Reasoning about Performance Intentions

Michael Freed, Bruce Krulwich, Lawrence Birnbaum, and **Gregg Collins**
Northwestern University, The Institute for the Learning Sciences
1890 Maple Avenue; Evanston, Illinois 60201
Electronic mail: {freed,krulwich,birnbaum,collins}@ils.nwu.edu

Introduction

For an agent to find and repair the faults that underly a planning failure, it must be able to reason about the intended behavior of its planning and decision-making mechanisms. Representations of intended decision-making behaviors, which we refer to as *intentions*, provide a basis for generating testable hypotheses about the source of a failure in the absence of complete information about its cause. Moreover, intentions provide measures by which beneficial modifications to cognitive machinery can be differentiated from harmful or useless ones. This paper presents several examples of these intentions and discusses how they may be used to extend the range of circumstances in which agents can learn.

Since our claim concerns the representations needed to learn from failure, we describe in section 2 a situation in which an agent should be able to learn, and discuss some of the obstacles to doing so. In section 3, we present the idea of *intentions* and show, using the example from the previous section, how they may be used to extend the range of circumstances in which machines can learn. Section 4 presents an implementation of our theory of failure-driven learning and discusses the role of intentions in several stages of the learning process. Finally, in section 5, we relate our claim to the work of other researchers and summarize the argument that the intended behaviors of decision-making mechanisms must be represented explicitly in order to learn from failure.

An every-day example

Consider the situation of a novice driver attempting to traverse a crowded and confusing road feature such as the rotary shown in figure 1. Lacking experience, the novice may end up waiting longer than more experienced drivers before entering the flow of traffic. It is not unreasonable to suppose that the novice notices the undesirably long wait, perhaps with the assistance of impatient drivers waiting behind. The situation implicates some shortcoming in the novice's driving skill, and so warrants an attempt to learn some improvement.

What should be learned?

Drivers can identify *collision-threats* such as road obstacles and moving vehicles, and must take them into account in deciding how and when to enter traffic. For instance, if at some moment traffic is heavy and moving quickly, a driver may notice many collision threats and choose not to enter. Alternately, s/he may employ some plan that avoids or neutralizes all of these threats, and proceed to enter traffic.

For the task of entering the flow of traffic, the category *collision-threat* is useful for constraining the set of plans which may be safely employed. However, the novice in our example may have been employing an overbroad definition of *collision-threat*. As a result, the set of seemingly safe plans will be overconstrained, thereby increasing the average amount of time before some plan seems safe. It is reasonable to blame the category definition for overconstraining the set of plans which can be used to enter traffic safely, thereby preventing timely action. Learning from the failure thus means narrowing the faulty definition of *collision-threat*.

There may of course be many ways in which the definition of collision-threat could be usefully narrowed. For instance, a novice may learn that other drivers tend to stay in their own lanes between exits; thus, the possibility that a vehicle will make a sudden and inexplicable lane change should not be considered a collision-threat.

Tracing the delay to its underlying cause

In our example, the novice's failure to expeditiously enter the flow of traffic stems from a failure to generate a safe plan for entry. This failure stems, in turn,

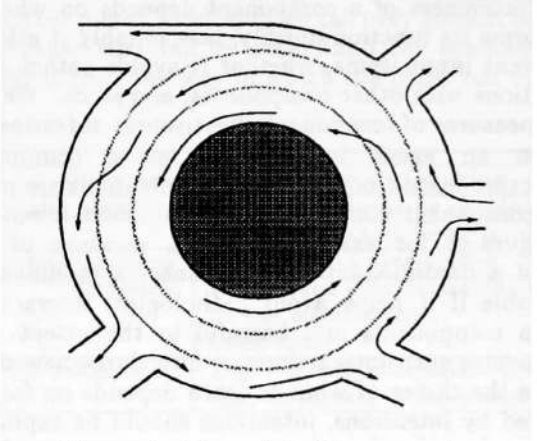

Figure 1: Flow of traffic on a rotary

from having noticed more threats than the novice could plan to counter or avoid. The novice can get better at entering traffic by learning to more accurately differentiate between threats and non-threats.

One way that this diagnosis could proceed is as follows: Suppose the novice receives feedback indicating that a particular perceived threat was not in fact a threat over some time interval. The circumstances during this interval could then be reviewed to show that a safe plan to enter the rotary would have been discovered had the misperceived threat (a "near miss") been ignored. Finally, the misperceived threat could be used to narrow the collision-threat category definition.

The preceding process is unrealistic for several reasons. First, it assumes that the agent can receive specific feedback regarding a *decision* that was made incorrectly, as opposed to an *action* that was incorrectly taken. Moreover, since the novice had no way to know *a priori* which of its decisions should be monitored, it would require that the novice receive continuous feedback on each of its decisions. Finally, it is highly implausible that the novice will be able to receive detailed feedback about particular perceived threats, since novices typically lack the knowledge necessary to evaluate such situations post-hoc [Fitts, 1964; Starkes and Deakin, 1985].

A more realistic diagnostic process begins as the driver of another car honks his horn and thereby leads the novice to question why no safe traffic entry plans have been generated. Hypotheses are developed as to why this might be the case, including for example that either the planning mechanism is inadequate, or its set of collision-threats is faulty. These hypotheses enable the novice to seek specific feedback through experimentation or advice.

Reasoning about agent intentions

As previously argued in [Collins *et al.*, 1991], it is is useful to divide a planner into *components*, each responsible for a task-independent function such as detecting threats or selecting among competing plans. The effectiveness of a component depends on whether it performs its function quickly, how reliably it attends to relevant input items, whether it avoids pathological interactions with other components, and so on. We call these measures of component effectiveness *intentions*.

When an agent implemented as a component architecture learns, one or more modifications are made to its constituent components. Component intentions (see figure 2 for examples) are a measure of the value of a modification. For instance, a modification is valuable if it helps avoid pathological interactions between components and harmful to the extent that it aggravates such interactions or introduces new ones. Because the choice of what to learn depends on factors measured by intentions, intentions should be explicitly represented so that learning processes can reason about them. To understand this point, consider again the rotary example of section .

Within components	Inter-component
No false positives	Don't flood other components
No false negatives	Don't monopolize resources
Efficient computation	Don't reproduce computation
Output values within acceptable ranges	Don't focus on areas of ultimate irrelevance
	Don't be a bottleneck

Figure 2: Sample planner component intentions

Recall that the principal difficulty in speeding traffic-entry performance lay in locating a sample misperceived threat (a near miss) on which to base a refinement of the collision threat classification. The most effective solution was apparently to hypothesize that the category is overbroad and to seek out specific feedback on future threat-detection performance through experimentation or advice. In the absence of an instance of a misclassification, this solution requires some other basis for formulating the fault hypothesis. Intentions provide this alternate basis.

Consider the component intention: ***avoid flooding another component with output.*** The negation of this intention represents a situation in which a component is producing too much output; this in turn indicates that some output-definer (category) may be too broad and that fixing the problem requires locating a false positive with which to narrow the category.

A second role of intentions is to evaluate candidate component modification for preventing failure recurrence. In our example, drastically narrowing the collision threat category (so that no collision threats are generated) would solve the problem of flooding the traffic-entry planner with output, but would lead to catastrophically faulty plans. The basis for rejecting this candidate modification is the intention that the threat-detection component's output be free of false-positives.

Implementation and second example

In this section, we describe a system, CASTLE[1], which implements important aspects of our theory. The system, which operates in the domain of chess, detects situations that are contrary to its expectations, and responds to these expectation failures by repairing the faulty planner components which were responsible for the failure. We view this learning as a knowledge-based process, in which the system uses knowledge of its own planning components to learn from events which led to expectation failures. More specifically, the system must reason along different dimensions of intentionality to determine what repairs should be made to its planning

[1] CASTLE stands for *C*oncocting *A*bstract *S*trategies *T*hrough *L*earning from *E*xpectation-failures.

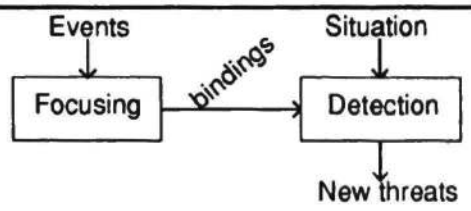

Figure 3: Incremental threat detection

```
(def-brule focus-new-source
 (focus focus-moved-piece ?player
   (move ?player ?move-type ?piece ?loc1 ?loc2)
   (world-at-time ?time))
<=
 (move-to-make (move ?player ?prev-move-type
                     ?piece ?old-loc ?loc1)
               ?player ?goal (1- ?time)) )
```

Figure 4: Focusing on new moves by a moved piece

rules.

The knowledge necessary for this repair process is expressed in the form of a planner self-model, which is used to diagnose and repair expectation failures [Davis, 1984; deKleer and Williams, 1987; Simmons, 1988]. More specifically, when the expectation fails, the system first examines an explicit *justification structure* which encodes the the reasoning which let to its belief in the incorrect expectation [deKleer *et al.*, 1977; Doyle, 1979]. This justification is used to isolate the faulty components of its architecture, each of which implements a particular sub-task in the decision-making process [Collins *et al.*, 1991; Krulwich, 1991]. It then uses a *specification* of the faulty components to guide the learning of new rules in response to the failure [Krulwich, 1992]. Each of these information sources must explicitly reference the planner's intentions.

Detection focusing

A central cognitive task in which CASTLE engages is that of noticing threats and opportunities as they become available [Collins *et al.*, 1991]. Rather than recomputing these at each turn, CASTLE maintains a set of active threats and opportunities that is updated over time. To accomplish this incremental threat detection, the system uses a *detection focusing* component, which consists of *focus rules* that specify the areas in which new threats *may have* been enabled. Then, a separate *threat detection* component, consisting of rules for noticing specific types of threats, detects the threats that have in fact been enabled. This relationship between the two components is shown pictorially in figure 3. A sample focus rule is shown in figure 4. This rule embodies the system's knowledge that the most recently moved piece, in its new location, may be a source of new threats. Another focus rule, not shown, specifies that the more recently moved piece can also be a target of newly enabled attacks. Using focus rules such as these, the actual threat detector rules will only be invoked on areas of the board which can potentially contain new threats.

Focusing intentions

What intentions does the system have regarding its detection focusing component? The primary intention that the system has is that there not be any newly enabled threats that are not within the scope of the bindings generated by the focusing component. This condition is clearly necessary for the incremental detection scheme to work. A more subtle intention is that the focusing component not generate *too many* bindings in which threats *do not* exist. If this intention is not met, the detection component will be invoked more than is necessary, and in the extreme case the entire point of the detection focusing is lost. Clearly the savings gained by only applying the threat detection rules in constrained ways (and not over the entire board) must be greater than the cost of applying the focusing rules. This will not be the case if the constraints given by the focusing component are too weak. It will also not be the the case if the computational cost of applying the focusing component is too high.

Another planner intention regarding the focusing component is that the division of the tasks shown in figure 3 be enforced. This means that the system should not incorporate information about different types of threats into the focusing rules.

Discovered attacks

To see how CASTLE uses representations of planner intentions in learning, let's first see an example of CASTLE enforcing its simplest intention, that there be no false negatives of its focusing component. Consider, in particular, the example of *discovered attacks* in chess, in which the movement of one piece opens a line of attack for another piece. Novices often fall prey to such attacks, not because they fail to understand the mechanism of the threat (i.e., the way in which the piece can move to

Intention	Application to focusing
No false negatives	Don't let a threat be enabled without detectors being invoked
No false positives	Detectors not over-applied
Efficiency	Incremental scheme shouldn't be less efficient than brute-force
No redundancy	Don't encode information about specific threats in focus rules

Figure 5: Planner intentions in detection focusing

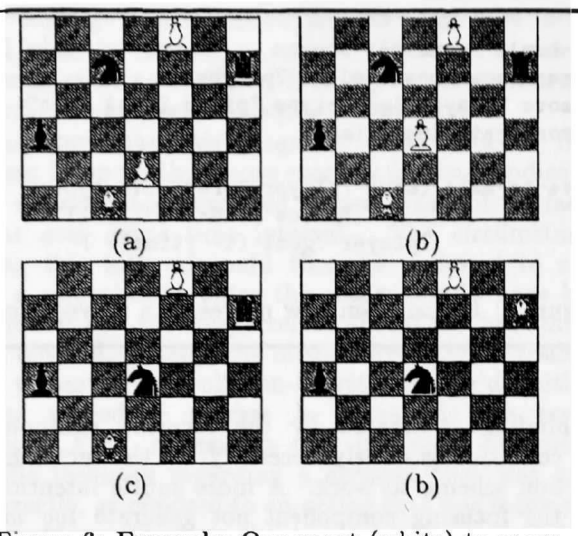

Figure 6: Example: Opponent (white) to move

```
(def-brule learned-focus-method25
  (focus learned-focus-method25 ?player
     (move ?player (capture ?taken-piece)
       ?taking-piece (loc  ?row1 ?col1)
       (loc  ?row2 ?col2)) (world-at-time ?time2))
<=
  (and (move-to-make
          (move ?other-player move ?interm-piece
                (loc ?r-interm ?c-interm)
                (loc ?r-other ?c-other))
          ?player ?goal ?time1)
       (loc-on-line ?r-interm ?c-interm
                    ?row1 ?col1 ?row2 ?col2)
       (at-loc ?player ?taking-piece
               (loc ?row1 ?col1)
               (- gen-time2.24 2)) ))
```

Figure 7: Learned focus rule for *discovered attacks*

make the capture), but rather because they simply fail to consider new threats arising from pieces other than the one just moved. The same is true of CASTLE if it is equipped only with the two focus rules described above.

The example in figure 6 shows the system falling prey to a discovered attack due to its lack of a necessary detection focusing rule. In the situation shown in figure 6(a), the opponent advances its pawn and thereby enables an attack by its bishop on the computer's rook. When the system updates its set of active threats and opportunities, its threat focusing rules will enable it to detect its own ability to attack the opponent's pawn, but it will not detect the threat to its rook. Because of this, when faced with the situation in figure 6(b), the computer will capture the opponent's pawn instead of rescuing its own rook, and it will expect that the opponent's response will be to execute the attack which it believes to the only one available, namely to capture the computer's pawn. Then, in the situation shown in figure 6(c), when the opponent captures the computer's rook, the system has the task of diagnosing and learning from its failure to detect the threat which the opponent executed.

Learning from the failure

To diagnose the failure, CASTLE examines an explicit *justification structure* [deKleer *et al.*, 1977; Doyle, 1979], which record how the planner's expectation was inferred from the rules that constitute its decision-making mechanisms, in conjunction with the policies and underlying assumptions which it has adopted. Diagnosing the failure then involves "backing up" through the justification structure, recursively explaining the failure in terms of faulty rule antecedents [Smith *et al.*, 1985; Simmons, 1988; Birnbaum *et al.*, 1990; Collins *et al.*, 1991]. This diagnosis process will "bottom out" by faulting either an incorrect planner rule or an incorrect assumption that underlies the planning mechanism. In our example, the fault lies in an assumption that the planner could enforce its first intention regarding its focusing component, that it would generate bindings for all enabled threats. CASTLE concludes from this that its set of focusing rules is incomplete and must be augmented.

To construct the new rule, CASTLE retrieves a *component performance specification* for each component. These performance specifications, a form of planner self-knowledge, describe the correct behavior of each component. The specification of the detection focusing component says roughly that *the focusing component will generate bindings that include any capture that is enabled by a given move.* This specification enables CASTLE to focus on the details of the example that are relevant to the component being repaired, by serving as an explanation-based learning target concept [Krulwich, 1991; Krulwich, 1992]. After retrieving the specification, CASTLE invokes its deductive inference engine to construct an explanation of why the possible capture of the rook should have been in the set of constraints generated by the focusing component. This explanation says roughly that the opponent's move should have been generated by the focusing component, *because* the opponent's previous move enabled the attack, *because* it was on a square between the bishop and the rook, *and* there were no other pieces along the line of attack, *and* emptying the line of attack is an enabling condition for the capture to be made. CASTLE then uses explanation-based learning techniques [Mitchell *et al.*, 1986; DeJong and Mooney, 1986] to generalize this explanation and to construct a new detection focusing rule shown in figure 7.

Back to intentionality

In the example of learning *discovered attacks*, the system is able to correct the failure of the detection

focusing component to generate bindings that included the new attack. The learning process that we described involves the construction of a new rule to enforce the intention that the focus component not generate any false negatives. Suppose, however, that instead of adding the rule shown in figure 7, the system's learning component added a focusing rule that returned completely unconstrained bindings. This would cause the detection rules to be applied to all board positions in computing the newly enabled threats. This would, of course, enforce the system's intention to have no false negatives, and would also satisfy the component specification, because all threats that could possibly be enabled are by definition within the unconstrained bindings. However, the whole purpose of the incremental threat detection scheme (figure 3) would be undermined, because not only will the system apply the threat detection rules over the entire board, but it will then proceed to integrate the threats that it finds into the set of previously available threats, which is clearly a waste of time. In short, this is a violation of the system's *no-false-positives* intention.

Unfortunately, while the violation of the no-false-negatives intention could be easily noticed by observing an enabled threat that was not in the focus bindings, it is much more difficult to notice the failure of the no-false-positives intention. In our example, after the opponent moves his pawn in figure 6(b), three classes of bindings constraints should be generated: *threats by the moved piece at its new location,* which are generated by the rule in figure 4, *threats against the moved piece at its new location,* and *threats through the square vacated by the moved piece,* generated by the learned discovered attacks rule in figure 7. Two of these in fact reflect new threats that have been enabled, but one of them, threats by the moved piece, in fact do not reflect any new threats.

We can see that this must be the case, because the division of labor between the focusing and detection components requires that no information about the types of threats themselves be present in the focus rules. Since the focusing component is only determining *where* to look for new threats, it is clear that there will be times when a correct place to look for new threats will not in fact contain any.

This complicates the problem of detecting false positives, since there is no direct test to determine whether the focus component was generating false positives. One approach would be for this intention to only come into play when the system is learning new focus rules. The no-false-positives intention could be used to force the learning mechanism to generate the most specific possible rule. Of course, this approach does not allow the system to reason explicitly about this intention.

Another approach would be to have the system generate expectations about the performance of its component that do not relate directly to false positives, but that are good indicators of the system's false positive rate. As we discussed above, the intention not to have any false positives is in service of system efficiency, because the design of the incremental threat detection scheme is based on the focusing component's sufficiently narrowing down the scope of the detection rule application. It follows from this that the system could monitor the computational effort spent on the detection focusing and compare it with the savings in threat detector application. If this tradeoff turned out not to be worthwhile, the system could examine its false positive rate in more detail. This is similar to the example in which the driver was unable to enter the intersection, causing him to examine his detection mechanism for sources of false threats. This requires that the system be able to make utility judgements about different tradeoffs between false positives, false negatives, and efficiency.

A similar learning process could be invoked if the system noticed that too much time was being spent considering pointless opportunities. This could arise if the system found that it was spending too much time considering pawn captures that were always being discarded by the plan selection component. If this were the case, the system could infer that its focusing component should be further constrained not to generate bindings for captures of pawns.

Discussion

We have shown that reasoning about the faults underlying a planning failure requires that an agent explicitly represent performance intentions which describe the desired behavior of its components. When one of its components is faulty, the agent must reason explicitly about its intentions to diagnose the failure and make a repair which is to its overall benefit.

This paper presents several examples of single-component intentions, such as completeness, soundness, and efficiency, as well as intercomponent intentions such as avoiding flooding and competition for global resources. We have discussed aspects of the learning process which require explicit reasoning about these intentions, thereby extending the range of concepts an agent can learn, and allowing it to learn in circumstances in which it could not otherwise learn. This work thus builds on previous research in failure-driven acquisition of new planning knowledge [Hammond, 1989; Birnbaum *et al.*, 1990; Collins *et al.*, 1991].

Previous research has dealt with several of the issues we have discussed. Minton [1988] discussed the need for learned planner rules to be sensitive to the global efficiency of the system. Our work builds on this idea by explicitly modeling and a variety of such intentions. Hunter's system [1989] reasoned about shortcomings in its diagnostic knowledge and explicitly modeled the intentions involved in that task to guide learning. Similarly, Cox and Ram [1991] have modeled several intentions of the case retrieval process for use in the task

of understanding. Our research extends these ideas to model intentions for a more general problem solver, as well as modeling the intercomponent intentions. Others have used representations of the system's intentions for planning [Jones, 1991] and understanding [Ram, 1989].

Our previous research has involved extending our model of planning and decision-making to include a variety of tasks and components, all in the domain of competitive games. To date we have developed models of threat detection, counterplanning, schema application, goal regression, lookahead search, and execution scheduling. Future research will elucidate the breadth of planner intentions, and will demonstrate the benefits of explicitly representing them for use in learning.

Acknowledgements: Thanks to Matt Brand, Bill Ferguson, Eric Jones, and Louise Pryor for many discussions on the research presented here. This work was supported in part by the Air Force Office of Scientific Research under grant number AFOSR-91-0341-DEF, and by the Defense Advanced Research Projects Agency, monitored by the Office of Naval Research under contract N-00014-91-J-4092. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of The Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech, an Institute Partner, and from IBM.

References

[Birnbaum *et al.*, 1990] L. Birnbaum, G. Collins, M. Freed, and B. Krulwich. Model-based diagnosis of planning failures. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 318–323, Boston, MA, 1990.

[Collins *et al.*, 1991] G. Collins, L. Birnbaum, B. Krulwich, and M. Freed. Plan debugging in an intentional system. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 353–358, Sydney, Australia, 1991.

[Davis, 1984] R. Davis. Diagnostic reasoning based on structure and function: Paths of interaction and the locality principle. *Artificial Intelligence*, 24(1-3):347–410, 1984.

[DeJong and Mooney, 1986] G. DeJong and R. Mooney. Explanation-based learning: An alternative view. *Machine Learning*, 1:145–176, January 1986.

[deKleer and Williams, 1987] J. deKleer and B.C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–129, April 1987.

[deKleer *et al.*, 1977] J. deKleer, J. Doyle, G.L. Steele, and G.J. Sussman. Explicit control of reasoning. *SIGPLAN Notices*, 12(8), 1977.

[Doyle, 1979] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231–272, 1979.

[Fitts, 1964] P.M. Fitts. Perceptual-motor skill learning. In *Categories of human learning*. Academic Press, New York, 1964.

[Freed, 1991] M. Freed. Learning strategic concepts from experience: A seven-stage process. In *Proceedings of the Thirteenth Annual Conference of The Cognitive Science Society*, pages 132–136, Chicago, IL, 1991.

[Hammond, 1989] K. Hammond. *Case-based planning: Viewing planning as a memory task*. Academic Press, San Diego, CA, 1989.

[Hunter, 1989] L. Hunter. *Knowledge-acquisition planning: Gaining expertise through experience*. PhD thesis, Yale University, 1989.

[Jones, 1991] E. Jones. *The flexible use of abstract knowledge in planning*. PhD thesis, Yale University, 1991.

[Krulwich, 1991] B. Krulwich. Determining what to learn in a multi-component planning system. In *Proceedings of the Thirteenth Annual Conference of The Cognitive Science Society*, pages 102–107, Chicago, IL, 1991.

[Krulwich, 1992] B. Krulwich. *Learning New Methods for Multiple Cognitive Tasks*. PhD thesis, Northwestern University, Institute for the Learning Sciences, 1992. (in preparation).

[Minton, 1988] S. Minton. *Learning Effective Search Control Knowledge: An Explanation-Based Approach*. PhD thesis, Carnegie Mellon University, 1988.

[Mitchell *et al.*, 1986] T.M. Mitchell, R.M. Keller, and S.T. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1), January 1986.

[Ram and Cox, 1991] A. Ram and M. Cox. Using introspective reasoning to select learning strategies. In *Proceedings of the First International Workshop on Multistrategy Learning*, 1991.

[Ram, 1989] A. Ram. *Question-driven Understanding: An integrated theory of story understanding, memory, and learning*. PhD thesis, Yale University, 1989.

[Simmons, 1988] R.G. Simmons. A theory of debugging plans and interpretations. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, St. Paul, MN., 1988.

[Smith *et al.*, 1985] Reid G. Smith, H.A. Winston, T. Mitchell, and B.G. Buchanan. Representation and use of explicit justifications for knowledge base refinement. In *Proc. IJCAI-85*, Los Angeles, August 1985.

[Starkes and Deakin, 1985] J.L. Starkes and J.M. Deakin. Perception in sport: A cognitive approach to skilled performance. In *Cognitive Sport Psychology*. Sports Science Associates, Lansing, N.Y., 1985.