypotenuse at the other two corners. It also "likes" nudges that push the trajectory along the hypotenuse when away from a trajectory corner. *Dislikes* include nudges that tug out at the corners close to the hypotenuse, tug straight-line motions away from the horizontal or vertical except near the hypotenuse, and nudges that impede motion. Each of these likes and dislikes become tropism elements.

For the first "like," *triangle* forms the fuzzy membership of the feature {*corner* AND *points-out* AND *4th-quadrant*}. If membership is sufficiently high, a match is made with the tropism element and an action that supports the hypothesis is taken. This action (the amount of increase in the agent's confidence) is calculated algebraically using the respective membership values and the magnitude of the nudge. The "neural characteristics" referred to above are these dynamic weights on the inputs and the accumulation of these actions and subsequent "firing" of the vote when an internal threshold is reached. (There is also an accommodation function that weights recent gestures more heavily than past gestures.)

The final vote of what the operator's intentions are and what user-level action to take is a multi-step process described in [8], the details of which have little bearing on this discussion. Essentially, the highest confidence wins.

6 Experimental trials

Many trials have been executed with the previous *ad hoc* implementation [8] and the fine-grained tropism architecture with similar performance. With all agents operating and a trained operator, Figure 5 illustrates a trial of the tropism system modifying width and trajectory shape. The top strip charts show time histories of the X and Z positions of the robot with the corresponding end effector forces superimposed (to make the nudges visible). The bottom boxes are spatial (X-Z) representations of actual cycles of the trajectory with force vectors showing points of application of the nudges. The bottom plots are spatial representations of snapshots the strip chart data.

In the first phase of the strip chart in Figure 5 (200 to 213 seconds), the robot is executing a narrow rectangular trajectory which is depicted spatially in the bottom left box. Gestures g1 and g2 push the rectangle wider. This effect is apparent in the top strip chart as the X envelope gradually grows with more gestures. Once the rectangle has been widened, nudges that tug in on the corners (g4 - g7 in the center picture) collapse the rectangle to the cross. This takes a total of five gestures. g9 and g10 further widen the cross.

Width and height adjustments can occur after a single gesture, as evidenced by g3, which further increased the width after g1 and g2 caused the initial increase. Most trajectory family changes occur after three to five nudges. This is typically what the systems takes to interpret the user's intentions, given a trained operator. "Pick" is consistently the most difficult because of its similarity to "rectangle" and the long path length between the distinguishing features (the end points).

7 Gestural HCI Applications

The implementation described thus far demonstrates the efficacy of the Tropism System Cognitive Architecture in interpreting context-dependent gestures for a novel approach to human/robot interaction. Admittedly, though, we have not demonstrated the usefulness of the interaction mechanism in this "toy" problem. To that end, we are also converting a Gesture-Based Interaction system, using many of the same agents, for robotic cable harnessing that was partially described in [9].

This system interprets not only tactile gestures but hand motion gestures and symbolic gestures using a CyberGlove (hence, the *CyberGlove* GRA shown in Figure 4). Hand motion gestures were used to train the planar trajectories the robot must follow to route the wires. Because of noise, it is not desirable to record and playback the actual training trajectories. Instead, gestures are interpreted to extract straight line segments.

Symbolic, sign language-like gestures were used to implement a menu because they can be interpreted instantaneously, making the entire interface gesture-based.

8 Conclusions

The Fine-Grained Tropism System Cognitive Architecture provides a useful and intuitive formalism for describing and implementing colonies of highly specialized software agents for tactile and motion gesture interpretation. Although the tropism sets were based on *ad hoc* rules developed previously, the learning facilities of the

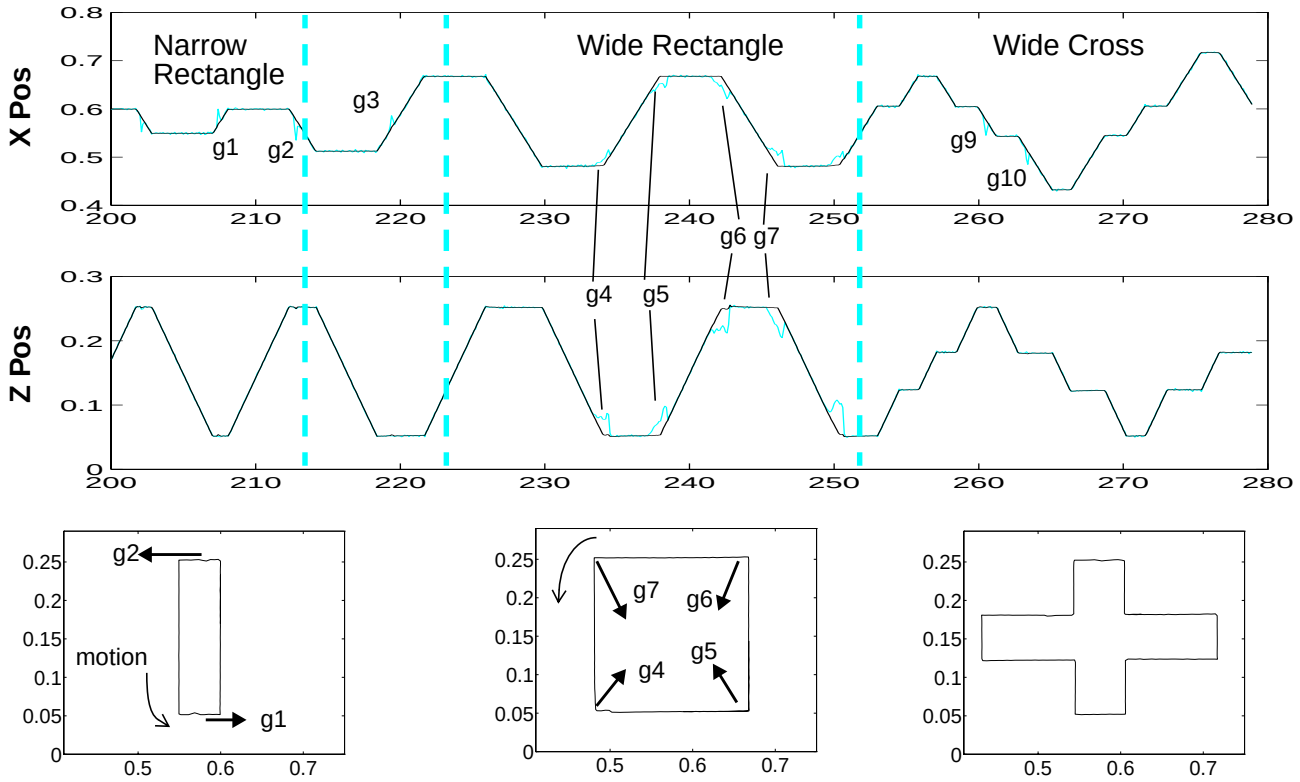

Figure 5: Annotated temporal plot (top) of the robot's position in the plane superimposed with force impulses. Spatial plots (bottom) show planar trajectory intervals with corresponding annotations.

architecture will provide opportunities for future enhancement. The gesture interpretation system described here was easy to implement, easy to reconfigure, and could be assembled in many topologies for different interpretations.

The fine-grained nature of the agents, implemented in the port-based object framework, allows for easy decomposition into reconfigurable and reusable elemental software agents. Of course, the decompositions described were done manually, which points out the need for an intuitive architectural representation. That is the key benefit to this type of task. Because there are no clear "symbolic gestures" that can be interpreted in dictionary fashion, there is a high degree of uncertainty implicit in the task. By design, inputs to the system are sparse and not precisely repeatable so interpretation is inexact. This is the type of application environment within which it can be beneficial to merely facilitate the programmer's cognitive abilities rather than attempt to replace them.

9 Acknowledgments

Mr. Voyles was supported, in part, by a National Defense Science and Engineering Graduate Fellowship and a DOE Integrated Manufacturing Predoctoral Fellowship.

10 References

[1] Agah, A, and G.A. Bekey, "Phylogenetic and Ontogenetic Learning in a Colony of Interacting Robots," *Autonomous Robots*, 1996, To Appear.

[2] Agah, A, and G.A. Bekey, "A Genetic Algorithm-Based Controller for Decentralized Multi-Agent Robotic Systems," in *Proc. of the 1996 IEEE International Conf. on Evolutionary Computation*, Nagoya, Japan, May 1996, pp. 431-436.

[3] Brooks, R.A., "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, v.RA-2, n.1, March 1986, pp. 14-23.

[4] Connell, J.H., "A Colony Architecture for an Artificial Creature," MIT AI Technical Report no. 1151, 1989.

[5] Morrow, J.D. and P.K. Khosla, "Sensorimotor Primitives for Robotic Assembly Skills," to appear in *Proc. of the 1995 IEEE Int. Conf. on Robotics and Automation*, May, 1995.

[6] Nilsson, N., "Shakey the Robot," SRI AI Center tech. note 323, April, 1984.

[7] Stewart, D.B., R.A. Volpe, and P.K. Khosla, "Design of Dynamically Reconfigurable Real-Time Software Using Port-Based Objects," CMU Robotics Institute tech. report, CMU-RI-TR-93-11, July, 1993.

[8] Voyles, R.M. and P.K. Khosla. "Tactile Gestures for Human/Robot Interaction," *Proc. of IEEE/RSJ Conf. on Intelligent Robots and Systems*, Pittsburgh, PA, v. 3, pp. 7-13.

[9] Voyles, R.M. and P.K. Khosla, "Multi-Agent Gesture Interpretation for Robotic Cable Harnessing," *Proc. of IEEE Conf. on Sys., Man, and Cyber.*, Vancouver, B.C., pp. 1113-1118.

[10] Walter, W.G. "The Living Brain," W.W. Norton & Company, Inc., New York, 1953.