Symbolic Action, Behavioral Control and Active Vision

R. James Firby and Michael J. Swain *
Computer Science Department
University of Chicago
1100 East 58th Street, Chicago, IL 60637
firby@cs.uchicago.edu, swain@cs.uchicago.edu

Abstract

This paper is about the interface between continuous and discrete robot control. We advocate encapsulating continuous actions and their related sensing strategies into structures called *situation specific activities*, which can be manipulated by a symbolic reactive planner. The approach addresses the problem of turning symbolic actions into continuous activities, and the problem of mapping continuous input into discrete symbols for use in planning and modeling.

Introduction

Planning is generally regarded as a symbolic task. An abstract model of the world is used as the starting point from which to construct a sequence of actions that will achieve some set of goals. Goals, actions, states, and the arguments they apply to, are all treated as discrete entities that can be given names for identification, representation, and reasoning.

In contrast, real robotic systems act over time in a complex, dynamic world. Sensors must be pointed and their data processed quickly and continuously for use as feedback to control motors and other actuators. Continuous control is the only way to actually get robots to do things.

This paper discusses the problems involved in connecting symbolic planning with continuous control. In particular, the issues involved in realizing discrete actions using fast concurrent behaviors are addressed. Task-specific, real-time perception is a fundamental part of these behaviors. While researchers have successfully used primitive touch and sonar sensors in such situations, it is more problematic to achieve reasonable performance with complex signals such as those from a video camera. Active vision routines are suggested as a means of incorporating visual data into real time control and as one mechanism for designating aspects of the world in an indexical-functional manner. Fast concurrent behaviors coupled with real-time indexical-functional designation bridge the gap between symbolic models and continuous action.

*This work has been supported in part by the Office of Naval Research under contract N0014-85-K-010.

Discrete Reactive Execution

Symbols give a system the power to build models and keep track of facts about the world that cannot be directly sensed. For example, a model might contain assertions that `item-23` is a hamster, that it is colored white, and that it is in `box-3`. Should a planner ever need a hamster or a box, it can consult these assertions and construct an appropriate plan. Given the task "get a hamster", a robot could decide to `(goto box-3)` and `(pickup item-23)`.

Actually executing such a plan requires filling in a great deal of detail at execution time. There may be obstacles near the box, there may be things on top of the box, and there may even be more than one hamster in the box. In each of these cases, the agent must refine its plan to include steps to deal with the difficulties as they arise. This refinement is often called reactive planning, or reactive execution, and a large amount of work has been done in the area recently [Firby, 1989a, Schoppers, 1987]. While these approaches differ in philosophy and implementation, they all agree on one point: planning and reactive execution bottom out in a set of discrete, symbolic actions.

Defining Symbolic Actions

Symbolic actions are intended to be the interface between discrete and continuous control. Unfortunately, the realization of discrete actions as real continuous processes is quite complex. In our example above, `(pickup item-23)` must be left as a single action because it must be realized in continuous motions and not a sequence of finer discrete tasks. Grabbing a moving hamster requires a sophisticated connection between tracking the hamster and moving the arm.

In fact, implementing symbolic actions involves two major problems in general: (1) Defining control algorithms that will implement the action on real hardware. (2) Mapping the symbolic arguments of the actions into sensory features that can be used easily by the control algorithms. A reactive planner, such as the RAP system [Firby, 1989a], which uses symbolic actions must address each of these problems explicitly. Rather than simply generate a symbolic primitive action with symbolic arguments, such a system must select a control algorithm and appropriate sensor features instead.

This paper discusses the issues involved in constructing control algorithms and sensing routines so that they can be combined into continuous activities that appear discrete to a symbolic planner.

Situation Specific Control

Control is typically *situation specific*, by which we mean that input is not interpreted in complex ways. The relevance of any input to the system's actions is direct and unambiguous, being computed by a fixed function that maps inputs directly into outputs. The behavior of the control system does not depend on the context in which input is received but only on the input itself. Thus, control systems are usually designed to work in a single, well-defined situation.

Controllers are structured this way because they must be fast. To ensure rapid response they are specifically constructed to avoid relying on the recognition of abstract states that require any significant sensor data processing. Information must flow directly from input to output without dependence on a world model or past history.

The most successful examples of implemented situation specific control systems for real robots are organized around the idea of behaviors. Rather than generate a single function that maps all inputs into all outputs, the control system is broken down into relatively independent processes that monitor a set of inputs and produce a specific result. These results are then combined to drive the outputs in a coherent fashion.

Behavior-based control systems have been very successful on quite sophisticated robots. Brooks has made particular progress using his subsumption architecture [Brooks, 1986]. Using subsumption, each behavior is described by a finite automaton and the overall behavior of the system arises from the interaction of the automata with the environment and each other. Typical behaviors in a mobile robot might be "move away from obstacles", "move forward", and "turn towards the light". The more complex task of "move toward a light without hitting anything" is achieved by running these simple behaviors together.

This architecture has been used by Horswill to build a robot which follows arbitrary moving objects [Horswill and Brooks, 1988] and by Mataric to build a robot that does navigation and learning in hallways [Mataric, 1990]. Gat has used a similar notion of interacting simple behaviors to construct a robot that gathers up small toys [Gat, 1990] and Soldo uses interacting simple behaviors to navigate robustly in office hallways [Soldo, 1989].

The advantages of behavior-based situation specific control systems are:

- Bounded response time from change of input to change of output.
- Modular representation and construction.
- Robustness in the face of small uncertainties and perturbations in the world.

These advantages are gained by using specialized sensing strategies and a single control program. As a result, the system as a whole exhibits behavior that might be ascribed to a single plan.

Our goal is to construct a control system that retains the advantages of situation specific control but can be used to implement multiple, varied and changing plans. We wish to construct activities with the moment to moment characteristics of continuous control but which have emergent discrete properties over longer timescales.

Situation Specific Activities

Using situation specific control systems for multiple plans in various contexts requires a shift in viewpoint. Rather than see the individual behaviors in the system as parts of an immutable whole, one can look at them as the building blocks for many different control systems that implement many different tasks. A symbolic action then becomes a particular set of primitive behaviors that can be activated together to achieve a specific goal in the outside world. Different actions activate different sets of behaviors.

Therefore, to define a symbolic action means to construct a set of behaviors that will predictably achieve a particular task. Sometimes these definitions will be easy because the desired goal will require only a single behavior. Usually, however, the symbolic actions useful in plans will be more complex and will require a whole set of behaviors for their definition. For example, an action like "go down the hallway" might require a behavior to move the robot forward, another to direct the robot away from obstacles, and yet another to keep the robot a constant distance from the right-hand wall. Executing "go down the hallway" as a single action means invoking all of these behaviors together. Once activated, their interaction with each other and the world will have the *emergent* property of satisfying the desired goal. Moreover, during the time it takes to move down the hallway, the control system doesn't change — the same behaviors are running doing the same thing all of the time. A short term stable situation has been generated. From the robot's point of view, a single control state persists over an interval of time and a discrete action has been carved out of the continuous world.

We define a situation specific activity as a collection of situation specific behaviors that work together over an interval of time to achieve a clear symbolic goal.

Guarded Situation Specific Activities

Situation specific activities are not enough to define symbolic actions by themselves. A critical feature of symbolic actions is that they are discrete and finite. Each action must be implemented as a stable control state with a well-defined beginning and end. When a situation specific activity is used to define an action, it begins when it is invoked but the constituent behaviors will run happily forever.

To be useful, activities must be finite; they must signal when they have successfully completed or when they discern that they can no longer achieve their intended goal. When the activities involve hardware actions with well defined completions this requirement is fulfilled simply by having the hardware signal when the operations are complete. The end of a sonar reading or a camera motion is unambiguous and clear.

However, when the purpose of an activity is emergent, there is no clear sign that it has been achieved. Moving forward, avoiding obstacles, and staying a constant distance from the wall has the effect of moving an agent down the hallway, but when is that action complete? Completion may mean many things: having traveled a specific distance, having reached the end of the hall, or having passed three doorways. Regardless of which of these goals is desired, noticing that it has been achieved requires monitoring states in the world beyond those needed just to carry it out. These additional monitoring behaviors must be included in any situation specific activity that implements a symbolic action.

The activities that define actions must not just signal completion; they must also signal reliably when they fail. For example, suppose a robot is executing a set of behaviors designed to move it down a hall 10 ft. What happens if the hallway is only 5 ft. long? Presumably it will eventually come to a halt under the direction of its various obstacle avoidance behaviors but its completion monitor will not yet be satisfied. At that point something should happen or the robot will just sit there forever, secure in the fact that its situation specific behaviors will move it reliably down the hall, not realizing that they have been stymied by the context the robot unwittingly finds itself in.

The solution to this problem is to augment the behaviors implementing the action to include monitor behaviors for failure states as well as for success states. This procedure is mandatory for every activity that is to define a reliable symbolic action. In the hallway, this may involve something simple like signaling that the robot has been stationary for 10 seconds, or something more complicated like signaling that the robot moving back and forth to go around the obstacle making up the end of the hall but is making no measurable progress. The important point is that the activity must *always* signal that it is wedged.[1] Only with such assurances can the planning system trust that its symbolic actions will interact with the world in a sensible fashion.

We define a guarded situation specific activity to be a set of situation specific behaviors that together achieve a particular goal and always signal when the goal has been achieved, or when it has become impossible. As the symbolic system knows which behaviors it has activated to signal success, and which it has activated to signal failure, it can tell whether an activity has completed normally or been interrupted simply by knowing which behavior is signaling. There is no need for the control system to understand the implication or purpose of its constituent behaviors in order for the symbol system to understand what has occurred.

Situation Specific Sensing

Developing guarded situation specific activities is only half the story in defining realizable symbolic actions. Authors of control systems for real robots routinely make the claim that their systems work because sensing is restricted to precisely those features of the world that are relevant to the task at hand [Horswill and Brooks, 1988, Soldo, 1989]. Underlying this claim is the realization that processing all available sensor data is both too slow to support control and unnecessary in most cases. Sensing strategies can and should be specialized to the particular needs of the situation specific behaviors that use them. Building a world model, or matching sensor readings against an existing world model, is typically not a bounded time procedure and would endanger behavior response time.

However, the interpretation of many forms of sensor data depends heavily on context. In particular, computer vision algorithms have typically required a significant application of previous knowledge, expectations about what is being viewed, and complex processing directed by this knowledge and these expectations. This would suggest that vision cannot be used as a reliable sensor in behavior-based control systems. However, recent research in *active vision* has suggested that, in fact, it can.

Active Vision

Active vision algorithms are designed specifically to be part of situation specific activities [Aloimonos *et al.*, 1988, Ballard, 1989, Swain and Ballard, 1991]. They can take advantage of the constraints provided by the motion of the observer, and are designed to extract precisely the knowledge that is required for the activity to function successfully, within the time constraints required by the agent and its environment.

For example, Ballard and Ozcandarli [Ballard, 1988] showed how an observer that maintains its gaze on an object in the world while moving perpendicular to the line of site can easily tell what was in front and behind the plane of fixation. The algorithm requires only local image computations and therefore can be executed in real-time. Similarly, Nelson [Nelson, 1989] showed that divergence (expansion) of image flow vectors can be used to detect objects that the agent is in danger of colliding with. These computations do not involve the

[1] This requirement may seem more stringent than it actually is; taking longer than expected to complete an activity can always be used as a default signal that things are going wrong.

intermediate step of calculating the three-dimensional position or relative velocity of the objects.

Active visual routines share many similarities with Ullman's idea of visual routines [Ullman, 1984] (see also [Chapman, 1990]), which have been proposed as computational models of human intermediate visual processing. Ullman's visual routines are a set of basic task-specific operations functioning on low-level representations generated from the image array which can, in theory, be composed to extract an unbounded variety of shape properties and spatial relations.

The data generated by active vision routines can be used directly by behaviors without complex, context dependent interpretation. For example, the center of an object being tracked can be used as a position for the robot to travel towards. The resulting emergent behavior has the robot follow the object even though following is not explicitly represented and neither recognition nor interpretation of the object is required.

Extracting Functional Attributes

Attributes of an object like its center, direction, or speed are important because they are useful in many different situations for accomplishing many different tasks. For example, the direction to an object can be used by behaviors designed to approach the object, to avoid the object, and to point in the object's direction. As long as the direction is known, these activities can proceed. It is the direction attribute that has a functional use in the behaviors, not the object itself or the way that the attribute is extracted.

Functional attributes are the features of the world that need to be computed in real time for use in feedback loops. Each attribute that a robot can extract, corresponds to a particular interpretation of its sensor data that is relevant to some activity that it might pursue. When behavioral control researchers suggest adding a new sensor when a new behavior is to be encoded, what they really mean is to add a new real-time functional attribute extractor. Active vision routines are a particularly useful idea in this respect because they are designed to extract different attributes from the world using the same sensor.

In fact, there will often be many different active vision routines for extracting the same functional attribute from the world. This allows an agent substantial leeway to instantiate its activities in different ways under different circumstances using different active vision routines. For example, tracking the location of an obstacle might be done using color, shape, or motion, depending on what is most reliable in the specific context.

Indexical-Functional Designation

Guarded situation specific activities define emergent symbolic actions that can be carried out in the world. Active vision routines compute the functional attributes of items in the world that are used within activities. Thus, if we think of an activity as describing a symbolic action, we can think of active vision routines as describing the arguments to that action.

Consider our robot's desire to fetch a hamster. The robot moves itself into position so that it can reach into the box containing the hamster, and it refines its plan to the action `(pickup item-23)` where `item-23` is the hamster. To instantiate this action, the robot looks up its guarded activity description for `pickup` and finds behaviors to move its hand to a location x, to warn if location x is out of reach, to signal when motion is complete, *etc.* These behaviors all refer to a location but do not define it.

The robot now examines its world model and determines that `item-23`, the hamster, is white and the box holding it is brown. Therefore, from the many possible active vision routines for tracking the hamster, the robot chooses a routine to track the position of a colored object (in this case white). The output of this tracking routine designates the position of the hamster. Putting these all together, the robot sets x to be the output of the a sensing behavior that generates constant updates of the hamster's location and activates action behaviors that move its hand toward that location. With luck, the hand will eventually end up at the hamster and the pickup will succeed.

[Agre and Chapman, 1987] describe attributes of the world like the hamster's position as functional-indexical references. The robot's choice of an active vision routine to extract an attribute from the world is precisely the generation of a functional-indexical designation for the symbolic description of that attribute in the robot's world model. The active vision routine operationalizes exactly what is needed to reliably refer to and use an abstract symbolic property of the world. The hamster becomes "the-position-I-am-tracking" and the pickup activity takes "the-position-I-am-tracking" as its argument.

Previous Work in Robotic Control

Several researchers have designed architectures for robot control, but few bridge the gap between symbolic and continuous control. If they do, they do not consider the problems of interacting with the world with high-bandwidth sensors that require context dependent interpretation.

Brooks's subsumption architecture [Brooks, 1986], Maes's competence modules [Maes, 1989], and Kaelbling's Gapps architecture [Kaelbling, 1988] contain no symbolic plan or model, and therefore are restricted in the tasks they can carry out and the flexibility of their responses. On the other hand, Robo-Soar [Laird, 1990] contains no continuous control, and often has lethargic, coarse-grained responses to events in the world. Theo [Mitchell, 1990] is an example of one system that does attempt to bridge the gap, using a set of control rules to define the continuous system augmented by a planner, which is invoked when there is no appropriate rule. However, the simplicity of Theo's sensors (a pair of sonar depth sensors) and actions (rotate a sonar, move the robot) make the discrete/continuous interface relatively straightforward.

Dean's recent survey of control theory and AI planning techniques for robot control shows a clear dichotomy between techniques for continuous and symbolic control [Dean, 1991]. We believe that the dichotomy is not a historical accident, but that it is a result of differing requirements of speed and flexibility at different levels of representation. Only by building an architecture that spans the gap will intelligent yet nimble robots be built.

Summary and Future Work

This paper proposes a model of symbolic action that includes a realistic translation between symbolic and continuous control regimes. Symbolic actions map into guarded situation specific activities which taken together have the emergent property of achieving the desired state in the world. The behaviors making up the activity refer to the world, not directly through sensors, but rather through functional-indexical attributes that can be computed quickly and easily.

The arguments to actions map into active perception strategies, such as active vision routines [2], designed to compute functional-indexical attributes in real time. The choice of both the activity for the action and the sensing strategies for designating its arguments are up to the robot's symbolic or reactive execution system. As long as the execution system has a reasonable idea of the context surrounding the agent, it will be able to assemble situation specific strategies that will achieve appropriate results.

Adopting the approach to planning and control advocated in this paper emphasizes research areas at the boundary between symbolic and continuous action that have traditionally been ignored in both the planning and vision communities. We must define situation specific activities and functional-indexical strategies that can be used together to achieve semantically meaningful discrete goals in the world. In other words, we must define a reasonable set of primitive actions for real life — a language for describing the steps we take to do ordinary things.

We believe that the ideas we have outlined in this paper provide a framework for developing such a language. Our current work is aimed at defining the activities needed to move a simple robot around our laboratory and search for specific items using visual feedback. Defining such a language will bring us significantly closer to understanding adaptable, reliable behavior, and building useful robots.

[2] Or any other fast method for extracting specific interpretations of data. As we mentioned, touch sensors and sonars compute some attributes of the world directly.

References

[Agre and Chapman, 1987] Philip E. Agre and David Chapman. Pengi: An implementation of a theory of activity. In *AAAI '87*.

[Aloimonos *et al.*, 1988] Yiannis Aloimonos, Isaac Weiss, and Amit Bandyopadhay. Active vision. *IJCV*, 1: 333-356, 1988.

[Ballard, 1989] Dana H. Ballard. Animate Vision. *Artificial Intelligence*, 48: 57-86, 1991.

[Ballard, 1988] Dana H. Ballard and A. Ozcandarli. Eye Fixation and early vision: Kinetic Depth. In *ICCV '88*.

[Brooks, 1986] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1), 1986.

[Chapman, 1990] David Chapman. Intermediate vision: Architecture, implementation, and use. Technical Report TR-90-06, Teleos Research, 1990.

[Dean, 1991] Tom Dean. Control theory and artificial intelligence. Manuscript in preparation, January 1991.

[Firby, 1989a] R. James Firby. Adaptive execution in complex dynamic worlds. Technical Report YALEU/CSD/RR #672, Computer Science Department, Yale University, January 1989.

[Firby, 1989b] R. James Firby. Task-directed sensing. In *Sensor Fusion II: Human and Machine Strategies*, Philadelphia, PA, November 1989. SPIE.

[Gat, 1990] Erann Gat. BDL: A language for programming reactive robotic control systems. Unpubl. doc., 1990.

[Horswill and Brooks, 1988] Ian Douglas Horswill and Rodney Allen Brooks. Situated vision in a dynamic world: Chasing objects. In *AAAI '88*, 796-800.

[Kaelbling, 1988] Leslie Pack Kaelbling. Goals as parallel program specifications. In *AAAI '88*, 60-65.

[Laird, 1990] John E. Laird. Integrating planning and execution in Soar. In *AAAI Spring Symposium Series: Planning in Uncertain, Unpredictable, or Changing Environments*, 1990.

[Lawton *et al.*, 1990] Daryl Lawton, Ron Arkin, and Jonathan Cameron. Qualitative spacial understanding and reactive control for autonomous robots. In *IROS 90*, Tsuchiwra, Japan, June 1990.

[Maes, 1989] Pattie Maes. The dynamics of action selection. In *IJCAI '89*, 991-997.

[Mataric, 1990] Maja J. Mataric. *A Model for Distributed Mobile Robot Environment Learning and Navigation*. PhD thesis, Department of EE & CS, MIT, 1990.

[Mitchell, 1990] Tom M. Mitchell. Becoming increasingly reactive. In *Proceedings of AAAI '90*, 1051-1058.

[Nelson, 1989] R. C. Nelson and J. Aloimonos. Obstacle avoidance using flow field divergence, *IEEE PAMI*, 11: 1102-1106, 1989.

[Schoppers, 1987] M. Schoppers. Universal plans for reactive robots in unpredictable environments. In IJCAI '87.

[Soldo, 1989] Monnett Hanvey Soldo. Fusion without representation. In *SPIE Vol. 1198 Sensor Fusion II: Human and Machine Strategies*, 1989.

[Swain and Ballard, 1991] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, to appear 1991.

[Ullman, 1984] Shimon Ullman. Visual routines. *Cognition*, 18:97-159, 1984.