Systems for Perception, Action and Effective Representation

Glenn S. Wasson, Gabriel J. Ferrer and Worthy N. Martin
Computer Science Department, Thornton Hall, University of Virginia, Charlottesville, VA 22903
{ wasson | ferrer | martin }@virginia.edu

1. Abstract

The situated control component of most autonomous agents has traditionally been either a planner or reactive system. However, there is general agreement that planning and reaction represent two ends of a spectrum. These action selection methods differ in, among other things, their treatment of representation. Planners perform projection on an internal world model, while pure reactive systems store no internal representation, using "the world as its own model". We seek to move situated controllers, or perception/action systems, toward the middle of this spectrum.

We have developed a perception/action (PA) system which incorporates task dependent representations. These representations do not constitute the full world model of classical planners, but they do provide action selection information beyond the current sensory data used by reactive systems. We will show that our representations are compact and maintainable and that they make our PA system more effective than a stateless reactive system. We will also discuss the implications a PA system using such representation has on the rest of an agent's architecture. The discussion will be carried out in the context of an application involving both planned and dynamic components.

2. INTRODUCTION

Most researchers would agree that a purely stateless autonomous agent is a strawman, yet there is little agreement about what should be represented and how that representation should be organized or accessed. We are interested in using representation at the perception/action layer (variously called the "skill layer" [5][3], or the "controller" [6]), of an agent architecture; that is, the layer where perception and action are most closely linked. We have developed a system of representation which can be used effectively by an agent's PA layer. In the next section, we will discuss the uses of representation in a PA layer. Then we will develop an organizational structure for our representation as well as an access method. We will then address the issues of representation maintenance and performance in the context of a task performed by our robot, Bruce. Finally, we will explore the impact that representation at the PA layer has on the other layers of our three-level architecture.

3. REPRESENTATION

Why have representation in a perception/action system? It can serve two inter-related purposes. First, representation can greatly facilitate some tasks by storing information which cannot be currently gathered from sensors. Consider the following example: you are driving your car and you come to an intersection. To make a left turn, you must determine if the cross street is clear in both directions. While it may be possible to create a reactive system which can make this determination by rapidly redirecting its sensors, it is more practical to have a representation for the-direction-I-just-looked-is-clear.

Now consider making a left turn in a different situation. You look right and see the way is clear, then you look left and see that direction is also clear. When you look back to the right, you see that a truck has pulled up next to you and your view is blocked. An agent with representation can remember that the way was clear (for some interval), but a pure reactive agent cannot.

Suppose instead that you look left and see that the way is clear, then you look right and see a car coming. Now the truck pulls up next to you in the right lane and blocks your view of the oncoming car. You need to remember that the car was there so that you don't pull out. Since the truck occludes the car, a reactive system (without x-ray vision) cannot use the knowledge of the car's presence in its action selection. In both of these cases, the safest strategy for a reactive agent is to wait for the truck to leave, just in case it was occluding any cars. However, this limits the agent to act only when it can gather *all* information necessary to select an action at the *current instant* in time. An agent with representation can act correctly in this situation (within some time frame) if it remembers whether there was a car coming before the truck arrived. A system with representation can act with limited extent sensors and in the presence of occlusion.

Representation can also serve as the basis for active perception. It can describe what to look for and where to look. Segmentation and recognition are more difficult without a priori knowledge. Our PA layer's representation describes a limited collection of task-dependent object's which an agent expects to find in its sensory field. In addition, representation provides information about where an agent should direct its perception. For example, a quarterback agent will not be able to simultaneously perceive all his receivers, but he can use his knowledge of their positions (based either on recent sightings or an understanding of their intended routes) to check each one and decide who is open.

4. THE EFFECTIVE FIELD OF VIEW

This section describes the overall organization of representation in our PA layer, as well as the specific structures of which it consists. We address how representation is accessed during the process of action selection.

Our PA layer selects actions based on information in its *effective field of view* (EFOV). We define the absolute field of view of a sensor as the area in which it can reliably extract information about the environment at any one time[1]. The effective field of view includes the absolute field of view as well as the location and identity of recently perceived objects which are important to the agent's task. For example, in the previous quarterback example, the quarterback's absolute field of view may contain only the receiver streaking toward the end-zone, while his effective field of view may contain that information as well as the expected position of the linebacker he saw blitzing from the left side.

It is important to note that his effective field of view does not contain the location of the coach, the fans or any players on the sidelines because their positions are unimportant for the complete-the-pass task. Note also that the priorities of the task: completing the pass above personal safety, dictate that the most reliable information, i.e. the most recently sensed data, be maintained about the receiver. Thus the absolute field of view tracks the receiver while the positions of blitzing linemen are extrapolated from where they were last seen.

Since the effective field of view contains information outside of the absolute field of view, we must begin to ask the question, what should be represented? This is a broad question with a range of semantic meanings for 'what'. Since we are interested in a PA layer that must rapidly interact with its environment, we have defined, for our representation, simple, task-dependent structures called *markers*.

4.1 Markers

The unit of representation in our system is the marker. Each marker corresponds to an object that the PA layer believes to be important to its current task. Ullman [12] uses the term "marking" to refer to remembering a location for later reference. He supposes the creation of a "marking map" that holds context dependent locations, in the visual field, that have been analyzed. Attneave and Farrar [2] suggest that locations outside the visual field can be marked. Pylyshyn describes a similar concept in his FINST model [8]. He describes a limited number of "reference tokens" which can be bound to visual features. This binding is a prerequisite to determining relational properties about those features. Agre and Chapman [1] use markers in their Pengi system to identify objects relevant to the penguin's current task, e.g. the-ice-block-blocking-my-escape. While Agre and Chapman used an overhead perspective and directly accessed the game's data structures, Brill considers issues involved in using markers with early vision in dynamic, 3D, 1st person domains involving occlusion [4].

1. We speak of fields of "view" because we are primarily interested in visual agents and we will concentrate on them throughout the rest of this paper. However, the arguments apply to any limited range sensor.

Markers are generally considered to denote the `what` and the `where` for important objects. The `what` of a marker describes the object's role in the current task. For example, a screwdriver may be marked as such for the task of tightening a screw, but may be marked as a pry-bar for the task of opening a stuck window. These two tasks require different roles to be filled, yet the agent can mark the same object for both. Selecting an object with the appropriate characteristics is a function of the marker's identification routines (discussed below). A marker's `where` component specifies the location of the marked object in an ego-centric coordinate system.

In addition to `what` and `where`, our markers contain two additional components: `identity` and `action`. Markers have primitive tracking capabilities and their `identity` routines are used to maintain the location of their associated objects. `Identity` is divided into two not necessarily distinct routines, `locate` and `track`. While `track` is concerned with the frame-to-frame identification of the marked object, `locate` is used to initially detect the object (or possibly to relocate the object if it is lost by `track`).

Markers may be in one of two states: instantiated and uninstantiated. An instantiated marker is one that the PA layer has associated with a specific object. These objects are tracked from frame to frame via the `track` routines of their associated markers. Uninstantiated markers represent the *expectations* of the agent's "higher" layers, i.e. what to expect and where to expect it. The PA layer attempts to associate an uninstantiated marker with an object in the absolute field of view using the marker's `locate` routine.

`Action` specifies what the agent wishes to do with the marked object. `Actions` specify goals to the PA layer. For example, a chair marker with a **goto** action would cause the agent to approach the particular chair designated by the marker's `what/where` components. Each `action` specifies a number of parallel routines which the PA layer executes to achieve the desired overall behavior. The exact `actions` that are available will depend on the types of roles within the current task, and the capabilities of the agent.

The components of markers are kept minimal so that they may be maintained more robustly. However, as we will see, markers provide just the information required for a great variety of tasks that the PA layer executes. "Higher level" knowledge is maintained by higher layers of the agent architecture.

Access to the PA layer's representation means examining the `what` and `where` components of markers in the agent's effective field of view. Our PA layer selects actions based on information contained in its effective field of view, i.e. its current sensor values and the information stored in its markers. The underlying PA layer keeps the markers up-to-date so that actions are based on current information.

4.2 The Effective Field Of View In Practice

So far we have developed the concept of the effective field of view as a system of representation for a PA layer. We will now describe an implemented agent performing a series of tasks to play a game of tag. This application allows us to discuss important issues such as maintenance of representation, assessment of performance and behavioral descriptions.

4.3 Tag: An Application

Our application is a two player game of tag. One player is "it" and must touch the other player, before that other player reaches a designated location called the home base. In our game, our autonomous agent, Bruce, is "it", while a human controlled vehicle (the truck) is the other player. Bruce begins the game with no knowledge of the truck's location. Bruce conducts a systematic search of his environment for his opponent, and attempts to touch ("tag") it upon discovery. The human controlled player is not allowed to "run for home" until Bruce has spotted it and begun a chase.

Bruce is an 68HC11-based agent possessing tactile and visual sensors. A single color camera is mounted on a pan/tilt platform on the agent's front and there are optical encoders on its wheels. Offboard processing is facilitated by video and data link transmitters.

4.4 Representation Maintenance

The primary reason that many agents are designed with no representation at the perception/action level is because of the difficulty of accurately maintaining that representation. "Stale" information leads to the selection of incorrect actions. The PA layer is not concerned with the "truth maintenance" of the sort of high-level predicates used by inference engines; rather it must keep the `where` components of its markers up-to-date (maintenance of the `what` component is discussed in [4]).

Since the effective field of view is used to select the agent's immediate actions, the agent must both execute its actions and update its markers in parallel. Consider the **chase** behavior which is initiated when Bruce detects his opponent. **Chase** consists of two parallel activities (termed PA processes). The first directs Bruce's pan/tilt unit, or "neck", to point the camera at his opponent. The second controls the wheels to move Bruce toward his opponent in order to complete a "tag". These PA processes consult the **chase-target** marker (which has its `identity` component set to **truck**) to determine where to direct the effectors under their control, so it is important that the representation be maintained effectively.

In our PA layer, the effective field of view is maintained by keeping the positions of all marked objects up-to-date. A marker's estimated position, in ego-centric polar coordinates, is calculated through a "hypothesis and confirm (if possible)" process. The hypothesis is created from the previously stored position and proprioceptive data, e.g. the most recent values read from the agent's drive shaft encoders. If the hypothesized position should be visible in the current absolute field of view, the agent looks for a correspondent in the current image. If a correspondent can be found, the object's position (based on its azimuth and elevation in the image with the assumption that all objects lie on the ground) is stored instead of its expected location. If the hypothesized position should not be visible, or no correspondent is found, then that hypothesis becomes the current position.

Visual correspondence is done by finding the ground/non-ground boundary, or groundline [7]. Vertical discontinuities in this line represent the edges of objects (see figure 1). Each segmented image region is analyzed by each marker's `track` routine (example routines include histogram intersection [10]). Matching of markers to objects is based on results of the match determined by `track` and the distance between this object's position and the position stored in the marker.

Figure 1. The Groundline and the Agent's Opponent

Now we can see one of the benefits of the effective field of view. The location of the truck may be computed from sensor data (when the truck is in the absolute field of view) or based on proprioceptive data (such as when the truck is at an azimuth not visible at the current camera angle or is occluded), but the PA processes operate the same in both situations. In this manner, the effective field of view provides a uniform access mechanism to all the information important to the agent's current task.

5. PA LAYER IN A COMPLETE AGENT

Many complex tasks, such as playing a game of tag, require more deliberative capabilities than we consider to be part of the PA layer. Many autonomous agent architectures, [3][5][6], have several layers operating at different "levels" of abstraction. Bruce's architecture also has three layers [13]. For this application, the architecture has a specialized planner (DL) at the highest layer, and the previously described PA layer at the lowest. In between is a management layer called the task executor (TE). In the following section we discuss how PA layer representation can be used by the TE. In particular, we address how the TE assists in playing the game of tag. Finally we discuss issues involved in the design of the TE and the DL to work with our PA layer.

5.1 Information Flow

A PA layer's representation provides the other layers with a means of specifying goals and a mechanism for receiving task-dependent, semantic knowledge about the world.

Consider the search task that is part of the tag game. Initially, Bruce does not know the location of his opponent and so must systematically search his environment. Bruce possesses a map of his environment from which the planner generates a search route (see figure 2) such that all of the game area will fall within the absolute field of view at some point [11]. However, the map's "global" coordinate system is fundamentally different than the local-space representation of markers. The agent's position in this global coordinate system is the type of information which the PA layer cannot effectively maintain. Thus, we need a means for the planner to communicate its goals to the PA layer in a manner suitable for effective action.

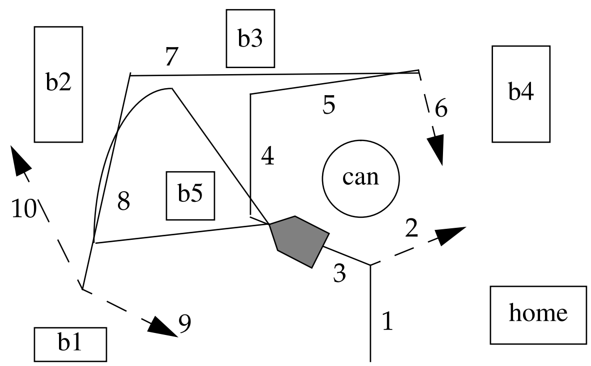

Figure 2. The Search Plan

Figure 2 shows a plan to search the game area. The solid lines represent paths to travel, while the dashed lines represent points where the agent must turn its camera in the direction indicated by the arrow. The agent is shown as a shaded pentagon with the point indicating its direction of motion. The cone indicates the agent's absolute field of view.

Figure 3a shows Bruce in the same position as the agent indicated in figure 2; near completion of step 3. In figure 3, the white lines extending from the agent's camera denote its absolute field of view. Solid arrows indicate objects associated with instantiated markers, while dashed arrows indicate locations at which the TE expects to find a specific object that can be associated with an uninstantiated marker. When Bruce completes step 3, he must go toward the object labeled **b3** which is outside of his current absolute field of view. The TE directs the PA layer to do this by placing an *uninstantiated* **destination** marker with identity **b3** (and a **goto** action) into the PA layer's effective field of view. This is shown by the dashed arrow in figure 3b.

As Bruce begins to execute the **goto** action of the current **destination** marker, as in figure 3c, the only information about **b3**'s position is that provided by the TE. The floating arrows in figures 3b and 3c show the general position indicated by the marker. When proprioceptive data indicates that the object on which Bruce would like to instantiate the marker should be within the current absolute field of view, the PA layer executes that marker's `Locate` routine to identify the object. If **b3** is detected, the position in the

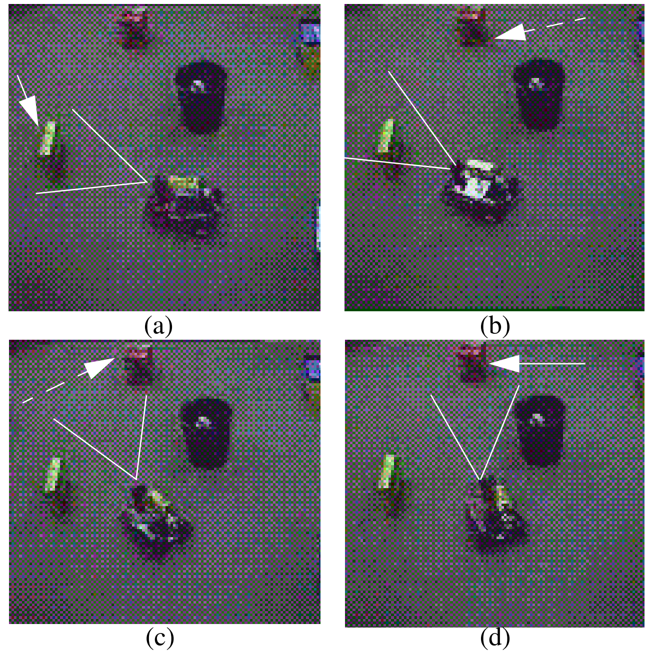

Figure 3. Marker Instantiation

marker is replaced by the newly detected position and the marker is said to be instantiated. This is shown in figure 3d. Now the object associated (**b3**)with the **destination** marker can be tracked. It is important to note that this behavior would be difficult for a stateless system because in the situation shown in figure 3b, there is no sensory information available to Bruce telling him the way to **b3**. The effective field of view allows the PA layer to use information that has not been acquired by the agent's sensors.

At the start of a round of tag, the PA layer is given an uninstantiated **chase-target** marker whose identity is **truck**. This marker contains no estimate of the truck's position, so the PA layer will try to instantiate the marker on any object matching the truck's sensory characteristics. The TE monitors the PA layer's effective field of view to see if the **chase-target** marker becomes instantiated. If so, the TE will stop the current PA layer action (by deleting the associated marker) and will alter the **chase-target** marker's `action` component to **chase**. Playing tag then involves the above TE/PA layer interaction directing the agent to follow each "leg" of the exploration route provided by the planning layer. As an action completes, the associated marker may be dropped. In figure 3a, there is a **destination** marker for **b5**, but in figure 3b (after **goto destination** has been completed) that marker has been dropped because it is no longer important to the agent's current task of going to **b3**.

5.2 Design Of Other Layers

We have described Bruce's three layered architecture as having a planner, a middle-manager and a PA layer using our effective field of view paradigm. This type of division is common among agent architectures today. However, representation at the PA layer provides new issues to be considered in the design of the other layers.

For example, the TE needs to bridge the gap between the knowledge used by the planner and the ego-centric, task-dependent representations of markers. For the search task of the tag application, the TE must compute ego-centric coordinates for the markers based on the map, knowledge of the agent's starting position and the `where` components of previous markers when their associated **goto** actions completed.

Planners need to create plans that take into account agent capabilities. The physical size and mechanics of our agent place limits on the range over which purely proprioceptive data can be accurately computed. Thus an uninstantiated marker which indicates its associated object to be outside of this range will likely not be effective for the **goto** task.

6. RELATED WORK

There are a number of autonomous agent architectures [3][5][6] utilizing multiple layers of activity. Their lowest layer, often analogous to our PA layer, is a collection of independent behavior processes which communicate with each other and the upper layers through some channels [6] or shared memory [5].

Designing these behaviors is mostly an art. While this paper does not alleviate the difficulties of behavioral design, it does seek to say something about the communication between behaviors and the non-PA layers of an agent's architecture. Work such as [6] and [5] allows communication to be unstructured and arbitrary. We have defined a more structured communication mechanism via the passing of markers associated with task-dependent roles. We believe that this interface allows the effective exchange of important information between behaviors or between behaviors and the upper layers of an architecture. This gives the upper layers a useful and straightforward paradigm for instructing the PA layer without over-constraining its options.

Spector's supervenience model [9] consists of a set of communicating levels in which lower levels pass facts about the world to higher levels while higher levels pass goals down to lower levels. Each level in the supervenience architecture contains a knowledge base accessible as a blackboard system. Uniform knowledge structures are used at each level. We have presented a knowledge structure, the marker, for use by a PA layer. We believe that the types of knowledge structures needed to perform the tasks required of higher levels are different than those that are usable by a PA layer in a dynamic environment. These "higher level" structures may contain more information than can be effectively maintained by a PA layer. It is important to design the knowledge structures of the PA layer carefully so that the advantages of the knowledge are obtained while real-time performance remains robust.

7. CONCLUSIONS

We have developed a situated controller which incorporates task dependent representations. This perception/action layer comprises a stateless reactive system with a limited amount of task-dependent representation.

We have developed an organizational structure for representation at the PA layer (markers) and described an access mechanism for that information (the effective field of view). These representations are compact, maintainable, and enable the PA layer to perform some task more effectively than a stateless system. Finally we discussed a complete agent architecture using a PA layer in the context of the tag task. Marker based representations allowed our agent's plans to be communicated to its PA layer. We have demonstrated the utility and effectiveness of this approach on a physical robot performing a non-trivial real world task containing both dynamic and statically planned components.

Representation has long been the subject of debate in the research community. We have created an agent which can use representation both efficiently and effectively in the lowest layer of its software architecture. We believe this constitutes an important step toward the integration of dynamic actions and cognitive activities.

8. REFERENCES

[1] Agre, P.E.; and Chapman, D. 1987. Pengi: An Implementation of a Theory of Activity. *AAAI-87*: 268-272.

[2] Attneave, F.; and Farrar, P. 1977. The Visual World Behind the Head, *American Journal of Psychology* 90(4): 549-563.

[3] Bonasso, R.P.; and Kortenkamp, D. 1995. Characterizing an Architecture for Intelligent, Reactive Agents. *AAAI Spring Symposium on Software Architectures*.

[4] Brill, F.Z. 1996. Representation of Local Space in Perception/Action Systems: Behaving Appropriately in Difficult Situations. Ph.D. Dissertation, Department of Computer Science, Univ. of VA.

[5] Firby, R.J.; and Slack, M.G. 1995. Task Execution: Interfacing to Reactive Skill Networks. *AAAI Spring Symposium on Software Architectures*.

[6] Gat, E. 1992. Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots. *AAAI-92*: 809-815.

[7] Horswill, I. 1993. Polly: A Vision-Based Artificial Agent. *AAAI-93*: 824-829.

[8] Pylyshyn, Z.W.; and Storm R.W. 1988. Tracking Multiple Independent Targets: Evidence for a Parallel Tracking Mechanism. *Spatial Vision* 3(3):179-197.

[9] Spector, L. 1992. Planning and Reacting Across Supervenient Levels of Representation. *International Journal of Intelligent and Cooperative Information Systems* 1(3 & 4): 411-449.

[10] Swain M.J.; and Ballard D.H. 1991. Color Indexing. *International Journal of Computer Vision*. 7(1):11-32.

[11] Taylor, C.J.; and Kriegman, D.J. 1994. Vision-Based Motion Planning and Exploration Algorithms for Mobile Robots. *Workshop on the Algorithmic Foundation of Robotics*.

[12] Ullman, S. 1984. Visual Routines. *Cognition* 18:97-159.

[13] Wasson, G.; and Martin, W. 1996. Integration and Action in Perception/Action Systems with Access to Non-local Space Information. *AAAI workshop on Theories of Planning Action and Control: Bridging the Gap*.