Title
Acquiring Rules for Need-Based Actions Aided by Perception and Language

Journal
Proceedings of the Annual Meeting of the Cognitive Science Society, 14(0)

Authors
Mani, Ganesh
Uhr, Leonard

Publication Date
1992

Peer reviewed

Acquiring Rules for Need-Based Actions Aided by Perception and Language

Ganesh Mani and **Leonard Uhr**
Computer Sciences Department
University of Wisconsin—Madison
1210 W. Dayton St., Madison, WI 53706.
{ganesh, uhr}@cs.wisc.edu

Abstract

The CHILDLIKE system is designed to learn about objects, object qualities, relationships among objects, and words that refer to them. Once sufficient visual–linguistic associations have been established, they can be used as foundations for a) further learning involving language alone and b) reasoning about the effect of different actions on perceived objects and relations, and internally sensed need levels. Here, we address the issue of learning efficient rules for action selection. A trial-and-error (or reinforcement) learning algorithm is used to acquire and refine action-related rules. Learning takes place via generation of hypotheses to guide movement through sequences of states, as well as modifications to two entities: the weight associated with each action, which encodes the uncertainty underlying the action, and the potential value (or vector) of each state which encodes the desirability of the state with respect to the current needs. CHILDLIKE is described, and issues relating to the handling of uncertainty, generalization of rules and the role of a short-term memory are also briefly addressed.

Introduction

Perception is crucial to any activity by intelligent agents in an environment. In dynamic environments, perception-mediated reasoning and acting can avoid the problems of planning that goes down blind alleys and expends massive efforts to anticipate situations that never occur. Perceiving the state of the world periodically can also save the embedded agent the trouble of keeping precise track of its moves to infer its position relative to other objects in the environment at each step.

On the performance side, it is instructive to note that humans are able to perceive and recognize scenes containing a few objects within a few hundred milliseconds. Such rapid perception is crucial to adequately fast reaction in an environment. (However, it is rarely necessary for a scene to be fully recognized before an agent reacts. Certain key features or salient objects in a scene may trigger reactions that have been associated with the features or objects by prior learning.)

Rules that facilitate choosing actions without extensive deliberation are important, since planning is time consuming and its utility is limited in dynamic environments. At the same time, it is also important to learn the effect of different actions or operators from experience (e.g., see [Drescher, 1987], [Mason *et al.*, 1989], [Shen, 1989]). Reactive planning (e.g., [Georgeff and Lansky, 1987], [Firby, 1987]) or iterative planning [Kaelbling, 1987] can be cast in a memory-based framework particularly when other tasks such as perception and language are being integrated into the system. (Integrating vision and language is gaining considerable attention in the AI and cognitive science community— e.g., see [Dyer, 1991], [Feldman *et al.*, 1990], [Okada, 1991], [Siskind, 1991].) In this paper we describe how rules can be learned that help the system to react appropriately to its needs, and how generalization of these rules is aided by prior learning of visual and linguistic constructs.

In the next section, we briefly describe the integrated system that we are developing to learn from simple experiences. The subsequent sections focus on the acquisition and refinement of hypotheses (structures of rules) that aid the system in reacting to its internal needs. Prior visual–linguistic associations act as powerful biases for the acquisition and refinement of rules that relate actions to the perceived environmental states and their need-satisfaction potential. The representation as well as generalization of rules is addressed.

Background about the CHILDLIKE System

The CHILDLIKE[1] system [Mani and Uhr, 1991a,b] [Mani, 1992] (in preparation) is a computational information-processing model (implemented in Common Lisp) designed to learn about objects, their qualities, and the words that name and describe them; and, further, to use this knowledge to act towards satisfying its internal needs (e.g., hunger, thirst, sleep, curiosity). Thus the CHILDLIKE system attempts to capture the entire perceive-reason-act-learn loop.

The system is subjected to a series of simple "experiences" from which it attempts to learn. An experience consists of several different types of input— for example, a visual pictorial scene, a short language utterance, an abstracted action, an internal need level.

One component of the system acquires visual–linguistic associations from experience. Initially, tentative associations are formed between words and visual features, and

[1] which stands for Conceptual Hierarchies In Language Development and Learning In a Kiddie Environment.

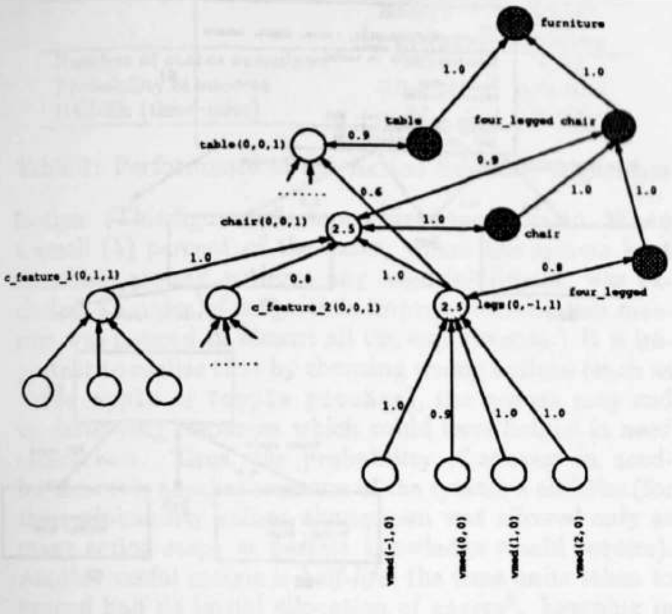

Figure 1: The Network Structure of Hypotheses

groups of these, using learning rules for extraction, aggregation and generation. These associations get strengthened with repeated co-occurrences. Such visual–linguistic associations are refined using ***de-generation*** and generalization mechanisms. For descriptions, see [Mani and Uhr, 1991b] and [Mani, 1992] (in preparation). For an example of the visual–linguistic associations learned by the system in response to inputs such as pictures of chairs and tables, along with words about them, see Figure 1 (only a cross section of the memories is shown). The hatched nodes correspond to structures learned from the linguistic channel; the others to visual features and their compounds. Only some of the highly weighted links are shown. The three numbers alongside a visual feature node represent, respectively, the x and y coordinates of the feature (in the *object*-centered coordinates of its parent node) and its relative size.

This paper concentrates on acquiring knowledge relating to actions and their effects. We also stress how the action-related rules can be generalized and improved using the visual–linguistic associations that have been acquired; integration of these different components is achieved by utilizing a memory-based framework. Mutually grounded representations — that consist of, for example, the visual representation of a fruit, the word that describes it and the action that can be performed on it (eat) to satisfy a certain internal need (hunger) — help in the attempt to span the wide variety of abilities that encompass everyday tasks and reasoning. The current version of the system is a starting point for a realistic architecture and implementation that integrates vision, language and action.

Representation of Rules About Actions

Knowledge for reasoning about actions is usually built into a system *a priori*, rather than learned. In contrast, CHILDLIKE attempts to learn the effect of various actions (currently, primitive actions are built in, but not

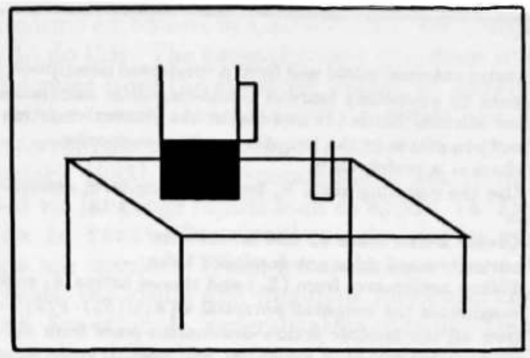

Figure 2: An Example Visual Scene

their effects). An action or an action sequence may impact both the externally perceived entities and the internally sensed need levels. Thus, each state that the system is in can be described in terms of (a subset of) the perceived objects, qualities of objects, relations among them and the internal need levels.

A suitable bias is required to translate the perceived state into an internal state in the plan memories. In CHILDLIKE, the bias used is prior learning from percept–language interactions.[2] Thus, the system translates a visual scene into object names and relations between the objects that are *known* to the system.

CHILDLIKE can learn about objects such as `table`, `pitcher`, and `glass`; spatial relations such as `on`, `above`, and `in`; and other relations such as `and`. Such knowledge is acquired by being trained on instances with language strings such as `brown table` and `pitcher on table` along with their corresponding visual scenes. Based on such knowledge, visual scenes (see Figure 2) presented as arrays representing medium- to high-level features are processed by the system to obtain, for example:

```
on(pitcher,table)
on(glass,table)
in(air,glass)
in(juice,pitcher)
yellow juice
.......
```

A hierarchical structure [Uhr, 1978;1987] is used to rapidly imply objects and relations. Needs are introduced by encoding the sensed internal need levels along with the entities perceived from the external world into each state in the long-term plan memories. These memories are graph structures wherein each link (or state transition) connotes an action. Every action has a weight associated with it, encoding the certainty of the action. Apart from the visually perceived information and the internally sensed need levels, each state also encodes its potential to satisfy each need. Currently, four internal needs are modeled: hunger, thirst, rest and curiosity. A state may not necessarily represent all the perceivable information, but simply the features the system is currently attending to.

[2]Note that other candidates may be useful biases. For example, children use perceptual knowledge alone before they acquire any language.

1a. Perceive external world and form a condensed description (VD) (biased by previously learned visual–linguistic associations).

b. Sense internal needs (I) and match the current situation (VD plus I) to states in the long-term action memories.
If there is a match **then**
  Use the resulting state S_c from the long-term memories.
else
  Create a new state S_c and initialize it.

2a. **if** curiosity-need does not dominate **then**
  Follow action-arcs from (S_c) and choose action A_i such that it maximizes the weighted potential $w(A_i)I(S_c) \cdot P(S_j) - \eta e(A_i)$ over all the feasible action–destination pairs from S_c. {$w(A_i)$ is the certainty that action A_i will lead to state S_j, $I(S_c)$ is the numeric need vector at state S_c, $P(S_j)$ is the potential of state S_j, $e(A_i)$ is the energy expended in or cost of executing action A_i and η is a normalizing factor.}
  Execute action A_i.
else
  Execute an action A_i randomly from the action repertoire of S_c.

b. Perceive the new state (S_n). {This is the same as Step 1 above.}
If there exists a link between S_c and S_n labeled with A_i **then**
  Update its frequency-based weight (also update the weights of other links labeled with A_i from S_c).
else
  Form a new link and initialize it.

c. Propagate the potentiality/need-fulfillment information at S_n back to the previous state S_c. Go to **2a**.

Figure 3: Algorithm that Creates and Refines the Action Memories

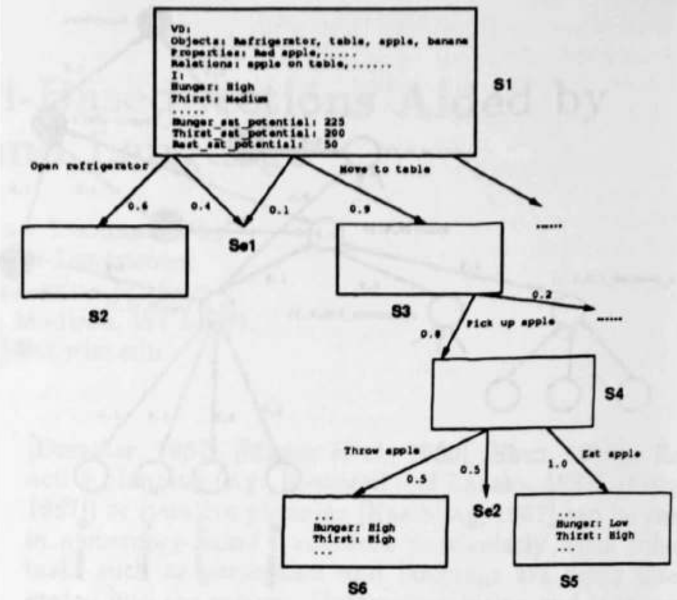

Figure 4: An Example of the Action Memories Acquired by CHILDLIKE

(*Se1* and *Se2* denote extraneous states which may not be explicitly represented in the system's memories.)

Action selection and refinement of the plan memories takes place using a trial-and-error learning algorithm (the current version used by CHILDLIKE is shown in Figure 3). The algorithm assumes abstracted actions such as `Pick up apple`, `Pour into glass`, and `Drink from glass`. (Future versions of the system will decompose these actions further, into sub-actions and the visual frames that bracket them.)

Rules are acquired implicitly, by updating the memories encoding knowledge about actions after each experience. Initially, all actions (in the set of possible actions associated with each state) are equivalent from the system's point of view, as it starts out with no knowledge about the effect of actions. States also usually have initial potential values of zero— exceptions are need-fulfilling states which have appropriately high potential values (these can be thought of as goal states or states where a reinforcement vector is sensed, changing the need levels). As the values corresponding to the potential of each state to satisfy particular needs get propagated through the learned network, and as the effects of actions are perceived and tabulated using a weight associated with each action (note that an action is represented as a link from one perceptual state to another), the performance of the system improves. The potential $P(S_j)$ of a subsequent state S_j is usually a vector, since there are multiple needs; a dot product with the need vector $I(S_c)$ of the current state S_c is used to reduce it to a single value (see Figure 3).

Figure 4 shows a snapshot of the action memories after a few tens of trials of one experiment. Note that the system starts out with all the weight-like certainty values associated with actions set to 0 and the values of variables such as `Hunger_sat_potential` (which stands for hunger satisfaction potential) for states like $S1$ set to 0 (or a low initial value). When hunger satisfaction takes place at a state such as state $S5$, the information propagates back (Step 2c in Figure 3); after a number of iterations, states such as $S1$ reflect their true potential. Currently, a simple average of the node's current potential and the potential of its successor (which is known after an action is performed) is used as the new value of the potential. One advantage of this is that a predecessor node's potential value moves towards that of the successor in a smooth trajectory, if there is a preferred action at the predecessor node (which is usually the case after a few trials).

Note that a learned weight such as 0.8 associated with an action such as `Pick up apple` reflects the fact that from S3 (where `apple on table` is perceived), when the action `Pick up apple` is executed, 4 times out of 5 the apple ended up in the hand. This simple approach to handling uncertainty appears to work well on these simple examples.

Notice that the internal need state is combined with the processed visual state in forming a rule for acting (see Figure 3). Approximate rather than perfect matches are usually employed while utilizing these learned rules.

CHILDLIKE's action-selection abilities clearly improve with learning. Table 1 summarizes the results of 20 experiments that involved action sequences of the sort shown in Figure 4. The shortest action sequence was of length 1 and the largest of length 6. Each state had between 1 and 8 possible actions, with an average of about 4. Actual need satisfaction occurred, typically, in two states. One determinant of performance is the number of states the system may have to look at in the course of need satis-

	Before Learning	After Learning
Number of states examined	120	< 12
Probability of success	0.0-0.15	0.75-0.90
Half-life (time units)	22	> 300

Table 1: Performance of the Action Selection Algorithm

faction. (This figure reflects a worst-case scenario. When a small (5) percent of the cases, where the system kept thrashing around without any need-fulfillment, was excluded, an order of magnitude improvement on this measure was noticed in almost all the experiments.) It is important to realize that by choosing wrong actions (such as `Throw apple` or `Topple pitcher`), the system may end up destroying resources which could have helped in need satisfaction. Thus, the probability of success in need-fulfillment is another measure of the system's abilities (for these probability values, the system was allowed only as many action steps as perfect knowledge would require). Another useful metric is *half-life*: the time units taken to expend half its initial allocation of *energy*[3]. Learning in these experiments usually involved between 30 and 100 trials (before performance leveled off).

Distinct memories are used to encode the action-related rules and their components; however, they are linked to memories containing encodings of related visual structures and words. Thus a pre-condition representing `on(pitcher,table)` in the action memories as part of a rule is connected to the corresponding visual structures and through them to words.

Discussion

The algorithm shown in Figure 3 is similar to the reinforcement learning algorithms that have been proposed recently (e.g., [Sutton, 1990], [Whitehead and Ballard, 1990]). However, the approach outlined here also has a number of significant differences. First, we attempt to handle uncertainty in the world by keeping track of the reliability of each action as a weight associated with each link that represents an action in the action memories. Second, since the knowledge encoded in the state is based on and linked to other acquired knowledge, it is easier to merge different states in an effort to keep the size of the memories reasonable. Such generalization in the plan memories (based on prior visual–linguistic associations) is one of the crucial mechanisms that stem the combinatorial explosion in the number of states. We are also planning to add a short-term memory component to the action memories to handle situations where action selection may also depend on information perceived at earlier times. We elaborate on some of these issues below.

Generalization, and the Effect of Action Words

Rules which encode actions pertaining to specific objects are initially acquired; as the number of objects experienced by the system grows, mechanisms are needed to compact the memory structures. The generalization mechanisms embodied in CHILDLIKE are designed precisely to do this. The generalization processes implement specific rules from the symbolic inductive learning realm such as the *turning constants into variable* rule, *the closing interval* rule or the *climbing generalization tree* rule [Michalski, 1983]. The generalization tree itself is often learned via language inputs such as `apple is fruit` and `banana is fruit` after visual associations for `apple` and `banana` are learned. When generalizing rules pertaining to actions, the visual and linguistic memories can be exploited. For example, generalization may involve merging states and actions that refer to `apple`, `pear` and `banana`. From these, using the information in the linguistic memories, the system can create states and actions that refer to `fruit`. The weights on the links as well as the need-fulfillment potentials associated with each state are suitably averaged; and further experiences attempt to improve these values, if necessary. An important capability that facilitates such generalization is that actions can be split into their components: a pure action part, objects referred to by the action, and so on. Note that actions described so far are represented using English words for clarity; but the object component of the action (e.g., `apple` in an action represented by `Pick up apple`) is actually an abstract internal symbol that is grounded in terms of both the visual representation and the language equivalent, thus permitting the kind of generalization described above. Using such generalization mechanisms, the system can form rules that refer to classes of objects and their need-satisfying potentialities. An important point to note is that these generalizations are not trivial, since a language input such as `apple is fruit` is not readily interpretable with respect to the plan memories (which typically encode visual features or pointers to them in each state). However, the previously acquired visual–linguistic associations enable going from linguistic descriptions to visual features (and vice versa); this mutual grounding seems to aid each component of the system. States are also merged based on other factors such as visual similarity (e.g., the states involved may refer to `stool` and `chair`, which share some physical attributes), common need-fulfillment (e.g., the states fulfill the `rest` need) and implication of the same action (e.g, the states share the action `Sit on ...`). Generalization keeps the sizes of the acquired memories down to realistic levels. A slight reduction in performance abilities may sometimes be manifested due to generalization. For example, a critical action such as `Peel banana` does not have a counterpart for apple and pear, as experienced by CHILDLIKE. So after generalization, it was found that the `Peel ...` step was skipped over implicitly for all fruits, including a banana. Figure 5 shows the qualitative effect of generalization on CHILDLIKE's action memories; results of experiments with a few hundred examples (or experiences) are described in [Mani, 1992] (in preparation).

Another mechanism that encourages parsimonious representations is the use of *extraneous states*— complete representations of these states are not stored, and they usually encode the results of actions with low weights (or certainty) attached to them. In Figure 4, two examples of

[3] Energy is just a simple function of the inverse of the need levels.

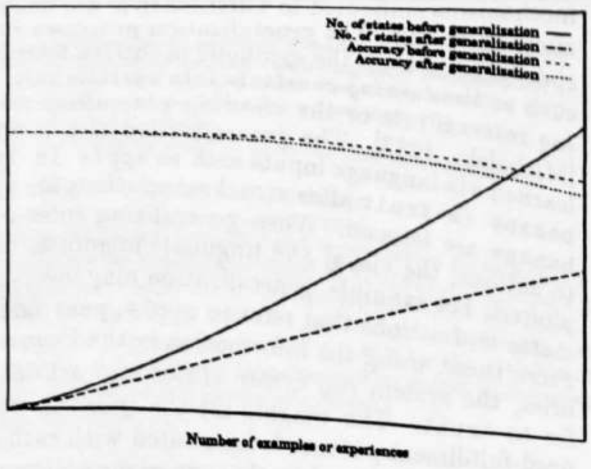

Figure 5: Effect of Generalization on Action Memories

extraneous states (denoted *Se1* and *Se2*) are shown. Extraneous states can be merged without much care; however, keeping a few extraneous states (as opposed to the extremes of storing just one, or all) has the effect of providing weak contexts for reasoning. Preliminary experiments using extraneous states indicate that *perceptual aliasing*[4] may actually turn out to be beneficial in certain cases (for a discussion of the problems usually caused by perceptual aliasing, see [Whitehead and Ballard, 1990]). Another important issue that is under examination is the learning of links between actions and simple linguistic descriptions of them. Teaching the system words about the various actions is important for two reasons. First, the memories acquired by CHILDLIKE will ground actions in terms of words. It is hoped that this will obviate the building in of actions. As noted earlier, in the current version, actions (but not their effects) are encoded in the system *a priori*. Second, the ability to use words about actions provides an effective way of communicating actions to be executed. Even when CHILDLIKE does not have the necessary motor capability, an action can be effectively achieved by dictating it to another agent that possesses the requisite motor skills. Such dictation can take place using the medium of a language that was acquired by the two agents in similar environments.

Using a Short-term Memory Component

The role of short-term memory in these tasks is obvious—e.g., if the system has to execute a number of "eating steps" each of which reduces hunger gradually, it should continue and finish eating before attending to other pressing needs such as `sleep`. Also, if the system notices the potential of a certain state to quench its `thirst`, while attending to the most pressing `hunger` need, it might be useful to remember the state so that it can come back to it after attending to the `hunger` need. For example, in Figure 4, this translates to going from state *S5* to state *S1* when an action such as `Move to refrigerator (from table)` is not highly implied. Note that this is not an unrealistic scenario even after the apple is eaten, since CHILDLIKE does an approximate match between world states and learned internal plan states. A simple implementation of short-term memory may consist of some selected state information (e.g., that state *S1* has a high `thirst`-satisfying potential) or that a particular action is reliable (i.e., has a very high weight associated with it). Storing and interpreting such information in the short-term memory can be achieved by modifying the action selection algorithm (Step *2a* in Figure 3). Several other approaches to using short-term memory to improve overall action selection are being tested. One is to perform deeper search into the plan memories (e.g., to see whether the target state *S1* above can be reached easily from *S5*)—the rationale behind this is that the current situation may be opportunistic and hence, not fully reflected in the backed-up values of the need-satisfying potentials.

[4] the phenomenon manifested by a many-to-one mapping from world states to the learner's internal states.

Other Issues

One point that needs to be stressed is that CHILDLIKE's `curiosity` need is slightly different from its other needs (such as `thirst`). The `curiosity` need dominates only when other needs do not, and aids in exploring perceptual states (and actions) that CHILDLIKE may otherwise ignore.

Simple trial-and-error learning appears to be a good mechanism for initial acquisition of knowledge (remember that CHILDLIKE starts out with no *a priori* rules). Once some initial planning knowledge has been acquired, a more deliberative algorithm could be introduced to perform action selection, since the system stores the effect of actions (or at least the effect of reliable actions).[5] Under such a scenario, the system may monitor the state of the world every n steps, where n is a parameter that encodes caution and is dynamically set as a function of the uncertainty expected in the environment and the reaction speeds required; learning an optimal value for this parameter is a good area for future explorations.

Using extraction and aggregation mechanisms, similar to those used in building visual and linguistic structures, to build macro-operators or hierarchies of actions is another issue that merits further exploration.

Conclusions

A survey of conventional planning techniques (for example, see [Allen *et al.*, 1990]) reveals two important related issues. The first is that planning is a hard problem, if only because of the potentially large search space involved. The second is that the environment the agent faces often changes in unpredictable ways, and this uncertainty greatly lowers the utility of planning.

An attractive approach is to learn rules for both perception and planning that can be quickly accessed and applied, choosing actions with very little (or no) deliberation, and constantly checking their effects via perception. This is exactly the design philosophy behind

[5] A purely reactive planning system learns the mapping from a set of states $\mathcal{S}$ to a set of actions $\mathcal{A}$ (the $\mathcal{S} \rightarrow \mathcal{A}$ mapping); CHILDLIKE, however, learns a partial $\mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ mapping in addition to the $\mathcal{S} \rightarrow \mathcal{A}$ mapping.

CHILDLIKE, which does not plan in the conventional sense. CHILDLIKE attends to the most pressing need at any point, keeping in short-term memory other pressing needs and possible satisfiers or actions. Moreover, it perceives the world after each step or action.

The current version of the system uses a high-level visual input, but interesting features can be propagated to this layer using a massively parallel, hierarchical structure of processes which starts with real images. Such a recognition-cone based architecture has been successfully employed for rapid recognition of large, digitized TV-frame-like images [Li and Uhr, 1987]. The serial depth of such a system is logarithmic in the size of the sensed input array, and such an architecture appears to roughly mirror the constraints established from neuroanatomical studies of the brain (for a fuller discussion, see [Uhr, 1987]). The CHILDLIKE system is an attempt to build on this parallel-hierarchical framework to handle language inputs and further (as described in this paper) to build and use memory structures that encode learned rules about actions and needs, exploiting the bias provided by similarly learned visual-linguistic associations.

References

[Allen *et al.*, 1990] J. Allen, J. Hendler, and A. Tate. *Readings in Planning.* Morgan Kaufmann, San Mateo, CA, 1990.

[Drescher, 1987] G.L. Drescher. A mechanism for early Piagetian learning. In *Proceedings of AAAI-87*, 1987.

[Dyer, 1991] M.G. Dyer. Symbol grounding in the blobs world with a neural/procedural model. In *Proceedings of the US-Japan Workshop on Language and Vision, CRL, New Mexico State University*, 1991.

[Feldman *et al.*, 1990] J.A. Feldman, G. Lakoff, A. Stolcke, and S.H. Weber. Miniature language acquisition: A touchstone for cognitive science. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society.* Lawrence Erlbaum Associates, 1990.

[Firby, 1987] R.J. Firby. An investigation into reactive planning in complex domains. In *Proceedings of AAAI-87*, 1987.

[Georgeff and Lansky, 1987] M. Georgeff and A. Lansky. Reactive reasoning and planning. In *Proceedings of AAAI-87*, 1987.

[Kaelbling, 1987] L. Kaelbling. An architecture for intelligent reactive systems. In M. Georgeff and A. Lansky, editors, *Reasoning about Actions and Plans.* Morgan-Kaufmann, Los Altos, CA, 1987.

[Li and Uhr, 1987] Z.N. Li and L. Uhr. Pyramid vision using key features to integrate image-driven bottom-up and model-driven top-down processes. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(2), 1987.

[Mani and Uhr, 1991a] G. Mani and L. Uhr. Integrating perception, language handling, learning and planning in the CHILDLIKE system. In *Working Notes of the AAAI Spring Symposium on Integrated Intelligent Architectures*, 1991. Also in the SIGART special issue on *Integrated Cognitive Architectures*, August 1991.

[Mani and Uhr, 1991b] G. Mani and L. Uhr. Perception-mediated learning and reasoning in the CHILDLIKE system. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society.* Lawrence Erlbaum Associates, 1991.

[Mani, 1992] G. Mani. *Learning Language about Objects and Using this Language to Learn Further: The CHILDLIKE System.* PhD thesis, University of Wisconsin, Madison, WI, 1992. In preparation.

[Mason *et al.*, 1989] M.T. Mason, A.D. Christiansen, and T.M. Mitchell. Experiments in robot learning. In *Proceedings of the Sixth International Workshop on Machine Learning*, 1989.

[Michalski, 1983] R.S. Michalski. A theory and methodology of inductive learning. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach.* Tioga, Palo Alto, CA, 1983.

[Okada, 1991] N. Okada. Aesopworld: An intellectual and emotional agent. In *Proceedings of the US-Japan Workshop on Language and Vision, CRL, New Mexico State University*, 1991.

[Shen, 1989] W. Shen. *Learning from the Environment Based on Percepts and Actions.* PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 1989.

[Siskind, 1991] J.M. Siskind. Naive physics, event perception, lexical semantics and language acquisition. In *Working Notes of the AAAI Spring Symposium on Machine Learning of Natural Language and Ontology*, 1991.

[Sutton, 1990] R.S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, 1990.

[Uhr, 1978] L. Uhr. 'Recognition Cones' and some test results. In A.R. Hanson and E.M. Riseman, editors, *Computer Vision Systems.* Academic Press, New York, NY, 1978.

[Uhr, 1987] L. Uhr. *Parallel Computer Vision.* Academic Press, Boston, MA, 1987.

[Whitehead and Ballard, 1990] S.D. Whitehead and D.H. Ballard. Active perception and reinforcement learning. *Neural Computation*, 2(4), 1990.