Generalizing Indexical-Functional Reference

Marcel Schoppers and **Richard Shu**
Robotics Research Harvesting, PO Box 2111, Redwood City, CA 94063

Abstract

The goals of situated agents generally do not specify particular objects: they require only that *some* suitable object should be chosen and manipulated (e.g. any red block). Situated agents engaged in deictic reference grounding, however, may well track a chosen referent object with such fixity of purpose that an unchosen object may be regarded as an obstacle even though it satisfies the agent's goals. In earlier work this problem was bridged by hand-coding. This paper lifts the problem to the symbol level, endowing agents with perceptual referent selection actions and performing those actions as required to allow or disallow opportunistic re-selection of referents. Our work preserves the ability of situated agents to find and track specific objects, adds an ability to automatically exploit the opportunities allowed by nonspecific references, and provides a starting point for studying how much opportunistic perception is appropriate.

Introduction

If an artificial agent is to interact with real objects, associations between references inside the agent and objects outside the agent must be maintained by the agent itself. To solve this basic problem, (Agre 1988) devised PENGI, which showed how to ground and manipulate indexical-functional references (IFR).

Subsequently, (Schoppers&Shu 1990) built the basic IFR capabilities into an execution engine for a symbolic plan representation, and reported that the plan's execution-time behavior was not what they had wished. They gave the agent a goal to stack any red block on any blue block. After waiting for the agent to find red and blue blocks, they then put a different red block on the blue block chosen by the agent. Instead of recognizing this as a serendipitous achievement of its goal, the agent put down its chosen red block, removed the unwanted red block, and then resumed the activity of placing its chosen red block atop the blue block. This behavior was of course appropriate for the agent's construction: the agent was designed to use all variables as vehicles for indexical-functional references, and to ground each such reference in any one suitable object, permanently. Such grounding corresponds to what might be expected if the agent were given an instruction using a definite noun phrase: "Put the (whichever) red block you see first on the (whichever) blue block you see first". But because the references associated with the plan's variables were initially ungrounded, it was easy to want behavior appropriate to a nonspecific indefinite noun phrase ("Put a red block on a blue block").

The foregoing analysis moves the problem into a linguistic realm where there are many varieties of reference, with IFR being only one special case. IFR happens to be basic to the tracking of physical objects, but is relatively rare in statements of things to be accomplished. Usually, any suitable object will do. Even when the language of an instruction identifies a specific object, this often occurs not because that one object must be used, but because the speaker believes one such object to be especially convenient. (Consider the written assembly instruction "With the Phillips screwdriver, ..." when the reader has several but was asked to fetch one in previous instructions.) Thus, the problem is that there remains a large gap to be bridged between implemented reference grounding capabilities (currently for IFR only) and the varieties of object reference available to humans when expressing desired behavior. If we are to produce agents capable of efficiently carrying out human instructions, whether verbal or programmed, we must find ways to make agents more responsive to the variety of object references used by humans. Such responsiveness must be provided both in instruction understanding and in instruction enactment.

It might be argued that PENGI, by using one marker to track the bee believed to be closest to the penguin and a second marker to compare other candidate bees, went beyond enactment of references of the form "that bee" to enact identification of "the nearest bee". While

true, this was accomplished by hand-coding a specific defensive behavior, where we wish to devise a mechanism that can be parameterized to allow an automatic choice from a variety of reference types.

The present paper is concerned with augmenting the varieties of reference that can be *enacted* by situated agents; it is not a Natural Language paper. We assume the existence of a Natural Language component capable of maintaining a discourse model, of resolving co-references, and of deciding what kind of reference grounding behavior is implied by received instructions. In the next section, where we are obliged to define a few terms for the sake of describing certain implemented capabilities, it is merely an accident that the terms we find most useful come from the domain of NL research.

In the next section we show that the kind of reference used in a goal restricts the range of situations that satisfy the goal. Then, since nonspecific indefinite reference is common in goals, we describe how we co-opted mechanisms for grounding IFR, to make them produce behavior appropriate to nonspecific indefinite references. As a result, our agent can behave sensibly when given goals combining nonspecific indefinite references with deictic references ("See to it that there's a red block on that blue block.")

Defining the Problem

Varieties of Reference

In English, noun phrases and their reference meanings can be distinguished in many ways, and we list a few distinctions below. Our objective in this and the next subsections is to exhibit some of the most obvious ways in which an artificial agent's behavior *must* be influenced by some basic varieties of reference that can occur in instructions. Our definitions are purposely *ad hoc*, i.e. for the sake of succintly describing what we have implemented we are making some distinctions in slighly different ways than linguists would. See e.g. (Quirk et al 1985) for a more complete classification.

- *Referential* noun phrases (NPs) expect the agent to identify the referent using either contextual or general knowledge. Since we are working on grounding references in physical objects, and since attributions occur at the linguistic level, we consider only referential NPs here.
- In a *specific* reference – and under the restriction to bounded physical objects – the instructor has a particular object "in mind". Only in the specific case can the speaker be expected to answer the question, "Who/what is it?"
- A *deictic* reference depends for its grounding on the instructor's place in space and time, and is used only when the instructor expects the referent to be immediately identifiable in context. Indexicals are deictic ("I", "tomorrow").
- The distinction between *definite and indefinite* NPs is syntactic, based on the use of such determiners as "the" and "a". However, a speaker using a definite NP expects one referent to be uniquely relevant, though perhaps only in some later context, e.g. "Go into the house and pick up the amulet [you will find there]" (Webber 1991).
- *Demonstrative* NPs are indicated syntactically by the presence of the determiners "this", "that", "these", or "those". The associated references are usually specific, definite, and deictic.

Specific reference is especially hard to isolate. For example, the NP "Olaf Palme's assassin" refers (perhaps) to a common knowledge entity, but no-one knows who that person is. Similarly, there is room to debate the specificity of superlatives such as "the biggest apple you can find," of ordinals such as "the first apple you touch," and of collectives such as "the 10,000 men around the castle." Since this paper is concerned with encoding and enacting plans that can identify and manipulate only singular physical objects, we gladly sidestep such complexities. To simplify the rest of this paper we propose that superlative, ordinal, and similar references which are not immediately groundable but may be made so in the future (including some functionals) be considered *nonspecific* references.

The distinctions just elaborated give rise to the following classification of referential, non-generic, non-coreferring, references to singular, physical, bounded, non-collective, objects. We include demonstrative NPs under deictic references.

- nonspecific, indefinite: "Pick up *a red block.*"
- nonspecific, definite, nondeictic: "Identify *the heaviest block* [on the table]" (when the blocks are otherwise indistinguishable). "Olaf Palme's assassin."
- nonspecific, definite, deictic: "Pick up *the block closest to your hand*" (when the instructor can't see it).
- specific, indefinite: "*A red block* just fell off the table" (while the instructor is looking at it).
- specific, definite, nondeictic: "Pick up *the red block on the floor.*"
- specific, definite, deictic: "Pick up *the block I'm pointing at.*" "Pick up *that block.*"

Varieties of Serendipity

Now let us consider what behavior we expect to see from an intelligent embedded agent when it is given instructions specifying goals (desired states) involving the kinds of reference listed above.

First, let us give the agent a goal to "Obtain a state in which a red block is resting on a blue block" (nonspecific indefinite reference). This goal is satisfied by any world state in which any red block is on any blue block; the agent need do no more than scan the table.

Next, let us consider the use of what we have decided to call nonspecific definite reference. Strictly speaking, only the object being "identified" will satisfy the request, but by definition, the speaker does not know which object that is and cannot verify that exactly the right one has been found (without executing her own instruction and assuming that the world has not changed)! That being the case, nonspecific definite instructions must be meant either as

1. requests to identify objects (e.g. find Olaf Palme's assassin) to save the instructor some time; or as
2. abstractions (e.g. retrieve the astronaut that just fell off the Shuttle, who it is doesn't matter); or as
3. simplifications that only approximate the instructor's real wishes (e.g. bring the next person in line, assuming no unruly behavior in the queue; or, use the first X you find, instead of "be quick").

In all cases, the definiteness of the reference serves only to communicate the instructor's (possibly false) expectation that one identifiable object is "the right one". Beyond that, it makes no difference to the desired behavior – the agent should find any person who is Olaf Palme's assassin, retrieve any astronaut who fell off the station, and bring whichever person is next in line. Consequently, the distinction between nonspecific indefinite and nonspecific definite instructions is not important to agent implementation.

Next, we find that specific indefinite reference cannot be used for posing goals. Any instruction forces the agent to choose objects to manipulate, but in a specific indefinite instruction the speaker already has specific objects in mind and refuses to say which objects!

Next, let us use specific definite reference (other than a coreference) and give the agent a goal to "Obtain a situation in which that (pointing) red block is resting on that (pointing) blue block." This goal is narrow: exactly the indicated blocks must be made to satisfy the goal condition, no other blocks will do. However, if the two indicated blocks are already in the desired configuration, the agent needs to do no more than look.

Finally, instead of describing a desired situation we can ask the agent to perform an action, such as "Put a red block on a blue block". In this case, no matter how many red blocks are already atop blue blocks, the agent is nevertheless obliged to build another such tower.

The two major kinds of reference (nonspecific and specific) usable in goals, and the distinction between requests for conditions and requests for action, together produce three kinds of behavior that may be distinguished from each other by the kinds of serendipitous circumstances an embedded agent may exploit while performing the requested task. We distinguish the following cases of admissible serendipity:

- *Nonspecific serendipity.* The agent may satisfy the goal using any suitable objects, and is not required to act if suitable objects already satisfy the goal.
- *Specific serendipity.* The agent may satisfy the goal using only the objects specified by the references included as part of the goal, but is not required to act if those specific objects already satisfy the goal.
- *No serendipity.* The agent must not exploit the presence of suitable objects that already satisfy the goal.

Problem Statement

The three kinds of serendipity also allow us to succinctly describe the problem with implementations of indexical-functional reference (IFR), as follows:

- IFR is a form of specific definite reference which naturally delivers only specific serendipity, and which can be made to deliver nonspecific serendipity only with considerable effort.
- Naive attempts to make implementations of IFR deliver nonspecific serendipity result in referential thrashing, i.e. an agent too confused about object selection to get anything done.

Our objective is to use explicit representations, not hand-coding, to specify the behavior of embedded agents, including behavior involving interaction with several indistinguishable objects at the same time. Two problems must be solved to attain our objective.

1. Symbolic plans using IFR must become capable of noticing and exploiting nonspecific serendipity.
2. Symbolic plan representations must become capable of expressing a demand for specific serendipity.

The second problem may be solvable by building on the distinction between rigid and nonrigid designators, as has been done in e.g. (Appelt 1982). In this paper we address the first problem. In particular, we wish to define general, *domain-independent* notations and plan execution machinery, such that the execution-time specificity of each of the agent's references can be controlled by domain-specific knowledge (which knowledge may therefore be regarded as a parameter to the agent's re-usable algorithms).

The Experimental Setup

Before we describe our work, the reader must understand some details of the plan encoding we worked with. Our agent consisted of a simulated arm, some simulated sensors, and a repetitively executed decision tree, all operating in a world of simulated blocks. The

```
on(A,B) ?
  T) NO-OP
  F) box(B) ?
       T) clear(B) ?
       .    T) holding(A) ?
       .    .    T) over(B) ?
       .    .    .    T) LOWER
       .    .    .    F) at(top) ?
       .    .    .         T) LATERAL
       .    .    .         F) RAISE
       .    .    F) [subplan to GRASP A]
       .    F) [subplan to CLEAROFF B]
       F) FAIL
```

Figure 1: Decision Tree Schema for Block Stacking.

simulated arm and camera moved horizontally and vertically, in increments that were much smaller than the size of a block. An example decision tree is shown in Figure 1. We refer to each traversal of the decision tree as a *execution cycle*. Blocks had names and colors, and could be indistinguishable. The human observer could move blocks about in the simulated world, thus bringing good or bad luck at will.

Notice that the given decision tree is a *schema* containing unbound variables (in the Prolog convention logical variables begin with an upper case letter). This was important in allowing us to invoke the plan (tree) with whatever parameters we wanted. (The parameters finally became records that contained two position vectors, one for predicted/believed position and one for perceived/known position; lack of information was indicated as a zero vector.)

Following (Schoppers&Shu 1990) we regarded object descriptions as goals. When we wanted a plan to put any red sphere on any blue box, we implemented a plan that achieved **color(X,red)** $\wedge$ **shape(X,sphere)** $\wedge$ **shape(Y,box)** $\wedge$ **on(X,Y)** and we expected the agent to find suitable objects for X and Y to refer to.

This view of descriptions is not as strange as it may seem at first. If we were to augment the agent's capabilities by introducing a painting action, the above goal might induce the agent to paint things, a potentially appropriate behavior. At the same time, one way of satisfying a color goal for a nonspecific indefinite object is to (physically) look for an object that already has the desired color; indeed, that is the only way to achieve a color goal when there is no painting action. Thus we came to regard painting and scanning as alternative ways of achieving color goals.

At the start of the agent's activities, the agent knew nothing at all about the state of the (simulated) world. In particular, when the plan specified that some block should be moved, the agent had first to find a block to move. To solve this problem we implemented a camera movement procedure that systematically scanned the table until the camera viewed a block. This whole scanning procedure was controlled by means of camera positioning coordinates.

As a result, the executing plan could refer to objects by knowing in what direction the camera should be pointed in order to make the object appear in the camera's field of view. That directional knowledge allowed the agent to verify visible properties of objects whenever the property verification could be implemented as a test on the camera image. To test relative positions of blocks, however, or to test the position of the robot arm relative to a block, it was necessary to know both the direction and the range to a block, i.e. the block's position in three dimensions. Thus, "referring to an object" came to mean "truly knowing the object's current 3D position" (versus having a belief or expectation). The required position information was stored (and updated) in records that were passed as parameters to the decision tree. Thus we could define a predicate **known located(X)** to test that variable **X** was bound to a record containing visually verified information about current position. We also defined **believed located(X)** to test that **X**'s record contained an *expectation* of current position. From there we could define actions that achieved **known located(X)** (see next section).

Referent Selection Actions

(Cohen 1981) examined dialogues in which speakers attempted to get hearers to identify specific objects, and argued for extending a plan-based theory of communication with explicit actions representing the hearer's identification of objects. The speaker would adopt a goal of getting the hearer to identify something, and would communicate that goal to the hearer, who would then try to achieve the goal by means of an IDENTIFY action. Our block-stacking agent's decision tree both tests object identification goals and achieves them with IDENTIFY-like actions. However, we have found it useful to endow our agent with many such actions, most of which come down to a visual search that is specialized to exploit known visual features of the object to be identified. Additionally, many of our "referent selection actions" exploit positional expectations.

Candidate referents must be found for each plan (object) variable before any of the conditions mentioning that variable can be tested. The very first thing most plans must do is cause the performance of perceptual searching activities to locate candidate objects. Once

candidate objects are found, plan execution can track them and apply perceptual tests to them. But clearly, if objects are selected only once, at the beginning of plan execution, and are tracked thereafter, the resulting behavior can exploit specific serendipity at best. To achieve nonspecific serendipity it must be possible to select new objects for references that already have referents. The cyclic execution paradigm makes this very easy by providing the opportunity to revise all referent selections once per execution cycle, and our definition of **known located(X)** plays the role of having to be constantly reachieved (because a remembered or predicted location is not a known location). The main challenge is to determine a suitable set of referent selection and reselection actions, and to integrate those actions with the existing tracking machinery.

Identifying Efficiently

There are many efficient ways to locate and identify objects. For example, given a goal to put a red block on a blue block, you will first look for either a red or blue block. If the perceptual apparatus can be primed to look for colors, this is already much more efficient than looking for any object at all. Now suppose you have found a blue block. Given that you have been asked to put a red block on top of it, the natural thing to do is: look for a red block on top of the blue block you just found. This behavior is efficient because it tells the perceptual system exactly where to look, and also because finding a red block there would allow you to consider yourself finished with the task.

The preceding paragraph implicitly states the heuristic that, to find referents for nonspecific indefinite references, it is efficient to limit perception by using knowledge of what's wanted, as indicated by current plan goals. This heuristic is the basis of a large body of work on task-directed perception. We are less concerned with the details of perception than with managing perception to support opportunistic task performance.

In our simulated domain we found that there were numerous goals that suggested perceptually efficient referent selection actions. For example:

on(X,Y) To find a referent for X when Y has a referent, look just above the location of Y.

on(X,Y) To find a referent for Y when X has a referent, look just below the location of X.

over(X) To find a referent for X (such that the agent's hand is over it), look at the agent's hand and move the camera in a vertical line downwards.

clear(X) To find a referent for X, start the camera at the left end of the table and follow the "skyline".

holding(X) To find a referent for X, look at the location of the agent's hand, which is always known. Similarly for **around(X)**.

shape(X,box) To find a referent for X, look for any rectangular thing (similarly for other shapes).

color(X,red) To find a referent for X, look for any red thing (similarly for other colors).

For the first few goals above, referents can be found efficiently because the goals are prepositions that identify a direction vector. The more general source of efficiency, as evidenced by the last few goals, is exploitation of capabilities of the perceptual system itself, e.g. the ability to locate only certain colors or shapes. The capability to rapidly look at specified lines or locations is responsible for the usefulness of spatial prepositions in referent selection.

Some Consequences

Because of the association between particular goals and particular actions for finding referents efficiently, we now give referent selection actions postconditions indicating their special suitability. For example, under a standard approach to action representation the action for visually locating red things should have the postconditions **known color(X,red)** $\wedge$ **known located(X)**, where **known located(X)** is as defined in the previous section: perceptually finding a red object also finds the object's position. But these postconditions would represent that the identify-a-red-thing action is also useful for locating nondescript objects, leading to inefficient behavior when the plan wants any nondescript object and all the available objects are blue. Indeed, many color-seeking, shape-seeking, and texture-seeking visual searches might be tried in vain. On the other hand, there are efficient ways to find nondescript objects. Consequently our representation of the identify-a-red-thing action omits the **known located(X)** postcondition. Conversely, even though an identify-anything action can eventually find a red object if there is one, our representation of identify-anything actions has no **known color(X,red)** postcondition. Thus, postconditions now have less to do with actual effects than with the utility of the described action. (We recognize that this is a poor substitute for explicit knowledge about the utility of actions, see the "Future Work" section.)

When a plan variable occurs in several goals, the referent selection actions associated with any of those goals might be used to find a referent. For example, when the plan's goal is a conjunction such as **color(X,red)** $\wedge$ **shape(X,sphere)** $\wedge$ **shape(Y,box)** $\wedge$ **on(X,Y)** there are two referent selection actions that can be used immediately to find a referent for X (using color and shape as guides) after which there are

also two referent selection actions for Y (using shape and the location under X).

To ensure that every object variable indexes at least one referent selection action, all goals are enlarged with additional conjuncts **known located(X_i)**, one per object variable X_i appearing in the goal. As a result, most referents can be selected with both goal-specific referent selection actions and a general visual search.[1]

Since the **known located(X_i)** conjuncts need constantly to be reachieved, every execution cycle offers an opportunity to change the object being referred to by X_i. To make the reselection, all of the goals and subgoals in which X_i appeared during the previous execution cycle may provide useful information. For example, **on(X,Y)** is achieved by LOWERING the hand when **holding(X)**, which is therefore a subgoal; this subgoal is achieved by GRASPING when **around(X)**, which is therefore a subsubgoal; this subsubgoal is achieved by **lowering** when **clear(X)**, and so on. To achieve **known located(X)** by possibly choosing a different **X**, it is useful to look at what's already on **Y**, at what's already in the agent's hand, and at the "sky-line" of the current block configuration.

This uniform relevance of everything in the goal stack was especially appreciated in the case of the goal **holding(X)**, whose associated referent selection action caused **X** to refer to "the-thing-I'm-holding" – a reference that is both indexical and functional.

Indeed, when the referential guidance provided by goals is taken seriously, the meaning of a reference comes to depend on which goals apply to the referent. Hence the meanings of our references change dynamically as the agent's goals change, and the semantics of our references is compositional.[2]

Recovering Opportunistic Behavior

Finally we come to the issue of integrating referent (re)selection with deictic tracking. The normal reactive execution cycle is constantly re-achieving **known located(X_i)** for each X_i. The first referent selection action that is relevant and that succeeds in finding a referent for the given X_i will have satisfied the goal, thus preempting the use of other referent selection actions (for the same reference and execution cycle). This means that we have to be careful with the order in which referent selection actions are tried:

- As soon as X_i becomes grounded, deictic tracking can maintain the grounding, but that produces exactly the single-minded tracking we are mitigating. To have any effect, goal-related referent selection actions must be tried before tracking. They can then look for objects in places that will cause the agent to notice nonspecific serendipities. A serendipitous object will ground the reference and preempt deictic tracking; otherwise deictic tracking can continue.
- Similarly, general visual scanning can always find a candidate object for any reference. If it were tried before tracking, then tracking would never be used. Hence, such general backup methods must be tried only after tracking has failed.

Thus we were obliged to try referent selection actions in the specific order **1)** goal-related referent selection actions, **2)** tracking (or, re-perceiving the referent of the preceding execution cycle), and **3)** general, goal-independent visual searching.

Nonspecific indefinite reference, and the corresponding ability to exploit nonspecific serendipity, resulted from trying all the relevant actions in the order just stated, until some action succeeded in finding a referent. Specific definite (deictic) reference could be recovered by disabling the use of (category 1) goal-related referent selection actions. Disabling the use of (2) deictic tracking produced referential thrashing; disabling the use of (3) goal-independent visual searching left the agent unable to find anything.

Summary and Future Work

In this work we have shown that the type of reference used in specifying agent behavior affects the kinds of serendipities the agent may exploit. We have harnassed IFR to provide two additional kinds of situated object reference, namely specific definite reference and nonspecific indefinite reference, as different points on a continuum defined by the presence or absence of three kinds of referent selection actions. The three kinds of referent selection actions are **1)** goal-related, which can detect that goals have been satisfied by objects other than those chosen by the agent; **2)** tracking, which provides referential stability; and **3)** general visual searches, which locate objects when none are known.

Our work raises many performance issues. An agent cannot always afford to re-perceive, in each execution cycle, every object it is already interacting with, let alone looking for serendipitous objects. There is a

[1] It does not matter that the same variable can appear in many **known located(X_i)** conjuncts. For each X_i, the first **known located(X_i)** conjunct encountered in any given execution cycle initiates perceptual activity. This achieves **known located(X_i)** and forestalls reachievement for the rest of that execution cycle. An equivalent approach could attach a **known located(X_i)** conjunct only to the goal at the root of the subtree containing X_i.

[2] The semantics of (Agre 1988)'s deictic representation was not compositional, i.e. there was no semantic relationship between the notations "the-thing-I'm-holding" and "the-red-thing-I'm-holding".

small body of work discussing when it is worth-while for agents to engage in sensing (Abramson 1993; Chrisman&Simmons 1991; Dean et al 1990; Doyle et al 1989; Draper et al 1994; Hansen 1994). Our own previous work (Schoppers 1995) made sensing intermittent and dependent on environmental dynamics. Our present implementation side-steps these issues by **a)** treating action descriptions as heuristics, and making postconditions encode our subjective judgments about the utility of actions for specific goals[3]; and **b)** restricting our goal-related referent selection actions to look *only* for serendipities involving previously selected objects. Thus, our agent will notice if another block is put down on one of the blocks the agent was using, but will not notice if someone builds a tower that satisfies the agent's goals by using only blocks the agent doesn't care about. In future work we intend to make the agent more conscious of the likelihoods that particular goals will be achieved serendipitously, of the time costs of perceiving those particular serendipities, and of the time costs of achieving the goals deliberately. Performing, in each execution cycle, the perception or action process having the highest probability of goal achievement per unit time cost, will then yield optimally efficient behavior (Simon&Kadane 1975). Since looking above a block already in the field of view is almost free, while looking in a random place for a particular block arrangement is both expensive and probably futile, we expect the resulting behavior to be very similar to what we have now.

Other relevant issues include:

- When the agent must find objects to ground several references, the order in which references are grounded may affect the agent's efficiency.
- There may be useful ways to order the referent selection actions applicable at a given time.
- There are choices to be made between: physically moving lots of things around to find an object that perfectly satisfies a description, versus purely perceptual searching for a perfect object, versus finding an object that is nearly right and then modifying it, versus just building a suitable object.
- Agents could be made very sophisticated about what serendipities they deem likely at any given time, and how much effort to spend on looking for them.

More long-range possibilities include, finding ways to mentally distinguish a selected referent (e.g. you can track one pigeon among a flock by using a distinctive feature); finding ways to *make* an indistinguishable referent distinguishable; how to handle plural references; and how to efficiently spot perceptually complex referents (since the predicate **block(X)** should automatically lead to a search for appropriate features such as lines and angles).

[3]Readers who feel that an effect is an effect no matter how expensive the action is, might yet hesitate to represent an effect having very low probability, and probability is merely the other factor in low utility.

References

Abramson, B. 1993. A decision-theoretic framework for integrating sensors into plans. IEEE Trans SMC 23(2):366–373.

Agre, P. 1988. *The Dynamic Structure of Everyday Life.* Tech rept 1085, AI Lab, MIT.

Appelt, D. 1982. *Planning natural language utterances to satisfy multiple goals.* Tech Note 259, SRI International, Menlo Park, California.

Chrisman, L. & Simmons, R. 1991. Sensible planning: focusing perceptual attention. Proc AAAI Nat'l Conf on AI:756–761.

Cohen, Phil. 1981. The need for identification as a planned action. Proc 7th IJCAI:31–36.

Dean, T. Basye, K. & Lejter, M. 1990. Planning and active perception. Proc DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control:271–276.

Doyle, R. Sellers, S. & Atkinson, D. 1989. A focused, context-sensitive approach to monitoring. Proc 11th IJCAI:1231–1237.

Draper, D. Hanks, S. & Weld, D. 1994. Probabilistic planning with information gathering and contingent execution. Proc Int'l Conf on AI Planning Systems AIPS:31–36.

Hansen, E. 1994. Cost-effective sensing during plan execution. Proc AAAI Nat'l Conf on AI:1029–1035.

Quirk, R. et al. 1985. *A Comprehensive Grammar of the English Language.* Longman Inc., New York.

Schoppers, M. & Shu, R. 1990. An implementation of indexical-functional reference for the embedded execution of symbolic plans. Proc DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control:490–496.

Schoppers, M. 1995. The use of dynamics in an intelligent controller for a space-faring rescue robot. *Artificial Intelligence* 73(1):175–230.

Simon, H. & Kadane, J. 1975. Optimal problem-solving search: all-or-none solutions. *Artificial Intelligence* 6(3):235–247.

Webber, B. 1991. Indexical-functional reference: a natural language perspective. Unpublished extended abstract.