Integrating Cognition, Perception and Action through Mental Simulation in Robots

Nicholas L. Cassimatis, J. Gregory Trafton, Alan C. Schultz, Magdalena D. Bugajska

Naval Research Laboratory
Codes 5513, 5515
4555 Overlook Avenue SW
Washington, DC 20375-5337
{cassimatis, trafton}@itd.nrl.navy.mil, {schultz, magda}@aic.nrl.navy.mil

Abstract

We propose that many problems in robotics arise from the difficulty of integrating multiple representation and inference techniques. These include problems involved in planning and reasoning using noisy sensor information from a changing world, symbol grounding and data fusion. We describe an architecture that integrates multiple reasoning, planning, sensation and mobility techniques by composing them from strategies of managing mental simulations. Since simulations are conducted by modules that include high-level artificial intelligence representation techniques as well as robotic techniques for sensation and reactive mobility, cognition, perception and action are continually integrated. Our work designing a robot based on this architecture demonstrates that high-level cognition can make robot behavior more intelligent and flexible and improve human-robot interaction.

Many problems in robotics are fundamentally integration problems.

We propose that many problems in designing intelligent autonomous robots arise from the difficulty of combining multiple representation and inference techniques into one robot that can construct and maintain rich, coherent and dynamic models of its environment. These include the problems involved in data fusion, symbol grounding and flexibly combining reasoning, planning, perception and action.

The data fusion problem involves integrating information from multiple sensors into a coherent model of the environment. For example, when sensor A in a robot detects an object and sensor B in the same robot detects an object, the robot must determine whether the two sensors are detecting the same or different objects. Since each class of sensor information has its own best representation, sensor fusion is in part a problem of combining multiple representation schemes. Since the decision about whether information from two sensors is about the same object can depend on world knowledge (e.g., that objects of a particular category are too slow to have moved from sensor A’s range to sensor B’s range in so short a time) that is often best represented using artificial intelligence reasoning techniques, robust sensor fusion also requires the integration of sensor information with traditional AI representations.

Some researchers (e.g., Harnad, 1990) have had difficulty understanding how symbols manipulated by traditional artificial intelligence algorithms can have any relation to objects, events and relations in the environment that are perceived through sensors yielding data in very different representations. This “symbol grounding” problem is fundamentally a problem of integrating different sorts of (“perceptual” and “symbolic”) representations.

A third problem in robotics involves the difficulty of using traditional artificial intelligence reasoning techniques to plan a robot's behavior using noisy information from a changing world that can invalidate a plan during the interval between its formulation and execution. Although one would like every step of a robot's reasoning and planning to be able to adapt to changing sensor information, the most effective reasoning and planning techniques use data structures and algorithms that are very different from and not obviously reconcilable with the most effective perceptual and motor techniques. It is not at all clear, for example, how to tightly integrate a STRIPS (Fikes & Nilson, 1971) planner with a Bayesian object localizer and a reactive obstacle avoidance system.

Behavior-based and reactive (e.g., Brooks, 1991 and Agre & Chapman, 1990) robotics researchers have reacted to this difficulty by dispensing with algorithms that are based on constructing and manipulating internal representations that are not closely (in time and space) tied to what is being immediately sensed. Since there is no "old" information and the robot never needs to pause to perform computations on internal representations, these systems appear to have the virtue of always taking the most relevant action given the current information. A major problem with these approaches is that so much of what determines the correct action for a robot to take involves the environment in past or future, occluded, spatially distant and/or hypothetical situations that cannot be immediately sensed. For example, the future consequences of an action (i.e., the future state of the hypothetical world in which that action is taken) help determine whether a robot should take that action. For tasks or environments with all but the most minimal complexity, one cannot anticipate all possible classes of sensor readings and precompile an appropriate reaction for each situation. Thus, representation-free approaches are severely limited by the complexity of the environments they can deal with and the tasks they can achieve.

Since so many problems in robotics involve the integration of multiple representation and inference techniques, we have developed a robotic architecture that supports the combination of these techniques. The architecture, which we call Polybot, is based on the Polyscheme cognitive architecture (Cassimatis, 2002) for solving integration problems in artificial intelligence and cognitive science generally. Polyscheme differs from traditional cognitive architectures based on one or a few data structures by enabling inference based on multiple data structures. Polyscheme differs from the many multi-agent system architectures that encapsulate specialized algorithms in modules by enabling every step of every algorithm to be executed using multiple representations and be potentially assisted by every other algorithm.

The next two sections explain how Polybot composes reasoning and planning techniques from cognitive science and artificial intelligence research by sequences of mental simulations. Since Polybot enables these sequences of simulations to be interleaved and since a particular simulation can be part of the execution of more than one algorithm, multiple reasoning and planning algorithms are easily integrated. Since each simulation is conducted using multiple representations, spanning the continuum from low-level perception and mobility techniques to higher-level knowledge representation schemes, reasoning and planning in Polybot are continuously integrated with perception, action and multiple knowledge representation schemes. The final section gives a brief overview of the initial results we have achieved with this approach in object tracking and human-robot interaction tasks.

An architecture for integrating multiple representation and inference techniques through the control of mental simulations

Polyscheme is motivated by the view that the difficulty of achieving the benefits of both high-level reasoning and planning and behavior-based and reactive systems is a problem of combining different representation and inference techniques into one system. In using Polyscheme to design Polybot, we aimed to create an architecture that could engage in reasoning and planning that is at every step responsive and adaptive to information from (potentially noisy) sensors and changes in the world.

Polyscheme's fundamental approach to this problem is to implement and execute reasoning and planning algorithms using mental simulations that are based on perceptual and reactive representations as well as traditional artificial intelligence representations. Polyscheme takes "reactive" components that typically choose their actions only with reference to the currently sensed state of the world, and allows them to "react to" represented or simulated states of the world. Using strategies for choosing which simulations to run (i.e., what time, place or hypothetical world to simulate), Polyscheme implements high-level AI algorithms by executing mental simulations. Because simulations are composed in part of reactive and perceptual subcomponents, reasoning is constantly and thoroughly integrated with perception and action. We now describe how these ideas are realized in Polybot.

Polyscheme encapsulates mobility and perception techniques in specialists.

Polyscheme encapsulates the functionality of robot perception and mobility techniques in modules called *specialists*. Because our experience has shown that multiple techniques for mobility and perception are useful, specialists may use any algorithm or data structure to implement their functionality. For example, Polybot includes a specialist that identifies the location and category of objects using color segmentation (Rehrmann & Priese, 1998). All of the inferences and actions in Polyscheme are executed by specialists. The rest of the Polyscheme architecture is aimed at coordinating and sharing information among specialists.

Specialists communicate using a representation-neutral language based on a common ontology.

Specialists must share information in order to perform their functions. For example, a mobility specialist will need information from a perception specialist in order to avoid obstacles. In order for specialists to be able to communicate such information, all Polyscheme specialists use the same common language to communicate information. Because the language is only used for communication and not computation, we focused on a simple, though expressive, propositional language. Specialists in Polyscheme must be able to translate between their internal data structures to this propositional language. In order for these specialists to communicate, the language must adhere to a standard known to all specialists. For example, an object recognition specialist indicates that a cone is at location `p` at time, `now`, with the propositions `Location(o,p,now)`, `xOf(p,3.4,now)`, `yOf(p,4.2,now)`, and `Category(o,Cone,now)`. If other specialists are to use this information (e.g., a mobility specialist that knows to avoid cones), then they must share predicates such as `Location`, `xOf`, `yOf` and `Category` and understand the units of distance and time used in their arguments. Specialists are not committed to using these units and these predicates in their own internal computations or even using representations based on predicates on objects. They must simply be able to *translate* between this representation and their own *internal* representation. By separating the representation used for communication from the representation used for inference, we achieve the benefits of a common ontology without the rigidity often associated with such knowledge representation schemes.

Specialists implement a common set of functions used to share information.

Specialists each implement a common set of functions that Polyscheme uses to coordinate information flow and behavior. By standardizing this set of functions and by using the common propositional language described in the last subsection, specialists do not need to take into account the implementation details of other specialists and Polyscheme can be extended with new specialists more easily. These functions are as follows:

- `ReportOpinion(prop,otherSpecialist,tv)`. A specialist learns that `otherSpecialist` believes that the proposition, `prop`, has truth value `tv`.
- `StanceOn(prop)`. Returns the truth value the specialist believes `prop` has.
- `RequestedFoci()`. Returns a set of propositions that the specialist would like to focus on. These include, but are not limited to, propositions that the specialists wants to assert as being true or false and subgoals (i.e. propositions whose truth values would help the specialist take a more accurate stance on another proposition.)
- `Groundings( prop )`. Returns a set of closed propositions that ground the open (i.e., its arguments have open variables) proposition, `prop`.

Specialists must alert the system to changes in their beliefs.

Because robotic sensors are noisy and because they generally yield incomplete information about the environment (because of limited sensor ranges and occluders), Polyscheme's specialists will at least occasionally change their stance on a proposition. When this occurs, it is important that other specialists are notified of this so that they do not continue to make inferences and take actions based on bad information. A specialist's `RequestedFoci()` function must therefore include among its assertions stances on propositions on which it has revised its beliefs.

Specialists must be able simulate non-immediate states.

Since robots' actions and inferences often depend on past, future, distant and/or hypothetical situations, specialists in Polyscheme must be able to react not just to the immediately sensed state of the world, but some representation (described in the next subsection) of past, future, distant, invisible or hypothetical states. We call any state, event, region, or situation that is not currently sensed (because it is distant, occluded, hypothetical at another time and/or at another place) *non-immediate*. To represent non-immediate situations, Polyscheme's ontology includes temporal intervals and hypothetical worlds. The last two arguments of every proposition input to and output from specialists are, respectively, a time and world argument. For example, the proposition that object, `o`, is located at point `p`, in hypothetical world, `w`, at time `t` is indicated: `Location(o,p,t,w)`. In the next subsection we discuss how Polyscheme represents non-immediate states.

Specialists simulate non-immediate states using multiple representations.

If specialists must react to non-immediate states of the world, then there must be some representation of those states that is different from the current state of the robot's sensors. There must be a memory for past events, properties of objects and relations between them; there must be a way to represent future and hypothetical states so their desirability can be evaluated and it must be possible to represent distant or occluded parts of the environment. Because different representational techniques are most appropriate for different aspects of the world (for example temporal constraint graphs for temporal intervals, spatial maps for object locations) Polybot includes multiple specialists that encapsulate representations for different aspects of the world. Representations that have already been implemented include spatial maps, temporal constraint graphs, scripts and directed graphs.

Given that robots must represent non-immediate states, how do they decide what the truth is about those states? There are several mechanisms for accomplishing this. Memory is a mechanism for deciding what was true in the past. Causal rules, constraint satisfaction algorithms, and dynamic simulation can predict what is true in future and/or hypothetical states. They can also decide what is happening in occluded regions of the environment, for example when the causal rule specialist predicts that in the absence of an obstacle or external force, a ball rolling behind an occluding object will continue to exist and roll behind the occluder even though the ball and its motion are not detected at that moment by the robot's sensors. We use Minsky's (1986) term *simulus* to refer to the input to specialists from a simulated world.

Although inferential specialists use high-level AI algorithms, the way they interface with the rest of the system has a reactive character because they implement the same specialist functions as reactive specialists. For example, Polyscheme's category hierarchy specialist learns new information about an object through its `ReportOpinions()` function, "reacts" to it by using its own internal data structures and algorithms to make inferences about the category membership of the object, and uses the `RequestedFoci()` function inform other specialists of these.

Figure 1 illustrates how several "high-level" artificial intelligence algorithms take on a reactive character when encapsulated in specialists.

Simulus	**Representation/ Algorithm**	**Action**
Sensors	[None for reactive systems]	Action
Cause	Causal Rules / Rule matching	Assert effect
Input layers	Neural Networks / Network propagation	Output layer
Object category information	Category hierarchies/ Graph walking	Assert other categories the object belongs to
Two temporal intervals	Temporal intervals / Constraint propagation	Assert the possible constraints between the two intervals

Figure 1. Reactions to simuli in multiple representations.

Specialists must focus their attention.

All specialists in Polyscheme react to the same proposition at the same time. There are two reasons for this. First, a surprisingly large number of specialists are relevant to any particular proposition. For example, when inferring whether a falling object will continue to fall, a causal specialist will predict that it will do so if the region underneath the object is empty, the perceptual specialist might be able to see whether there is a supporting object, the object location specialist might be able to remember if there was a support, etc. Second, if a specialist acts on an inferred or perceived proposition before checking other specialists' stances on it, the specialist might be acting on incorrect information that can lead to harmful mistakes or at best require a time-consuming retraction of incorrect actions and inferences.

For these reasons, at every time step in Polyscheme, specialists focus on the same proposition. Once a proposition, `P`, is chosen (as described in the next section), the following sequence occurs:

- Polyscheme calls `StanceOn(P)` for each specialist to determine the consensus truth value of all the specialists on `P`.
- For each specialist, Polyscheme calls the function `ReportOpinion(P,specialist,tv)` to report specialists' truth values for `P` to each other.
- Polyscheme calls `RequestedFoci()` to get propositions the specialists would like to focus on soon.

The chief form of communication among specialists in Polyscheme is through the focus of attention. Through functions such as `StanceOn()`, `ReportOpinions()`, `Groundings()` and `RequestedFoci()`, specialists communicate information to and request information from each other. When a proposition becomes the focus of attention, each specialist learns the other specialists' opinions of its truth and has an opportunity to ask questions that flow from or assert propositions that follow from the focal proposition.

Specialists must quickly react to the focus.

Because the environment can change quickly or because new sensor information may become available at any moment, specialists must quickly execute functions such as `StanceOn()` and `ReportOpinion()` so that the specialist can constantly focus on and make inferences with the newest information. This is an important component of Polyscheme's solution to the problem of inferences and plans that are invalidated before they are completed or executed.

Simulations result from the focused attention of specialists.

One result of the architectural principles discussed in this section is that Polyscheme's specialists collectively perform a kind of simulation. When Polyscheme focuses on a particular time and world, the architecture forces all specialists to focus on that time and world. The sum effect of this attention will be that specialists will make inferences about that world and therefore elaborate Polyscheme's representation of it. Since some of these inferences will involve the consequences of states and events in this world, Polyscheme will focus on subsequent times in the same world. The result is that specialists will perform a dynamic simulation of that world.

Focus schemes guide simulation.

Nonrepresentational robotic systems must constantly (implicitly or explicitly) make a choice, "Where do I look now?", because sensors are inherently directed towards a local region of space. Even sensors which uniformly monitor the region surrounding a robot can be directed to another region by the robot's motion. For Polybot, the number of choices is even greater because specialists can focus on many possible simulated worlds in addition to the immediate world itself.

As indicated in the previous subsection, when Polyscheme specialists focus on a proposition, they ask for a set of propositions to focus on through their

RequestedFoci() function. Polyscheme's *focus manager* chooses from these requests (based on their level of urgency (as indicated by the specialist) and several other factors). If the proposition is open or "ungrounded", i.e., if it contains an open variable, the focus manager chooses a proposition that grounds the proposition. Once the focal proposition is chosen, Polyscheme calls the various specialist functions as indicated in the last subsection.

Polyscheme thus continuously chooses a proposition to focus on, allows specialists to communicate about and make inferences about this proposition and then chooses the next proposition to focus on based on specialists' requests. Through their ability to request Polybot to focus on a proposition, specialists can influence the flow of attention and hence computation. We call different strategies for guiding attention *focus schemes*. We have already encountered one focus scheme implemented by all specialists:

> *Resimulation focus scheme*. When a specialist infers that a proposition P has a truth value that is the opposite of the truth value it returned during the last call of StanceOn(P), include P in the return set of RequestedFoci()the next time that function is called. Less formally, when a specialist changes its stance on P, it should request that P be focused on again.

Another example is the prediction focus scheme. It is tells Polybot to simulate the results of an action before executing it:

> *Prediction Focus Scheme*. When the motor specialist is about to take an action, A, simulate Occurs(A,t,w) where t is the next time step and w is the hypothetical world in which A is taken at time t.

When the system focuses on Occurs(A,t,w), all of the specialists in the system will infer what else is true in that world, i.e., what the consequences are of the action A, and request that these consequences be focused on through their RequestedFoci() function. If w is a world that contains damage or harm, the motion specialist will not execute A.

Algorithms are implemented by strategies for choosing simulations.

The most fundamental point of this paper is that many "high-level" artificial intelligence algorithms can be implemented by focus schemes for choosing simulations that Polyscheme's specialists execute. We illustrate this by showing how to implement backtracking search with the counterfactual simulation focus scheme.

> *Counterfactual Simulation*. When uncertain about A's truth value (because of a lack of information or because of conflicting information), simulate the world in which A is true and the world in which A is false.

If when simulating one of these worlds, say where A is true, one of the specialists infers a fact that contradicts what is already known for certain, then the world where A is true is contradictory and hence A can be inferred to be false in the real world.

Consider the case where Polybot is uncertain of two propositions, A and B. In the simulated world in which A is true, there is still uncertainty about B. Thus the counterfactual simulation focus scheme still applies and imagines the world in which A and B are true and the world in which A and not-B are true. When one of these leads to a contradiction, that world is not simulated any longer. Thus, the counterfactual simulating focus scheme can lead to nested simulations which effectively implement backtracking search.

Algorithm	**Focus scheme that implements it**
Case-based reasoning	Memory-based simulations
Prediction	Forward simulation
Counterfactual reasoning	Counterfactual simulation
Backtracking search	(nested) Counterfactual simulation.
Backward chaining/Theorem proving	Antecedent simulation
Truth maintenance	Resimulation
Bayesian inference	Stochastic simulation

Figure 2. Inference algorithms and the focus schemes simulations that implement them.

Figure 2 lists several focus schemes that implement important artificial intelligence algorithms. The resimulation focus scheme of the last section implements a form of truth maintenance since the result of focusing on a proposition whose truth value has changed will be to resimulate events and states the proposition relates to and hence change incorrect inferences based on the initial false belief. Cassimatis (2003) has shown that the *antecedent simulation* (roughly, "when P implies Q and you want to know Q, simulate the world where P is true") implements a form of resolution theorem proving. The *stochastic simulation* focus scheme ("when P has probability *m*/(*m*+*n*), simulate the world where P is true *m* times and the world where P is false *n* times) implements an approximate form of Bayesian inference that has been used widely by the uncertain reasoning community. Finally, *memory-based simulation* ("when G is a goal, simulate in the current situation actions you have previously taken that have achieved goals similar to G") implements a form of case-based reasoning.

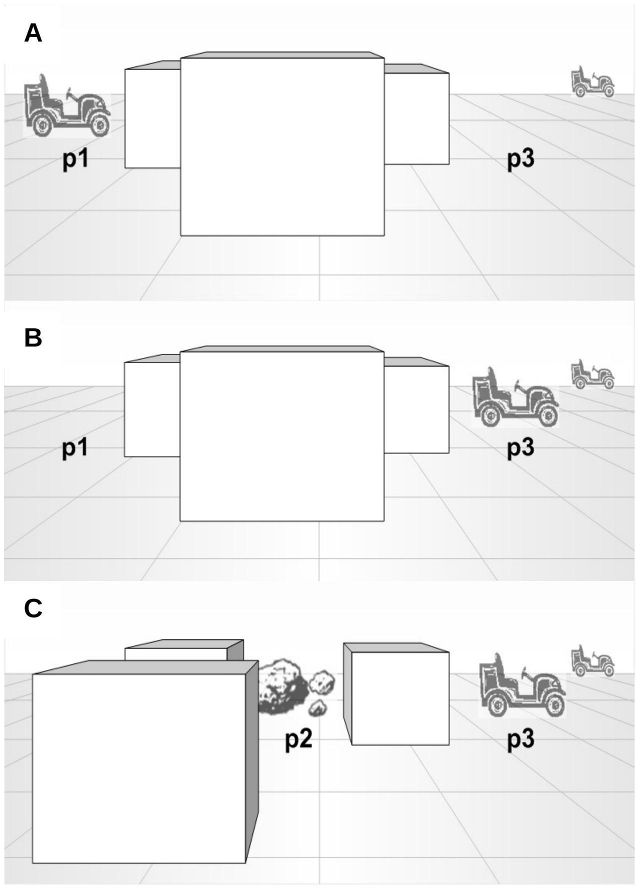

Figure 3. Polybot's view of a scenario in which it adapts a plan to new sensor information while it is being executed.

Combining mental simulations resolves many integration issues in robotics.

Now that we have described Polyscheme's approach to supporting multiple representation and inference schemes, we discuss how this helps resolve many problems in robotics. We do so by presenting an example, illustrated in Figure 3, which will illustrate several of this paper's themes.

An extended example

In 3A, Polybot observes one of two visible carts roll behind some boxes. Polybot's task is to go to the cart. In 3B, Polybot sees a cart move out from behind the boxes and assumes that it is the cart it is tracking. As Polybot moves towards the cart in 3C, it notices that behind the boxes there is an obstacle that the cart could not have moved through and therefore assumes that the cart it sees is in fact not the cart it is tracking and moves towards the last place it saw the cart go. Figure 4 traces Polybot's focus during this scenario.

The following subsections use this example to illustrate how using Polyscheme to combine multiple representation and inference techniques helps resolve many of the problems surrounding building flexible robots that can engage in high-level reasoning.

Simulations are a medium for integrating multiple representations.

The example shows that sharing information between different representations is fairly straightforward in Polyscheme. Each specialist has its own representation and can translate back and forth between it and the representation-neutral language used to encode the focus. For example, during step 9, the perception specialist perceives that `p2` is flat and encodes that in its own perceptual representation, returns `true` as a stance and the other specialists learn of this through their `ReportOpinion()` function. One of these specialists is the causal specialist which encodes that `p2` is flat in its causal rule language. Thus, through the focus of attention and the representation-neutral language, specialists can share information easily.

In this context, symbol grounding is less puzzling. The causal rule specialist can manipulate symbols representing flatness without fear of losing touch with "physical reality" because each simulation it performs combines (through the focus of attention) information from sensors and information from the rule specialist. This is made possible because of the representation-neutral language the specialists share.

Combining simulations combines algorithms.

In this example, Polybot executes three different algorithms: prediction, backtracking (path) search and truth maintenance. Each is composed of several foci. Prediction is composed of foci 5-7 and 11-13, back tracking search is composed of 6, 9 and 10 and truth maintenance is composed of 9-13. Note that each algorithm's foci overlap with the others. This overlap or sharing of foci is a key to combination of algorithms in Polyscheme. When, for example, Polybot focuses on `Flat(p2,E,R)` in step 9, the specialists' inferences about whether that proposition is true are shared by both the search and truth-maintenance algorithms. In general, since algorithms implemented by focus schemes are merely composed of foci, the data they operate on resides in the specialists which make inferences on the foci. Since foci can be shared by any algorithm, sharing information between these algorithms is simple.

	Focus	Simulation	Algorithm	Explanation
1	…	Immediate		Perception specialist observes the cart move from behind the boxes (`Location(cart1,p1,t1,R)` and `Category(cart1,Cart,t1,R)`) and then observes a cart come out from behind the boxes (`Location(cart2,p3,t3,R)` and `Category(cart2,Cart,t3,R)`).
2	`Category (cart2,cart, t1,R)`	Immediate		Identity hypothesis specialist's neural network infers that `Same(cart1,cart2,E,R)` and requests focus for that proposition.
3	`Same (cart1,cart2, E,R)`	Immediate		The difference specialist detects a difference in the location of tracked cart, infers an change event and requests for a focus on `Exists(d,E,R)`, `Category(d,ChangeEvent,E,R)`, etc.
4	`Category (d,Change,E,R)`	Recent	Prediction	The causal rule specialist infers that since `p1` and `p3` are not adjacent points, there must be an intermediate point, `P`, that cart1 visited and that `P` must be flat because carts cannot roll over non-flat surfaces.
5	`Flat(P,t2,R)`	Recent	Prediction	The space specialist infers that there `P` might be the intermediate point and requests for a focus on `Same(p2,P,w)`, where `w` is the world where `p2` and `P` are the same.
6	`Same(p2,P,E,R)`	Immediate Hypothetical	Prediction, Search	The tracking specialist infers that since the location `cart1` is now at `P` that the system should move there.
7	…	Future Hypothetical	Prediction	The prediction focus scheme simulates that motion and finds no problems. (This takes several steps.)
8	…	Immediate		While moving towards that location, the perception specialist sees `p2` for the first time and sees that is flat.
9	`Flat(P,t2,R)`	Current Hypothetical	Search, Truth maintenance	The difference specialist notices the difference between `Flat(P)` and not `Flat(P2)`. Since `p2` is certainly flat, and `P` is certainly not flat, then the difference specialist assumes that in fact `Same(p,p2,E,R)` is false and requests focus for that proposition because of the resimulation focus scheme.
10	`Same(p2,P,E,w)`	Hypothetical	Truth maintenance, Search	The space specialist cannot find any other places that `P` might be equal to and thus assumes it does not exist and request focus for `Exist(P,E,R)`.
11	`Exists(P,E,R)`	Immediate	Truth maintenance, Prediction	Since no intermediate point exists, the causal rules specialist retracts the existence of the event that implies it and request focus on it because of resimulation.
12	`Exists(d,E,R)`	Past	Truth maintenance, Prediction	Since delta is retracted, the identity of `Same(cart1,cart2,t2,E,w)` is retracted
13	`Same (cart1,cart2, E,R)`	Immediate	Truth maintenance, Prediction	Thus, `cart1` and `cart2` are different and cart1 is still behind the box on the left.
14	…	Immediate		Motion specialist initiates movement in that direction.

Figure 4. A trace of Polybot's focus.

Simulations integrate reasoning, planning, perception and action.

Inference algorithms are composed of sequences of simulations executed by specialists. Since these specialists include specialists for perception and mobility, the combination of high-level and low-level computational techniques is constant in Polybot. For instance, the focus in step 9 of the example is part of a path search. The focus, `Flat(p2,E,R)`, is perceived to be false and modifies the course of the path search. This is a simple example of a system's sensors being able to influence the course of a high-level artificial intelligence algorithm as it is being executed.

Reasoning and planning with information from noisy sensors in a dynamic world.

Three features of this approach greatly reduce the tension between the flexibility of representation-free, reactive systems and the power of high-level artificial intelligence algorithms. First, since algorithms are composed by simulations executed by specialists and because these specialists include perceptual specialists, every step of inference is always being checked against new sensor information. Thus, revisions in sensor readings or changes in the world will be detected immediately. Second, because each specialist is obligated to immediately broadcast these changes to the rest of the specialists as soon as they occur, inference can be adjusted immediately. Finally, because algorithms are composed of reactions that are required to follow quickly from simuli, there is no long lag between the formulation and execution of a plan during which the plan can become invalidated by changes in the world.

In the example above, as soon as Polybot's initial assumption that P is flat is seen to be false in step 9, that change is broadcast to the rest of the system and search is immediately altered. The revision does not need to wait until the end of prediction and truth maintenance in step 13 for it to influence inference.

Results and Conclusion

Although Polybot is a relatively new project, we have been able to demonstrate many of the benefits discussed in this paper in object tracking and human-robot interaction tasks. This work has been based on several specialist we implemented, including those for perception (which uses CMVison), movement (a reactive planner), temporal constraints (Allen's (1983) temporal intervals), spatial location (cognitive maps), change events, causation (production rules), ontology (category heterarchy), uncertainty, perspective and object identity (neural networks).

In our object tracking work, Polybot has been able to improve upon a statistically-based object tracker by using common sense reasoning about what kinds of actions a particular category of object is capable of. Our existing vision system could only track an object when there was an overlap between its pixels on one image and those in the subsequent image. By incorporating in high-level reasoning about what kinds of paths objects could take (e.g., carts could only roll over a flat surface whereas people would move over rougher terrain) we were able to track objects that became occluded over longer distances and periods of time.

In our human-robot interaction work, Polybot has been able to improve upon previous work on an object reference tasks by using its ability to simulate the world from a person's perspective in order to more accurately understand and predict his actions. We created a task in which a human would use language to refer to an object and it was the robot's task to go to that object. Since there were many instances of the same object (e.g., many cones) in the room and since humans could see objects that the robots could not, and *visa versa*, many of the human's utterances were ambiguous when taken literally. By using Polybot's simulation abilities, robots were able to take the perspective of the human in the task and significantly improve the accuracy of their understanding.

We believe that these results are a first step toward demonstrating that this approach towards integrating multiple algorithms and representations enables a significant reduction in the tension between using sophisticated reasoning and knowledge representation techniques in robots that can flexibly react to new sensor information and to changes in the environment.

Bibliography

P. Agre and D. Chapman 1990. "What are plans for?". *Journal for Robotics and Autonomous Systems*, vol. 6, 1990, pp. 17-34.

J. F. Allen 1983. *Maintaining knowledge about temporal intervals*. Communications of the ACM, 26(11):832–843, 1983.

R. A. Brooks 1991. Intelligence without Representation. *Artificial Intelligence*, Vol. 47, 1991, pp. 139-159.

N. L. Cassimatis 2002. *Polyscheme: A cognitive architecture for Integrating Multiple Representation and Inference Schemes.* Ph.D. diss, Media Laboratory, Massachusetts Institute of Technology.

N. L. Cassimatis 2003. A Framework for Answering Queries using Multiple Representation and Inference Techniques. In *Proceedings of the 10th International Workshop on Knowledge Representation meets Databases*, 2003.

R. Fikes and N. Nilson 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*. 2(3-4):189-208, 1971.

S. Harnad 1990. The Symbol Grounding Problem. *Physica* D 42: 335-346.

M. L. Minsky 1986. *The Society of Mind*. Simon and Schuster, New York, New York, 1986.

V. Rehrmann and L. Priese 1998. Fast and robust segmentation of natural color scenes. *Computer Vision - ACCV98* (R. Chin and T.-C. Pong, eds.), vol. I, (Hong Kong, China), pp. 598-606, January 1998.