Integrating Unsupervised Learning, Motivation and Action Selection in an A-life Agent

Mark Witkowski

Department of Electrical and Electronic Engineering
Imperial College of Science Technology and Medicine
Exhibition Road
London SW7 2BT
United Kingdom
m.witkowski@ic.ac.uk

Abstract.

How can we expect an A-life Agent to learn how to perform tasks when it is not told what those tasks are, and it is not provided any indication or feedback as to its performance? This is at the heart of the unsupervised learning problem. If the Agent were able to learn in this manner, how could specific tasks be communicated to it? This is the Goal setting problem. Having been set a task, how would the Agent go about choosing things to do that will lead it to perform those tasks in an orderly manner? This is at the heart of the action selection problem.

1 Introduction

This paper presents an ingenious approach to solving these three closely related problems, unsupervised learning, goal setting and action selection, for a class of A-life Agents. The method presented, the *Dynamic Expectancy Model* (DEM), builds on the "sensori-motor" ([9]), "intermediate level cognitive" model ([3]), or "cognitive action theory" ([11]) approaches. Each uses a three-part representation for "knowledge" within the Agent – the "Schema". Such schema are formulated as "context – action – outcome" triples. The method presented here overcomes some significant technical problems of these earlier models. In particular we present considerably improved strategies for the creation of new schema-like objects and a robust and flexible mechanism for action selection.

This paper builds on the conjecture that the proper interpretation of this triple is that of an *expectation*, and that the strength of the connection between context and action, and the outcome should depend only on the predictive performance of the unit. A conventional view would hold that the strength of the connection should be related to any goal or task specific "desirability" of the outcome. In the DEM these expectation triples are referred to as *μ-hypotheses*. So called because each is a "micro observation" about the Agent and its world that can be created, verified ("corroborated") and used by the DEM. By adopting the predictive view strength changes can be made internally, and task independently, just by testing whether the predicted event did or did not occur, independently of reward or reliance on an external agent to indicate correctness. Strength changes can be applied immediately the prediction is corroborated, and are always attributed to the specific μ-hypothesis that made the prediction. This leaves the μ-hypothesis uncommitted to any particular goal; learning is not task directed. In this manner the unsupervised learning problem is addressed. Prediction is used in two ways to learn. First to corroborate μ-hypotheses – *tactical learning*, and second, by detecting unexpected events, to trigger the creation of new μ-hypotheses – *strategic learning*. This ability to learn in the absence of motivation is closely related to the *latent learning* phenomena, ([6], [10]).

From time to time the Agent will be called upon to perform certain tasks. By associating a degree of motivation to any particular outcome the Agent is signalled that it should try and achieve that outcome. In this manner, the goal setting problem is addressed.

There may be many possible choices of context and action that will lead to the desired outcome. The Agent may select amongst these primarily on the basis of the corroboration between these components. Where no action may be selected because none of the required contexts are available, then, as context and outcome are uniformly represented, the Agent may chain μ-hypotheses to form a "policy map" of options where μ-hypothesis contexts match the current circumstances. This chaining process is referred to as *spreading activation* ([8], [10]) in the DEM. The DEM provides a uniform measure (the *policy value*) which allows the Agent to select between competing alternatives for selection. In this manner the action selection problem is addressed.

Action selection and learning models based on Reinforcement Learning ([7], [13], [16]) propagate the effects of occasional reward backward to rank sense-act pairings in the form of a "policy map" according to iteratively developed estimates of future reward. Neural networks provide a sensed input to output mapping function which may be refined by a variety of well known learning methods ([5], [14]). Genetic Learning algorithms rank and refine sets of condition-action rules according to some "fitness function" (past payoff in the case of [4]). Classifier Systems ([1], [10]) combine the notion of propagating the effects of occasional reward, using a "bucket-brigade" method, for behavior selection with a genetic algorithm approach to create new classifier selection elements. Alternatively, action selection may be pre-defined, with no little or no learning component ([2], [8] and [15] for a useful review).

The primary purpose of this paper is to describe and define the Dynamic Expectancy Model mechanism, and the next sections provide a description of the individual processes that comprise the DEM. Each process is performed once in an extended "sense, corroborate, evaluate goals, create policy map, act, predict, learn" cycle. All the activities of the Agent are encapsulated into this cycle, which repeats *ad infinitum*. The paper concludes with a simple, but illustrative, example of the DEM in operation, demonstrating both latent learning and its ability to respond rapidly to changing motivations, followed by some discussion.

2 μ-Hypotheses, Signs and Actions - Making Predictions

All working information in the DEM is held in five main "Lists", the Hypothesis List, Sign List, Action List, Goal List and the Prediction List[1]. During each cycle the system tests the sensory apparatus of the Agent and every element of each list, modifying List and List elements (and selects an action to perform) according to the processes described next.

All μ-hypotheses known to the DEM at any time are retained in the *Hypothesis List* (abbreviated to $\mathcal{H}$). The form of the μ-hypothesis (h, $h \in \mathcal{H}$) is:

$$h\text{:}\;\; s' \wedge a \xrightarrow{t\pm\tau} s'' \qquad \text{(eqn. 1)}$$

Each μ-hypothesis should be read as an expectation of the form "performing the action a in the context of s' predicts the occurrence of the condition s'' at time *t* in the future". The time *t* is bracketed by ±τ, forming a range a times to generalise the prediction in the temporal domain ($t >> \tau$) and to overcome the effects of sensor sampling aliasing. The overall predictive ability for each μ-hypothesis, the strength of the predictive connection "→" is recorded in a numeric variable, the *corroboration measure* (C_m).

The context (s') and outcome (s'') terms in eqn. 1 are *Signs* drawn from the *Sign List* (abbreviated $\mathcal{S}$, thus $s' \in \mathcal{S}$, $s'' \in \mathcal{S}$). Signs both define and detect *situations* that can be recognised by the Model. Signs are derived from, and form the interface to the sensory apparatus available to a physical Agent. Signs are compound items, conjunctions of elemental sensory items ("Tokens"). At each cycle in the execution of the algorithm each Sign is either detected or is absent and so evaluates to *active* or *inactive* for that cycle. All active Signs for a cycle are held on the Active Sign List $\mathcal{S}^*$, a subset of $\mathcal{S}$. The first time a Token is encountered the DEM automatically creates a Sign (containing only that Token) and adds it to the Sign List. New Signs are also appended to a working list $\mathcal{S}^{new}$. This list drives the structural learning process. Tokens, Signs and actions have no *a-priori* meaning to the learning mechanism in the Model, and may be named arbitrarily to suit experimenter or user.

As a special case Signs will equate directly to a "State", a unique situation detected reliably by the Sign. A Sign may also equate to a "partially observed state" (P.O. State), a unique situation unreliably detected by the Sign. Equally, a Sign may just represent a collection of sensory conditions available to the Agent, from which it must create predictions of increasing reliability and repeatability. This notion of "State" is useful when evaluating the model formally, but it is a poor metaphor for a realistic agent problem. The latter case allows Signs to act as abstractions. Adding tokens to a Sign makes it more specific; deleting them makes it more generally

[1] *A note on notation.* Each of the lists is denoted by a single, upper case, calligraphic letter ($\mathcal{H}$ (Hypothesis), $\mathcal{S}$ (Sign), $\mathcal{A}$ (Action), $\mathcal{G}$ (Goal) and $\mathcal{P}$ (Prediction)). Individual elements are denoted by lower case letters (h, s, a, g and p respectively). Sub-sets of lists, and individual elements that must be identified across steps or cycles are indicated by superscripts (e.g. $\mathcal{S}^*$). The symbol '$\in$' may be read as "member of", '$\cup$' as "union of", '$\cap$' as "intersection of" and '$\wedge$' as "concurrently with". Other symbols are explained as they are encountered.

applicable. In these circumstances we would expect the formation of many "candidate" µ-hypotheses to be created, verified and possibly discarded before a viable Hypothesis List is formed.

Actions, (a; $a \in \mathcal{A}$, the *Action List*) define the activities the Agent may perform. Where Signs defined the interface to the sensory apparatus, actions connect the DEM to the Agent's physical actuators. Selected actions are placed onto the $\mathcal{A}^*$ (active) sub-list. Every action a has associated with it an *action cost*. The action cost measure indicates the relative effort that will be required to complete the action. Action costs may be expressed in any units (such as elapsed time or energy expended) that may be determined and consistently applied across all the elements of $\mathcal{A}$. The action cost measure will be used in the "cost estimation" process for goal directed action selection. The DEM also maintains a memory of recent activations and their associated timings for both Signs (from $\mathcal{S}^*$) and actions (from $\mathcal{A}^*$). Information held on these *activation traces* is used by the structural learning component to construct new µ-hypotheses.

A µ-hypothesis is deemed active (and so placed on $\mathcal{H}^*$) whenever both its context Sign (s') and its action (a) are active simultaneously, ($s' \in \mathcal{S}^* \wedge a \in \mathcal{A}^*$). A new prediction p is created and added to the *Prediction List* $\mathcal{P}$ for every instance of an activated µ-hypothesis. Note in particular that this mechanism is invoked for all µ-hypotheses that meet these criteria and that a prediction records three items. First the identity of the Sign predicted (s^p, s'' from the active µ-hypothesis). Second the time (derived from *t*, eqn. 1, and the current time) that the Sign is predicted to occur and third the identity of the µ-hypothesis that made the prediction (h^p). Elements of the Prediction List are active (and so placed on $\mathcal{P}^*$) when the time element recorded matches the current time within the bounds defined by τ. The presence of active predictions drives the corroboration process.

3 Corroboration - Tactical Learning

For each active prediction, the corroboration measure (C_m) of the µ-hypothesis responsible for the prediction (h^p) is modified according to:

$$C_m = C_m + \alpha(1 - C_m) \qquad \text{(eqn. 2)}$$

where the prediction was successful (that is, when $s^p \in \mathcal{S}^*$, s^p being the Sign recorded at the time of making the prediction), and

$$C_m = C_m - \beta(C_m) \qquad \text{(eqn. 3)}$$

where the prediction was unsuccessful ($s^p \notin \mathcal{S}^*$). Active predictions are discarded from $\mathcal{P}$ once this step is complete.

The positive *reinforcement rate,* α ($0 \le \alpha \le 1$), defines the rate at which successful predictions will strengthen C_m. Similarly, the *extinction rate,* β ($0 \le \beta \le 1$), defines the rate at which C_m will be weakened by failed predictions. Where no prediction was made the value of C_m remains unchanged. Sequences of successful (or unsuccessful) predictions give rise to the familiar negatively accelerating learning curve, the values being normalized such that C_m rises asymptotically toward 1.0 (or falls toward 0.0).

4 µ-Hypothesis Acquisition - Structural Learning

Prediction, or rather the failure to predict an event, drives the structural learning component of the DEM, which is responsible for forming new µ-hypotheses. The opportunity to create new µ-hypotheses is indicated by appearance for the first time of a previously unknown (“novel”) Sign or by the appearance of a known but unpredicted (“unexpected”) Sign. New, unknown, Signs trigger the *creation by novelty* method, recall that the first occurrence of a previously unknown Sign is recorded in $\mathcal{S}^{new}$ specifically to invoke this learning method. The appearance of an unpredicted, but previously known, Sign invokes the *creation by unexpected event* method. Unexpected Signs are detected by comparing the active Prediction List to the active Sign List and applying the method to the unpredicted residue ($\mathcal{S}^*$ - ($\mathcal{P}^* \cap \mathcal{S}^*$)).

In either method a new µ-hypothesis is constructed from the novel or unpredicted Sign as ‘s”, and a Sign (s’) and action (a) drawn respectively from the recorded activation trace of values in the Sign and Response Lists. The timing relationship (t and hence τ in eqn. 1) is derived from their relative positions in the respective memory traces. Note that the structural learning mechanism is independent of the source of the Signs and actions it will employ. Riolo ([10]) and Shen ([12]) have described broadly similar strategies for “rule” creation triggered by “surprise” events.

To limit the rate at which new µ-hypotheses are created the user may specify a *learning probability rate*, λ, which determines the probability with which a new µ-hypothesis will be formed given one of these opportunities to do so. The Dynamic Expectancy Model also defines methods for differentiating partially effective µ-hypotheses by making their component Signs more or less specific by adding (or removing) Tokens, and removing ineffective µ-hypotheses. The requirements for such additional processes are considered further in [17].

5 The Action Selection Mechanism

At each execution cycle the Agent must have some action to perform. Normally, the Dynamic Expectancy Model operates in two distinct modes for action selection (1) *Goal directed selection*, (2) *Exploratory selection.* Whenever goal directed action selection is selected, the algorithm attempts to construct a *Dynamic Policy Map* (DPM) from which it may select an action. The construction and use of the DPM is described later.

Where no goal is set the system selects actions in exploratory mode. In the current implementation, exploratory selection is made on a random basis. Regardless of how the action was selected the learning mechanism continually monitors the activities of the Agent, and corroborates existing µ-hypotheses and creates new ones according to the learning strategies described.

Goals - instructions to the system to select actions with respect to some purpose - are held on the *Goal List* ($\mathcal{G}$). Individual goals are drawn from the Sign List. Placing a Sign on the Goal List is a signal to the Agent that it should be motivated to select actions that cause that Sign to become activated (i.e. appear on $\mathcal{S}^*$). Each goal Sign on $\mathcal{G}$ is assigned a *priority*. The goal with the highest priority at any time is referred to as the *top-goal*. Once a goal Sign has been activated (i.e. it appears on $\mathcal{S}^*$), it is deemed *satisfied* and removed automatically from the Goal List. The next highest priority goal becomes top-goal, or the Goal List becomes empty.

Purposive goals are, by definition, largely domain specific - they serve some purpose. The Dynamic Expectancy Model provides two distinct routes to setting goals. (1) Goals may be programmed into or be inherent to the Agent, or (2) they may be imposed externally by directly manipulating $\mathcal{G}$. The former route equates directly to our intuitive notion of primary reinforcer. Some things, such as food for a hungry animal, or water for a thirsty one, inherently motivate because they are "programmed" to do so. In a mobile robot context the detection of a "battery_low" Sign may cause an "on_charge" Sign to be placed on the Goal List, possibly with a priority related to the extent of battery discharge. The latter route provides an experimenter with a method with which to manipulate the goal-driven behaviour of the Agent directly.

6 Building the Dynamic Policy Map

Whenever a top-goal is available, the DEM will attempt to create a Dynamic Policy Map (DPM) to form a sequence of links from every other Sign in $\mathcal{S}$ to the Sign currently set as top-goal. The DPM is conveniently represented as a graph where context Signs associated with individual µ-hypotheses represent the nodes and actions embedded within individual µ-hypotheses the arcs. The DPM is created by a process of *spreading activation* from the top-goal. The method used to construct the DPM is a modified form of the standard breadth-first graph search/construction algorithm. Each arc has associated with it a *cost estimate*, C_e, value. This cost estimate is computed from the given action cost of α and the C_m (eqns. 2 and 3) value defined earlier:

$$C_e \leftarrow \text{action\_cost}(\alpha) / C_m \qquad\qquad \text{(eqn. 4)}$$

Consider a situation where C_m is simply p(number of successful predictions|total predictions made) by a µ-hypothesis - the probability that the µ-hypothesis predicts correctly. The cost estimate value C_e is then reasonably interpreted as the total estimated cost for the average number of attempts that must be made with the given µ-hypothesis to achieve the transition. A similar interpretation may be placed on the case for C_m shown in eqn. 4, with the proviso that the "averages" are now biased towards recent experiences.

Each node (Sign) in the graph will acquire a *valence level*, *v*, indicating the number of arcs, *n*, that must be traversed to reach the top-goal "node". The top-goal has a valence level of zero, the s' Sign of any µ-hypothesis that leads directly to the goal (i.e. where s'' = top-goal) a valence level of 1, and so on. The *policy value*, P_v, of any node s' at level n in the DPM is then expressed as a summation of individual estimated costs ($(C_e)^v$) by:

$$P_v(s') \leftarrow \min \left(\sum_{v=1}^{v=n} (C_e)^v \right) \qquad \text{(eqn. 5)}$$

The policy value for each Sign s' implicated in the DPM is computed by adding the cost estimate for its transition to the minimum cost of the path to its s'' node. If a lower cost path is encountered the spreading activation is re-activated for that node to minimize path costs at higher valence levels.

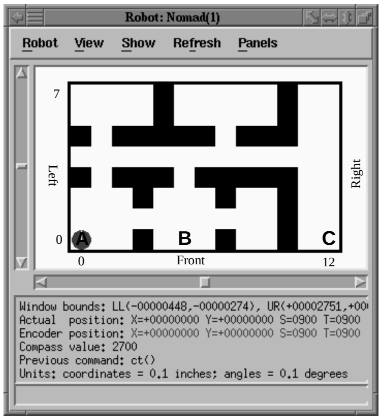
Construction of the Dynamic Policy Map is complete when there are no further µ-hypotheses that can be implicated, and no further path cost minimization can occur. Following construction of the DPM the Agent has an estimate of the total "cost" to attain the top-goal for every s' implicated in the map. Once DPM construction is complete, the DEM may simply select the action associated with ($\min(P_v(s' \in S^*))$ in the µ-hypothesis containing that s' and route it to the Agent.

If a currently active Sign is included as a node in the DPM (DPM $\cap$ S^*), the action α included in the µ-hypothesis arc associated with the Sign node with the lowest P_v (eqn. 5) is selected. This is the action with the lowest overall estimated cost to achieve the top-goal. Where there is no intersection between the set of active Signs and nodes on the DPM, an exploratory action is selected. These new actions will either (1) reach the goal directly, (2) lead to a situation where a action selection from the DPM may continue, or (3) cause new µ-hypotheses to be created, which in turn expands scope of the DPM. The DPM is recomputed frequently, whenever goals change, new µ-hypotheses are formed or existing ones have undergone sufficient corroboration to indicate that a different solution path may be preferable.

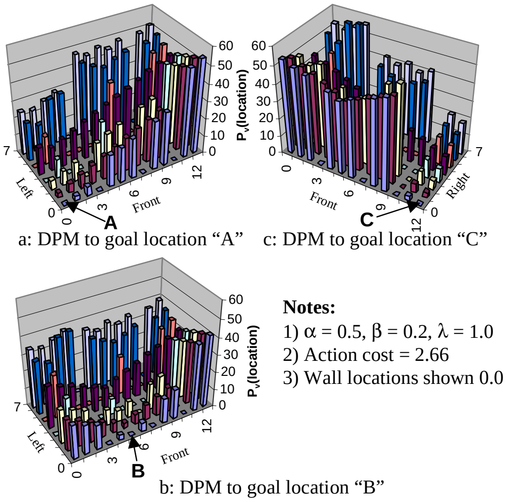
7 An Example

To illustrate the two essential of the properties of the DEM algorithm, unsupervised learning and policy map generation, figure 1 shows a simulated robot learning task for navigation. The robot may recognize some 74 individual locations on a grid within the environment. These equate directly to individual Signs (and, in this instance, to "states"). The robot is supplied with four actions with which to traverse between these locations. These experimental conditions (but not the actual layout) accurately reflect those described by Sutton ([13]). We note that, in simulation, each action takes 2.66 seconds on average. This is used as the action cost. Initially the robot is allowed 2000 exploratory knowledge gathering (randomly selected) actions. This is a period of latent learning, no goal is set nor any other form of reward provided during this period. Nevertheless a Sign List and a corpus of µ-hypotheses is constructed in the Hypothesis List by the novelty and unexpected event methods, and subsequently corroborated by the tactical learning method. Initially both Lists were empty. Random exploration is inefficient in this environment, the robot tending to become "trapped" in "rooms" for extended periods, but this number of actions ensures that every location is visited more than once. Ordered exploration techniques can improve the time to complete the exploration procedure, but otherwise have marginal impact on the underlying learning mechanism.

Immediately following this period of exploration the Sign detecting location “A” is established as top-goal by the experimenter. The DEM computes the Dynamic Policy Map cost estimates, which are visualized in Figure 1a. The column heights corresponding to the total cost estimate from that Sign/location (their position) to the goal. After so much exploration, the task is learnt well. The Dynamic Policy Map corresponds closely to our intuition of a “cost gradient”, flowing from “room” edges, through “doors”, along the central “corridor” etc. The use of a navigation based example here allows us to think of the Dynamic Policy Map as a “Cognitive Map” ([11]) in a quite literal sense. In other agent based applications there is no clear mapping of cost estimate to place and satisfactory visualization is harder to achieve. Next the Sign detecting location “B” is made top-goal, and the DPM immediately adopts the cost estimate configuration of Figure 1b. Figure 1c shows the DPM cost estimates when the Sign for location “C” is established as top-goal.



```
Window bounds: LL(-00000448,-00000274), UR(+00002751,+00
Actual  position: X=+00000000 Y=+00000000 S=0900 T=0900
Encoder position: X=+00000000 Y=+00000000 S=0900 T=0900
Compass value: 2700
Previous command: ct()
Units: coordinates = 0.1 inches; angles = 0.1 degrees
```



a: DPM to goal location “A”

c: DPM to goal location “C”

b: DPM to goal location “B”

Notes:
1) α = 0.5, β = 0.2, λ = 1.0
2) Action cost = 2.66
3) Wall locations shown 0.0

Figure 1: Robot Task and Dynamic Policy Maps for Different Goal Locations

Not shown in these visualizations is the action that is associated with each Sign for each of the three different Dynamic Policy Maps. In each case the action that will be selected with the agent at any particular location represents the first action on the path with the lowest total estimated cost. As the goal, and hence the DPM changes, the action selected in response to any particular Sign stimulus may therefore change dramatically. The spreading activation algorithm is fast, the DPM in these examples being computed in “real-time” (<10mS on a 166MHz Pentium P5 running Linux). [17] describes an extensive series of investigations using the DEM under a variety of learning conditions, imposed uncertainty and varying environments.

8 Discussion

The Dynamic Expectancy Model addresses and unifies three issues central to the construction of a fully autonomous A-life Agent. First, that of true unsupervised learning based on particular properties of the prediction process. Second, the ability to define motivations that drive the overt behavior of the Agent. Third, a mechanism to select actions according to a uniform measure relative to the Agent's current state of knowledge (the Hypothesis List), its current motivations (the top-goal) and the situation as the Agent perceives it (S^*).

The contribution afforded by this work is not so much in the knowledge that all the things it does are possible, but in the careful selection and tight integration of each of the parts to produce an Agent controller capable of robust action selection performance and self-sustaining, self-sufficient learning. The DEM is distinguished from the overwhelming majority of learning reactive systems[2] in its decisive adoption of internal prediction to corroborate schema like connections and away from reliance on external or task specific reward.

The structural learning component of the DEM represents a significant advance over that used by its (arguably) closest precursor system, due to Drescher ([3]). Drescher's computationally intensive "marginal attribution" schema learning process is discarded in favor of the use of the learning by novelty and unexpected event. Marginal attribution requires extended periods of exploration to first construct viable action – outcome pairs, followed by further periods of exploration to establish and subsequently extend the context part of the new schemas. In the DEM µ-hypotheses (the schema like objects) are created as the opportunity arises, but always from a combination of events that can occur in the environment (they did occur – and were stored in the activation traces.)

This method is also clearly distinct, in its directness, from the mutation and crossover approach to structural learning used in genetic algorithms and classifier systems ([1], [4], [5], [10]). Riolo ([10]) describes CFSC2, extending the classifier system model with a three part schema like representation, which adopts a forward chaining approach to detecting possible future rewards rather than the goal directed backward chaining approach used in the DEM.

The Dynamic Policy Map is not a plan; it is a temporary mapping between sensory conditions and actions to take. It does not define a path from start to finish; rather it is a characterization of the Signs known to the system according to an estimated cost from that Sign to the primary source of motivation, the top-goal. In use, it is similar to a reactive look-up table: if this is the current situation, then select this action. In this respect, it is similar to the policy map of, say, Watkins' ([16]) or Sutton's ([13]) *Q*-learning based algorithms. It is profoundly different in that the policy is computed (and re-computed) frequently and quickly, relative to a specific goal, rather than as an iteratively formed static policy estimating future discounted and anonymous rewards.

In comparative tests, [17] using the highly stylized navigation tasks of the type described in section 7, the DEM can show dramatic performance improvements (up to the order of 40:1) over a conventional *Q*-learning algorithm (compared to results presented in [13]). This is due almost entirely to the fact that the connection strength (C_m) can be updated at every step, rather than occasionally at the end of a long

[2] A notable exception being Tani's neural network based "prediction learning" [14] method.

sequence of actions. In more realistic environments other factors tend to mask these apparent gains.

The DEM can be seen as an exemplar of Grefenstette’s notion of “Anytime Learning” ([4]). New μ-hypotheses can be formed at any point in the Agent’s existence, immediately extending its behavioral repertoire. Corroboration is also an ongoing process throughout the Agent’s “lifetime”.

The Dynamic Expectancy Model defines a style of autonomous Agent controller, it has been implemented and tested in a range of conditions. It learns autonomously and has been found to be flexible and responsive to changing conditions. Work on the Model continues.

References

[1] Booker, L.B., Goldberg, D.E. and Holland, J.H: Classifier Systems and Genetic Algorithms, in: Carbonell, J.G. (Ed.) Machine Learning: Paradigms and Methods, The MIT Press, (1990) 235-282

[2] DeCugis, V. and Ferber, J.: An Extension of Maes’ Action Selection Mechanism for Animats, 5th Int. Conf. on Simulation of Adaptive Behavior (1998) 153-158

[3] Drescher, G.L.: Made-up Minds: A Constructivist Approach to Artificial Intelligence, The MIT Press, Cambridge, MA (1991)

[4] Grefenstette, J.J. and Ramsey, C.L.: An Approach to Anytime Learning, 9th Int. Workshop on Machine Learning (1992) 189-195

[5] Grossberg, S.: Birth of a Learning Law, Boston University Technical Report CAS/CNS-TR-97-017 (1997) [http://cns-web.bu.edu/Profiles/Grossberg/Learning.html]

[6] Hergenhain, B.R. and Olson, M.H.: An Introduction to Theories of Learning, Prentice Hall, New Jersey (1993)

[7] Humphrys, M.: Action Selection Methods using Reinforcement Learning, 5th Int. Conf. on Simulation of Adaptive Behavior (1998) 135-144

[8] Maes, P.: A Bottom-up Mechanism for Behavior Selection in an Artificial Creature, 1st Int. Conf. on Simulation of Adaptive Behavior (1991) 238-246.

[9] Mott, D.H.: Sensori-motor Learning in a Mobile Robot, Dept. Comp. Science and Statistics., Queen Mary College, University of London, Ph.D. Thesis (1981)

[10] Riolo, R.L.: Lookahead Planning and Latent Learning in a Classifier System, 1st Int. Conf. on Simulation of Adaptive Behavior (1991) 316-326

[11] Roitblat, H.L.: Cognitive Action Theory as a Control Architecture, 1st Int. Conf. on Simulation of Adaptive Behavior (1991) 444-450

[12] Shen, W-M.: Autonomous Learning from the Environment, Computer Science Press, New York (1994)

[13] Sutton, R.S.: Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming, Proc. 7th Int. Conf. on Machine Learning, 216-224 (1990)

[14] Tani, J. and Nolfi, S.: Learning to Perceive the World as Articulated: An Approach for Hierarchical Learning in Sensory-Motor Systems, 5th Int. Conf. on Simulation of Adaptive Behavior (1998) 270-279

[15] Tyrrell, T.: Computational Mechanisms for Action Selection, University of Edinburgh, Ph.D. thesis (1993)

[16] Watkins, C.J.C.H.: Learning from Delayed Rewards, King' s College, Cambridge University, Ph.D. thesis (1989)

[17] Witkowski, M.: Schemes for Learning and Behaviour: a New Expectancy Model, Dept. of Computer Science, Queen Mary Westfield College (Univ. London), Ph.D. Thesis (1997)