Opportunity recognition in complex environments

Louise Pryor
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
Scotland
louisep@aisb.ed.ac.uk

Abstract

An agent operating in an unpredictable world must be able to take advantage of opportunities but cannot afford to perform a detailed analysis of the effects of every nuance of the current situation on its goals if it is to respond in a timely manner. This paper describes a filtering mechanism that enables the effective recognition of opportunities. The mechanism is based on a characterization of the world in terms of *reference features*, features that are both cheap and functional and that appear to be prevalent in everyday life. Its use enables the plan execution system PARETO to recognize types of opportunities that other systems cannot. Reference features can also play a rôle in the detection of threats, and may be involved in the development of expertise.

Introduction

The world is unpredictable—it is impossible to tell in advance exactly what its state will be at any future time—so plans on their own are not sufficient to govern the behavior of a goal-driven agent. For example, a robot roaming the surface of a strange planet will have to be able to decide for itself where it should take soil samples, what areas to explore and what routes to take. The samples that should be taken may depend *inter alia* on the terrain that is encountered, atmospheric conditions, and the results of tests on earlier samples, none of which can be predicted in detail. Moreover, the number of possibilities is huge: it would be impossible to construct contingency plans for all combinations of circumstances that could be encountered, even if it were possible to predict what they might be.

The traditional approach to AI planning assumes that plans are guaranteed to work and that nothing unexpected can happen during their execution. However, in many real-world domains the inevitability of the unexpected means that any plans that are made in advance will have to be changed during execution (Alterman 1986; Firby 1987; Hammond 1989; Pryor & Collins 1994). Expending a great deal of effort on constructing elaborate and detailed plans by trying to predict exactly what will happen is therefore often unproductive. A more effective approach is to choose simple plans and adapt them when unforeseen circumstances are encountered An agent following this approach must be able to determine when it should change its current plan. This determination is not trivial. Any aspect of the world may in principle affect any of the agent's goals: any change in the world may thus make it desirable to change plans. The agent cannot afford to analyze in detail every circumstance that it encounters if it is to respond appropriately.

This paper presents a filtering mechanism that can be used to decide when a detailed analysis of the agent's circumstances should be performed. The mechanism has been implemented in the plan execution system PARETO.[1] The mechanism used in PARETO is based on the observation that the world, although unpredictable in detail, is in essence very regular. It uses the fact that particular aspects of the world are often associated with the achievement or frustration of certain types of goals. For example, the presence of a sharp object is often implicated in the ability to achieve a goal of cutting something. The *sharp*ness of an object indicates its effect on goals whose achievement is affected if the object cuts something else. There are many concepts such as *sharp* that describe effects on goals: they form the basis of an effective method of opportunity recognition.[2]

The problem

An agent should change its plans when an unpredicted environmental factor affects the achievement of its goals. Goals can be affected in two ways: either opportunities are presented, or threats are posed. This paper concentrates on the recognition of opportunities; the principles involved are much the same for threats.

Opportunity recognition is complex in two ways. First, there is an enormous number of elements in every

[1] Planning and Acting in Realistic Environments by Thinking about Opportunities. The economist, sociologist and philosopher Vilfredo Pareto (1848–1923) is best known for the notion of *Pareto optimality* and for the *Pareto distribution*, neither of which is used in this work.

[2] This work is described in more detail in (Pryor 1994).

situation that no agent can possibly predict (Brand & Birnbaum 1990). None of these elements can be ruled out *a priori* as never being relevant to any goal. Suppose, for example, that you have a goal to open a can of paint. The objects in your garage include your car, the shelves on the wall, engine oil, *etc.* Few of these are relevant to your current goal, but they may be relevant to other goals at other times. Second, there are many subgoals involved in achieving a goal. For instance, to pry the lid off the paint can you must find something that is the right size and shape to act as a lever, a strong rigid surface on which you can rest the can at a convenient height, and so on.

The analysis of each situation element of a complex environment in the light of each of the agent's many goals would involve huge numbers of subgoals and situation elements and would preclude a timely response to unforeseen situations. Moreover, such an analysis would require the determination of each goal's subgoals, thus demanding the existence of a plan to achieve that goal. However, the recognition of an opportunity may trigger a radical change of plan or the construction of a plan for a hitherto unplanned-for goal. PARETO can recognize and take advantage of opportunities in these circumstances.

Related work

Most research on the issues of replanning during plan execution fails to address the issue of opportunism. The need to replan is usually determined by projecting the agent's current plans explicitly (Ferguson 1992; McDermott 1992); as projection may involve arbitrarily complex reasoning, the problem of the timely recognition of opportunities is not addressed.

Hayes-Roth and Hayes-Roth (1979) look at opportunism in plan construction but do not extend the concept to plan execution. Birnbaum and Collins (1984) present a theory of opportunism in plan execution based on the idea that opportunity recognition should be goal-driven rather than environment-driven. They suggest that each goal should be an active *mental agent*, performing the necessary reasoning to recognize opportunities to achieve itself. They do not address the issue of how agents can recognize opportunities with little reasoning: sharing the reasoning among the goals does not necessarily reduce the amount required.

Hammond *et al.* (1993) suggest that each goal should have associated with it the features in the environment that will be involved in achieving it. Such features might include tools and resources, locations at which the goal can be achieved, *etc.* The agent then recognizes an opportunity for the goal when the relevant feature appears in the environment. This approach relies on having specified the plan that will be used in enough detail that the relevant features are already known. It does not allow an agent to recognize opportunities that require a method of achievement other than that in the current plan or for goals that it has not yet decided how to achieve.

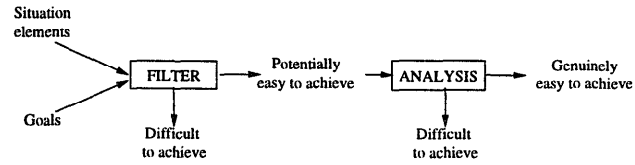

Figure 1: A filter for potential opportunities

The RAPS plan execution system recognizes opportunities only when their possibility has been represented in the plans being executed (Firby 1987; 1989). Reactive systems such as PENGI require the system designer to have encoded potential opportunities into the reaction rules (Agre & Chapman 1987).

PARETO's approach extends the range of opportunism by enabling the speedy recognition of opportunities for goals for which a plan has not yet been chosen and of opportunities whose possibility has not been explicitly foreseen.

A filter for opportunity recognition

PARETO uses a filtering mechanism that indicates those situations in which there are likely to be opportunities and that will therefore repay further analysis (Figure 1). It is based on two key observations: first that there is a large class of opportunities whose presence is indicated by a single critical factor; and second that the causal properties of situation elements, and thus their effects on goals, are in general predictable.

The critical factor hypothesis

Although an opportunity usually arises as the result of a number of factors, in many cases few of these factors by themselves indicate the presence of an opportunity. For example, the presence of a table does not usually constitute an opportunity to open a can of paint. There are many different ways in which the requirement of having somewhere to rest the can could be met, and it happens that the presence of the table is one of them. However, the presence of a screwdriver is more significant; there are comparatively few objects that are suitable for use as a lever. The *critical factor hypothesis* states that the presence of a single factor is often crucial for the existence of an opportunity.

The critical factor hypothesis relies on the observation that many situation elements are stable across many different situations. There is more stability at the functional level than at the purely physical: while there may not always be a table, there is usually something that you can use to support an object. It is thus usually easy to meet most of the preconditions for a given goal and the presence of situation elements that allow you to meet these preconditions does not greatly affect goal achievement. However, there is often one precondition that is more difficult to achieve, such as the availability of a lever that will fit under the lid

of the can. The presence of a situation element that enables the achievement of this precondition is then sufficient to indicate the presence of an opportunity.

The critical factor hypothesis appears to hold for a large number of opportunities and forms the basis of an effective filtering mechanism for recognizing opportunities. An agent can take the presence of a factor that is critical for one of its goals as an indication that the goal is worth analyzing in more detail.

Reference features

A filtering process is only effective if it is cheap and adequately predictive.[3] The critical factor hypothesis means that an agent need only recognize the presence of the critical factor for a goal to realize that there may be an opportunity for it. However, the filtering process cannot involve a detailed functional analysis of all the elements of the current situation to determine whether they are critical factors for any of the agent's goals. Instead, we can use the fact that the causal properties of objects tend to be stable across situations: *e.g.*, knives and scissors tend to cut things, tables to support things, and cups to hold liquids.

These functional tendencies can be labeled; for instance, objects that cut other things are labeled as being `sharp`. There are many similar descriptive terms that label functional properties of objects. Words such as `absorbent`, `sturdy` and `fragile` indicate that the object with the property either affects other objects in some way or is itself affected. These properties can be used when planning to achieve goals: something sharp can be used to cut another object, something absorbent to mop up a spill, and something sturdy to support a heavy object. We call these labels *reference features*.

Reference features are cheap to compute To be effective the filtering process must involve minimal reasoning. An important characteristic of reference features is thus the ease of infering them in most situations in which the associated causal effects are present.

Reference features are often associated with perceptual cues: *e.g.*, a `sharp` object has a well-defined edge between two nearly parallel planes. In such cases, the sight of an object is enough to bring the reference feature to mind. There are obvious links to the theory of *affordances* proposed by Gibson (1979): an environment's affordances are the functionalities it offers to an animal in it. There are two main differences between affordances and reference features. First, reference features are not limited to interactions between the agent and its environment: *e.g.*, a surface can have the reference feature `support` even if it could not bear the agent's own weight, as long as its ability to support other objects is useful to the agent. Second, reference features need not be directly linked to perceptual features: *e.g.*, we do not have to feel or even see a rubber band in order to apply the term `elastic` to it.

[3] It is in the nature of heuristic filtering processes such as this that some mistakes will be made.

Surfaces: `smooth`, `sticky`, `slippery`, `rough`, `gritty`, `dull`, `shiny`, `greasy`	Load-bearing: `flimsy`, `sturdy`, `tough`, `heavy`, `light`, `soft`, `hard`, `solid`
Containment: `bulky`, `dense`, `tiny`, `large`, `small`, `big`, `hollow`	Liquids: `impermeable`, `permeable`, `absorbent`, `viscous`
Breaking: `delicate`, `brittle`, `fragile`, `robust`	Deformation: `rigid`, `flexible`, `elastic`
Cutting: `sharp`, `blunt`, `soft`, `hard`	Temperature: `hot`, `cold`, `frozen`
Optics: `transparent`, `translucent`, `opaque`	Use as tools: `lever`, `pointed`, `graspable`
Tasks and plans: `urgent`, `fiddly`, `lengthy`	Stability: `stable`, `precarious`, `balanced`

Table 1: Some reference features in everyday domains

Sloman (1989) proposed compact functional descriptions derived from the possibilities for motion provided by physical descriptions of objects. Reference features are more general, as they are not concerned with motion alone but with any type of relevant functionality. Some reference features from everyday domains are shown in Table 1.

Reference features are highly predictive The filtering process used in PARETO is predictive because it is based on the critical factor hypothesis. A critical factor in the achievement of a goal is characterized in functional terms, *i.e.*, in terms of its causal effects on goals. For example, objects that can cut things are critical factors for the goal of cutting string. Reference features, which label the functional tendencies of objects, can thus be used to recognize critical factors.

It is important to note that reference features do not form a complete functional classification of situation elements. For example, not all objects with the reference feature `sharp` help with the goal of finding something to cut a cake. Reference features are useful only because they form the basis of a heuristic filter that indicates some goals as being potentially easy to achieve. Goals that pass the filter can be analyzed further: although the filtering mechanism must reduce the amount of detailed reasoning required, it need not obviate the need for all reasoning. Reference features form a simplified classification of objects by their functionally interesting properties.

Reference features form the basis of an effective filter for opportunities because they constitute an intermediate level of conceptualizing the world between the physical vocabulary provided by perception and the functional vocabulary required to reason about goals. They are thus both cheap and highly predictive.

Recognizing opportunities

Reference features are attached to the situation elements that tend to cause the effects that they label. They can also be attached to goals by labeling each goal with the reference feature of its critical factor. If

there is a situation element that shares a reference feature with one of the agent's goals, that goal is likely to be easily achievable. An agent can then use the following filtering process to spot potential opportunities:

1. Find the reference features in the current situation, indicating the easily achievable effects.
2. Find the reference features of its current goals, indicating the effects needed for goal achievement.
3. Compare the two sets of reference features. The goals whose reference features are found in the current situation are likely to be easily achievable.

The agent then analyzes the indicated goals to see whether they are genuinely easy to achieve.

PARETO

PARETO's world was built using the TRUCKWORLD simulator (Firby & Hanks 1987; Hanks, Pollack, & Cohen 1993). A delivery truck travels between locations, encountering and manipulating objects as it goes. There are three building sites whose workers use the truck to run delivery errands such as "fetch something to carry my tools in." There are over 30 types of object in PARETO's world, of which 20 are used for deliveries. At any moment, there are typically well over 100 different objects at the various locations. PARETO receives delivery orders at random intervals, with a typical run involving between seven and twelve separate deliveries. The world is unpredictable: the truck can sense only those objects at its current location; objects may spontaneously change location; and actions may fail to have the desired results.

How PARETO works

PARETO is based on the RAPS plan execution system (Firby 1987; 1989). When PARETO acquires a new goal, it looks in its library of RAPs (sketchy plans) for one that will achieve the goal. A RAP (Reactive Action Package) specifies all the different *methods* that might be used to achieve a goal. Each method consists of subgoals that must be achieved to execute the plan.

Having chosen a RAP for the goal, PARETO adds a new task to its *task agenda.* A task consists of a goal and a RAP that will achieve it. PARETO's execution cycle consists of choosing a task and processing it:

Either If it has succeeded remove it from the agenda;

Or If it is described by a RAP choose the appropriate method, based on the state of the world at the time that the processing takes place, and add a new task to the task agenda for each new goal;

Or Perform the primitive action specified by the task.

The original task is reprocessed after all its subtasks have succeeded. Their success does not guarantee the success of the original task as some time may pass between their execution and its repeat processing.

At any time during execution the agenda may hold tasks at varying levels of abstraction, ranging from tasks that have not been expanded at all to those consisting of single primitive actions. PARETO can choose any task on the agenda for processing: in general, however, constraints are observed that ensure subgoals are addressed in the correct order and that a parent task is not reprocessed until its subtasks have all succeeded.

PARETO's task selection algorithm, unlike that of the RAPS system, incorporates an opportunity recognition mechanism. Reference features are used to indicate tasks for which there are potential opportunities; they are then analyzed in more detail to determine whether the opportunities are genuine. Finally, PARETO chooses a task from among the set of those for which there are genuine opportunities together with those that are ready for expansion according to the constraints on the task agenda. A set of heuristics is used to ensure that opportunities are pursued when desirable without abandoning the original goal.

Reference features in PARETO

The reference features of situation elements are directly related to the goals they help achieve. PARETO is asked to deliver objects to building sites. Requests are often specified in terms of the type of object required, so every object has its type as a reference feature. Other goals are more loosely specified: *e.g.*, PARETO may be asked to deliver something that can be used to carry tools. Several types of object can be used to carry things; all have the reference feature `carrier`. The functional aspects of objects are relevant only insofar as they affect PARETO's goals; PARETO's reference features may not be those we would attach to the objects' real-world counterparts.

The critical factors for the goals that involve delivering objects are the objects to be delivered; the goals' reference features reflect the objects' properties. A delivery goal that specifies only the class of object to be delivered has the reference features of that class of object. More complex criteria have their own specific reference features: *e.g.*, a task to deliver something with which to carry tools has the reference feature `carrier`. Other tasks include those to travel to the location of a building site or unload an object at a building site, both of which have the reference feature `site`, and the task to refill the fuel tank, which has the reference features `fuel` and `fuel-drum`. There is an inheritance mechanism that ensures that subtasks inherit relevant reference features from their parent tasks.

Opportunity recognition in PARETO

This section gives a detailed example of PARETO's opportunity recognition.[4] PARETO has just received two requests: a `carry-tools` goal for `maple-ave` and a `cut-twine` goal for `sheridan-rd`. Each delivery goal has a `deliver-object` task on the task agenda. The first task PARETO chooses to process

[4]Program output is edited for brevity.

is <deliver-object carry-tools> which it expands into several subtasks, one of which is a find-object task, which is processed next. Boxes (which are suitable for carrying tools) are generally to be found at the lumberyard, so a <truck-travel-to lumberyard> subtask is added to the agenda and is then processed.

On the way to the lumberyard PARETO arrives at warehouse-2 and scans its neighborhood as it always does when it arrives at a new place. Most of the objects present are irrelevant to its goals but two of them, a pair of scissors and a bag, are directly relevant to its cut-twine and carry-tools goals. The first step of PARETO's filtering process is to find the reference features of the current situation, which include:

```
Reference features for ITEM-20: (BAG CARRIER)
Reference features for ITEM-13: (SCISSORS SHARP)
```

The second step is to find the reference features of its current goals. PARETO attaches reference features directly to tasks on the task agenda: the carry-tools tasks have the reference feature carrier, while the cut-twine task has sharp.

The third step is to match the two sets of reference features. The bag indicates potential opportunities for several of the subtasks of the carry-tools goal:

```
Potential opportunity for
  <DELIVER-OBJECT CARRY-TOOLS MAPLE-AVE>
  ITEM-20 (BAG) has reference feature CARRIER
Potential opportunity for <FIND-OBJECT CARRY-TOOLS>
  ITEM-20 (BAG) has reference feature CARRIER
Potential opportunity for
  <LOAD-PAYLOAD-OBJECT CARRY-TOOLS>
  ITEM-20 (BAG) has reference feature CARRIER
```

The top level <deliver-object cut-twine> task has not yet been processed; it is therefore the only task for its goal. It has the reference feature sharp, as do the scissors that are present at the current location:

```
Potential opportunity for
  <DELIVER-OBJECT CUT-TWINE SHERIDAN-RD>
  ITEM-13 (SCISSORS) has reference feature SHARP
```

Finally, PARETO analyzes the potential opportunities to determine which are genuine. There are two ways in which one of PARETO's task may be easily achievable: if it has already succeeded or if it is ready for processing, *i.e.*, not waiting for any others to be completed (Pryor & Collins 1994). In this case the <find-object carry-tools> task has already succeeded. It therefore constitutes a valid opportunity, and PARETO loads the bag into its cargo bay.

```
Has already succeeded <FIND-OBJECT CARRY-TOOLS>
Taking unexpected opportunity:
  <FIND-OBJECT CARRY-TOOLS>
```

PARETO also decides that there is an opportunity for the <deliver-object cut-twine> task: the presence of the scissors prompts it to expand the task.

```
Taking expected opportunity:
  <DELIVER-OBJECT CUT-TWINE SHERIDAN-RD>
Processing task:
  <DELIVER-OBJECT CUT-TWINE SHERIDAN-RD>
```

It then loads the scissors. Since there is now no need to go to the lumberyard, PARETO now makes its way directly to maple-ave where it delivers the bag and then to sheridan-ave to deliver the scissors.

In this example we have seen how PARETO uses reference features to recognize potential opportunities, regardless of whether the goals are being actively pursued. The opportunities that are recognized might involve abandoning an existing plan or choosing a plan for a goal that has not hitherto been addressed.

Discussion

This paper has described a filtering mechanism for opportunity recognition, and its implementation in the plan execution system PARETO.

Reference features and threats

This paper has described how reference features can be used in the recognition of opportunites: they can also be used to recognize threats. Many reference features indicate possible threats to goals. For example, in PARETO's world a sharp object, such as a knife, may cut a soft object, such as a ball of twine thus rendering it unsuitable for its purpose and making it unacceptable to the worker who has requested it.

A limited form of threat detection through the use of reference features has been implemented in PARETO (Pryor & Collins 1992). PARETO's analysis of potential opportunities is performed in two steps: the goal is first checked to see whether it is easily achievable and then its potential interactions with other goals are examined. A goal with problematic interactions does not constitute an opportunity. Detecting problematic interactions involves the same computational problems as recognizing opportunities: the agent may have many goals, each with a large number of subgoals, and in any case might not have chosen plans for all of them. PARETO uses a filtering process that is very similar to the one used for opportunity recognition. Reference features are used to indicate potentially problematic interactions. The potential interaction is then analyzed in more detail to determine whether it is genuine. The reference features not only indicate that a problematic interaction is possible, they also indicate its probable form thus assisting its avoidance.

Reference features and expertise

The concept *sharp* is useful just because many commonly arising human goals involve structural integrity. If, however, we lived in a world in which structural integrity was unimportant, we might well not even have such a concept, let alone find it useful. The reference features that an agent finds useful depend on the tasks that it habitually performs. Agents performing different tasks in the same world may attach completely different reference features to objects in their environments. Properties that are significant to one agent may be completely meaningless to another.

An important corollary to the task-related functionality of reference features is their rôle in the develop-

ment of expertise. A domain expert is certainly expected to be able to recognize opportunities, both routine and novel. A domain novice, on the other hand, may not recognize even routine opportunities. The difference between the two is due to their relative familiarities with the domain. An interesting hypothesis is that an expert is one who has a comprehensive set of reference features for the domain: a novice has a limited or nonexistent set. Testing this hypothesis is an important area of future work. A related issue is the question of how reference features are acquired. The development of expertise involves learning about the functional aspects of the domain and learning to associate perceptual cues or other easily inferable features with them. This learning may consist of associating functional tendencies with features that are already available, or of learning to recognize new features.

Reference features appear to provide a bridge between the perceptual cues an agent receives and the functional judgments it must make. However, the question remains as to whether they are a genuine psychological phenomenon or simply the basis of an effective mechanism totally unlike anything actually used by people. Further work is needed to investigate their reality, both by examining their possible use by domain experts and by investigating their use in opportunity recognition, possibly along the lines of the experiments performed by Patalano *et al.* (1993).

Conclusion

The implementation of PARETO has demonstrated that opportunity recognition based on reference features is feasible. To demonstrate its effectiveness the mechanism must be scaled up to a real world domain. This will involve the construction of a set of reference features for that domain. Scaling up will also involve the use of computationally efficient matching algorithms in the filtering stage of the process.

The mechanism described in this paper can recognize only those opportunities indicated by critical factors. PARETO cannot, for example, make a detour while traveling to a specific location. This type of opportunity can be recognized only through detailed analysis and projection of a type that PARETO is not designed to perform. Instead of deciding in advance how the plans for its various goals will be combined, PARETO combines them on the fly through opportunity recognition. This enables robust and reactive behavior because its method of opportunity recognition does not rely on a plan already having been chosen, and allows the recognition of opportunities whose pursuit involves abandoning existing plans.

Acknowledgments. This work was principally performed at the Institute for the Learning Sciences, Northwestern University. Thanks to Gregg Collins for criticism and encouragement, and to Matt Brand, Will Fitzgerald, Eric Jones and Bruce Krulwich for many useful discussions.

References

Agre, P. E., and Chapman, D. 1987. Pengi: An implementation of a theory of activity. In *Proc. 6th Nat. Conf. on Artificial Intelligence*. AAAI.

Alterman, R. 1986. An adaptive planner. In *Proc. 5th Nat. Conf. on Artificial Intelligence*, 65–69. AAAI.

Birnbaum, L., and Collins, G. 1984. Opportunistic planning and freudian slips. In *Proc. 6th Ann. Conf. Cognitive Science Society*. Boulder, CO: LEA.

Brand, M., and Birnbaum, L. 1990. Noticing opportunities in a rich environment. In *Proc. 12th Ann. Conf. Cognitive Science Society*. Cambridge, MA: LEA.

Ferguson, I. A. 1992. TouringMachines: An architecture for dynamic, rational, mobile agents. Technical Report No. 273, Computer Laboratory, University of Cambridge.

Firby, R. J., and Hanks, S. 1987. The simulator manual. Technical Report YALEU/CSD/RR 563, Department of Computer Science, Yale University.

Firby, R. J. 1987. An investigation into reactive planning in complex domains. In *Proc. 6th Nat. Conf. on Artificial Intelligence*, 202–206. Seattle, WA: AAAI.

Firby, R. J. 1989. Adaptive execution in complex dynamic worlds. Technical Report YALEU/CSD/RR 672, Department of Computer Science, Yale University.

Gibson, J. J. 1979. *The ecological approach to visual perception*. Boston, MA: Houghton Mifflin.

Hammond, K.; Converse, T.; Marks, M.; and Seifert, C. 1993. Opportunism and learning. *Machine Learning* 10:279–309.

Hammond, K. 1989. Opportunistic memory. In *Proc. 11th Int. Joint Conf. on Artificial Intelligence*, 504–510.

Hanks, S.; Pollack, M. E.; and Cohen, P. R. 1993. Benchmarks, testbeds, controlled experimentation, and the design of agent architectures. *AI Magazine* 14(4):17–42.

Hayes-Roth, B., and Hayes-Roth, F. 1979. A cognitive model of planning. *Cognitive Science* 3(4):275–310.

McDermott, D. 1992. Transformational planning of reactive behavior. Technical Report YALEU/CSD/RR 941, Yale University, Department of Computer Science.

Patalano, A. L.; Seifert, C. M.; and Hammond, K. J. 1993. Predictive encoding: Planning for opportunities. In *Proc. 15th Ann. Conf. Cognitive Science Society*, 800–805. Boulder, CO: LEA.

Pryor, L., and Collins, G. 1992. Reference features as guides to reasoning about opportunities. In *Proc. 14th Ann. Conf. Cognitive Science Society*, 230–235. Bloomington, IN: LEA.

Pryor, L., and Collins, G. 1994. Opportunities: A unifying framework for planning and execution. In *Proc. 2nd Int. Conf. on Artificial Intelligence Planning Systems*, 329–334. Chicago, IL: AAAI Press.

Pryor, L. 1994. Opportunities and planning in an unpredictable world. Technical Report 53, Institute for the Learning Sciences, Northwestern University.

Sloman, A. 1989. On designing a visual system: Towards a Gibsonian computational theory of vision. *Journal of Experimental and Theoretical Artificial Intelligence* 1(4):189–337.