Integration and Action in Perception/Action Systems with Access to Non-Local-Space Information

From: AAAI Technical Report WS-96-07. Compilation copyright © 1996, AAAI (www.aaai.org). All rights reserved.

Glenn S. Wasson and Worthy N. Martin

Computer Science Department, Thornton Hall, University of Virginia, Charlottesville, VA 22901

{wasson | martin}@virginia.edu

Abstract

Actions and objects are closely tied together (Russell & Norvig 1995). We often think of the actions we can perform on objects as properties of those objects. The theory of action presented here, contains the belief that actions are defined in terms of the objects they effect. Intuitively then, actions consist of a verb and a noun, for example, pick-up-the-soda-can, go-to-the-car, or paint-the-canvas. We are interested in two different roles for nouns, as direct objects which the agent manipulates (the-can in pick-up-the-can) and as indirect objects which specify destinations (the-car in go-to-the-car). We refer to these nouns as m-role objects or d-role objects. In our theory of action, before performing any action, an agent first develops a deictic representation of the m-role or d-role object. So, the agent identifies the-soda-can, the-car, or the-canvas. When this is completed, the specified action can be taken using the object indicated by the representation. A plan is, therefore, a set of objects and the actions to perform.

We contend that this deictic organization is an effective means of expressing actions because an action cannot take place without identifying the m or d-role objects. Our research will show that actions can be performed effectively and plans can be organized appropriately using such a representation. We will present a system that uses deictic representations as a means to integrate a deliberate planner and a perception/action system. Non-local-space information in the plans will be conveyed to the perception/action system, where it can be acted on.

Emergent from our theory of activity is a representation system that integrates agent control system components. Our theory is centered on the real world objects effected by action. We believe this provides sufficient and effective organizing principles for designing perception/action systems.

Background

There are two schools of thought in the AI planning community, the classical planning camp [see (Tate, Hendler & Drummond 1990) for a survey] and the reactive camp (Brooks 1986)(Firby 1987). The classical planning proponents claim that complex tasks cannot be completed without reasoning about the state of the world in advance, while the reactive proponents claim the world is too dynamic to reason about or even represent. Clearly, a little of each is true.

One source of information, common to many classical planners, but inaccessible to pure perception/action (i.e. reactive) systems, is non-local-space information, such as is present in a map. A perception/action system operating in a large environment may not be able to perceive its entire route space a priori. Some deliberative entity must plan a course for the agent and activate appropriate behaviors at the correct time. This is because a perception/action (PA) system has no memory or representation. It cannot create a plan, it only knows what action should be taken *now*. Although PA systems can perform some navigation tasks without consulting a deliberator, when the agent reaches a T-junction for example, the perception/action system must appeal to its route plan in order to deterministically choose the correct direction. This paper's agenda is not to discuss better methods of plan generation, but to discuss how to better communicate the information expressed in plans to a perception/action system. That is, how to integrate a non-local-space system component and a PA system component by transferring non-local-space information to the PA component, in such a way that the PA component can use the information efficiently and effectively.

Integration

Traditionally, perception/action systems are thought of as determining their actions directly from their perceptions. This can be expressed as $a = f(P)$, where a is the current action and P is the set of current precepts. Work by Brill (Brill 1995) has shown that perception/action systems can exhibit greatly improved performance by incorporating small amounts of task-dependent representation into their framework. In such systems, $a = f(P, M)$, where a and P are as before and M is the PA component's memory or representation. Brill proposes a deictic, local-space representation system based on Ullman's (Ullman 1984) markers, as used in (Agre & Chapman 1987).

Since perception/action systems can use such local-space representations effectively to achieve certain complex behaviors, then the task of the entity possessing the non-local-space representation is to transfer this information into the PA system's local-space representation. Given the formulation, $a = f(P, M)$, the most efficient way of influencing a PA system's actions is through M.

However, there is a temptation to pass arbitrarily complex data structures to the PA system through M. In order for a PA system to remain effective in dynamic environments, it cannot be asked to manipulate large data structures. We present a minimalist interface between a non-local-space component and a perception/action system, using markers.

Markers

Reactive systems, as conceived by Brooks (Brooks 1986), do not attempt to model the world because any stored representation will become stale. In fact, the design philosophy of such systems is that the world is a better model of itself than an agent could ever build. Therefore, an agent needs no representation because it can just "look" at the world to determine whatever it needs to know.

Clearly, there are some simple tasks which require some form of representation. For example, when I sit in an easy-chair, I align myself with the chair, but when I actually sit down, I am facing away from the seat. However, because I have a representation of where the seat is, which I developed while looking at it, I am able to maneuver myself into the chair. If a pure reactive system were trying to sit down, once it is no longer facing the chair, there is nothing in its perceptive field to indicate that it should sit down. Without representation to tell the system that it was trying to sit, it never will.

A more sophisticated system which could sit in the chair at my desk might use the computer monitor as a visual cue to decide when to sit, i.e. when you see the monitor, the chair is behind you, so sit down. However, my desk chair has wheels and can move around. In order to decide what visual cue tells the agent when to sit for various instances of the sitting task, the sit routine must be parameterized. Another system component must pass this parameter to the reactive system. This interaction is discussed in section titled "Transferring Non-Local-Space Information to a PA System".

Trying to preserve the robustness of PA systems, while incorporating representation involves using a minimalist representation for any given task. Task dependent representations give us a minimalist deictic representation in that the agent models only those objects whose roles are relevant to its current task. The reason that only task dependent objects are modeled is that the high cost of keeping an elaborate model up-to-date limits PA system effectiveness.

Markers are the basis of the representation in our theory of action. Ullman (Ullman 1984) uses the term "marking" to refer to remembering a location for later reference. He supposes the creation of a "marking map" which holds context dependent location, in the visual field, that have been analyzed. Attneave and Farrar (Attneave & Farrar 1977) suggest that locations outside the visual field can be marked. Pylyshyn describes a similar concept in his FINST model (Pylyshyn & Storm 1988). He described a limited number of "reference tokens" which can be bound to visual features. This binding is a pre-requisite to determining relational properties about those features. Agre and Chapman (Agre & Chapman 1987) use markers in their Pengi system to identify objects relevant to the penguin's current task, e.g. the-ice-block-blocking-my-escape. Brill considers issues involved in using markers in dynamic 3D domains involving occlusion (Brill, Wasson & Martin 1996).

Markers are a task dependent representation in which each marker is associated with an object having some role (m or d) in the task. Markers are generally thought to contain a 'what' and a 'where' component. The 'what' component describes the object's role in the current task. The marker associated with a particular object may have different 'what' components for different tasks. For example, an electric socket could be marked as an *obstacle* for navigation, unless the agent needed to recharge, in which case, the same socket could be marked as *goal*. The 'where' component contains the location of the associated object in some coordinate system useful to the current task. The markers in our system have other components which are discussed in the implementation section. However, the total number of components in a marker is purposely kept small in order to avoid the data manipulation problems discussed earlier.

Expectants

Since a marker is associated with an object, an important question is how the association is made. Pylyshyn says the association is made through a "primitive operation which is pre-attentively performed on the visual display" (Pylyshyn & Storm 1988). This suggests that there is a process examining the visual field for properties which can be detected in a pre-attentive fashion. However, it is unlikely that all the pre-attentive properties which humans can detect are being sought at all times. Rather, we only detect the features that are important to our current task. Given this task dependence, it is likely that the current task can dictate not only which features one expects to find, but where one might expect to find them.

In order to facilitate non-local/local space integration, We define the term "expectant". An expectant defines a set of objects which match some description of an object. The more specific the description, the fewer objects will be in the expectant set. For example, suppose my task was to tighten a screw. I know that my screwdrivers are in the basement on the workbench. There may be multiple screwdrivers on the workbench and each of these can match the expectant description. The expectant describes the object I need, a screwdriver, and where I may find that object, on the workbench. The difference between a marker and an expectant is that a marker is associated with a specific object, while an expectant describes a set of objects.

Transferring Non-Local-Space Information to a PA System

Communication between our system component possessing non-local-space (NLS) information and our perception/action component occurs via uninstantiated markers. "Uninstantiated" reflects the fact that the marker is associated with an expectant and hence a single object has not yet been associated with the marker. The NLS component can pass uninstantiated markers to a PA system for instantiation. The PA system must detect and select a single object from the set described by the expectant and associate it with the marker. Making this association means the marker is now instantiated. So, it is now within the local-space representation of the PA system, where it can be used effectively.

Consider an agent whose task is to get a cup of coffee. One of the steps in this plan is to get a coffee mug. The coffee mugs are in the cabinet above the coffee maker, on the bottom shelf. When the cabinet door is closed, the agent cannot perceive any mugs which may be in the cabinet. The NLS component passes an uninstantiated 'mug' marker to the PA system. The 'where' slot of this marker tells the PA system that the mug will be inside the cabinet, on the bottom shelf. The PA system can open the cabinet (whether this emerges from some behavior in the PA system or is initiated by the TE passing in a 'cabinet door' marker and telling the PA system to execute an open action on it, will be determined by the PA system's basic capabilities). At this point, the PA system can instantiate the 'mug' marker. This deictic representation can be used to effectively manipulate the mug in later stages of the plan, such as pour-coffee-into-mug.

It should be noted that the PA system is not limited to representing only those objects indicated by the NLS component. The PA system may have its own routines actively looking for task dependent features. Representation acquired and maintained by the PA system is referred to as local-space representation.

Architecture

In the proceeding discussion about our theory of action and how it leads to a form of representation, we discussed an agent whose control system consists of a perception/action system and some other system component possessing non-local-space information. A diagram of this general architecture appears in figure 1. Figure 2 shows a more structured architecture, which our system uses. Our architecture consists of 3 layers, the planning module, the task executor, and the perception/action system. The job of the planning module is to formulate high-level goals for the agent and create plans. The task executor sequences these plans to the perception/action system as necessary. The perception/action system controls the agent's effectors. This layer performs all actions and forms all percepts from the environment.

In the architecture of figure 2, the means of integration between the NLS component and the PA component is the Task Executor (TE). The planning module has access to non-local-space representations which it can incorporate into the plans it generates. The TE takes plans and creates uninstantiated markers for each object represented in the plan. These uninstantiated markers are passed to the PA system for instantiation when the appropriate step in the plan is reached.

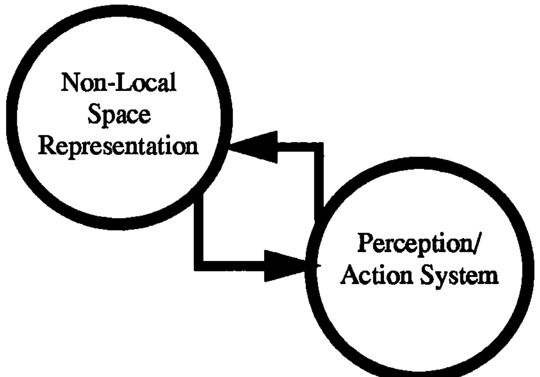

Figure 1. General Architecture

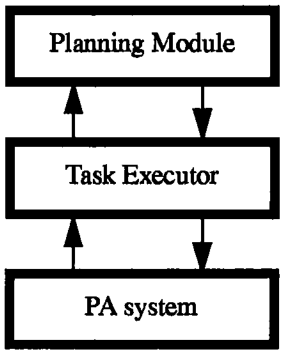

Figure 2. Our Architecture

The specific architecture of figure 2 is similar to a number of other 3 layered architectures, such as (Bonasso & Kortenkamp 1995)(Connell 1992)(Gat 1992), and is not the subject of this research.

Features of Integration

There are several important features of the form of representation that emerges from our theory of activity. First, our theory leads to the development of integrated agent control systems with a variable level of autonomy in their action sub-systems. The perception/action system is capable of taking actions based only on its own percepts and representation, i.e. without any information from other levels of the architecture. When there is little control information flowing from the TE to the PA, the PA is more autonomous. In fact, the PA system is capable of completely autonomous operation if communication with the TE is completely cut off. The system is then reduced to a Brill-like system. This form of graceful degradation was one of Brook's original goals for subsumption.

For tasks in which it is necessary for the TE to more tightly control the actions of the PA system, the TE can send more control information via expectants. Some situations require decisions that are more correctly made using non-local-space information (and therefore more correctly made outside the PA system). Deciding which way to go at a T-junction is such a situation. So, if the agent's plan was to navigate to the water fountain which is to the left at a T-junction, when the agent arrived at the T-junction, the TE would pass an uninstantiated marker corresponding to the water fountain to the PA system. The 'where' slot of this marker would indicate that the water fountain is to the agent's left.

In order for the agent to be as robust as possible, the amount of control exercised over the PA system should be minimized. This is because of the belief that the information which the PA system acquires is the most up-to-date and therefore the most believable. The non-local-space information used by the planning module is potentially stale. The TE communicates expectants to the PA system as necessary, to allow the PA system to operate more autonomously when this is appropriate.

Another feature of this integration methodology is the ability to use multiple maps of differing scale. Planning becomes more complex for an agent operating in a large environment. As the number of features in the environment grows, so does the complexity of the planning task. Route finding and feature selection may well become intractable for even moderately large areas. Consider an agent driving across the country. While on the interstate, many details about the various localities are not needed. Having multiple maps of differing scale allows unimportant details to be abstracted away from the planning process.

There are also situations when having access to two different scale representations of the same area is useful. Consider an agent which is walking through a university campus in the rain. In order to stay dry, the agent enters a near-by building. Now, the agent still wishes to reach its original destination, but in order to stay dry, the agent wants to follow a route which stays inside buildings as much as possible. The agent can use its coarse scale map of the university to form the context, i.e. the entrance and exit, for the planning process which selects a route through a given building from the agent's fine scale map of that building.

Implementation

In order to decide what to do, the PA system accesses its percepts and looks at its markers. These stimuli may trigger multiple possible behaviors and the PA system must prioritize its responses. Practical implementation of this system requires an expanded view of markers, from that of earlier marker systems (Agre & Chapman 1987)(Brill 1995). In addition to the 'what' and 'where' slots, markers have 'appearance', 'action' and 'action support' slots.

Appearance

Appearance denotes the sensor signature characteristics of the object. This involves two, not necessarily distinct, routines for initially locating the object and for tracking the object once it is marked. The initial location routine can be a visual routine which the agent begins executing when its PA system receives an expectant. After the marker is instantiated, the PA system can begin using the tracking routine to keep the marker's position up to date.

Action

The 'action' slot is what the system wants to do with the marked object. This is probably why the system marked the object in the first place. The action can be any which the system knows about for a particular object, i.e. for a particular entry in the marker's 'what' slot. The possible actions will be determined by the capabilities of the system. Some actions will be common to many objects. Examples of common 'actions' include, go-to and get. Examples of actions specific to a particular object could be 'pour' and 'fill' for a pitcher. The agent may have a database of action routines which the PA system can access.

Progress. The notion of task progress is essential to the agent's ability to determine when an action is complete. Obviously, the measure of progress will vary from task to task. Each action which the system knows about can have its own progress routine.

The progress may also feed back into the action routine. For example, when pulling a car up to a stop sign, the closer the vehicle gets to the sign, the more strongly the breaks should be applied.

While the PA system can execute the progress routine to determine how close to completion it is on some task, it may not be possible to resolve a lack of progress at the PA system level. The PA system may be able to detect that a given action is failing, but may have no idea why or what else to try. At this point, it must appeal to higher layers of the architecture for guidance.

Action Support

'Action support' holds information about the action which the perception/action system does not possess. This could be parameters to the action's progress routine or a pointer to another marker marking an object to be "operated" on. There may also be information applicable to the particular instance of a task which the system is about to perform. For example, if the agent wanted to fill a glass half full of water from a pitcher, the pitcher would be marked with a marker whose 'action' slot contained 'pour' and whose 'action support' slot contained a reference to the glass marker. The task also specifies a stop condition for the pour action, i.e. when the glass is half full, which can be checked by the progress routine. This condition also appears in the 'action support' slot of the pitcher marker because it is the pour action that must be stopped. The pour-progress function must be applied to the object marked by the marker in the 'action support' slot, i.e. the glass. Another example of a parameter to a progress routine might be for an open-door task where the door just has a handle which can be pulled or pushed. If the agent knows this particular door, it can notify the progress routine that it should expect a push motion to cause the door to open from the side the agent is on. If the progress routine reports a lack of progress, the agent can try pulling the handle.

System Platform

We are currently developing a system based on this theory of action. The platform is a mobile robot based on a MC6811 CPU. The robot, Bruce, has a single camera on a pan/tilt platform for sensing. The perception/action system runs on a workstation sending commands to the robot's effectors over a wireless data-link. More details and examples of the system in action are available in (Brill, Wasson & Martin 1996).

Conclusions

We have discussed a theory of action which identifies actions by associating them with the objects involved. This theory encompasses a representational system and an architectural organizing principle. Actions are performed by acquiring a deictic representation of the necessary objects and then performing the action. This theory leads to a system of representation which can express task dependent local-space information and be used to integrate perception/action systems and non-local-space information, via expectants.

We have described our minimalist deictic representation system, markers. The marker interface is minimalist in both the number and size of markers. Uninstantiated and instantiated markers facilitate communication between a perception/action system and a system component containing non-local-space information. These two states of markers allow non-local-space information to be placed into the local-space representation for a perception/action system.

The theory of action presented in this paper can be used to create systems which can operate robustly in large scale domains. Variable levels of PA system autonomy allow rapid execution of primitive actions when the environment is sufficiently dynamic. Multiple maps of possibly differing scale can be used to abstract unnecessary details away from the planning process. Our communication between system components can express non-local-space information at any scale.

We believe that designing agents in accordance with our theory of action will produce systems which are both sufficient and effective for operation in large dynamic environments.

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*, Vol. 90, No. 4, 549-563.

[3] Brill, F.Z. 1995. Representation of Local Space in Perception/Action Systems: Behaving Appropriately in Difficult Situations. Ph.D. Dissertation, Department of Computer Science, University of Virginia.

[4] Brill, F.Z.; Wasson, G.S.; and Martin W.N. 1996. Sensing and Representing Dynamic, Three-Dimensional Environments: Judicious Use of Representation Measurably Improves Performance, *IROS-96*, submitted.

[5] Brooks, R.A. 1986. A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation*, RA-2(1):14-23.

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

[7] Connell, J.H. 1992. SSS: A Hybrid Architecture Applied to Robot Navigation, In *Proceedings of the 1992 IEEE Conference on Robotics and Automation*, Nice, France, 2719-2724.

[8] Firby, R.J. 1987. An Investigation into Reactive Planning in Complex Domains, *AAAI-87*, 202-206.

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

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

[11] Russell, S.J., and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*, Prentice Hall, Englewood Cliffs, New Jersey.

[12] Tate, A.; Hendler, J.; and Drummond, M. 1990. A Review of AI Planning Techniques, *Readings in Planning*, Allen, Hendler and Tate ed., Morgan Kaufman.

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