Anticipatory Behavior in Adaptive Learning Systems

ABiALS 2004 Workshop Proceedings

Martin V. Butz

Olivier Sigaud

Samarth Swarup (Eds.)

Table of Contents

Program Committee

Anticipatory Brain Potentials: An Electrophysiological Insight into the Anticipatory Behavior of Adaptive Learning Systems

Stevo Bozinovski, Liljana Bozinovska[1]

Mathematics and Computer Science Department, South Carolina State University, USA
and Electrical Engineering Faculty, Sts Cyril and Methodius University, Skopje, Macedonia
[1]Institute of Physiology, Medical Faculty, Sts Cyril and Methodius University, Skopje, Macedonia
sbozinovski@scsu.edu

Abstract: The paper describes a new approach toward the study of anticipatory behavior in adaptive learning systems, an approach based on electrophysiological evidence of anticipatory behavior of the human brain. The basic idea is to study brain potentials related to anticipation of some event. Here we describe our work with the CNV anticipatory potential. We extended the classical CNV paradigm introducing a brain-computer interface that allows observation of the adaptive change of the cognitive environment of the subject. We obtained a cognitive phenomenon which generates a trace which we denoted as Electroexpectogram (EXG). It shows a learning process represented by the learned anticipation. Emotional Petri Nets are used in explanation of the EXG paradigm in terms of consequence driven systems theory.

1. Introduction

We understand anticipatory behavior as the one that considers its consequence; the mechanism itself may be learned or inherited. Related definition [11] is that anticipations are predictions of future values that have an effect on a current behavior of the agent. The classical work of Rosen [17] defines "anticipatory system as a system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accord with the model's predictions pertaining to a later instant" (p. 339). As pointed in [10] Rosen related the anticipation to the concept of final causation of Aristotle. We emphasize the concept of consequence [5] rather the concept of prediction in explaining the concept of anticipation. As pointed in [15] anticipation is expressed through consequences. In another view [8] anticipatory learning system is a system that learns a predictive model of an encountered environment specifying the consequences of each possible action in each possible situation. Various approaches have been used in study of anticipatory behavior and a good review of current efforts is given in [9].

Here we will focus on an electrophysiological approach. We will observe some electrophysiological phenomena that are evidence of anticipatory behavior. It is reported in [15] that a previous work [14] reports on a readiness potential which is observed in an EEG trace as result of preparation for a voluntary event. Instead of observing the readiness potential (BP, Bereitschaftspotential, [13]), in this work we observe the CNV [19] potential, which actually contains evidence of anticipation of a voluntary action.

We will start our presentation with taxonomy of brain potentials, emphasizing existence of anticipatory potentials. The classical CNV (Contingent Negative Variation) potential is an example of an anticipatory brain potential. Then we will describe our experimental setup which introduces a feedback in the CNV paradigm. We describe a design of a brain computer interface system that will allow a study of learned anticipation in a human brain. In such an experimental we obtain a chart which we denote as electroexpectogram (EXG). The whole procedure of obtaining the electroexpectogram we denote as EXG paradigm. EXG is an evidence of *adaptation of a brain to its cognitive environment*. At the end, using Emotional Petri Nets, we discuss the relation of the EXG paradigm to consequence driven systems theory.

2. Anticipatory Brain Potentials

Figure 1 shows a taxonomy of the brain potentials. In general, brain potentials could be divided into spontaneous and event related. Event related potentials can be divided into pre-event and post event. Post-event (evoked) potentials can be divided into exogenous (reflex reaction to the event) and endogenous (cognitive processing because of the event). Pre-event (anticipatory) potentials can be divided into expectatory and preparatory.

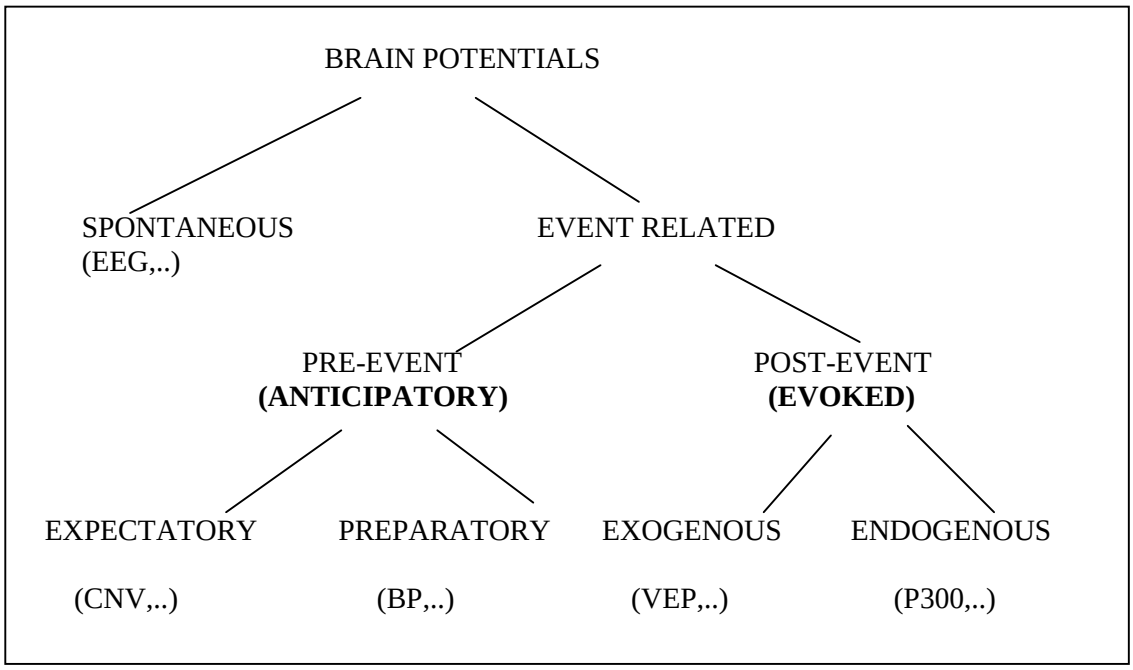

Figure 1. A taxonomy of the brain potentials

According to Figure 1 *we introduce a structure* into the concept of anticipation. Anticipation includes at least expectation and preparation. Preparatory potentials are initiated internally, by the *will*. The expectatory potentials are initiated by some *expected external event*. A prominent example of this group is the well-known Contingent negative Variation (CNV) potential, but actually as we can see the CNV potential includes also the preparatory components. Figure 1 also shows the place of the well known brain potentials such as BP, VEP, and P300, in this taxonomy.

3. The CNV Potential

The Contingent Negative Variation (CNV) potential appears in an experimental procedure known as CNV paradigm, originally proposed by Walter et al. [19]. It is a procedure (e.g. [18], [12], [1]) in which, after several repetition of a sequence of two stimuli S1 and S2, a slow negative potential shift (the CNV) appears in the interstimulus S1-S2 interval. The negative slow potential shift is interpreted as expectancy wave and is related to learning. Figure 2 shows the procedure, result, the time scale, and the original interpretation.

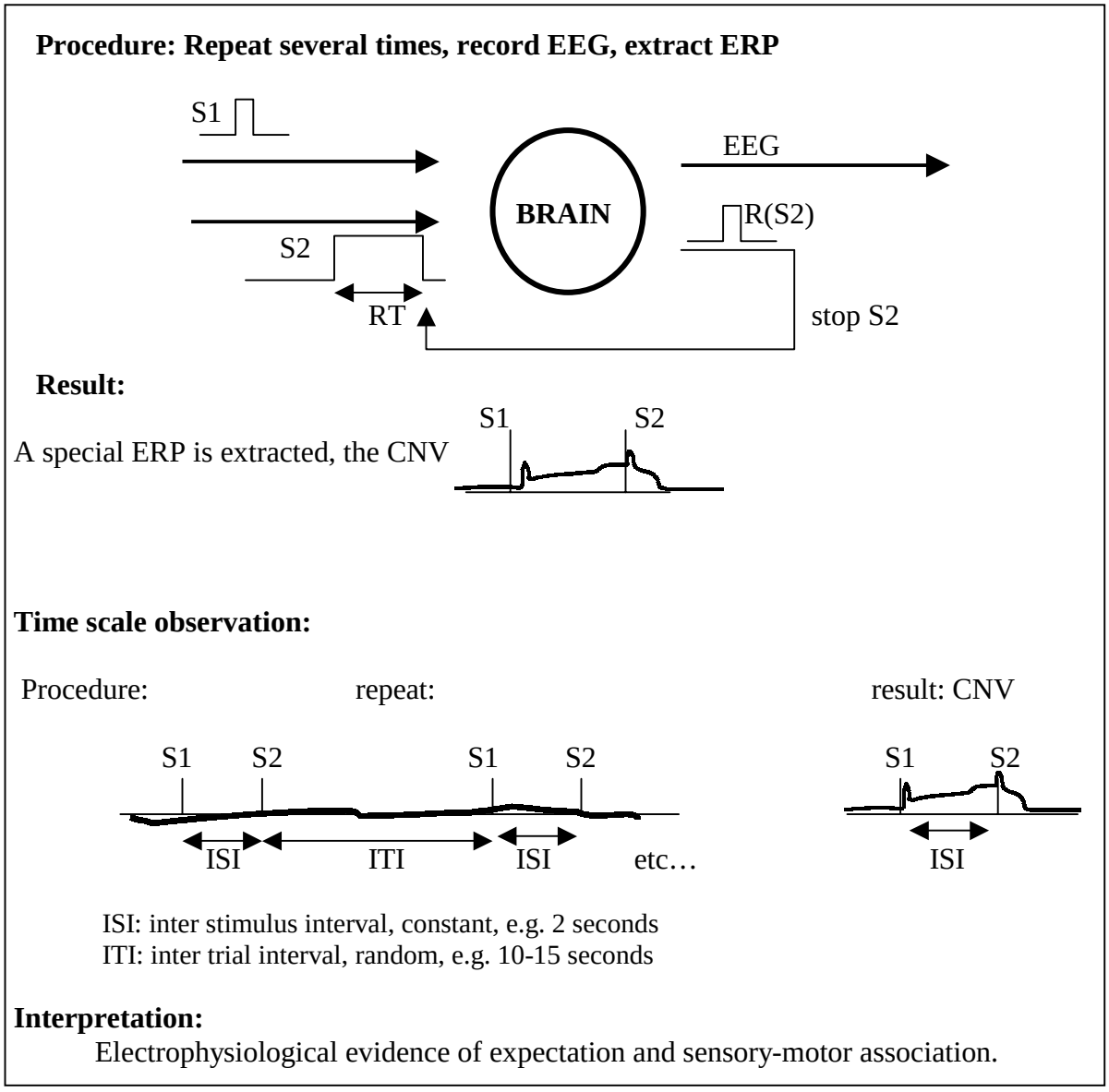

Figure 2: The CNV paradigm: Procedure and interpretation

Figure 2 shows the classical [19], *open-loop design* in which after S1, the brain is expecting S2 and is preparing to produce reaction R on S2 (denoted as R(S2)). After repeating several times S1-S2, the ERP between S1 and S2 gradually develops to be a recognizable CNV.

Contingent Negative Variation (CNV) can be defined as an increasing negative shift of the cortical electrical potentials associated with *an anticipated response to an expected stimulus.* It is a bioelectrical evidence of an anticipatory state of the brain.

In terms of cognitive psychology, the classical CNV paradigm can be viewed as a reaction time (RT) task. There is a neutral warning stimulus S1, and an annoying stimulus S2 to which the subject should react by pressing a button R(S2) in order to stop the unpleasant event. The subject *learns to anticipate action at S2 given warning at S1*. This is clearly an anticipation learning task, *A*(R(S2)|S1).

In our experiments the subject is only told that whenever an unpleasant sound is heard, to pres the button. The subject rather quickly learns that the unpleasant sound S2 comes after S1. That learning process generates an anticipatory process that expects occurrence of S2 after S1 and prepares an action toward S2. The anticipatory process is recorded as CNV potential.

The CNV potential has a morphology as shown in Figure 3. The form shown here is obtained in our experimental work, where S1 and S2 are sound stimuli. Slightly different CNV shape is obtained when S2 is a visual stimulus [19].

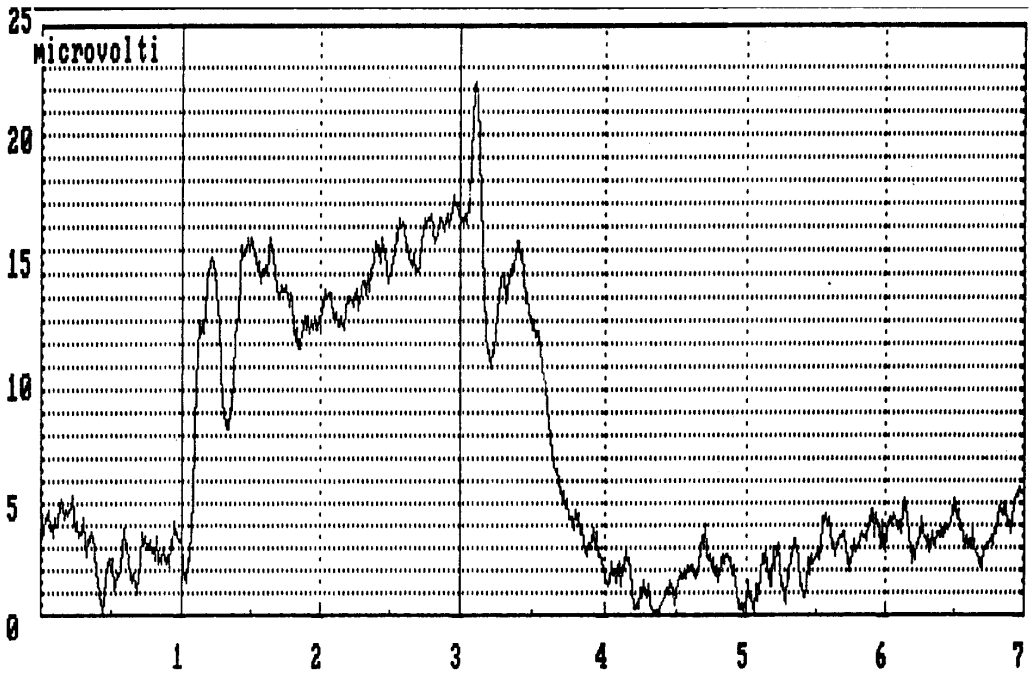

Figure 3. The CNV morphology. S1 is applied at second 1 while S2 is applied at second 3 of a 7-seconds CNV trace shown here. In the interstimulus interval S1-S2 we can assume signals exogenous to S1, endogenous to S1, anticipatory to S2, and possibly other cognitive components. In some cases, not in this particular one, after S2 appears additional signal negativity, denoted as post imperative negativity (PINV).

The CNV paradigm produces a number of evoked potentials related to S1 and S2. However we are interested only in the *anticipatory ramp* between S1-S2. It contains the expectancy components, preparatory components and possibly other cognitive components.

4. EXG paradigm: Designing an electrophysiological agent-environment interaction

In order to study anticipation and learning process in a human brain we designed a closed loop, brain-computer interface as shown in Figure 4.

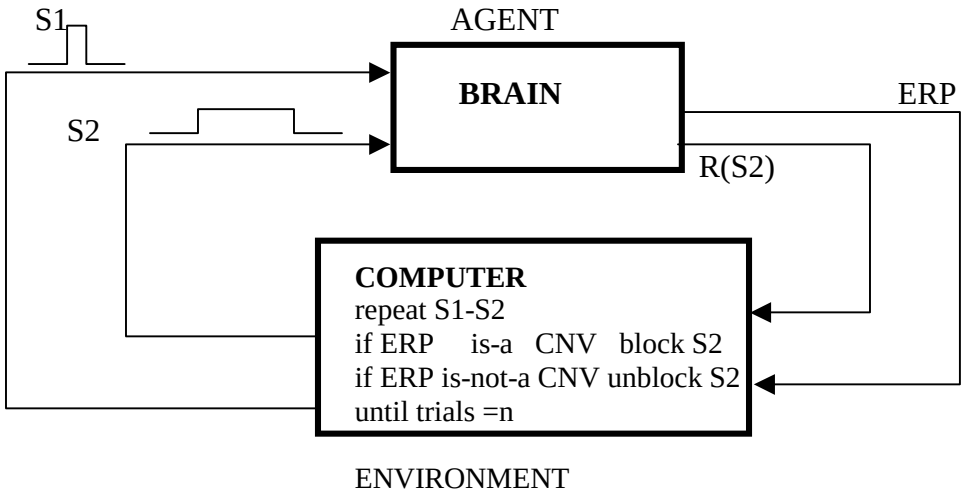

Figure 4. The EXG paradigm: An agent environment interaction setup

As Figure 4 shows we designed a brain-computer interface which we call EXG paradigm, in which the brain receives two types of stimuli and produces two types of actions. One action is a direct brain-computer interface action, an event related potential (ERP), and the other is press of a button. The environment produces two types of sound stimuli: S1 has neutral emotional value and S2 has unpleasant emotional value, which the agent (brain) would leave with a press of the button.

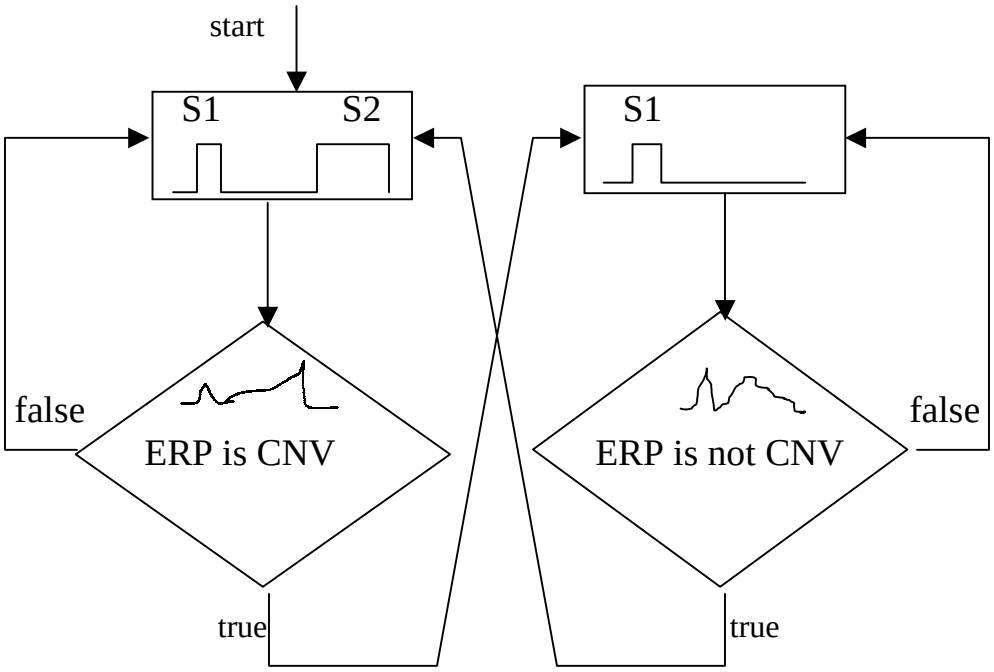

Figure 5. The EXG paradigm: Flow-chart representation

In this design, whenever the computer recognizes that ERP is a CNV potential, it blocks the S2. That produces the CNV to disappear. When that is observed by the computer, the S2 is unblocked and the whole procedure is repeated again. A parameter that represents the CNV potential can be either the CNV amplitude at S2, or CNV slope between S1 and S2, or some other parameter. Using the control theory terminology, the EXG paradigm shows a *cognitive servo system* maintaining *the anticipatory level* at some setpoint. Figure 5 shows another, algorithmic representation of this agent-environment interaction.

5. Observing the anticipatory process: Electroexpectogram

Figure 6 is a snapshot from a computer screen from our experimental work, monitoring the described agent environment interaction and the learning process in the agent (brain). Each small window represents a trial.

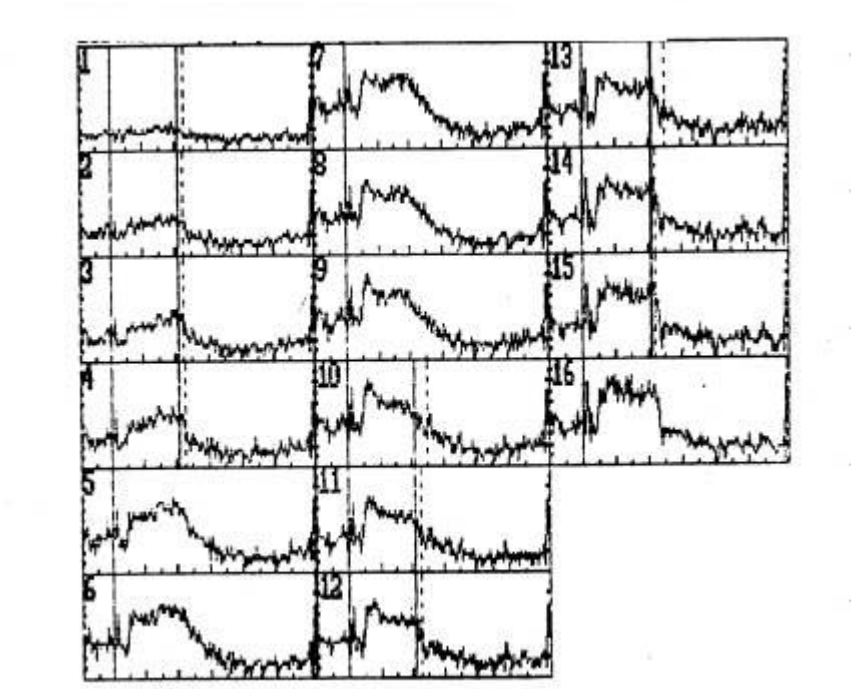

Figure 6. The learning process represented by the anticipatory brain potential

Figure 6 shows the first 16 trials of an ongoing experiment, which would last 100 trials. We can see how an ERP is rising up from the EEG, and develops a recognizable CNV. Here the first CNV is formed rather fast, after 4 trials. (Usually subject learns after more than 8 trials. Here is a case of a subject familiar with the procedure.) The S1 signal is always present (the left vertical bar in each trial window), while S2 is sometimes omitted (the second solid vertical bar). The S2 is followed by the subject's reaction time (the dashed vertical bar following S2).

In this experiment the computer blocks S2 in the fifth trial, unblocks it in the 10-th trial, blocks it again in the 16-th trial. Note that in trial 10, the subject is rather *surprised* by unexpected occurrence of S2 which can be seen by a much *longer reaction time*. We can see how ERP develops toward CNV, then degrades, then develops again and so on, a cognitive oscillatory phenomenon is in progress

Having series of trials in which we observe the CNV dynamics, we can plot a parameter that would represent that dynamics. Plotting a representative CNV parameter across trials gives a representation of a *cognitive state of anticipation across trials*. The obtained curve is shown in Figure 7.

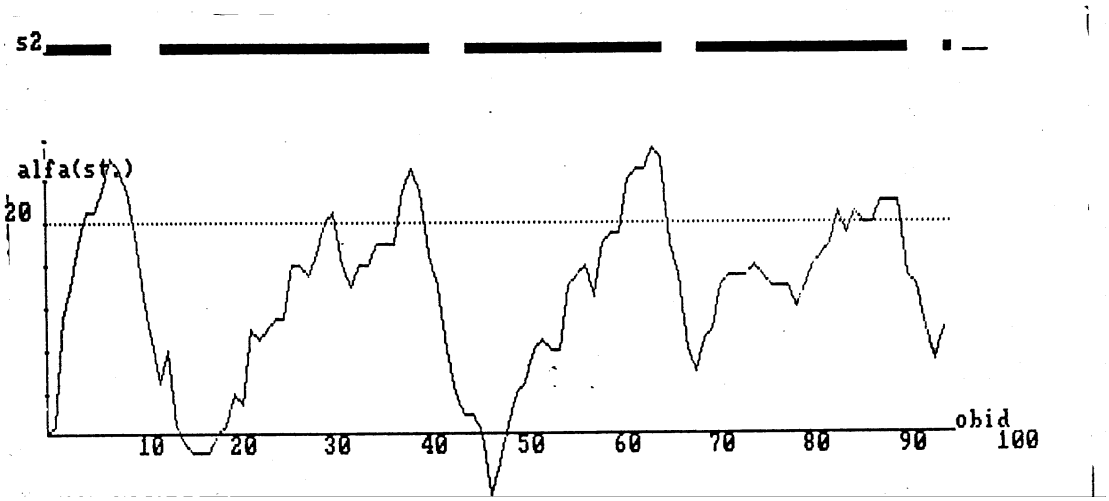

Figure 7. The Electroexpectogram (EXG). Monitoring an anticipatory process in a human brain

In this particular case, the CNV slope is used as a representative parameter of a CNV trial. When the slope has a threshold value (e.g. 3,6 μV/sec) the computer switches off S2, and switches it on again when the angle is below the threshold. In other words, whenever CNV appears, it turns off the reason of its appearance, the signal S2. The EXG represents the anticipation level of the agent, measured across trials, and shows *how fast the subject adapts to the changing environment*, i.e. how fast it learns to expect and not to expect (which is also an expectation of the complement event).

Figure 8 shows a polygraph chart, the same picture but shown together with other measured parameters, in this case with the amplitude of the CNV just before S2, and the reaction time of pressing the button.

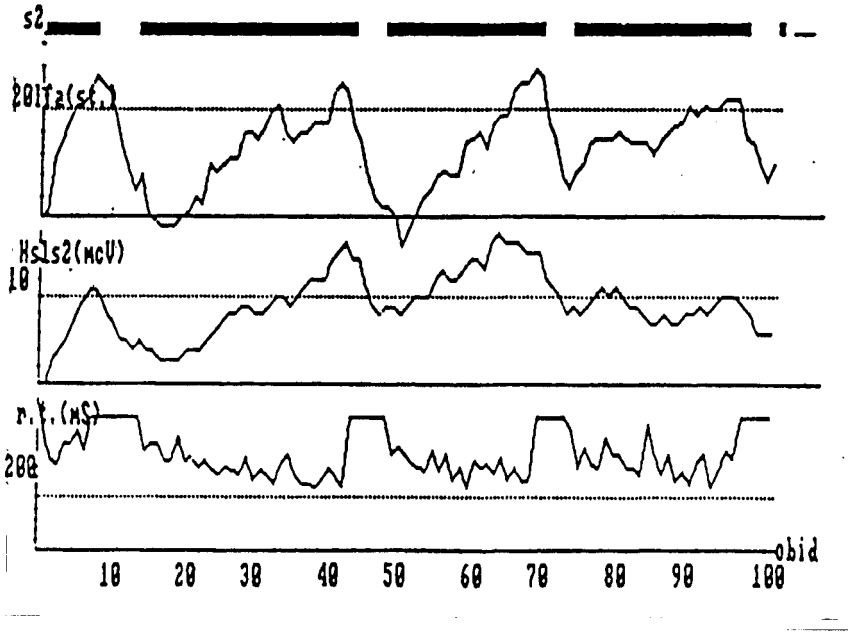

Figure 8. EXG polygraph: CNV slope, CNV amplitude, reaction time

Figure 8 shows examples of EXG curves obtained from healthy subjects. Our investigation shows that a healthy subject always develops an anticipatory oscillator. We also observed that it is not always the case in patients with diagnosed neurosis or with diagnosed epilepsy

Observing Figure 8 we would like to point out the relevance of reaction time to the concept of anticipation. It is shown that whenever the subject anticipates action to S2 the reaction time is shorter, and when the subject is surprised by an appearance of S2, the reaction time is longer. An useful insights from the EXG experiments would be that in addition to the EXG curve, the RT curve also represents a kind of learning curve, giving evidence of anticipation in an adaptive learning systems.

The technical details of our work in terms of signal processing and pattern recognition algorithms have been described elsewhere [2]. Let us note that anticipatory behavior oscillators in mathematical terms were studied in [10].

6. Electroexpectogram : Anticipatory behavior in a consequence driven system

In this chapter we will very briefly discuss an EXG trial in terms of consequence driven systems theory [3], [5], [6], [7]. We introduced Emotional Graphs [3], Emotional Petri Nets [5], and Motivational polynomials [6],[7] as a graphical tools for describing *emotional states* and *motivated behaviors* in consequence driven systems. The basis of our Emotional Graphs and Emotional Petri Nets are emotionally colored facial expressions (Figure 9). Petri nets [e.g. 16] are often used to represent processes with states and events that change states.

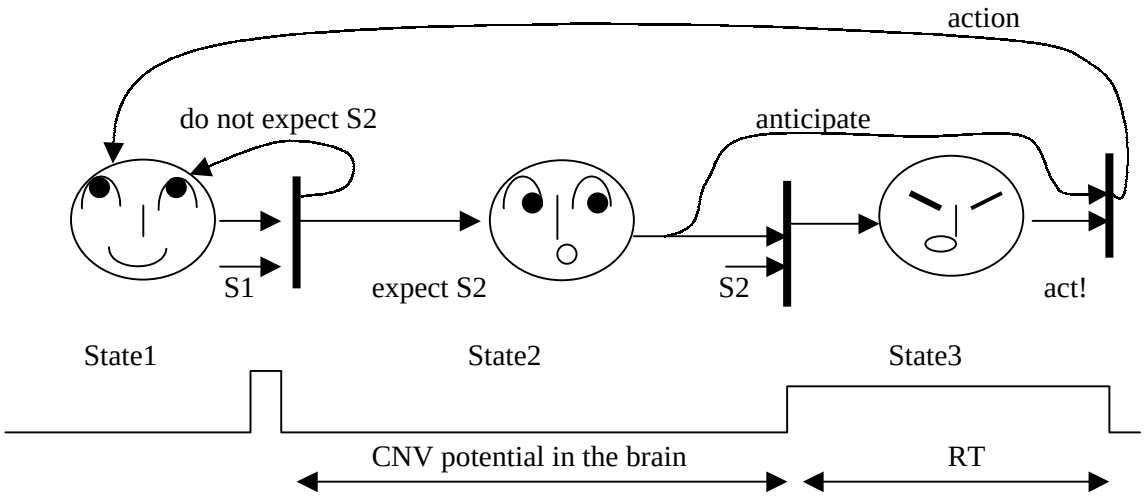

Figure 9. Emotional Petri Net representation of the anticipatory behavior during an EXG trial

Figure 9 shows an Emotional Petri Net consisting of three states (Petri circles) and three transitions (Petri bars). The Petri bars compute the AND function in order to make a transition possible. The transitions are nondeterministic either/or type.

Here is represented a subject being in State1 until S1 signal is received. That event will either leave the subject in the same state, or will transfer the subject into State2, representing the fact that the subject has learned to expect the event S2 after S1. State2 will wait for event S2 but will also feedforward an anticipation of action that will respond to S2. Once S2 is received, both the previous preparation for action and the present signal saying it is time for action will contribute toward generation of an action that will transit the subject back to State1.

7. Discussion and Conclusion

Here is presented a new, electrophysiological approach toward the study of anticipatory behavior in adaptive learning systems. We provided experimental results that show brain electrical activity in anticipatory behavior both in terms of expecting something and in terms of preparing to act in order to avoid/approach that. The paper also presents our taxonomy of brain potentials emphasizing the anticipatory ones. We recognize the CNV potential as the best example of a brain potential that could be related to the concept of anticipation.

In such a way the paper sheds light on a *structure into the concept of anticipation*, proposing that anticipation is related to an action that is prepared to meet an expecting event. We propose that while prediction is an event-oriented concept, anticipation is an action-oriented concept. That is a possible solution in the dilemma of distinction between anticipation and prediction. Another view may be that anticipation is the prediction of a given event including the preparation to act towards that event; in such a way anticipation may be viewed as special kind of prediction.

Here we described a brain-computer interface (BCI) as a model of agent-environment interaction. Contemporary the research in BCI is mostly concerned in using the brain waves to control devices such a robots, an early work in the area being our work of moving a robot by brain potentials [4]. Here we described a novel idea showing that BCI techniques could be used to modify a brainwaves based task in order to manipulate the subject, rather then having a subject using brainwaves to directly manipulate a task.

Finally we described the EXG experimental trial within the framework of Consequence Driven Systems theory. Here we used Emotional Petri Nets to model the EXG trial, to recognize *emotional states* of the agent, and to emphasize the anticipation feedforward phenomenon in a consequence driven system.

We believe that this work shows a new direction of research in anticipatory behavior of adaptive learning systems. It point toward the structure of an anticipatory process and a distinction between anticipation and prediction. It also connects the research in the area of brain-computer interface (BCI) to the research in the area of adaptive learning systems.

Acknowledgement. The authors would like to thank the anonymous reviewers for useful comments that were used to improve the paper.

References

[1] Bozinovska L. The CNV paradigm: Electrophysiological evidence of expectation and attention" Unpublished term paper in physiological psychology, Beth Powell, instructor, Psychology Department, University of Massachusetts at Amherst (1981)

[2] Bozinovska L., Bozinovski S., Stojanov G. Electroexpectogram: Experimental design and algorithms. Proc. IEEE International Biomedical Engineering Days, Istanbul, (1992) p. 58-60

[3] Bozinovski S.: A Self-learning System using Secondary Reinforcement. In: R. Trappl (ed.) Cybernetics and Systems Research. North-Holland (1982) 397-402,

[4] Bozinovski S., Sestakov M., Bozinovska L. Using EEG alpha rhythm to control a mobile robot. Proc 10th Annual Conference of the IEEE Engineering in Medicine and Biology Society, New Orleans, (1988) vol 3: 1515-1516

[5] Bozinovski S. Consequence Driven Systems. Gocmar Press (1995)

[6] Bozinovski S. Motivation and emotion in the anticipatory behavior of consequence driven systems. In M. Butz, O. Sigaud, P. Gerard (eds.) Proceedings of the Workshop on Adaptive Behavior in Anticipatory Learning Systems. Edinburgh, (2002) p. 100-119

[7] Bozinovski S. Anticipation driven artificial personality: Building on Lewin and Loehlin. In M. Butz, O. Sigaud, P. Gerard (eds) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003) p. 133-150

[8] Butz M., Goldberg D. Generalized state values in an anticipatory learning classifier system. In M. Butz, O. Sigaud, P. Gerard (eds) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003) p. 282-301

[9] Butz M., Sigaud O., Gerard P. (eds.) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003)

[10] Dubois D. Mathematical foundations of discrete and functional systems with strong and weak anticipations In M. Butz, O. Sigaud, P. Gerard (eds) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003) p. 110-132

[11] Fleischer J., Marshland S., Shapiro J. Sensory anticipation for autonomous selection of robot landmarks. In M. Butz, O. Sigaud, P. Gerard (eds) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003) p. 201-221

[12] Holscher C. Long-term potentiation: a good model for learning and memory? Progress in Neuro-psychopharmacology and Biological Psychiatry (1997) 2:47-68

[13] Kornhuber H., Deecke L. Hirnpotentialanderungen bei willkurbewegungen und passive bewegungen des menschen: Bereitschaftspotential und reafferent potentiale. Pflugers Archive(1965) 284: 1-17

[14] Libet B. Unconscious cerebral and the role of conscious will in voluntary action. Behavioral and Brain Sciences (1985) 8: 529-566

[15] Nadin M. Not everything we know we learned. In M. Butz, O. Sigaud, P. Gerard (eds.) Anticipatory Behavior in Adaptive Learning Systems. LNAI 2684, Springer Verlag, (2003) p. 23-43

[16] Peterson J. “Petri Nets”, ACM Computing Surveys 9(N3), 1977

[17] Rosen R. Anticipatory Systems. Pergamon Press (1985)

[18] Tecce J. CNV and physiological process in man” Physiological Bulletin (1972) 77:73-108

[19] Walter G., Cooper R., Aldridge V., McCallum W. Contingent negative variation: An electric sign of sensory-motor association and expectancy in the human brain. Nature (1964)

The dynamics of appropriation

Emmanuel Dauc

UMR Movement and Perception
Faculty of Sport Sciences
University of the Mediterrannean
163 avenue de Luminy, CP 910
13288 Marseille cedex 9
France

Abstract. In the framework of the embodied approach, this paper tries to emphasize that, aside from its closed loop organization, a body is a *growing* object. A body can indeed recruit external elements to make them part of its own structure. The general question we address is the following : to what extent does the process of recruitment relate, in a direct or indirect fashion, to the cognitive activity? We show in this paper that the structure of the interaction between an agent and its environment is comparable to a recruitment process, taking place both at the level of the nervous activity, and at the level of the group of individuals. During an interaction process, the coordination between the agent's body and its surrounding is interpreted as an intermittent expansion of the agent's body. For that reason, we say that the agent *appropriates* the surrounding elements. The choice of a relevant partner among various partners illustrates the ability to anticipate, through the co-construction of action, some desirable forthcoming interaction. The agent indeed takes part in the realization of its own predictions. The paper finishes with some remarks about the learning processes, seen as the crosswalk between nervous-environment processes and body growth processes.

1 Introduction

The "embodied" approach is a well-known alternative to classical cognitive models. Under that approach, an agent identifies with its body, and the cognitive activity identifies with the continuous trade-off between the dynamics of self-construction and the body/environment structural couplings [1]. Under the embodied approach, a cognitive process is not something located inside the agent. It relies on the continuous interleaving of physiological and environmental autonomous flows, with the body's skin acting as a barrier or a frontier preserving the body's operational closure [1].

We will try in this paper to explore some of the embodied approach conceptual extents, according to the questions of learning and anticipation, and their implementation in artificial devices.

2 Model setting

The aim of this paper is not to give definite or elaborate models, but to accompany some of the proposed ideas with a light mathematization. We thus introduce this paper with a global model of interaction, with the use of some of the concepts of classical control theory (input, output, functional diagrams) and dynamical systems formalism. A control system is thus a *model* of interactions between a model of controller (or agent) and a model of the environment. The agent perceives the environment through sensors and acts on the environment through actuators. The environment evolves under the actions of the agent, and those actions are updated according to the sensory flow.

In this presentation, the agent's and environment evolutions are co-dependent, i.e. belong to the same process, whose evolution both originates from the two sides. It results, formally speaking, by a simple splitting in two parts of a single autonomous dynamical system composed of an environment and an agent for which we suppose we have a precise state description (see also [2]). An *interaction system* can classically be described by the set of equations:

$$\begin{cases} u_{\text{out}}(t) = k_{\text{out}}(x_{\text{in}}(t)) \\ \frac{dx_{\text{in}}}{dt} = f_{\text{in}}(x_{\text{in}}(t), u_{\text{in}}(t)) \\ u_{\text{in}}(t) = k_{\text{in}}(x_{\text{out}}(t)) \\ \frac{dx_{\text{out}}}{dt} = f_{\text{out}}(x_{\text{out}}(t), u_{\text{out}}(t)) \end{cases} \tag{1}$$

Where X_{in} is the internal state space, U_{in} is the internal input state, $f_{\text{in}} : X_{\text{in}} \times U_{\text{in}} \rightarrow X_{\text{in}}$ is the internal transition function, $x_{\text{in}} \in X_{\text{in}}$ is the internal state, $u_{\text{in}} \in U_{\text{in}}$ is the internal command, X_{out} is the external state space, U_{out} is the external input state, $f_{\text{out}} : X_{\text{out}} \times U_{\text{out}} \rightarrow X_{\text{out}}$ is the external transition function, $x_{\text{out}} \in X_{\text{out}}$ is the external state, $u_{\text{out}} \in U_{\text{out}}$ is the external command,

- The mapping $k_{\text{out}} : X_{\text{in}} \rightarrow U_{\text{out}}$ represents the transformation of the agent's state space to the commands space, i.e. the various forces which activate the agent's body. The outer process is thus dependent on the internal state, through its "actions" $u_{\text{out}}(t)$.
- Conversely, the mapping $k_{\text{in}} : X_{\text{out}} \rightarrow U_{\text{in}}$ represents the transformation from the external space to the agent body-centered space, basically corresponding to the signal sent to the agent by its various sensors. The inner process is dependent on the external state, through its "perceptions" $u_{\text{in}}(t)$.

The functional scheme of this system is represented in figure 1.

In the most general case, every subsystem (agent and environment) owns one or several hidden processes, whose dynamical evolution actively take part in the production of new perceptions and new actions. Given some perceptual configuration, the agent's reaction is not strictly predictable. Conversely, the environment does not always react the same. The global evolution is not exclusively under the control of the agent, or under the control of the environment. The global evolution is as much driven by the hidden processes than by the commands.

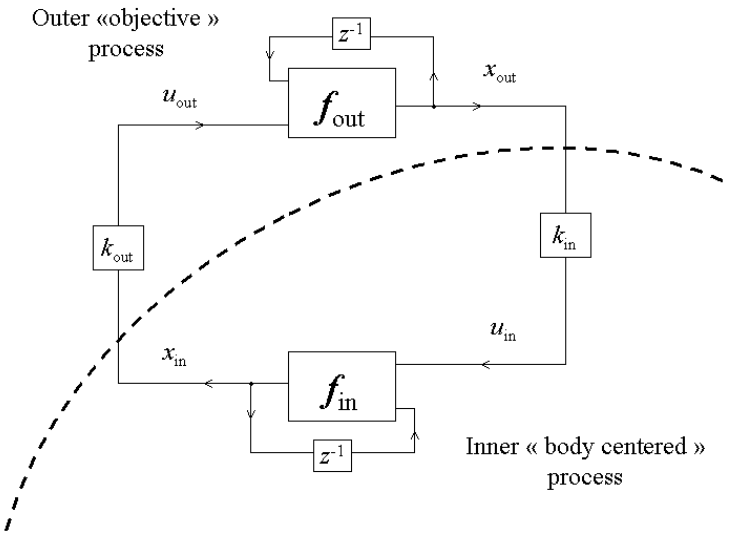

Fig. 1. Functional diagram of an interaction system.

3 The processes of recruitment

This section is about growth processes and the ability of dynamical models to relate to such processes.

3.1 Recruitment and growth

Animals or plants life relies on the permanent process of body self-construction and growth. A distinction can be made between the body macroscopic structure and the process of recruitment and extension which continuously modifies the body. A body "survives" as long as it can access to a minimum amount of nutriments and energy. The assimilation of external elements is thus a fundamental process by which a given body *recruits* external element as new body compounds. The necessity of a permanent renewal of internal components makes the body very dependent on its environment. Our proposal in this paper is to identify *the process of recruitment and growth* as the fundamental level from which every further cognitive refinement develops.

Under this schematic view, the bodies can face different situations which may facilitate or prevent the fundamental growth process. Any nutritive assimilation facilitates this internal growth process. Any aggression or nutriment lack is in conflict with the current internal growth process. Reciprocally (in a recurrent fashion), the internal growth process is selective and opportunist. The body develops in a way which facilitates compatible assimilations, and prevents destructive intrusions. For instance, a plant basically grows where the energy

and nutriments sources are the more abundant (roots extension, foils development,...). Different mechanism prevent and repair intrusions and aggressions.

In terms of modeling, the most popular models of plant growth relate to a fractal process of replication of branch growth patterns [3]. Those models however only care about the structures, and do not take into account the properties of the substrate. Here, we use as a reference the models of aggregation processes, which have been given for instance in Turing reaction-diffusion equations [4]. In an aggregation process, the various particles belonging to the substrate obey to two antagonist tendencies, i.e. a strong local attraction tendency and a weak distant repulsion tendency, which results in the constitution of macroscopic aggregates. In that model, the current aggregates can recruit new individuals in the constitutions of its own "body".

3.2 Nervous dynamics

Contrarily to plants, animals own a nervous system, so that they can produce movements. Plants thus passively consume the energy of light and soil nutriments, while animals actively seek for sources of food and energy. The nervous system allows high speed processes which operate in parallel with the original self-construction growth process. The nervous process is accompanied by muscles contractions and body movements. The nervous activity (neurotransmitters release and ionic electric transmission) is persistent during the whole life of the body (i.e. more or less quiescent). This persistent activity needs external stimulation (i.e. perception-induced neurotransmitters release) in order to maintain its activity, like bodies need nutriments.

This analogy helps to to analyze the nervous dynamics in the terms of a recruitment/aggregation process. New stimulations thus "feed" the nervous dynamics, and a lack of stimulations diminishes the nervous activity. Conversely, the nervous activity propagates through the nervous system, like fire, and recruits new stimuli as components for its "combustion". The activity grows better where the stimulations are, and thus extends toward the current stimulations. The property of aggregation also relates to the property *synchronization*, which has been extensively observed within local assemblies [5] or between different assemblies [6]. The dominant hypothesis is that such synchronization expresses the involvement of a structure in a process taking place at the global level. This involvement manifests in a measurable cooperation between the local and the global level, so that a certain dynamical pattern "recruits" participants to its own activity (and conversely some neurons "choose" to take part in the global process, modifying their dynamics and being modified by the global dynamics).

This property of synchronization has been extensively established as a rather common phenomenon taking place in various models of artificial neural networks [7–11], from binary models [7] to elaborate stochastic and sparsely connected integrate and fire models [9]. More generally, the model given in eq.(1) is well suited for the representation of such dual nervous/body interactions. The internal variables may correspond to the nervous dynamics, while the external variables may

correspond to the agents body and also partly to the environment. The internal process of recruitment may be obtained with topologically organized neural maps [12, 13], where incoming stimuli can operate as seeds for new aggregates of neural activity. The two subsystems may mainly differ by the integration times, i.e. the internal integration time may be of one or several order faster than the external ones, mimicking the difference between internal and external "reaction times".

4 Interaction and appropriation

4.1 Partners and perceptibility

The way the nervous system reacts to external events is dependent on the structure of the animal's body. I call here "*partner*" an object which is affordant [14] with the body, i.e. which facilitates a structural coupling with the agent's body. It can touch its senses, or even make an attempt on its life (ravine, poisoned food, predator)[1].

The way bodies perceive their surrounding objects or partners is first determined by their respective physical properties, forms and spatial extension. Some surrounding objects are perceptible, others are not. In the natural world, perceptibility is basically rooted on the symmetry/disymmetry of the facing bodies in one or several physical dimensions, i.e. relative spatial extension, relative speed, relative illumination, and also in the symmetry/disymmetry of the individuals sensors. So, two facing bodies may easily ignore each other for basic physical reasons, possibly colliding by chance.

The main aspect however of perceptibility is that bodies are built in a way that favors the perception of the most relevant features in their environment. For evolutionary and/or adaptive reasons, attractive or aversive sources of food, attractive or aversive partners, are more salient in their perceptual field. This basically means that bodies are prepared to interact with *elective* partners. The bodies are predisposed to perceive the items they can interact with. So, in a schematic view, a body is surrounded by various partners, which are potentially eligible for interaction. The eligibility of a partner means that a given partner does not by itself necessarily trigger a pre-definite reaction. At a given moment, a process of decision takes place where a partner is elected, among others, for an interaction.

How do a particular body "take the decision" to select an elective partner? This relates to the question of action selection and decision processes. From a global point of view, one can not say that a certain decision is strictly taken "inside" the body. One should better say that the environment dynamics facilitates a certain series of interaction patterns, and reciprocally the body's internal dynamics facilitates a certain series of interaction patterns, and the decision relies on a mutual process of convergence toward a compromise:

[1] On the contrary, an infectious agent may not be considered as a partner, as it can not touch the agent's senses

– At the nervous scale, only the more *desirable* perceptual compounds are chosen by the nervous process. Reciprocally, the desirable items are defined by the process of selection which occurs in the current nervous activity.
– At the body scale, only the more desirable partners are chosen. Reciprocally, the desirable partners are the ones which find themselves chosen by the animal.

The election of a partner for interaction is comparable to a recruitment process. In a particular bodies/environment context, an interaction pattern is formed and various partners are recruited to participate to that process. Such recruitment process is a mix between individual choices and global entrainment.

Given a certain interaction system (1), can we measure whether the two subsystems cooperate or, on the contrary, disturb each other? This question relates to the question of the *coupling*, or matching, between the two sub-processes. Such matching may be measured by the way the two sub-processes display common features in their state space, like periodicity, synchronicity[2]... In the general case, one can not strictly define a causal path in the process giving rise to a certain interaction pattern. The two parts are equally involved in this process, i.e the two subsystems may end up on a compromise, so that they mutually resonate with the other, or, on the contrary, end up on a dissension, so that they tend to produce, for instance, a chaotic pattern of interaction (see also [16]). In the first case, the two processes are easily penetrated by the other's influence. In the second case, the two processes remain blind to the other's influence.

4.2 The appropriation process

At a given moment, some of the elective partners are "elected" to take part in the ongoing interaction process. That moment is accompanied by a specific nervous pattern. A perceived partner fundamentally appears in the form of various sensory compounds (i.e. neurotransmitters). At the local scale of the nervous activity, some of those latent sensory compounds are integrated in the current nervous dynamics ("chosen" as relevant nervous compounds), and take part in the current nervous dynamics. The nervous activity thus recruits some new sensory compounds among several sensory appeals. The elected partners (their sensory compounds) are "used" and manipulated by the nervous dynamics, i.e. they are integrated in the internal/external process of action construction. They thus "belong" to the nervous dynamics.

In parallel, the body participates to an interaction pattern. Under this interaction pattern, the various partners are found to act coherently according to the body's current dynamics. In other terms, the body and its surroundings are synchronized. In accordance with the internal dynamics, it appears that the various partners *virtually belong to the animal's body*. In that sense, the animal

[2] The measure of the coupling between the two dynamics may empirically rely on a comparison between the embedding dimension of the global trajectory D, and the embedding dimensions of every local trajectory D_{in} and D_{out}. This point will not be developed in this paper. See also [15].

appropriates its partners. For that reason, the interaction moments can also be called "appropriation moments", where the agent's body virtually extends to one or several partners taking part in the interaction. This virtual body and the neurons internal dynamics operate at the same speed. There is thus a correspondence to be found between the neuronal aggregation dynamics and the "aggregation" that comes with the current interaction.

5 Anticipation and learning

5.1 An uncertain body in an uncertain environment

A world item, as a partner, can locally represent a future moment (a prey for instance represents a future meal). This is true only in a particular context (a fed up predator does not consider a potential prey as a future meal). The current partner can thus be seen as an anticipatory clues of the following events, in the particular context of the current interaction. The agent's anticipations are closely linked to the agent's decision. The election of a particular partner corresponds to the election of a particular forthcoming pattern of interaction. The agent thus actively takes part in the realization of its own predictions, and there is no separation between anticipation and action decision.

The way the agents *choose* partners is often simple and non controversial. In many situations, there is no serious trouble in doing what the senses suggest to be done, so that almost automated responses can be triggered. The set of familiar partnerships may be seen as what the agent feels as "belonging" to its own world : usual places, usual faces, usual habits. The usual partners trigger usual responses and usual interactions.

On the contrary, unknown territories, unknown environments, unpredictable reactions are the major part of everyday life. The persistent environment uncertainty requires persistent attention! Reciprocally, internal processes often present some uncertain aspects : the strong complexity of internal processes can lead to unstable and/or chaotic internal patterns.

The global uncertainty of the agent/environment coupling are the reasons why a significant part of structural couplings do not issue as they were anticipated! This is often referred as "cognitive dissonance", i.e. a lack of congruency between the agent actions and the environment reactions. The process of election, i.e. the process by which elective partners are chosen, is driven by an internal nervous process which is known to be highly unpredictable, presumably chaotic [17]. The precise moment of action decision is thus growing on a moving ground in an uncertain surrounding.

5.2 Reward dynamics

Reward and learning dynamics are precisely at the crosswalk between the body growth dynamics and the nervous dynamics. We give here some tracks toward a modeling of the learning process according to the primitive process of growth

and nutriment assimilation. From a global point of view, the body growth is favored when the movements that accompany the nervous activity orientate the body toward sources of nutriments, and avoid major dangers. The reciprocal is not obvious. In which fashion does the slow process of assimilation orientate the nervous activity toward the facilitation of body persistence? Due to the difference of speed between growth and nervous dynamics, the direct dependency between nutriments and growth is not operant. The nutriments are not assimilated at the same place and at the same speed than where the nervous activity is.

The point I want to suggest here is that the plant tendency to grow better where the nutriments are is mimicked and extended by animals *in the behavioral domain*. Some behaviors tend to be consolidated for they give a better access to power sources. Some parts of the internal construction (neural circuits mainly) are consolidated and grow, for they participate to a body-environment coupling which gives access to a rewarded moment. Other circuits degenerate for they don't bring such access. The important point is that a new domain of growth is defined, which is not related to the body mass, but on the body's abilities and skills. These new abilities correspond to *an extension of the agent appropriation capabilities*, i.e. knowledge extension. The knowledge of the environment increases with the number/variety of eligible objects and partners. The more the agent can identify various partners, the more it can take a part in the construction of future events, and the more it can avoid to face cognitive dissonances.

The learning process must rely on a local storage of neurotransmitters, which are released and then recruited by the current nervous pattern. Those neurotransmitter (dopamine for instance) may be seen as the substitutes of real nutriments, following the path of the current nervous activity, and possibly stimulating local weight reinforcement mechanisms. That moment of neurotransmitters release is often described as a "reward". It is a signature that something positive has been identified by the body. In neural networks modeling terms, if we decompose internal input u_{in} in various components, i.e. $u_{\text{in}} = (u_{\text{signal}}, u_{\text{weights}}, u_{\text{other}})$, the weights vector $u_{\text{weights}} = \{W_{ij}\}_{i,j\in\{1..N\}^2}$ directly relates to the interconnection pattern of the neural network[3]. The weight evolution rule, even directly depending on the internal activity, may thus be attached to the slow external process.

The question is now to define the nature of the operational core from which new partners may emerge through the global and local body-environment processes. A preliminary implementation of that principle using the properties of an internal chaotic dynamics as a generative process for action production and environment appropriation can be found in [18].

[3] where for instance the neural activation dynamics may be updated according to

$$x_i(t+1) = f(\sum_{i=1}^{N} W_{ij}(t)x_j(t) + u_{\text{signal},i}(t))$$

References

1. Varela, F.: Principles of Biological Autonomy. North Holland, Amsterdam (1979)
2. Beer, R.D.: Dynamical approaches to cognitive science. Trends in Cognitive Science **4** (2002) 91–99
3. Lindenmayer, A.: Mathematical models for cellular interaction in development i+ii. Journal of Theoretical Biology **18** (1964) 280–315
4. Turing, A.: The chemical basis of morphogenesis. Philos. Trans. R. Soc. London (1952) 37–72
5. Singer, W.: Time as coding space in neocortical processing : a hypothesis. In Buzsáki, G., ed.: Temporal Coding in the Brain, Berlin Heidelberg, Springer-Verlag (1994) 51–79
6. Rodriguez, E., George, N., Lachaux, J.P., Martinerie, J., Renault, B., Varela, F.: Perception's shadow: long-distance synchronization of human brain activity. Nature **397(6718)** (1999) 430–433
7. Amari, S.: A method of statistical neurodynamics. Kybernetik **14** (1974) 201–215
8. Gerstner, W., Ritz, R., Van Hemmen, L.: A biologically motivated and analytically soluble model of collective oscillations in the cortex. i - theory of weak locking. Biol. Cybern. **68** (1993) 363–374
9. Brunel, N., Hakim, V.: Fast global oscillations in networks of integrate-and-fire neurons with low firing rates. Neural Computation **11** (1999) 1621–1676
10. Hansel, D., Sompolinsky, H.: Chaos and synchrony in a model of a hypercolumn in visual cortex. J. Comp. Neurosc. **3** (1996) 7–34
11. Daucé, E., Moynot, O., Pinaud, O., Samuelides, M.: Mean-field theory and synchronization in random recurrent neural networks. Neural Processing Letters **14** (2001) 115–126
12. Amari, S.: Dynamics of pattern formation in lateral-inhibition type neural fields. Biological Cybernetics **27** (1977) 77–87
13. Daucé, E.: Short term memory in recurrent networks of spiking neurons. Natural Computing **2** (2004) 135–157
14. Gibson, J.J.: The ecological approach to visual perception. Houghton-Mifflin, Boston (1979)
15. Penn, A.: Steps towards a quantitative analysis of individuality and its maintenance: A case study with multi-agent systems. In Polani, D., Kim, J., Martinez, T., eds.: Fifth German Workshop on Artificial Life: Abstracting and Synthesizing the Principles of Living Systems, IOS Press (2002) 125–134
16. Tani, J.: Symbols and dynamics in embodied cognition: Revisiting a robot experiment. In Butz, M., Sigaud, O., Grard, P., eds.: Anticipatory Behavior in Adaptive Learning Systems, Springer (2003) 167–178
17. Skarda, C., Freeman, W.: How brains make chaos in order to make sense of the world. Behav. Brain Sci. **10** (1987) 161–195
18. Daucé, E.: Hebbian reinforcement learning in a modular dynamic network. In: SAB 04. (2004)

Robustness in the long run: Auto-teaching *vs* Anticipation in Evolutionary Robotics

Nicolas Godzik, Marc Schoenauer, Michèle Sebag

TAO team, INRIA Futurs and LRI, UMR CNRS 8623 bat. 490, Université Paris-Sud
91405 Orsay Cedex, France
{Nicolas.Godzik,Marc.Schoenauer,Michele.Sebag}@lri.fr

Abstract. In Evolutionary Robotics, auto-teaching networks, neural networks that modify their own weights during the life-time of the robot, have been shown to be powerful architectures to develop adaptive controllers. Evolved auto-teaching networks have demonstrated better performance than non-learning controllers when facing different environments during a given life-time. Unfortunately, when run for a longer period of time than that used during evolution, the long-term behavior of such networks can become unpredictable. This paper gives an example of such dangerous behavior, and proposes an alternative solution based on anticipation: as in auto-teaching networks, a secondary network is evolved, but its outputs try to predict the next state of the sensors of the robot. The weights of the action network are adjusted using some back-propagation procedure based on the errors made by the anticipatory network. First results show a tremendous increase in robustness of the long-term behavior of the controller.

1 Introduction

One key challenge of Evolutionary Robotics [6] is the controller robustness, defined as the ability of the controller to deal efficiently with changing environments and previously unseen situations − in other words, to adapt itself to some real world.

One prominent approach aimed at robust controllers is based on the so-called auto-teaching networks[9]. In this approach, the controller is made of two parts, simultaneously optimized by evolutionary algorithms. The first part, referred to as Agent Model, is fixed offline. The second part, the Agent, actually controls the robot; in the same time, the Agent is modified on-line to get closer to the agent model (section 2). This way, evolution constructs a dynamic decision system, the trajectory of which is defined from an attraction center (the model) and a starting point (the agent at time 0).

At this point, two time scales must be distinguished. During the training period, the agent is adjusted to the model, the fitness associated to the pair (agent, model) is computed and will serve to find optimal couples of (agent, model).

During the robot life-time, referred to as generalization period, the agent is still adjusted to the model in each time step.

However, for feasibility reasons, the training period only represents a fraction of the robot lifetime.
Therefore, the long term dynamics of the controller is not examined during the training period. This would make it possible for (opportunistic) evolutionary computation to select controllers with *any* dynamics in the long run, compatible with a good behavior in the short run...

This paper first focuses on the long-term behavior of auto-teaching networks, making every effort to reproduce as closely as possible the experimental setting described in [9]. Though results could not be exactly reproduced, interesting phenomena appear. Intensive experiments show that, not infrequently, auto-teaching networks with good fitness (good behavior during the training period) are found to diverge (repeatedly hitting the walls) as time goes by.

A tentative interpretation for this fact was based on the cumulative effects of agent adaptation with respect to a fixed model, gradually attracting the agent toward bad performance regions.

Along this interpretation, it came naturally to consider another source of adaptation, more stable in the long term than a fixed model. Such a source of adaptation is proposed, inspired from the cognitive sensori-motor framework [10]. Specifically, we propose a controller architecture where adaptation is centered on an anticipation module; the anticipation module predicts the next state of the environment (the sensor values) depending on its current state *and the agent action*. As the true state of the environment becomes available in the next time step, the anticipation module actually provides a key signal: either everything is as predicted (the anticipated values match the actual ones); or something "went wrong". Adaptation is achieved as the agent uses this signal as additional input.

Finally, the source of adaptation is provided by a World Model (the anticipation module). The important point is that the world model can be evaluated with respect to the world itself, available for free (in the next time step) during both the training and the generalization period. In opposition, the agent model in the auto-teaching architecture could not be confronted to the "true actions" during the generalization period.

Implementations of this architecture, termed AAA, for *Action, Anticipation, Adaptation*, demonstrate an outstanding robustness in the long run, comparatively to the reference results.

The paper is organized the following way. For the sake of completeness, section 2 briefly describes the auto-teaching architecture, the goal and experimental setting [9], and presents and discusses the results obtained along the same settings when observing the behavior of auto-teaching networks in the long run. In section 3, the AAA architecture is presented, focusing on the anticipation module and its interaction with the decision module. Section 4 reports on the experimental validation comparatively to the reference results. The paper ends with a discussion of these first results, and points out the numerous avenues for research opened by this study.

2 Long Term Behavior of Auto-teaching networks

2.1 Settings

The architecture of auto-teaching networks used in [9] involves two modules, implemented as feed-forward neural nets without any hidden layer. During the lifetime of the robot, the first module is fixed, while the second module uses its error (difference with the first module output) to adapt its weights using back-propagation.

The scenario of the experiment is also taken from [9]. During evolution, the training period is made of 10 epochs. At the beginning of each epoch, both the target and the robot are set to random positions; the robot then explores the arena (60×20 cm, where the robot is a 5cm diameter Khepera), until i) it hits a wall, with no gain of fitness ; ii) at some time step $t < 500$, it finds the target area; in this case, its fitness is increased by $500 - t$; iii) $t = 500$; in this case the fitness is not increased. It must be emphasized that the target is not "visible": the robot has to roll over the target zone (2cm radius) to find it.
The initial goal of the experiments was to examine how the auto-teaching networks adapts to rapidly changing environments. To this aim, the color of the walls was changed every generation, alternating dark and white walls.

As in [9], we used an ES-like Evolutionary Algorithm in which 20 parents generate 100 offspring. The weights were real-coded genotypes, and their values were limited to $[-10, 10]$. The mutation rate was 10%. However, in order to approach the performances, some slight modifications were necessary: using (20+100)-ES (the parents in the next generation are deterministically selected from all 120 parents plus offspring) rather than a (20,100)-ES (the parents are selected from the 100 offspring). We used Gaussian mutation (with fixed standard deviation .5) instead of uniform mutation in $[v - 1, v + 1]$, v denoting the current value. We used some crossover (rate 40%) while none was used in [9]. Finally, all individuals were reevaluated every generation to avoid very lucky individuals to take over by chance (see below).

2.2 Training period

A first remark is that we failed to fully reproduce the results of Nolfi and Parisi [9]. The main reason for that is the very high variability of the fitness, too much dependent on the respective starting positions of the robot and the target. This variability was clearly visible when post-processing the best individuals from the last generation: out of 10 epochs, it never came even close to the same fitness than it had been given during its last evaluation. The only way to get over that variability and to come close to that was to run 10 times 10 epochs and to take the best results out of those 10 evaluations.

Nevertheless, the on-line results (best and average of 11 runs) resemble those of [9], as can be seen on Figure 4-left, and show a rapid increase toward a somehow stationary value of 2500 (though the variance is higher, for both the best run and the average of 11 independent runs).

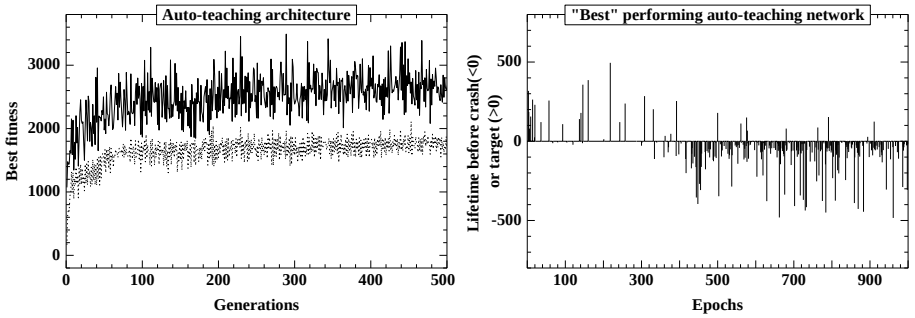

Fig. 1. Experiments with auto-teaching architecture: Left - On-line results (peak and average from 11 independent runs). Right - Life-times for the "Less Disastrous" results on long-term adaptation: a negative life-time indicates that the epoch ended with a crash.

2.3 Generalization period

However, when we started to investigate the behavior of the controllers over a very large number of epochs, we discovered that after 10 epochs of a "normal" lifetime (same length as the training period) their behavior began to change, and became increasingly chaotic. In particular, most robots began to hit the walls soon after epoch 10.

The initial goal of the experiment was to discover how the auto-teaching networks adapts to rapidly changing environments - and the scenario was to change the color of the wall every 10 epochs, alternating dark and white walls.

Figure 1-left and Figure 2 show typical results in our setting for that experiment. For each epoch (x coordinate), a bar shows whether the robot found the target (positive bar, the smaller the better) or hit the wall (negative bar). To make the figure more readable, no bar is displayed when the robot neither found the target nor hit the wall.

The "most decent" individual (Fig. 1-left) only starts hitting the walls after 300 epochs − though it does not find the target very often after the initial 10 epochs. Figure 2 is a disastrous individual, that starts crashing exactly after 10 epochs.

More precisely, the best individuals (gathered after a re-evaluation on 10 times 10 epochs of all individuals in the final population) hit the walls on average for 400 epochs out of 1000, the one displayed on Figure 1-left being the best one with only 142 crashes.

The interpretation offered for these results is that once again, evolution found a mouse hole to reach the goal: because what happens after the training period does not influence selection, anything can indeed happen then. The underlying dynamical system modifies the weights according to some evolved model that only has to be accurate during 10 epochs. On the other hand, increasing the training period by an order of magnitude appears to be unrealistic.

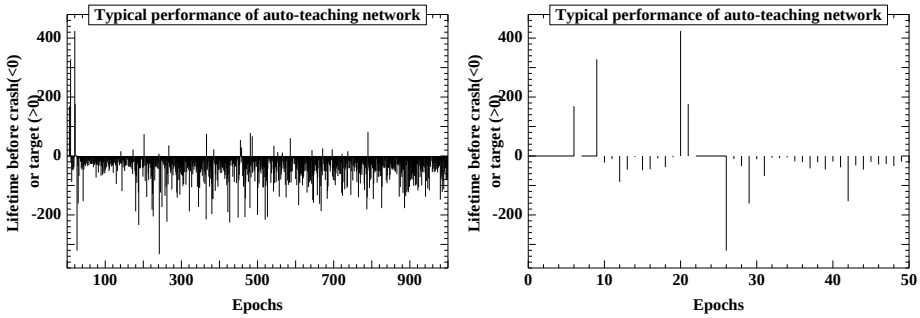

Fig. 2. More typical behaviors on long-term runs: Left - for 1000 epochs. Right - Zoom on the first 50 epochs.

The anticipatory architecture presented in the next section is an attempt to address the above limitations.

3 Action, Anticipation, Adaptation

This section presents the *AAA* architecture for neural controllers, achieving three tasks: action (controlling the robot effectuators); anticipation (based on the robot sensors, and the action output); adaptation (based on the difference between the sensor values anticipated in the previous time step, and the current sensor values).

As mentioned in the introduction, the basic underlying idea of the *AAA* architecture is that the adaptation mechanism must apply only when needed, and must be based on "true errors" rather than errors coming from an arbitrary model purposely evolved. In other words, rather than build a declarative model of the world, the idea is to give the robot a procedural model that will allow him to predict the consequence sof his own actions. And the simplest description of these consequences is through the values of its sensors. Such views are inspired from the cognitive sensori-motor framework [10].

In the framework or neural controllers, and in the line of auto-teaching networks [9], a second neural network, the Model network, is added to the Agent controller, and its goal is to predict the values of the robot sensors at next time step. The inputs of this Model network are both the hidden layer of the actuator controller and the actual commands given to the actuators. Its outputs are, as announced, the values of the sensors (see the dark gray part of Figure 3).

Those predictions are then compared with the actual values sent by the sensors at next time step, and the results of these comparisons are used for a back-propagation algorithm that adjusts the weights of both the Model and the Agent networks, as described on Figure 3.

Yet another possible use of those errors is to add them as direct inputs to the actuator network, possibly increasing its adaptive possibilities. This is the light gray part of Figure 3. Note that the intermediate architectures in which

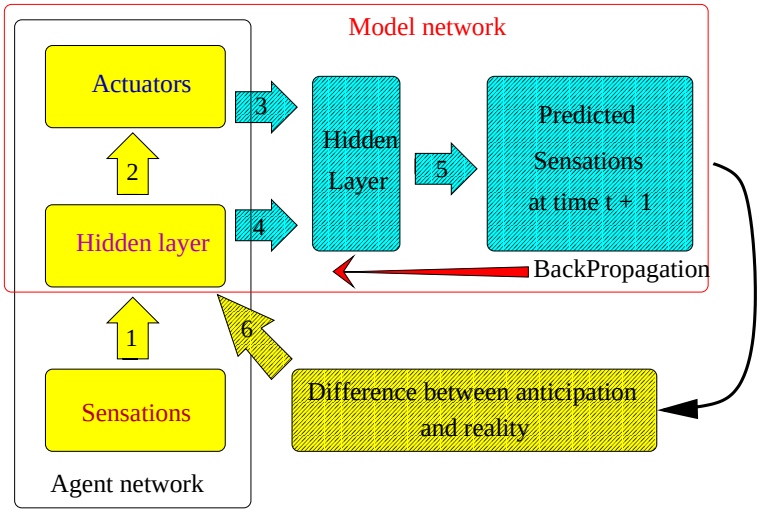

Fig. 3. The complete Anticipatory Neural Architecture. Rounded boxes are neurons, large arrows are connection weights. The classical Agent network goes gives actuator commands from the sensations. The Model network predicts values for the sensors from the intermediate layer of the actuator network and the actual actuator commands. Back-propagation is applied using the prediction error to all weights backward: 5, then 3 and 4, then 2, then 1. The prediction errors are also added as inputs to the Agent network.

either the back-propagation is suppressed, or the prediction errors are not taken as inputs are also being considered in the forthcoming first author's PhD.

4 Long Term Robustness of *AAA* Architecture

4.1 The neural network

In the experimental framework considered in this paper, the Khepera has 4 inputs (pairs of infra-red sensors), 2 outputs for the two motors. Hence the anticipatory part of the network must also have 4 outputs, while the complete neural network should have 8 inputs: the 4 sensors, plus the 4 errors computed from the outputs of previous time step. Both hidden layers of the actuator network and of the anticipatory network are composed of 5 neurons. Considering that all neurons also have a bias as input, the resulting network hence has 9×5 (arrows 1 and 6 + bias on Figure 3) + 6×2 (arrow 2 + bias) weights on the actuator network, plus 8×5 (arrows 3 and 4 + bias) + 6×4 (arrow 5) weights for the anticipatory network – 121 weights altogether. All those weights are submitted to back-propagation when some error occurs on the sensor predictions.

All evolution experiments described in next section for the anticipatory architecture were run using the Evolutionary Algorithm described in section 2.1.

4.2 Long-term robustness

The same experiment than that of [9] was run with the *AAA* Architecture. The learning curves along evolution are given on Figure 4-left, averaged on 11 independent runs: they are not very different from the same plots for the auto-teaching network (Figure 4-left). But the results about long-term adaptation are by no way comparable. Whereas the auto-teaching networks show unpredictable behaviors after the initial 10 epochs the anticipatory controllers stay rather stable when put in the same never-ending adaptation environment (e.g. the color of the walls change every 10 epochs, while adaptation though back-propagation is still going on). A typical summary of the behavior of the best individual of an evolution of the anticipative architecture can be seen on Figure 4-right: apart from a few crashes due to starting positions very close to the wall, almost no crash occurs in that scenario.

More precisely, out of 11 independent runs, 8 never crash in 1000 epochs (plots not shown!) while the 3 others had a behavior similar to that displayed on Figure 4-right: they never crash when the walls are white, and start hitting the dark walls after 700-800 epochs of continuous learning.

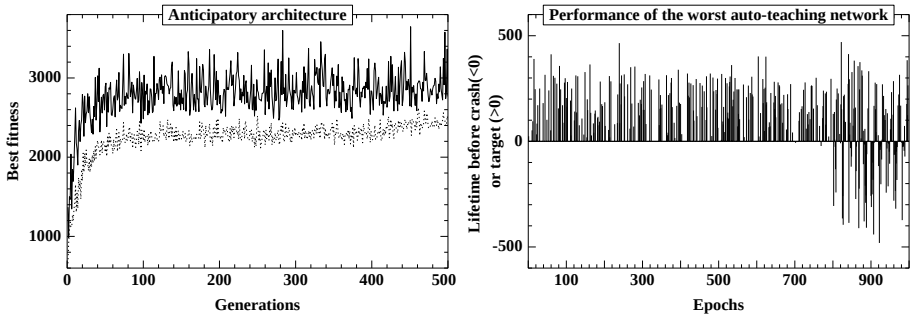

Fig. 4. Experiments with the anticipating architecture: Left - On-line results (average best and average from 11 independent runs). Right - **Worst** result on long-term adaptation (the wall color changes every 10 epochs).

All those results clearly demonstrate that the anticipatory architecture does not suffer from the same defects that the auto-teaching networks, and exhibit very stable behaviors even after thousands of epochs (the 8 crash-free best individuals were run up to 50000 epochs with no crash at all).

4.3 Adaptation in *AAA* networks

An important issue, however, is that of the adaptivity of the anticipatory architecture. Indeed, more sophisticated architectures than the simple auto-teaching network described in section 2.1 (like for instance a 3 layers network with one fully recurrent hidden layer) can be evolved to be robust in both the white and

black environment – the robots will simply stay further away from the walls in the white environment. But such architectures do not have any adaptive mechanism, and the experiments presented now will demonstrate that the anticipatory architecture does behave adaptively.

The initial a posteriori experiments described in previous section (let the robot live for 1000 epochs, alternating dark and white walls every 10 epochs) did not give any evidence of adaptivity: all anticipatory controllers behave similarly, crashing very rarely against the walls, and behaving almost the same in both dark and white environments: due to the large number of weights, their values change rather slowly.

It was hence decided to let the weights adjust during 100 epochs in the same environment, and some interesting phenomena started to appear. First, after 100 epochs, some individuals began to have trajectories like the ones plotted on Figure 6: whereas the initial weights allow a cautious behavior in case of dark walls (the robot stays farther from the walls, see the thick line on the top plot), this is no longer the case after 100 epochs of weight modification, as witnessed by the bottom plot of Figure 6, where the red cross indicates that the robot hit the wall (dark walls) while it still avoids the walls when they are white. Indeed, after 100 epochs of adaptation to the white walls, the immediate epochs in the dark environment always resulted in a crash. Note that the reverse is not true, and individuals that have spent their first 100 epochs in the white environment never hit any wall, black or white afterward.

But more importantly, after some time in the dark environment, the behavior of the robot comes back to collision-free trajectories. Figure 5 shows two situations in which this happens. When, after the initial 100 epochs in the white environment, the walls remain black forever (left), the number of crashes gradually decreases, and no crash takes place during the last 100 epochs. More surprisingly, when the wall change color every 100 epochs, the rate of crashes also decreases (Figure 5-right), and in fact, it decreases even more rapidly than in the previous scenario – something we still cannot explain.

Note that control experiments with the auto-teaching networks gave exactly the same results than those of section 2.1: an enormous amount of crashes, whatever the scenario.

5 Discussion and further work

The idea of using anticipation to better adapt to changing environments is not new, and has been proposed in many different areas. Anticipatory Classifiers Systems [11] are based on anticipation, but in a discrete framework that hardly scales up.

Trying to predict some other entity's action also amounts to anticipation, but does not really try to anticipate on the consequences of one's own actions (e.g. a program playing "psychological" games [1], or multi-agent systems [13]).

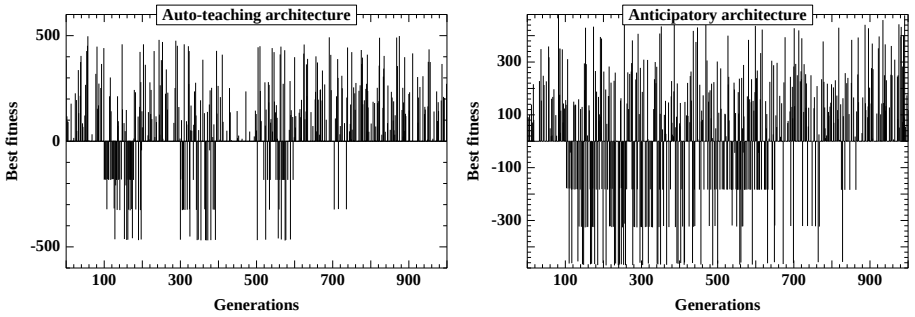

Fig. 5. Adaptation by the Anticipating architecture: 100 epochs are run with white walls; the robot is then put in the dark environment for 900 epochs (lest) or is put alternatively in dark and white environments by periods of 100 epochs (right).

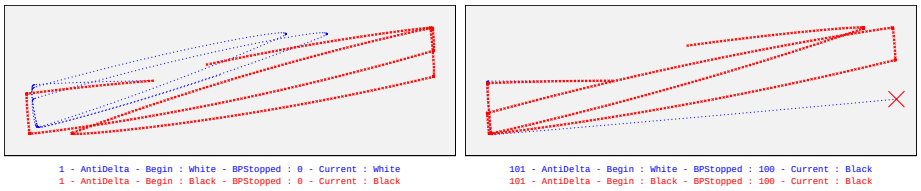

Fig. 6. Trajectories of the best individual of an evolution of an anticipative architecture, where the thick lines correspond to dark walls and the thin line to white walls. The stating points are the same for all trajectories. Top: Initial behavior during the first epoch, before any adaptation could take place. Bottom: behavior after 100 epochs of adaptation in the white environment.

Trying to directly predict its own sensor values has also been tried to help building Cognitive Maps in Robotics [2]: the prediction error is then used as a measure of interest for an event.

But the architecture the most similar to *AAA* has been proposed in the Evolutionary Robotic domain by Nolfi, Elman and Parisi [5] in a simpler framework, for artificial organisms foraging food. First of all, no long term experiment was described in that work. Moreover, in that work, their architecture did not use the prediction errors as supplementary inputs - but on the other hand it did use the last output commands ... And a final remark about that work os that the sensors and the actuators were closely related: the sensory inputs of the network were the direction and distance of the nearest food, while the commands for the actuators were given in terms of the angle and distance to advance, whereas in *AAA* architecture, the only relationship between the actuator commands and the predicted outputs is through the Agent network itself.

This raises another interesting issue is that of the outputs of the Model network. It has been argued by Nolfi and Parisi [8] that the best teaching inputs are not the correct answers for the network (i.e. the exact predictions of the next sensor values). But this might be because of that link mentioned above between

the predicted outputs and the actuator commands. Indeed, some preliminary investigations inside the *AAA* neural networks during the lifetime of the robot seem to show that its predictions are here rather accurate most of the time: for instance, when the robot is far from any obstacle, the predicted values are indeed very close to 0 (and hence not modification of the weight does take place). But here again deeper investigations are required.

Looking at the behavior of adaptive systems from a long-term perspective asks new questions beyond the traditional debate between Nolfi's model of interaction between learning and evolution [4] and Harvey's claim that the success of learning + evolution only comes from the relearning of weights that have been perturbed by some mutation [3]. Indeed, the successful re-adaptation observed after a long period in the white environment (Section 4.3 seems to suggest that the learning is not limited to correcting some weight modifications. However, more work is needed to understand how such re-adaptation has been made possible by evolution.

Finally, another important issue is that of the scaling up of the *AAA* architecture with the number of sensors (e.g. if the robot is equipped with some vision system). A possible answer might come from the *information bottleneck* theory [12]: this model tries to compress the sensor information as much as possible, while still maintaining feasible a reconstruction of the world that is sufficient for the task at hand. In that perspective, the hidden layer of the Agent network (Figure 3) could then be viewed as the set of *perceptions* of the robot, and the Model network could then try to predict this minimal compressed information rather than the numerous sensor values.

6 Conclusion

After having pointed out a major weakness of auto-teaching networks, the unpredictability of their long-term behavior, we have proposed the *AAA* architecture to remedy this problem: the evolved oracle is replaced by a Model network that will learn to predict the values of the sensors of the robot. The modification of the weights of the Agent network is then based on the error made for those predictions.

The first results in terms of long-term robustness are outstanding compared to those of the auto-teaching networks. Moreover, at least some of those networks do exhibit a very interesting adaptive behavior: after having evolved during 100 epochs in a white environment, they can gradually re-adapt to dark walls.

However, a lot of work remains to be done to assess the efficiency and usefulness of the *AAA* architecture, starting with a better understanding of how and why such anticipatory networks can re-adapt their weights on-line without any incentive or reward for collision avoidance. We nevertheless hope that anticipatory networks can somehow help bridging the gap between fully reactive controllers and sensori-motor systems.

References

1. Meyer C., Akoulchina I., and Ganascia J.-G. Learning Strategies in Games by Anticipation. In *Proc. IJCAI'97*. Morgan Kaufmann, 1997.
2. Y. Endo and R.C. Arkin. Anticipatory Robot Navigation by Simultaneously Localizing and Building a Cognitive Map. In *ICOS'03 – Intl. Conf. on Intelligent Robots and Systems*. IEEE/RSJ, 2003.
3. I. Harvey. Is there another new factor in evolution? *Evolutionary Computation*, 4(3):311–327, 1997. Special Issue on Evolution, Learning, and Instinct: 100 Years of the Baldwin Effect.
4. S. Nolfi. How learning and evolution interact: The case of a learning task which differs from the evolutionary task. *Adaptive Behavior*, 7(2):231–236, 2000.
5. S. Nolfi, J.L. Elman, and D. Parisi. Learning and evolution in neural networks. *Adaptive Behavior*, 3(1):5–28, 1994.
6. S. Nolfi and D. Floreano. How co-evolution can enhance the adaptive power of artificial evolution: implications for evolutionary robotics. In P. Husbands and J.A. Meyer, editors, *Proceedings of EvoRobot98*, pages 22–38. Springer Verlag, 1998.
7. S. Nolfi and D. Floreano. *Evolutionary Robotics*. MIT Press, 2000.
8. S. Nolfi and D. Parisi. Desired answers do not correspond to good teaching input in ecological neural networks. *Neural Processing Letters*, 2(1):1–4, 1994.
9. S. Nolfi and D. Parisi. Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior*, 5(1):75–98, 1997.
10. J. Kevin O'Regan and Alva Noë. A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences*, 24(5), 2001.
11. W. Stolzmann. An introduction to anticipatory classifier systems. In P.L. Lanzi, W. Stolzmann, and S.W. Wilson, editors, *LCS'99*, pages 175–194. LNAI 1813, Springer Verlag, 2000.
12. N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.
13. M. Veloso, P. Stone, and M. Bowling. Anticipation as a key for collaboration in a team of agents: A case study in robotic soccer. In *SPIE Sensor Fusion and Decentralized Control in Robotic Systems II*, volume 3839, 1999.

Dream Function as an Anticipatory Learning Mechanism

J.C. Holley,[1] A.G. Pipe[2] and B. Carse[2]

[1] Clares MHE Ltd., Wells, Somerset, England
julian@holley.uklinux.net
[2] University of the West of England, Bristol, England
{anthony.pipe,brian.carse}@uwe.ac.uk

Abstract. How are the internal structures for anticipatory based behaviour developed, refined and retained in humans and animals? This paper proposes arguments that such development could occur during sleep, specifically dream sleep which mostly occurs during rapid eye movement sleep (REM). Justification for this view is supported by increasing evidence from relevant sleep and dream research in aspects of neurology, psychology, philosophy and evolutionary development. In support of these arguments, results from initial experiments with a machine learning architecture, the Anticipatory Classifier System (ACS) are presented.

1 Introduction

There is increasing evidence [8][18][24] that sleep and in particular rapid eye movement sleep[3] (REM) [1] where dreaming mostly occurs has important roles in memory and learning amongst both animals and humans. A recent neurophysiological study has demonstrated that rats do *dream* or at least replay or rehearse maze experiments while asleep [25]. Psychological studies with animals and humans have shown that REM deprivation (REMD) can impair the learning of complex tasks [17]. Studies, particularly with rats, have demonstrated a distinct correlation between task complexity and onset and longevity of REM sleep [16]. A recent human sleep study has also demonstrated the influence of sleep on insight and reaction time after learning to solve a sequential numerical problem [24]. This report attempts to relate philosophical concepts of dream sleep with physiological and psychological evidence to improve learning in the machine learning arena [21].

Specifically a tentative start is made by exploring a modified latent learning architecture, the Anticipatory Classifier System (ACS) [3][19] to simulate simple dreaming.

[3] Also known as Paradoxical Sleep, reflecting the paradox of waking brain activity during sleep. For reviews see [10][15].

2 Background

This current research direction has been derived from previous studies controlling unstable dynamic mechanical systems in a failure avoidance framework. Adaptive learning systems such as the Adaptive Heuristic Critic (AHC) [22] modulating Artificial Neural Networks (ANNs) were used to control complex systems by avoiding failure [12]. The paradox of learning from failure, experience of the undesirable, led to thoughts into how best off-line or cognitive strategies could be developed in such cases. The context of the original problem further shaped development by the batch orientated structure of expensive and dangerous on-line experience and cheap and safe off-line adaption. The natural analogy between the normal wake-sleep cycle and this work set the stage for the investigations into sleep and especially dream sleep adaptation.

3 Dreaming as Anticipation Preparation

It is well established that humans and higher animals interact with the world through a cognitive abstraction layer. Visual, auditory, olfactory etc. inputs are combined with memory to form an internal representation and meaning. It is this internal representation that the agent sees and makes decisions upon. Considering the visual sense, it is difficult for us to accept that the surrounding world does not actually look as we see it, but what we actually *see* is an internal representation of that a world updated by external changes.

Combination and internalisation of the environmental stimulus is useful for real time interaction and also essential for more abstract functions such as anticipation, planning and imagination. The ability to predict or estimate the future is clearly a desirable survival attribute and sophisticated planning can be observed in many animals [7]. During sleep the brain is closed off from its usual source of environmental stimulus and during REM sleep (where dreaming mostly occurs) the normal outputs are also inhibited. In such a configuration it is possible that the animal or agent could interact with the internal representation of the world, (conceptualisation of reality) and adapt as though interacting with reality. In the sleep and dream research arena there are many arguments and counter-arguments as to purpose of such behaviour.

The proposal here is that this delusional interaction has evolved in order to better prepare an agent for the unknown future, a cognitive maintenance program organised to update a predictive model of the future by using some but not all of the existing cognitive mechanisms. In short, one purpose of sleep and especially dreaming is to generate and maintain a predictive model of the future. By definition an agent's anticipatory mechanism is based on a predictive model which in a constantly changing world never reaches the state of a true model. Rare or undesirable and yet critical scenarios can be generated and safely rehearsed, expanded and extrapolated in preparation for the real future.

4 Improved Convergence Through Model Learning Using the ACS

The long term goal of this work is to improve an agent's cognitive abilities inspired by developing theories from dream research. As an interim step toward this goal the ACS has been employed to reduce learning interactions with the real world by abstractly or cognitively revisiting situations and confirming or weakening state relationships. As the ACS interacts with the environment a generalised model is developed, this knowledge can be exploited to perform some model learning Dyna style [23] and reduce environmental interaction. As with the Dyna architecture the goal is to minimise environmental interaction and utilise information held in the model in order to obtain an accurate model with the least real world interactions. However, the ACS does not hold an explicit world model, but a generalised representation and this represents a challenge when employing the model to generate an abstract representation of the real environment. Model learning is applied to the ACS by switching ACS experiences between the real world and the developing model. The model is a snapshot of the classifier list at the time between leaving interactions with the real world and entering the abstract model world. Consider the following example; an ACS system has been interacting with an arbitrary plant and has formed the classifiers shown in Table 1.

Table 1. Example Classifier Set

Rule	C	A	E	q
1	`####`	0	`####`	0.2
2	`####`	1	`####`	0.2
3	`01##`	0	`10##`	0.9
4	`10##`	1	`01##`	0.9

For classifier `3`, the list indicates that with a high confidence ($q = 0.9$) if the agent is currently in a state where the first (left most) attributes are '`01`' and a '`0`' action is performed, then the first two attributes of the following state will be '`10`'; the rest of the attributes remain unchanged.

To present the ACS with a model of the real world, an algorithm must extract developing environmental information from the classifier list and present a response in the same format back to the ACS (Fig. 1). The current input state and action are taken as inputs and the next state is presented to the ACS as though the action has taken place in reality. The difference is that the real world response is based in reality whereas the model is an estimate derived from current experience. The algorithm for the model world operation and response is similar to that of the normal ACS selection process. From the snap copy of the classifier list the model must select the most appropriate next state given the current

Table 2. Illustration of State Cycling

Step 1	
Input state	= 0100 (matches all classifiers except C4)
C3 Expectation state	= 10##
Expected next state	= 1000 (last two attributes passed through)
Step 2	
Input state	= 1000 (matches all classifiers except C3)
C4 expectation field	= 01##
Expected next state	= 0100 (last two attributes passed through)

state and action. Consider that the ACS is running with the previous classifier list. Assume the current state is '0100' and through the normal selection process classifier 3 is selected with the action '0'. The classifier expects the next state to become '1000'. At this point before executing the action on the real world, the switch 'S' changes state to allow the ACS to interact with the model world (Fig. 1). The model algorithm takes a copy of the current classifier list and current input state and waits for an action from the ACS. Without distinguishing between the two worlds, the ACS executes action '0' on the model world instead of the real world and the model responds. The model must now reply to the ACS as though the ACS is interacting with the real world. The model replies by performing a selection process (see Table 3) on matching classifiers and uses the winning classifier's expectation field to generate the next state response. After the first cycle the input state therefore becomes uncoupled from reality.

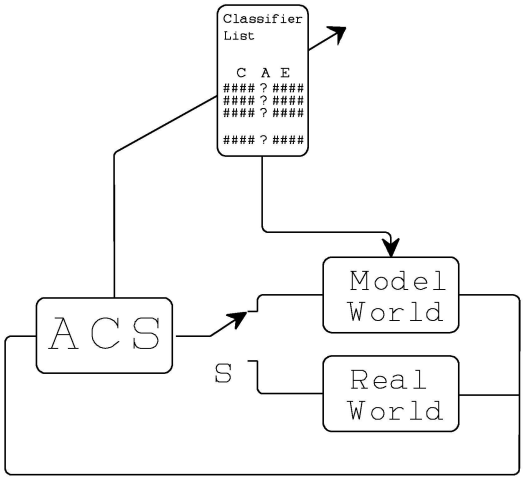

Fig. 1. The 'ACS' switching between the real world and the model of the real world

From the previous list only two classifiers match in both the condition and action part, classifier `1` and classifier `3`. If (as with the ACS) a high bias is given to classifiers that are more specific, the likelihood that classifier `3` will win the bid is high (good prediction history, $q = 0.9$ and contains less hash terms). If classifier `3` wins the bid the model algorithm simply responds to the ACS with the next expected state by presenting the current input with a pass through operation using the expectation part of classifier `3` to form '`1000`'. If classifier `1` wins (even with the small bid), then the current input state will be passed through as the next state (the model presents no change, an *incorrect* response)[4]. When the ACS receives the 'predicted' response from the model, this is taken as the actual response and learning occurs. The input state presented to the ACS is now '`1000`'. In response to this input the ACS is most likely to choose classifier `4` resulting in issuing action `1` to the model. As with the previous criteria the model now uses classifier `4` to respond, resulting in a return to the first '`0100`' state as illustrated in Table 2. This can therefore reduce the amount of interactions that are required for the ACS to build up an accurate model in a static environment. When a novel situation is presented to the model and no input-condition and action pair exists, the root classifiers will always win the bid and predict no change. To explore this strategy, the adapted ACS was applied to a simple maze learning problem, the T-maze adapted from Stolzmann [19].

5 Experimental Procedure

The object of the experiment is to record the quantity of total real interactions that are required to reach a confident world model when ACS operation is split between the real and model worlds. Sessions run until a complete sweep of each state with each action predicts the next state correctly and with a classifier that has a high prediction quality (¿90%). Each of the 10 sessions have a decreasing interaction with the real world and increasing interaction with the developing model. Basic ACS operation and parameterisation are as in Stolzmann's T-maze experiment [19]. Sessions are limited to 5000 cycles, results are averaged over 100 runs[5]. Four experiments are conducted with different selection policies for both the ACS and the model selection policy listed in Table 3.

A probabilistic bid is by the roulette wheel selection technique and deterministic selection is a winner takes all technique. Exploration in experiment 3 is only possible due to a linear random selection amongst competing classifiers when all bids are equal. Results are illustrated in Fig. 2 and Fig. 3. The X-axis represents each of the 10 sessions with varying percentages of real and model interaction. Session 1 starts with 100% real world interactions (normal ACS) and session 10 ends with just 10% real world interactions and 90% model.

[4] The ACS requires environmental causality
[5] Comprehensive experimental detail is publicly available [9]

Table 3. Experimental Exploration Strategies

Experiment	ACS Selection Policy	Dream Selection Policy
1	Probabilistic bid	Deterministic
2	Probabilistic bid	Probabilistic bid
3	Deterministic	Deterministic
4	Deterministic	Probabilistic bid

The Y-axis represents the following recordings :-

`Real` = total real world interaction.
`Model` = total model world interaction.
`Correct` = independent correct responses.
`Restarts` = useless responses from the model phase.

5.1 Experiment T-maze 1 and 2

The ACS classifier selection process of probabilistic bidding operates consistently throughout interaction with the real and model worlds. In the first experiment the model selection process is deterministic. In the second experiment the model selection process is probabilistic.

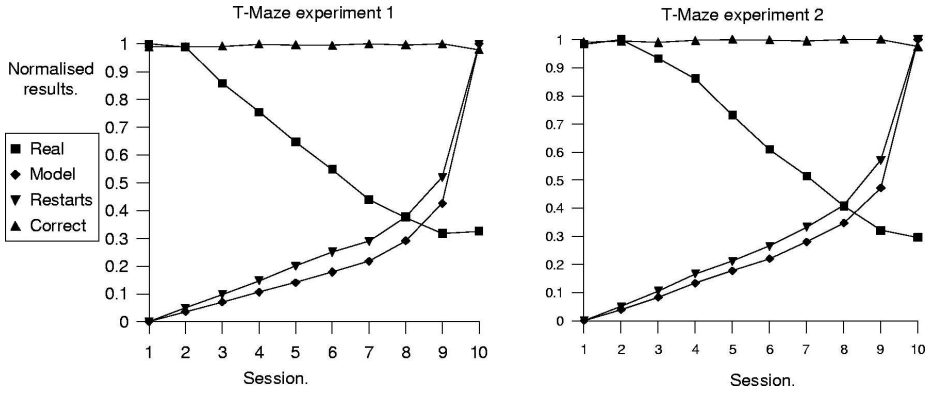

Fig. 2. T-Maze experiments 1 and 2

5.2 Experiment T-maze 3 and 4

Again, the ACS classifier selection process of deterministic bidding operates consistently throughout interaction with the real and model worlds. In the first experiment the model selection process is also deterministic. In the second experiment the model selection process is probabilistic.

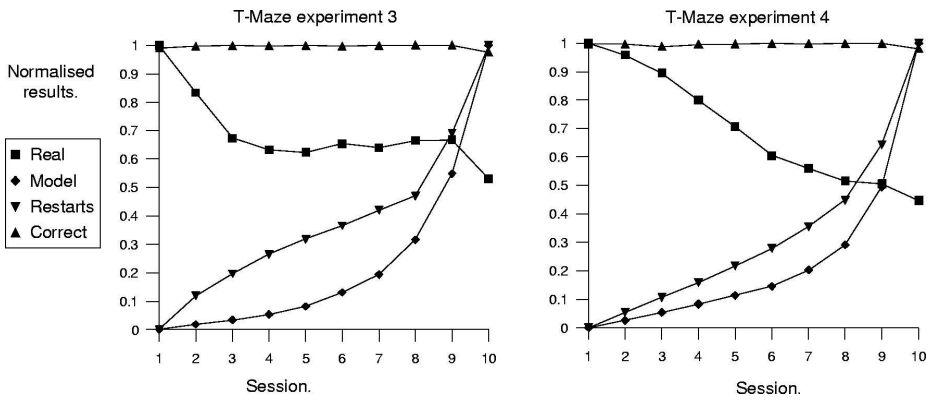

Fig. 3. T-Maze experiments 3 and 4

5.3 Interpretation of the Results

For each session the ACS interacts between the *real* world (T-Maze) and then with a *model* of that environment. The model responses are generated from a static copy of the current classifier list at the point of the switch. Switching between the two is distributed throughout the session. For example, in session 10 there are 10 initial interaction steps with the real world followed by 90 steps with the model. Throughout the session the accumulated real and model interactions are collated until the ACS has developed a confident model of the environment. In all cases the requirement to interact with the real world decreases in order to generate a confident world model. As interactions with the real world decrease the ability of the model to generate realistic responses decreases, significantly increasing model cycles, tending towards infinity as real interactions approach zero.

In Experiment 1 and 2 (Fig. 2) the ACS interacts normally, i.e. employing a selection process based on the historical prediction quality of the classifier proportioned by a specificity factor. In Experiment 1 the model selection process replies with a response based on the classifier with the highest bid without introducing an exploratory component implicit in the ACS roulette wheel selection. From the perspective of minimising real world interaction (for reasons outlined in Section 2) this strategy is the most logical and subsequently performs well. This has parallels with Sutton's prioritised sweeping in the Dyna framework [21]. The strategy only allows exploration within the real world session and only allows reiteration by selections of the best generated classifiers in the model session phase. Experiment 2 allows a probabilistic response during the model session, allowing occasional *exploration* by selecting other matching classifiers. Effectively the ACS real world interaction is mirrored in the model world interaction. This is reflected by the slight increase in real world interaction. Due to small problem size this does not adversely change from the response achieved in Experiment 1.

Interesting results are shown in Experiments 3 and 4 (Fig. 3). In Experiment 3 both the ACS and the model response choose the best classifiers upon which to use and with which to respond. The only way this configuration can explore is when the selection process is presented with identical bids. In that case a random selection is taken between them. This pairing produces sequential loops around first discovered sequences failing to explore further. The final configuration illustrated in Experiment 4 reduces this problem by allowing the model phase to *explore* by occasionally choosing random responses that break the cyclical behaviour observed in Experiment 3. (Experimental details are publicly available at [9]).

6 Discussion

The structure presented reduces real world interaction in a similar method to other Dyna style model learning systems [23]. Other classifier systems have incorporated latent learning [2][13][14] and have also combined Dyna style extensions [5][6][20]. However these systems have developed from a machine learning perspective (with perhaps the exception of Stolzmann et al. [20]). This work is contrasted in 3 areas, firstly in the model generation scheme. The model generates sequence responses based current experience represented in the classifier list. Realistic experience or practice sequences are an emergent property of the selection requiring no explicit guidance. Secondly the structure maintains the wake-sleep analogy. The agent learns without differentiating between the real world and the simulated world, the adaptive emphasis falls on the content of the generated model thread. Finally this work differs in the source of direction and aims. Artificial dreaming has previously been simulated to promote a particular functional stance [4]. It has also been associated with the behaviour of some machine learning systems including the ACS [20]. This work takes the view that dreaming *has* evolved some useful function (probably multifaceted)[11]. Those propositions are being explored with existing and new machine learning architectures to develop novel cognitive adaptations.

The function of generating useful threads and how the threads modify existing rules is the subject of continuing work. The selection process could easily be modified further by various methods, such as mutating attributes from classifiers that have a common condition part (in a match or action set) or filling in non-specified attributes from other similar classifiers in uncertain situations. During the dream thread execution several actions could be taken to generate new rules, assimilate rules or delete rules.

Rules generated during the model interaction present the opportunity for anticipatory behaviour based on real prototypical sequences in situations that have not been actually been experienced (dreams). Rules are generated in advance, their generation being guided by real world adaptation (waking). Explicitly the agent receives no new information during model sessions, but implicit preparation or anticipation could be developed. Research on rodent sleep behaviour recently reported that subjects where more likely to dream about maze run-

ning *before* maze runs when sessions where scheduled rather than just after the session. The rodents appeared to be rehearsing a anticipated event [25].

7 Conclusion

Inspired by developing sleep and dream research an extension has been applied that allows the ACS agent to replay abstract dreamlike state to state transitions as though real on a simple maze problem. The results have shown that this reduces the amount of real world learning interactions that would otherwise be required. Clearly much more work needs to be done to modify the ACS or similar model building architecture in order to create a system that proves beneficial in solving dynamic problems. In respect to the long term goals of this current research it is interesting to conclude with a quote from neurological researchers Kenway Louie and Matthew Wilson [25] on results from their research into rodent sleep behaviour. In their recent study they were able to detect temporal reactivation of spatial hippocampal place cells during REM sleep periods that were very similar to activations when awake, in effect they could view rodent dreams during REM sleep:-

> "... This reactivation of previous behavioural representations may be important for the learning of procedural tasks, which is dependent upon REM sleep. Mnemonic information that may have shared characteristics along a particular behavioural axis such as emotion could be juxtaposed and evaluated for common causal links, allowing adaptive behaviour change based on prior experience ...".

References

[1] Aserinsky, E. and Klietmen, N. (1953) Regularly Occurring Periods of Eye Mobility, and Concomitant Phenomena, During Sleep. Science 118: 273-274

[2] Bull, L. (2001) Lookahead and Latent Learning in ZCS. UWE Learning Classifier Systems Group Technical Report UWELCSG01-004, University of the West of England.

[3] Butz, M. V. (2001) Anticipatory Learning Classifier Systems, Genetic Algorithms and Evolutionary Computation, 4. Kluwer Academic Publishers. ISBN 0-792-37630-7

[4] Crick, F. and Mitchison, G. (1986) REM Sleep and Neural Nets. Journal of Mind and Behavior 7. 229-250

[5] Gerard, P. and Sigaud, O. (2001) YACS: Combining Dynamic Programming with Generalization in Classifier Systems. In Advances in Classifier Systems, Vol. 1996 of LNAI, Springer-Verlag. 52-69

[6] Gerard, P., Meyer J. A. and Sigaud, O. (2003) Combining Latent Learning with Dynamic Programming in the Modular Anticipatory Classifier System. European Journal of Operation Research (submitted 2003)

[7] Heinrich, B. (2000) Testing Insight in Ravens. In Heyes, C. and Hiber, L. (eds.): The Evolution of Cognition. The MIT Press, Cambridge, MA. ISBN 0-262-09286-1. (2000) 289-305

[8] Hobson, J. A., Pace-Schott, E.F., Stickgold, R. and Kahn, D. (1998) To Dream or Not to Dream? Relevant Data from New Neuroimaging and Electrophysiological Studies. Current Opinions in Neurobiology, 8. 239-244

[9] Holley, J. (2004) First Investigations of Dream-like Cognitive Processing using the Anticipatory Classifier System. UWE Learning Classifier Systems Group Technical Report UWELCSG04-002, University of the West of England, England.

[10] Jouvet, M. (1994) The Paradox of Sleep: The Story of Dreaming. Translated by Laurence Garey (1999). The MIT Press, Cambridge MA. ISBN 0-262-10080-0

[11] Kavanau, J. L., (2004) Sleep Researchers need to bring Darwin on Board: Elucidating Functions of Sleep via Adaptedness and Natural Selection (Editorial). Medical Hypotheses Vol. 62. 161-165

[12] Miller, T. W., Sutton, R. S. and Werbos, P. J. (1990) Neural Networks for Control, The MIT Press, Cambridge, MA. ISBN 0-262-13261-3

[13] Riolo, R. L. (1991) Lookahead Planning and Latent Learning in a Classifier System. Proceedings of the First International Conference on Simulation of Adaptive Behavior. Cambridge, MA: The MIT Press. 316-326

[14] Roberts, G. (1993) Dynamic Planning for Classifier Systems. Proceedings of the 5th International Conference on Genetic Algorithms (ICGA93). Morgan Kaufmann. 231-237

[15] Rock, A. (2004) The Mind at Night, Basic Books, Cambridge MA. ISBN 0-7382-0755-1

[16] Smith, C. and Lapp, L. (1986) Prolonged Increases in both PS and Number of REMs Following a Shuttle Avoidance Task, Physiological Behavior, Vol. 43. 1053-1057

[17] Smith, C. (1995) Sleep States and Memory Processes, Behavioral Brain Research, Vol. 69. 137-145

[18] Stickgold, R., Hobson, J. A., Fosse, M. and Fosse. M. (2001) Sleep, Learning and Dreams: Off-line Memory Processing, Science, Vol. 294. 1052-1057

[19] Stolzmann, W. (1998) Anticipatory Classifier Systems. Genetic Programming 1998: Proceedings of the Third Annual Conference, July 22-25, 1998, University of Wisconsin, Madison, Wisconsin, San Francisco, CA: Morgan Kaufmann. 658-664

[20] Stolzmann, W., Butz, M. V., Hoffmann, J. and Goldberg, D. E. (2000) First Cognitive Capabilities in the Anticipatory Classifier System, From Animals to Animats 6, Proceedings from the Sixth International Conference on Adaptive Behavior. 285-296

[21] Sutton, R. S. and Barto, A. G. (1998) Reinforcement Learning. The MIT Press, Cambridge, MA. ISBN 0-262-19398-1

[22] Sutton, R. S. (1984) Temporal Credit Assignment in Reinforcement in Reinforcement Learning. Ph.D. Dissertation. University of Massachusetts, Amherst.

[23] Sutton, R. S. (1991) Dyna, An Integrated Architecture for Learning, Planning and Reacting. SIGART Bulletin, 2: ACM Press. 160-163

[24] Wagner, U., Gals, S., Haider, H., Verleger, R. and Born, J. (2004) Sleep Inspires Insight, Nature, Vol. 427. 352-355

[25] Wilson, M. A. and Louie, K. (2001) Temporally Structured Replay of Awake Hippocampal Ensemble Activity During Rapid Eye Movement Sleep, Neuron, Vol. 29. 145-156

Anticipatory Learning for Focusing Search in Reinforcement Learning Agents

George Konidaris and Gillian Hayes

Institute of Perception, Action and Behaviour
School of Informatics, University of Edinburgh
James Clerk Maxwell Building, King's Buildings
Mayfield Road, Edinburgh EH9 3JZ
Scotland, UK

Abstract. We introduce the use of an anticipatory learning element that ranks novel state-action pairs in reinforcement learning agents, replacing the use of uniformly optimistic initial values for exploration. We argue that reinforcement learning functions as a search process for situated agents, and that anticipatory learning can be viewed in this context as the process of learning a heuristic that guides search behavior. The discussion is grounded with some gridworld experiments that demonstrate that a simple learning element can learn to rank actions sufficiently accurately to rapidly improve search behavior.

1 Introduction

Although the concept of anticipatory learning is intuitively appealing, its role and the potential behavioral benefits it brings to situated agents are not yet well understood [2]. In this paper, we consider one particular way in which an anticipatory learning element can provide useful behavioral benefits – that of learning to rank novel state-action pairs in reinforcement learning agents.

Reinforcement learning provides an intuitively appealing agent learning model because it is well suited to the problems that situated agents face [10], and because it provides a principled way to build agents motivated by a set of internal drives. Unfortunately, the use of such drives inevitably results in situations where reward is delayed because it results from a long sequence of unrewarded actions. In such cases, the agent often spends a significant amount of time initially searching for any solution at all, since the impoverished nature of the reward signal is not sufficient to quickly guide it to a solution.

In this paper, we show that a reinforcement learning agent with the additional capacity of being able to learn to distinguish good actions from bad, and thereby desirable novel states from undesirable ones, acquires a behavioural advantage that can be characterised as analogous to guided rather than blind search, with the additional advantage of being able to learn the heuristic online.

Previously [5], we have shown that the use of associative learning to model such a heuristic can improve the performance of a realistically simulated robot that uses reinforcement learning to learn to find a puck in a maze, and return

home. The approach presented here differs in that state-action pair rankings are learnt explicitly (rather than implicitly, using a state similarity metric), and in that we explicitly evaluate such behavior as anticipatory. In addition, the computational experiments presented here utilise a simple, abstract model of a situated agent, and thus allow for a more thorough evaluation of the model and its behavioral characteristics than would be possible with a real or realistically simulated robot.

2 Background and Preliminaries

In this section we briefly discuss what we mean by anticipatory learning, outline the particular model of situated reinforcement learning that we assume, and introduce the gridworld environment that we employ as an experimental domain throughout the remainder of this paper.

2.1 Anticipatory Learning

Anticipatory behavior is a broad term – virtually any form of learning can be considered anticipatory because it acquires knowledge about the world and how to act in it. In this paper, however, we are concerned specifically with learning mechanisms that, given an agent's past experiences, predict something useful about situations that it has never encountered before. Such mechanisms could be classed as *payoff anticipatory* or *state anticipatory* mechanisms, depending on what it is that they predict [3]. The mechanism introduced here falls into the payoff anticipation category, since it ranks novel state-action pairs with the aim of maximising future total payoff. Note that although reinforcement learning mechanisms by themselves are often considered anticipatory [3], such anticipation is usually either based on an agent's direct prior experience of each state, or an explicit model given in advance. For the purpose of this paper we do not consider either anticipatory, as the first deals with states the agent has experienced before, and the second deals with states that it has been given explicit prior knowledge of.

Following Butz, Sigaud and Gérard [3], we are interested in establishing how such mechanisms could be used to generate useful behavior, and under what circumstances they are feasible. The model we introduce here is one example of an anticipatory mechanism that produces useful behavior, in the form of guided search, and the remainder of this paper is concerned with demonstrating the advantages it provides and discussing the conditions under which it is able to provide them.

2.2 Situated Reinforcement Learning

Reinforcement learning refers to a family of methods that aim to solve the problem of learning to maximise a numerical reward signal over time in a given environment (Sutton and Barto [11] give a thorough introduction to the field).

In control tasks, reinforcement learning is usually used to learn the values of *state-action pairs*, so that the agent can select the action that should be taken at each state in order to maximise total reward over time. It is attractive because it provides a principled way to build agents whose actions are guided by a set of internal drives, has a sound theoretical basis, can allow for the principled integration of a priori knowledge, handles stochastic environments and rewards that take multiple steps to obtain, and is intuitively appealing.

In this paper, we assume that we are dealing with a situated agent using the reinforcement learning model described in Konidaris and Hayes [4]. This model differs from standard temporal difference reinforcement learning models in three ways. First, reinforcement learning is layered on top of a topological map, so that it takes place at the level of the problem space (in this case, at the level of navigation), rather than at the level of the robot's sensors. Second, learning is distributed, so that updates may occur in parallel if the agent has parallel hardware. Finally, learning is asynchronous, in that each of the states performs updates constantly, whatever state the agent happens to be in, and whatever else it is doing. The first difference means that learning occurs in a small state space, and that there is a separate learning element that is able to learn to distinguish between aliased states and form an internal map of the resulting state space. The last two differences mean that thousands of updates may occur while the agent completes an action, so that we can assume (as has been shown to occur on a real [4] and a simulated [5] robot) that the learning process *converges between decisions*, so that the agent makes the best possible decisions given its experiences.

Exploration in reinforcement learning is usually performed using ϵ-greedy methods, where the agent makes a random move some fraction (ϵ) of the time, which guarantee asymptotic coverage of the state space. In addition, the agents typically assign optimistic initial values to new states to encourage exploration, so that agent is more likely to explore unknown regions in the state space because they have higher values [11]. These values are gradually replaced as the agent learns the actual value of each state. However, we claim [4] that ϵ-greedy methods (along with all other methods that emphasise asymptotic coverage) are not appropriate for situated agents that must act as best they can given the knowledge, time, and resources they have available. Rather than occasionally making random decisions, an agent should learn how to satisfy its drives as soon as it can, and explore when it has time. We therefore separate exploration in situated reinforcement learning into two separate concerns: first, how to find any solution path at all, and second, how to thoroughly explore the state space in order to improve that solution or find new ones. In this paper we use anticipatory learning to improve the first, since mechanisms already exist to handle the second (e.g., using an exploration drive [4], or modelling curiosity [9]).

2.3 The Gridworld

We use a simple gridworld domain throughout this paper to illustrate our arguments and compare standard reinforcement learning to a model with an antici-

patory learning element. A randomly generated example gridworld is shown in Figure 1.

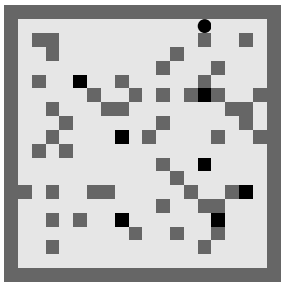

Fig. 1. An example gridworld

The agent starts in the lower left corner, and at each open square is able to take one of four actions (move up, down, left or right), unless that would result in it hitting one of the walls (dark gray squares, and here scattered over 15% of the grid and forming its outer perimeter). The agent's task is to reach the puck, (the black circle), whereupon it receives a reward of 1000. It should also avoid falling into a hole (black squares, here scattered over 2% of the grid), which would cause it to lose 10 points. Every other action costs the agent a single point. We assume that the agent has sensors sufficient to be able to determine when it cannot execute an action due to the presence of a wall, and because we assume the agent has a topological map, that it can unambigously determine its state and store data corresponding to it. In this very simple domain, this means that we can use a lookup table with an element for each grid square as the state space. Reinforcement learning is accomplished using an asynchronous temporal difference equation [4] similar to TD(0) [11].

In addition, we assume that the agent has a hole sensor for each direction which can sense when there is a hole directly adjacent to the agent in a given direction, and four "smell" sensors that provide an intensity reading (inversely proportional the distance to the puck) for each direction. Note that although the agent possesses these sensors, it does not a priori know how to use them.

Although this is obviously a highly simplified environment, and agents in it can hardly be considered situated, it serves as a clean and clear experimental domain where the points we discuss here can be easily illustrated.

3 Reinforcement Learning as Systematic Search

We can view the process of trying to find an initial solution as a search process where the agent itself is physically, rather than mentally, searching through the state space. This is similar to the classical idea of searching as problem solving

[8], except that since the search is actually being carried out (rather than being simulated in memory) only one node in the search tree can be considered "open" at a time. Korf [6] considered this problem within the framework of the classical deterministic search problem where an exact model of the state space is available, and Barto, Bradtke and Singh [1] showed that dynamic programming is a generalisation of this idea for stochastic problems. Although both papers included work on learning heuristics for the search space, these were simply more accurate values for states that had already been encountered, and neither paper considered the use of other sensors to predict the value of each state.

When searching a new state space, the simplest useful solution to the search problem is to have the agent select moves at random. This obviously does not perform very well because the agent spends much of its time in territory that it has already covered, until it finally stumbles across the puck.

An immediate improvement is to make the agent *explore greedily with respect to novelty*, where it chooses an action that has not been taken from its current state if one is available, and moves randomly otherwise. Although we would expect this agent to perform better than a purely random agent (because we would expect it to explore more efficiently) it nevertheless will often waste time randomly wandering through fully explored regions of the space until it chances across a state which has an unexplored action, because it is a greedy explorer. Note that this strategy requires a structure capable of recording which actions have been taken at previously visited states – a requirement easily fulfilled if a topological map is present but which could create difficulties in systems using function approximation techniques to compress the state space.

Reinforcement learning using (uniformly) optimistic initial values [11] for exploration is an improvement over this search strategy, because it propagates the optimistic exploration values of unvisited states through the state space, and thus causes an agent that has not found a solution to take the shortest route to an unexplored action from its current state. It can thus be considered a *systematic exploration strategy* because it thoroughly explores the state space, and does not waste time wandering around already explored regions of it. In addition, in more general situations where movement penalties are not uniform or where negative reward states are present, it is able to find paths to unexplored states without incurring unnecessary penalties. This is illustrated by the sample gridworld and coverage maps shown in Figure 2. Figure 2a shows a randomly generated sample gridworld, with Figures 2b, 2c and 2d showing the relative density (darker squares having been visited more) of visits over 100 runs of random, greedy exploration, and reinforcement learning agents, respectively, with each run starting in the lower left corner with no prior knowledge of the gridworld, and ending when each agent first finds the puck. The reinforcement learning agents clearly explore systematically and more thoroughly than the other agents, and in addition they are able to learn to avoid holes, as shown by the lighter colouring of the hole squares. The other searches cover the entire search space roughly uniformly, although they both appear to get stuck in the top left corner and very lightly cover the bottom right corner, probably due to the trap-like configuration of the

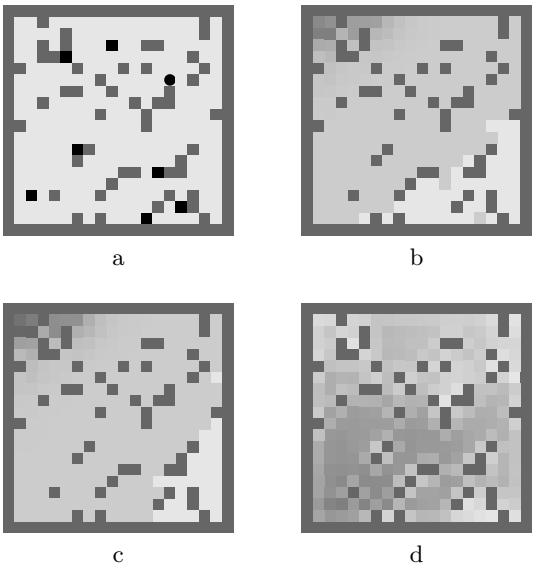

Fig. 2. A sample gridworld (a) and coverage maps for it using the random (b), greedy (c) and reinforcement learning (d) exploration strategies.

walls in each. This systematicity means that the agents using it find the goal in an average of 586 moves, whereas the greedy-exploration and random agents require an average of $13,595$ and $87,916$ moves, respectively.

The reinforcement learning agents still explore (on average) in a breadth-first manner – they cannot do any better because they have no a priori way to evaluate potential actions, and thus must consider all potential actions equal. Such search strategies can be considered *unguided* [8] because they do not use any form of heuristic to inform the search process. Without such a heuristic, we can expect to do no better – but if one is present, we can use it to perform a guided search of the state space.

4 Anticipatory Learning: Providing a Heuristic

When reward is necessarily delayed, we can expect reinforcement learning by itself to do no better than systematic search, because it simply does not have the information available to do any better. Taking the search analogy further, if we would like our agents to exhibit behavior similar to A* search [8], then we must provide them with a heuristic that can be used to rank novel state-action pairs in order of interest, and thereby guide search.

Unfortunately, building such a heuristic directly into the agent requires a high level of a priori knowledge of the task, entails significant design effort, and results in agents with fixed, rather than adaptive, search patterns. Using the

agent's sensors directly as the state space is also difficult because it removes the advantages of using a topological map – the Markov property will be lost, it will be difficult to link high level actions to states, and most of the agent's sensors will not be relevant to the task, resulting in an infeasibly large state space. The natural solution to this is to employ a learning element that attempts to rank all of the possible actions at a newly discovered state. Such a solution would ideally be sufficiently rapid to learn quickly, but sufficiently flexible to be able to adjust to new environments. To implement this, we connected four output sigmoid units [7] to the agent's four hole sensors and four smell sensors. Unless the smell sensors were all zero or all equal, they were pre-processed so that:

$$s_i = \frac{s_i - s_{min}}{s_{max} - s_{min}} \tag{1}$$

where s_i was the ith sensor, s_{max} the one with the highest reading and s_{min} the one with the lowest. The sensors were therefore scaled relative to each other, with the one with the highest reading returning 1, and the one with a lowest reading returning 0. The output units were then used to rank potential novel actions, with each untried action given an action value of:

$$v_i = kn_{novel}\frac{o_i}{o_{total}} \tag{2}$$

where k was the default optimistic initial value (set to 10), o_i was the ith sigmoid unit's output, n_{novel} was the total number of untried actions at the current state, and o_{total} the sum of their sigmoid outputs, resulting in values scaled against the total output for all novel actions. The action with the highest action value was taken, although this was not always a novel action since a previously taken action from the same state may lead to a higher reward.

In order to evaluate the benefits of this element, we compared the average total reward obtained by 100 agents using the standard reinforcement learning search method in the gridworld shown in Figure 2a with that obtained by 100 agents with the extra learning element as they were exposed to more and more training examples. Each agent with the anticipatory learning element was given 70 trials, consisting of a training stage and an evaluation stage.

Each training stage took place on a randomly generated gridworld, using the following procedure. First, the agent was left to find the puck using reinforcement learning and using its existing sigmoid units to estimate optimistic values. The ranking element weights were frozen during thise phase so that no learning took place. If the puck was found in fewer than 1000 steps, the agent made the remainder randomly, without learning, in order to further explore the gridworld. Finally, the agent was replaced in its initial position, and allowed to follow the path it had found to the puck. For each action that it took, the sigmoid units were updated towards a value of 0.95 for the chosen action, and 0.05 for the remainder, using gradient descent [7]. Following each training trial, the agents were evaluated in the gridworld shown in Figure 2a, with their ranking element weights frozen. The results obtained thus indicated performance in an environment in which the agents had never been allowed to learn in.

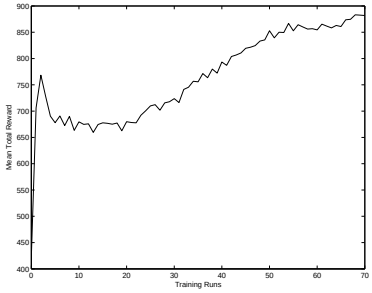

Fig. 3. Mean Reward over Time

Figure 3 shows the resulting improvements in reward over time, where it is clear that the anticipatory learning element is able to improve the agent's behavior and allow it to obtain a higher mean total reward as time progresses. The agent learns a relatively effective strategy fairly quickly, but is also able to improve steadily over time. The mean reward for the standard reinforcement learning agent over the same task was slightly less than 400, which matches with the level at which the learning agents start out. Figure 4 shows the drop in mean ranking error over time for the learning element, showing consistent but diminishing improvement.

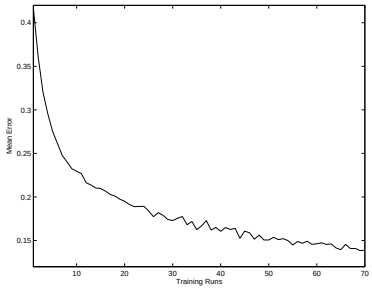

Fig. 4. Mean Ranking Error over Time

Finally, Figure 5 shows the resulting average search pattern after 0, 5 and 25 training experiences, which is a good indicator of the agent's expected search behavior. Initially, the agent spends most of its time in a breadth-first outward expansion, similar to the coverage map generated by an agent using reinforcement learning with uniformly optimistic initial values, as shown in Figure 2d. This is to be expected, since the untrained ranking network can be expected to rank novel actions randomly, which is equivalent to ranking them uniformly and then breaking ties randomly. Later, the agents learn to move towards the

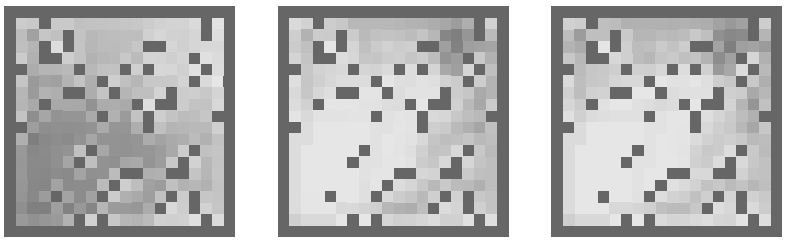

Fig. 5. Frequency Maps after 0, 5, and 25 Training Runs

smell while avoiding obstacles, and spend most of their time in the area where their smell sensors have very high readings (in the upper right corner near to the puck). It is clear from these patterns that the learning element is successfully able to focus the agents' searching behavior, even though it is never allowed to learn in this particular gridworld.

The anticipatory learning element is thus able to rapidly learn to significantly decrease the time taken by a reinforcement learning agent to find a solution, and to avoid holes. The resulting search behavior can be characterised as a guided search, with the anticipatory learning mechanism providing a form of heuristic.

5 Discussion

The mechanism demonstrated in this paper for anticipatory learning was very simple, but it supported relatively interesting, goal-directed behavior and provided some obvious behavioral benefits. Despite its simplicity, we feel that the work presented here highlights two interesting aspects of anticipatory learning systems. First, the system demonstrated here works because of the existence of a behavioral regime that provides a context in which anticipatory learning is useful. The ability to forsee the consequences of action is not useful in and of itself – rather, it is useful because it provides a means by which good actions may be chosen over bad. In this paper, the form of the behavioral strategy and learning mechanism used both afforded the opportunity for anticipatory learning and provided a framework in which it could provide behavioral benefits.

Second, although the sensors and learning element used in this paper were deliberately kept simple in order to demonstrate clear results, we would expect a kind of synergy between sensing and learning: that sensors and learning systems would be co-evolved to make learning and the generation of adaptive behavior flexible and computationally easy. In this case, anticipatory learning does not have to be exact – that would probably require a much more complex learning element – it just needs to rank novel actions in roughly the right order. Explicitly learning a predicted value for each state is likely to require significantly more computational effort, and may be difficult to accomplish within a reasonable portion of the task lifetime.

6 Conclusion

In this paper, we have introduced the use of an anticipatory learning element for novel state-action pair ranking in reinforcement learning agents. We have shown that anticipatory learning is effective as a heuristic that is capable of focusing search for such agents, and thus provided a concrete example of the behavioural benefits of adding anticipatory learning to an existing control system.

Acknowledgements

We would like to thank the Institute of Perception, Action and Behaviour at the University of Edinburgh for the use of its resources, the three reviewers for their useful comments, and the organisers of ABiALS 2004 for some good ideas. George Konidaris is employed under the HYDRA project (EU grant IST 2001 33060).

References

1. A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72, 1995.
2. M. V. Butz, O. Sigaud, and P. Gérard. Anticipatory behavior: Exploiting knowledge about the future to improve current behavior. In M. V. Butz, O. Sigaud, and P. Gérard, editors, *LNCS 2684 : Anticipatory Behavior in Adaptive Learning Systems*. Springer-Verlag, 2003.
3. M. V. Butz, O. Sigaud, and P. Gérard. Internal models and anticipations in adaptive learning systems. In M. V. Butz, O. Sigaud, and P. Gérard, editors, *LNCS 2684 : Anticipatory Behavior in Adaptive Learning Systems*. Springer-Verlag, 2003.
4. G.D. Konidaris and G.M. Hayes. An Architecture for Behavior-Based Reinforcement Learning. To appear, *Adaptive Behavior*, 2004.
5. G.D. Konidaris and G.M. Hayes. Estimating future reward in reinforcement learning animats using associative learning. In *From Animals to Animats 8: Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior*, July 2004.
6. R.E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42, 1990.
7. TM Mitchell. *Machine Learning*. McGraw-Hill, 1997.
8. S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Eaglewood Cliffs, New Jersey, 1995.
9. J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In J. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of the International Conference on Simulation of Adaptive Behavior*. MIT Press, 1990.
10. R.S. Sutton. Reinforcement learning architectures for animats. In J. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of the International Conference on Simulation of Adaptive Behavior*. MIT Press, 1990.
11. R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

Anticipation of Periodic Movements in Real Time 3D Environments

Vincent Labbé, Olivier Sigaud, and Philippe Codognet

poleia-LIP6, 8, rue du Capitaine Scott
75015 Paris France
{labbe,sigaud,codognet}@poleia.lip6.fr

Abstract. In this paper we present a method to anticipate periodic movements in a multi-agent reactive context, of which a typical example is the guard who patrols. Our system relies on a dynamic modeling of motion, based on a state anticipation method. This modeling is achieved with an incremental learning algorithm, dedicated to 3D real time environments. We analyze the performance of the method and demonstrate its usefulness to improve the credibility of pursuit and infiltration behaviors in front of patrolling agents.

1 Introduction

In a dynamic multi-agent environment, the reactive anticipation of movements of an opponent may be crucial to survive. In this paper, we focus on the anticipation of the motion of an agent who follows a well-defined periodic path. The guard who goes his rounds is a typical example. This guard is going to face another agent, for whom he will be either a prey or a predator. If the guard is a predator, it may induce several problems for the other agent: how to cut his trajectory without being seen, how to mislead him in another direction, etc. In the other case, the agent has to intercept the guard in what he thinks is the most favorable way for himself. Various criteria can be used, such as the quickest way, the one with the least hazard, etc. The method presented in this paper deals with this kind of matters. It is based on an incremental modeling of periodic movements carried out by a learning algorithm, dedicated to 3D real time environments.

Hereafter, we will use the verb “to patrol” in the meaning of “to patrol along a well-defined periodic path, while no major perturbation stops one’s behavior”.

Modeling this kind of patrol is interesting for video games, particularly in FPS[1] and strategic games. When a player commands several agents, it is interesting to be able to give them high level orders such as “watch a zone”, “infiltrate the enemy’s area” or “lay an ambush”. The “zone watching” behavior already exists in strategic games like StarCraft or Conflict Zone. But players miss infiltration or ambush behaviors

[1] “First Person Shooter”

orders. Our anticipation method allows one to get this kind of behaviors facing patrols.
This paper is organized as follows. In the next section we present related work regarding movement anticipation, especially in 3D real time surroundings. Then our method is explained in section 3. In section 4, our main results are presented and discussed. The following section discusses the possibility to apply our algorithm for the video game prey/predator context. In section 6, we discuss the benefits and limits of our method. Finally, we highlight the role of anticipation in our approach.

2 Background

A classification of anticipatory mechanisms is proposed in [1]. It describes four kinds of anticipation: implicit, payoff, sensorial and state-based. Our approach deals with the fourth category: state anticipation. To anticipate the movement of a patrolling agent, we build a model of the movement. This model allows the explicit simulation of the future motions of the agent. These predictions are directly used to improve escape or pursuit abilities and to obtain infiltration and ambush behaviors.

Christophe Meyer's SAGACE method [4] provides another example of state anticipation. His system learns how to anticipate the actions of a human opponent in the context of repeated games with complete or incomplete information. It is based on two Learning Classifier Systems (LCSs) whose rules can evolve thanks to a Genetic Algorithm. The first one models the opponents' behaviors, the other one plays depending on this model. These systems refine themselves during each game. Tested on the game ALESIA, with opponents using a fixed strategy, this method gives very good results because it determines an adapted strategy. However, it is designed to be used in "turn by turn" games. It seems that the SAGACE method is difficult to adapt to continuous real time games because LCSs scale poorly to continuous domains [2].

For continuous environments, Craig Reynolds developed algorithms to simulate motion behaviors in a visually credible way, notably of groups of animals, such as a fish shoal or a bird flock. These algorithms use simple and very quickly calculated mechanisms and which, combined, well adjusted and applied in a reactive way (about 30 times a sec) offer realistic motion results. A pursuit behavior endowed with a capacity of anticipation is proposed in [5]. It consists in leading the predator towards an estimation, at T steps of time ahead, of the prey's position. To realize this prediction, the predator supposes that the prey's speed will remain constant. The issue is to determine T. Reynolds suggests a simple method based on the distance between the prey and the predator, which is: $T = Dc$, where $\boldsymbol{D}$ is the prey/predator distance and $\boldsymbol{c}$ a parameter. This prediction is carried out at each time step, using the last observed velocity vector of the prey. This method does not pretend to be optimal, but improves the credibility of the pursuit behavior. Indeed, the extremely quick calculation does not reduce the agent's reactivity, and it seems to pursue his prey in a more efficient manner.

However, such anticipation finds its limits in the periodic motion framework. As an example, if the prey is following a circular closed path, if it is going faster than the predator and the parameter c is not well adjusted, this one will move on a smaller circle inside the one followed by the prey, without ever reaching it (see fig 1).

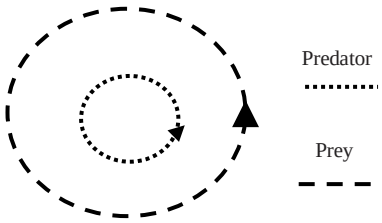

Fig. 1: A pathological case for Reynolds' pursuit behavior

Our method solves this kind of problem thanks to a more sophisticated model of the prey's motion than Reynolds' one (which is limited by the fact that the velocity vector is supposed constant). This model is learned in an incremental and reactive way and provides an estimation of a crossing point with the prey's trajectory without reducing the predator's reactivity.

Cyril Panatier [6] exposes another motion anticipation method, based on potential fields. The method is tested in a real time 3D environment with agents divided into two categories: friends and opponents. Each side has to push some pucks towards an opposed zone. Only one agent, the "adaptive" one, tries to interpret the others' movements in order to guess who his friends are. It assumes that the agent's velocity results in a combination of attraction/repulsion forces caused by other agents. To calculate the others' attraction/repulsion coefficients, the adaptive agent uses his surrounding perception, i.e. every agent's position, and his memory of the last perception and calculation. Thus it learns the potential field functions of every agent, in an incremental manner. Once the coefficients are calculated, the adaptive agent gets a global transition function letting him simulate the movement of each agent. In order to choose his next behavior it performs simulations of each agent motion. Then the selection is made by comparing the outcomes of each behavior.

This kind of anticipatory mechanism eventually endows one with infiltration or ambush abilities, but not facing a patrol. Indeed, by definition, the patrol's path is defined independently of the current behavior of other agents. Therefore the interpretation of a patrolling guard, by the mean of potential fields, would not result in a correct model.

In the video game Quake III, more centered on the prey/predator relationship, John Laird explains how his quakebot [7], based on Soar architecture, can anticipate some movements and other actions of his opponents. The bot starts by modeling its opponent from its observations. The model consists in observable variables like position, health, and current weapon, as well as non observable ones such as the current aim, which has to be guessed. Then he predicts the future behavior by simulating what he should do if he was in his opponent's situation with his own tactical knowledge. Thus Laird assumes that the enemy's goals and tactics are basically the same

as the quakebot's. Using simple rules to simulate his opponent's actions, the bot anticipates until he finds an interesting or too uncertain situation. The prediction is used to lay an ambush or to deny the enemy of a weapon or life bonus. This anticipatory mechanism, whose calculation takes a lot of time, is used only under some conditions in order not to reduce the bot's reactivity. Designed to anticipate human players' actions, this method seems unable to anticipate patrol movements. Indeed, in order to anticipate the guard's periodic motion, the bot should be able to consider the "patrol" mode like an internal state and above all, to model the path. As far as we know, this modeling capacity is not part of to the quakebot's characteristics.

We propose to address this problem thanks to our periodic motion modeling method viable in a highly reactive 3D environment such as Quake III's.

3 Movement learning algorithm

Our algorithm is dedicated to a real time modeling of periodic motion in 3D worlds. The model performs a linear approximation of motion so as to anticipate the next position. The velocity is supposed constant on each segment, equal to the average of velocities observed between both extremities of the segment. When the agent accelerates, the anticipation happens to be wrong and the model is updated. In order to do so, edges are adapted permanently to fit to the motion points that present the strongest accelerations. This dynamical adaptation is made through experience, justifying the notion of learning. It is driven by comparisons between predictions and observations and implies a motion abstraction. Hereafter, we call "track" the motion model.

3.1 Model

The model itself consists of a circular chained list. The circularity of the list means that the motion is considered periodic a priori. The maximal size of the list is the only model parameter. List elements are called "edges", they are made of four components:

1. a position vector in 3D space
2. a distance: estimation of the distance between last and current edges.
3. a time: estimation of run time between last and current edges.
4. a radius: estimation of the maximal distance between prediction and observation on the segment.

Figure 2 illustrates the relationship between a 2D motion and the track. There are:

- The motion trajectory: undulant pull.
- The corresponding learned model: arrows linking A to B, B to C etc., edges and the grey circles that symbolize uncertainty radius of these edges.
- The position of the observed object at instant t: the star
- The prediction of the model for the same instant t: the dotted circle.

As shown by the orientation of the arrows, A is before B in the list. If A is the last point of the trace reached, then the motion prediction is made according to the [AB]

segment and the uncertainty radius in B. The prediction is made of a position and a radius representing an uncertainty. In other words, it points out that the object should be in a perimeter around an accurate point. This uncertainty can be interpreted like a motion abstraction carried out by the algorithm.

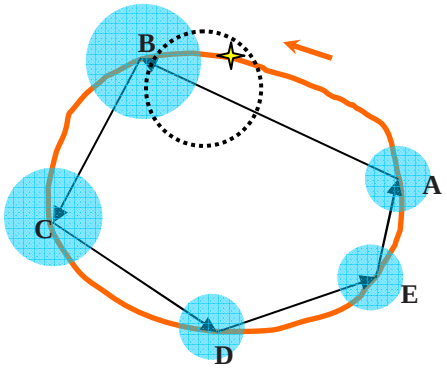

Fig. 2: example of track

On this scheme, the prediction is a success because the star is inside the dotted circle. Along time, the dotted circle moves from A to B where its radius will become the next edge's one, etc. Its velocity is constant on a segment: it represents an approximation of the average speed of the observed object on this segment.

3.2 Algorithm outline

At any time, the algorithm compares the position vector and observation date with the prediction realized thanks to the current model, and the model is updated according to the comparisons. When the prediction is wrong, a new edge corresponding to the current observation is added in the list. Then, in order to keep the model size constant, the algorithm must suppress an edge in a relevant way. The selected edge is the one whose suppression would involve a minimal loss of accuracy in prediction. The rest of the list is then updated in order to maintain the reliability of the model.
We define the model uncertainty as the average of the edges' radius weighted by the length of the corresponding segments. The model accuracy is the opposite of uncertainty. The selection of an edge is computed by anticipating the update of the list. Indeed, the uncertainty of the model grows during the update. Let A, B and C be three consecutive points of the track (see Fig. 3). Considering that B is selected then C is updated in the following way:

- $D_{AC} = D_{BC} + D_{AB}$
- $T_{AC} = T_{BC} + T_{AB}$
- The radius R_{t+1} is computed as:

$$R_{t+1} = \max(R_t, \|BH\|) \quad \text{where} \quad H = A + AH \quad \text{with} \quad AH = \frac{AC}{\|AC\|} \cdot \frac{T_{AB}}{T_{AB} + T_{BC}}$$

Where D_{AB} and D_{BC} are the distances from A to B and from B to C; T_{AB} and T_{BC} indicate the running time in the same way. H points to the position estimated when the real object is in B. At this moment, the gap between prediction and reality is theoretically[2] maximal on the segment [AC].

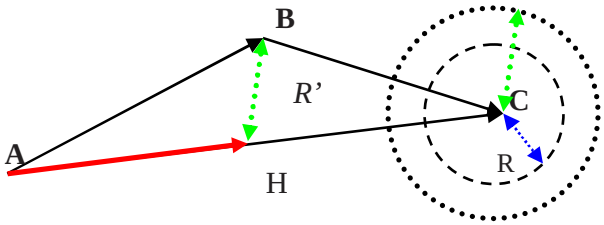

Fig. 3: uncertainty calculation

Another problem to solve is the discovery of the period. To know when the starting point is reached, the algorithm uses an elementary distance based on the average of the observed movements. If, for any reason (noise, sampling, etc.), the algorithm does not detect when the starting point is reached, then a dilation phenomenon appears, the track trying to model two or more periods as a single one. To deal with this problem, we have developed a mechanism that deletes the current first edge of the model when the position of a non-predicted observation is very close to another edge. This heuristic works quite well but can involve the inverse effect: the retraction phenomenon, when the trajectory has many intersection points with itself. The more intersection points, the more possibilities to meet a situation where the anti-dilation mechanism is triggered. It can result in a track reduced to a single segment to model the whole trajectory. The algorithm complexity in time and space is ***O(n)*** where ***n*** is the maximal size of the list. We also designate this size as "model complexity".

4 Experiments and results

We have tested our algorithm on several noisy periodic movements. The motion learning performance is evaluated according to two criteria linked with predictions:

- The track *reliability* is the right predictions rate, during one period.
- The *uncertainty*, as defined above.

[2] « theoretical » meaning in case where one starts exactly from A, what's not always right; so this is why we need to increase the radius.

Hereafter, we present results on two particular trajectories, in 8 and in W shapes. We define critical points as the points that present an outstanding acceleration. The intersection points of the trajectory with itself are also important. They make the discovery of period more difficult because of the mechanism used to solve the dilation problem. Thus the number of critical and intersection points determine the modeling difficulty. For these examples, the guard moves with a constant speed. Therefore the critical points correspond only to the curves' acute turns. Figure 4 shows the 8-trajectory endowed with 4 critical points (white squares in the initial curve picture) and the three tracks obtained after two periods. These tracks consist of 4, 12 and 24 edges. The circles symbolize the uncertainty radius of tracks' edges.

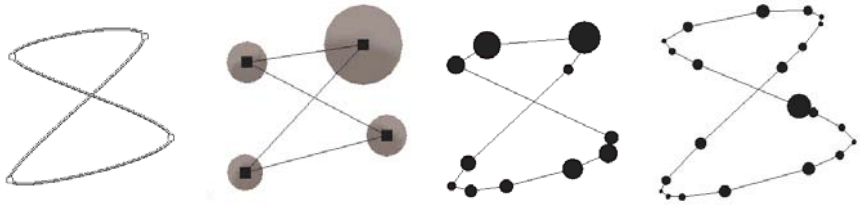

Figure 4: initial trajectory and models with 4, 12 and 24 edges

We can see that the edges are distributed around the critical points. Light curves contain only few edges. It illustrates the fact that the algorithm selects the trajectory points that present strong acceleration. Figure 5 presents the W-trajectory and some track obtained after two periods with 5, 10 and 20 edges models.

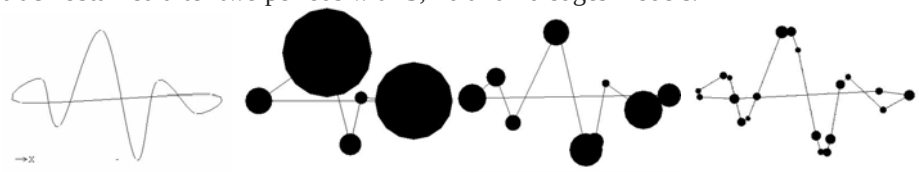

Figure 5: Initial trajectory and models with 5, 10 and 20 edges

This more complex trajectory has height critical points and five intersections. These pictures corroborate the trend brought to mind by the 8-trajectory: the more complex the model, the more accurate the predictions. The following graphs present the comparisons of uncertainty and reliability of three models of increasing complexity, along 10 periods regarding the two motions presented above. The values represent the averages from a sample of 10 tests.
Figure 6 presents the evolution of uncertainty:

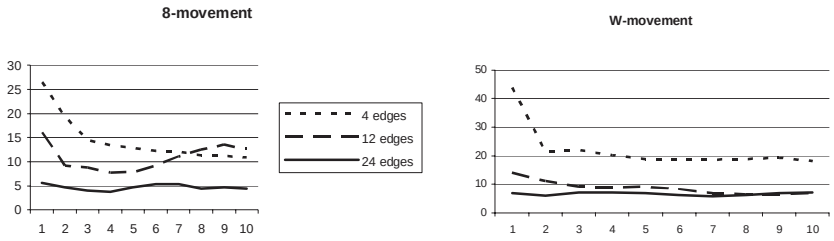

Fig. 6: uncertainty evolutions

The average uncertainty on predictions tends to increase with the trajectory complexity and to decrease with time and model complexity. After 3 periods, the uncertainty is relatively stationary. The curve that represents 12 edges model evolution on the 8-movement gets higher after the fifth period. This corresponds to the dilation phenomenon. Figure 7 deals with reliability evolution:

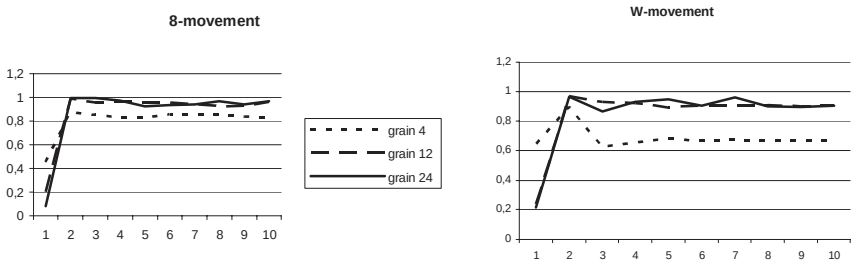

Fig. 7: reliability evolutions

We can see that the reliability of the predictions is almost stationary after the third period. Furthermore, the 4 edges model is not complex enough to allow a reliable anticipation of the W-movement.

5 Patrol anticipation application

As outlined in the beginning of this paper, our learning algorithm has been designed to anticipate patrol movements in a real time 3D environment, such as rounding guards in a video game surrounding. We can anticipate the guard's motions according to two main objectives: interception or avoidance.

5.1 Interception

Our interception method is based on the following principle: from the motion model of the guard, the agent decides to intercept him at the first edge that he can reach

before the guard by optimizing a particular criterion. The nature of this criterion determines the type of behavior. For instance:

- Minimizing interception time to obtain a pursuit behavior
- Maximizing position benefits to obtain an ambush behavior

The choice and computation of this criterion depend on the application, which is not the point of this paper. To check if the agent can reach a position before the guard, one must estimate the running times. The guard's running time is easy to estimate thanks to the track model corresponding to its trajectory. As far as the agent is concerned, it depends on his own model. From a simple calculation in function of the distance and average speed, the estimation remains dependent on the application and computation possibilities. In order to demonstrate the feasibility of our method, we have implemented a simple demonstration of a pursuit behavior in the context of a "toy" video game environment with a 3D real time surrounding.

5.2 Avoidance

The guard's avoidance behaviors are investigated from the point of view of infiltration, i.e. when an agent has to cross the guard's trajectory without being seen nor touched. We propose a simple anticipation method allowing infiltration behaviors.
We assume that the agent does not adapt his reactive steering parameters: once he starts moving, he will not stop anymore until he reaches his goal. The challenge is to start moving at the right moment to avoid being seen or touched. In order to determine this moment, the agent must simulate his movement as well as the guard's on several steps. From the current situation, he checks at each step if he is not in a critical situation. Critical situations can be collisions, inclusions in the guard's perception area, etc. If the agent fails, then he starts again until he finds a favorable moment to move. Here again, we have demonstrate this behavior in the same context as above.

6 Discussion

The anticipatory mechanism presented in this paper endows an agent with interception or avoidance capabilities in front of a patrolling guard without equivalent in the video game research literature. We have empirically demonstrated how this mechanism could be used, but we did not implement it yet in a actual video game. Actually, we must confess that there are not so many video game contexts in which our mechanism could be used. In most cases, the agents are designed to anticipate the human players' behavior [4] [7]. But the human player seldom behaves in a periodic way. The only favorable context is when a software agent is confronted to another software agent. This is the case, for instance, when the human player can give high level orders like "keep a zone" or "invade this building" and the agent must realize

the behavior on its own. Very few video games offer this possibility so far[3] but we believe that this situation will change with the raise of interest in AI in the video games industry.
From a more technical point of view, our algorithm still suffers from some limitations. Given the dilation and retraction problems outlined in section 3, one needs to determine the horizon of the simulation, the number of steps and the complexity of the problem in a concrete way. In order to keep the algorithm reactive and to improve the behaviors significantly, we must find a good compromise between the resources used and the accuracy of predictions. One way to act on this compromise is to tune the model complexity. Experiments show that to obtain reliable and accurate predictions, the model's complexity must be sufficient in regard to the movements that must be anticipated. The more numerous the critical points and self-intersections are, the more complex the model of the periodic motion must be. As we have shown, this concern is not critical since the algorithm complexity is in ***O(n)***. Another matter of discussion is the possibility to generalize the algorithm. An extension to the case where several guards patrol in the same area appears quite straightforward. On the contrary, its extension to non-periodic movements seems to be more difficult and should be studied in greater detail.

7 Conclusion

In this paper we have dealt with anticipatory mechanisms in two ways: on the one hand, anticipation is used to drive learning mechanisms, on the other hand, the learned model results in the possibility to anticipate periodic movements through mental simulations. Our system uses a dynamic modeling of movement, driven by an anticipatory mechanism. It is a clear case of state anticipation where the prediction error is used as a feedback signal to improve the model at each time step. Thanks to an incremental learning algorithm dedicated to real time 3D environments, it endows an agent, among other things, with infiltration and pursuit behaviors. Based on distances computation, the algorithm can easily be extended to N dimensions space. Though our system has proven efficient on toy problems, its applicability to an actual video game still needs to be demonstrated, which we hope to do in the near future.

References

1. Butz, M.V., Sigaud, O. and Gérard, P. (2003) *Internal Models and Anticipations in Adaptive Learning Systems* In Butz et al. (Eds) LNCS 2684 :*Anticipatory Behavior in Adaptive Learning Systems* , pp 87-110, © Springer Verlag.

[3] StarCraft and Conflict Zone are notable exceptions.

2. Flacher, F. and Sigaud, O (2003). Coordination spatiale émergente par champs de potentiel, *Numéro spécial de la revue TSI : Vie Artificielle*, A. Guillot et J.-A. Meyer (Eds), Hermès, 171-195.
3. Parenthoën, M., Tisseau, J. and Morineau, T., in Rencontres Francophones de la Logique Floue et ses Applications (LFA'02), 219{226, Montpellier, France, 21-22 Octobre 2002.
4. Meyer, C., J.-G. Ganascia, and Jean-Daniel Zucker. (1997). Modélisation de stratégies humaines par Apprentissage et Anticipation génétiques. Journées Française de l'Apprentissage, JFA'97, Roscoff, France
5. Reynolds, C. W. (1999) Steering Behaviors For Autonomous Characters, in the proceedings of Game Developers Conference 1999, Miller Freeman Game Group, San Francisco, California, pages 763-782.
6. Panatier, C., Sanza, C., Duthen Y. *Adaptive Entity thanks to Behavioral Prediction*. in: *SAB'2000 From Animals to Animats, the 6th International Conference on the Simulation of Adaptive Behavior, Paris*. Meyer, ... , p. 295-303, 11 septembre 16 septembre 2000. Accès: http://www-poleia.lip6.fr/ANIMATLAB/SAB2000/
7. Laird, J. It Knows What You're Going to Do: Adding Anticipation to a Quakebot. in AAAI 2000 Spring Symposium Series: Artificial Intelligence and Interactive Entertainment, March 2000: AAAI Technical Report SS-00-02.

The role of epistemic actions in expectations

Emiliano Lorini[1]; Cristiano Castelfranchi[2]

[1]University of Siena, Cognitive Science Doctorate
Via Roma 47, Siena, 53100 Italy
e.lorini@istc.cnr.it
[2]Institute of Cognitive Science and Technology–CNR
Viale Marx 15, Rome, 00137 Italy
c.castelfranchi@istc.cnr.it

Abstract. The goal of this paper is to analyse the central role of the epistemic activity in anticipatory mental life. A precise typology of different kinds of epistemic actions will be presented. On theses basis, a precise characterization of expectations about the success in the achievement of an intended result will be provided. Moreover, two specific kinds of epistemic actions (Epistemic Control and Epistemic Monitoring) will be defined and their functions in the goal-processing will be identified.

1 Introduction

A crucial feature of intelligent Agents is their being pro-active not only reactive i.e. their ability to deal with the future by mental representations or specific forms of learning. For guiding and orienting the action a representation of the future and more precisely a representation of future effects and of intermediate results of the action is needed. Moreover, having a mind means to have anticipatory representations, i.e., predictions and goals; not just perception, beliefs, memory. In our previous works we have provided an ontology of anticipatory mental states, we have analyzed them both in terms of "epistemic" components and in terms of "motivational" ones and we have analyzed their two quantitative dimensions (the strength of the belief and the value of the goal)[1].

In the present work we will argue that when an agent believes that some result *p* will be true in the future and he is concerned with that result, always he needs to acquire information in order to verify whether that result will be (is being) achieved or not. We will argue that every expectation has a dynamic component, therefore every expectation should be conceived as an activity. The dynamic component of every expectation is the Epistemic Action (EpA): the action aimed at acquiring new information, at verifying the truth value of a proposition, at testing beliefs. In section 2 we will propose two dimensions for categorizing EpAs: a modal dimension (how

[1] For the definitions of the different types of expectations see [11]. For the formal theory of anticipatory mental states and the related semantic see [6]. In that work the multi-modal intentional logic given in [7] extended to cover probabilities (for representing the strength of the beliefs) was used.

are EpAs effectively performed?) and a functional one (why are EpAs performed? Which is their purpose?). Moreover, a schema of Discovery Learning will be used as a possible application of our EpAs ontology. Finally in section 3 an analysis of two specific kinds of EpAs will be given: the expectation-based notions of Epistemic Control (EpC) and Epistemic Monitoring (EpM).

2 Epistemic Actions

"Epistemic Actions" (EpAs) are *actions aimed at acquiring knowledge from the world*; any act of active perception, monitoring, checking, testing, ascertaining, verifying, experimenting, exploring, enquiring, give a look to, etc.
Epistemic Actions cover both:

- *rule-based, goal-directed action*, like anticipatory classifiers (see [4]), which contain an anticipatory representation of the result (*proto-intention*) and are reinforced by the positive match of it with the perceived result; and
- *intentional actions*, where the goal is an anticipated mental representation of a result that is the starting point for selecting the appropriate plan or action, guiding planning, guiding performance, stopping behavior (see [12]).

In this paper we are interested in giving an account of intentional EpAs.

2.1 The modal dimension

Epistemic intentional actions can be distinguished in terms of modus operandi: how a specific epistemic action is realised. There are 2 general macro-modal categories of epistemic actions:

- *Pure Epistemic Actions* (Waiting for);
- *Parasite Epistemic Actions* (Looking for).

Pure epistemic actions are identifiable through verbs of sensory actions: to see, to observe, to hear etc…We can define them as: *actions that are specialised for epistemic functions: to the acquisition of knowledge, the verification (confirmation) of beliefs etc*...Since actions are identifiable in terms of the results they are done for, we could define pure epistemic actions as: *Actions that are specialised for achieving epistemic results: to the acquisition of knowledge, the verification (confirmation) of beliefs etc*...Pure epistemic actions never change the state of the world, they just change the knowledge of the agent (see [10]).
Parasite epistemic actions on the other side are: *pure epistemic actions that exploit pragmatic actions in order to achieve the epistemic result they are done for*. Parasite epistemic actions change both the state of the world and the knowledge of the agent. In principle it would be possible to define the pure epistemic intentional action as the deliberated *fixation* of attention. There are two forms of parasitism of pure epistemic actions on pragmatic ones. Pragmatic actions and epistemic actions can be done *sequentially* or in *parallel*. The pragmatic action is a mean for achieving the epistemic goal (to know the truth value of *p*). In order to know whether *p* is true or not the agent has to check whether *p* is true or not (pure epistemic action) and before it the agent

has to execute a pragmatic action *a1*. The pragmatic action (the first component of the plan) creates the conditions for the execution of the pure epistemic action (second component of the plan). For example, imagine an agent wants to know whether "outside it is snowing or not" (epistemic goal). The agent will first open the door of the house (action *a1*) and will move his own attention towards the external environment (pure epistemic action).

There is another important distinction that should be addressed at this point. There is a difference between "Checking whether" and "Checking what".

- *Proposition-based epistemic actions* (Check whether). I test the truth value of a specific proposition;
- *Proposition-free epistemic actions* (Check what).

In the first case (for example: "is a (any) mail arrived?" "some mail is arrived; right?") the answer is Yes/No, True/False. In the second case the question (test) is open (for example, "some mail is arrived: which one?").

In psychology of attention filtering tasks have been proposed as experimental platforms. Filtering tasks are suitable experimental domains for analysing both previous kinds of EpAs (Check whether and Check what). For instance consider the following task discussed in [14]. A subject has to say what is written in a RED word that appears in a screen with other words of different colours. The subject has to filter out more than one word that appear in the board. He can adopt one of the following strategies. On the one side he can check in parallel "whether" a word *1* is a RED word,…, "whether" a word *n* is a RED word one till he finds a RED word. The strategy is realised through deliberately *dividing* and *shifting* the attention on different items in the screen. There are many *perceptual objects* that are matched with one *epistemological representation.* On the other side the agent can *select* a given word *1* in the screen, *fixate* the attention on it and check in sequence "whether" the word *1* is a colour *1*,…, "whether" the word *1* is a colour *n*. There is only one *perceptual object* that is matched with many *epistemological representations.*

The parallel or sequential execution of Proposition-based epistemic actions (either pure or parasite) aimed at testing reciprocally excluding propositions is considered in our framework a Proposition-free epistemic action.

The final important distinction that we want to present concerns the sources of beliefs and the general organization of the belief network of an agent. Indeed we can distinguish:

- *Pure Epistemic Actions on perceptive sources;*
- *Pure Epistemic Actions on inferential sources.*

An agent executes a Pure EpA on a perceptive source in order to test a belief that *p* if and only if he is simply testing the belief that *p* "by matching the *epistemological representation* of the object designated in *p* with the *perceptual object* (the referent) and no inferential process is involved in the process of verification"[2]. On the other

[2] This way to approach to the problem is similar to the one given in [13] where the process of testing trough perception the appropriateness of a Frame (for representing categories and situations) is analyzed. We assume here that a proposition p can be specified as a predicate of the following form "the |perceived object| is a |dog|" where the category |dog| can be organized (as in [9]) in terms of an *epistemological representation* (which is used to recognize through perception the instances of a category, for instance a 3D model) and an *inferential*

side an agent executes an EpA (either Pure or Parasite) on an inferential sources in order to test a belief that *p* if and only if he is testing the belief that *p* through some inferential process and no match with perceptual *object* is involved in the process of verification[3].

2.2 The functional dimension

Let us give here a functional typology of Epistemic actions. We distinguish two general functional categories of EpAs:

- *Goal-driven epistemic actions*;
- *Curiosity-driven epistemic actions*.

The category *Goal-driven epistemic actions* can be further decomposed in several sub-categories. There are different EpAs that are aimed at verifying different kinds of beliefs. Those different kinds of beliefs have a specific role in different stages of the goal processing where different kinds of goals/intentions can be identified.
In the model presented in [5] eight phases are identified and nine different categories of beliefs, goals (or intentions) play a role in each of them. Beliefs in each phase are *reasons* for selecting a certain motivation *G1* rather than another motivation *G2*. For instance if the agent believes that goal *G1* is more achievable than goal *G2* then the agent will select *G1*. *At each phase of the goal processing a specific kind of Epistemic goal can be activated and that goal determines the execution of a specific kind of Epistemic Action: an EpA aimed at testing a specific belief involved in the goal processing.* We do not describe here the full model and we do not consider all beliefs involved. We merely focus on the two following Macro-phases.

Macro-Phase 1: **From the commitment with a single result to the last evaluation before the execution of the action**.
There are several beliefs that play a role as *bases for deciding what mean to select for achieving an intended result*: *beliefs about instrumentality* (Belief-set 1) (belief about the possibility of achieving the intended result through the action), *Beliefs about the possibility of obtaining additional rewards and paying additional costs* (Belief-set 2), *beliefs about ability and opportunity* (Belief-set 3). BEL-INSTR x a1 p, BEL-RewCost x a1 p, BEL-AbilOpp x a1 p indicate beliefs concerning *p* that belong to Belief-set 1, 2, 3.[4]. Once an action has been selected on the basis of the previous beliefs, the action becomes the object of a *future-directed intention*. Before executing the action (and so going to Macro-phase 2) the agent must believe that necessary conditions and resources for executing the intended action are holding. Let us call *beliefs about necessary conditions and resources for executing the intended action*

---

representation (collection of encyclopedic knowledge about the category) and a *designator* (the symbol used to refer to the category).

[3] We do not consider in the present analysis *Pure Epistemic Actions on mnestic sources* that are according to Audi [2] another important kind of Basic sources of belief (together with perceptive sources) where Basic Sources of belief are those experiential sources whose justificatory power is not derivative (it is not obtained by some inferential source).

[4] We do not use in this paper an extensive formal analysis. Different formalisms are available for different kinds of beliefs in the goal processing. For instance see [17] for a logic of ability, opportunities and power.

that are *bases for deciding whether acting or not* BEL-CondRes x a1 p. Those beliefs belong to Belief-set 4. In Macro-phases 1 EpAs are executed: 1) for verifying whether the final intended result will be achieved or not; 2) for verifying whether a certain action is a good mean for achieving the final intended result.

Macro-Phase 2: From the "command" of execution of the action to the end of the action.

Once an action has been sent to execution EpAs can be executed at different levels of the temporal trajectory that goes from the beginning of the action to the expected time point of the achievement of the final intended result.

Beliefs about success or failure in the achievement of the final result (Belief-set 5) play a central role in this Macro-phase. The agent believes that *p* will be successfully achieved while he is intentionally doing the action relative to (instrumental) the result *p*. Let us call a *belief about future success and failure in the achievement of the final result* in Belief-set 5: BEL-FUT-SUCC x a1 p.

Beliefs in Belief-set 5 are always objects of verification in Macro-phase 2 (Epistemic Monitoring) (see Thesis 1 in section 3). It is crucial to argue that the verification of success can also be executed in Macro-phases 1.

In Macro-phase 1 and 2 *belief about success or failure in the achievement of intermediate results* play a role too. Since not only simple actions but also plans can be sent to execution, EpAs can be applied recursively for verifying the success in the achievement of those results that stand between the beginning of the *complex action* (or *plan*) *a* and the final intended result *p* (tests on the intermediate results of the plan $a = a_1,\ldots, a_n$). Finally in Macro-phase 1 and 2 *beliefs about necessary intermediate conditions for executing* $action_i$ *of a full plan* $a = a_1;\ldots; a_n$ play a role too. Generally EpAs on beliefs about intermediate conditions have a role in *conditional planning*. Before executing a plan or during its execution the agent can test whether intermediate conditions for executing $action_i$ are holding or not in order either: 1) to execute the sub-component $action_i$ of the plan when the conditions are holding or 2) to execute a different sub-component $action_l$ when the conditions are not holding (let us call it *corrective action*).

The different kinds of belief that we have described can be organized through a *Dynamic Belief Network* with (positive or negative) mono-directional and bi-directional links of support between different beliefs.

Belief-sets about the success in the achievement of the result: Belief-sets 1, 3, 4, 5. We assume that both in Macro-phase 1 and in Macro-phase 2 beliefs in Belief-sets 1, 3, 4 are "signs" (supports) of beliefs in Belief-set 5. In Macro-phase 1, 2 an EpA on a belief in Belief-set 1 or 3 or 4 can be executed in order to have a verification of a belief in Belief-sets 5. For instance a high strength belief about "necessary conditions and resources for executing the intended action *a1*" (belief in Belief-set 4) supports the belief that "*p* will be successfully achieved" (belief in Belief-set 5). Moreover, a high strength belief about "the ability to execute the intended action *a1*" (belief in Belief-set 3) supports (together with the belief in Belief-set 1 about "the degree of instrumentality of *a1* with respect to *p*") the belief that "*p* will be successfully achieved" (belief in Belief-set 5).

Previous beliefs in Belief-sets 1, 3, 4 are special kinds of *inferential sources* (see section 2.1) with respect to beliefs in Belief-set 5.

Belief-sets about the "goodness" of the mean: Belief-sets 1, 2, 3, 4, 5. We assume that both in Macro-phase 1 and in Macro-phase 2 beliefs in Belief-sets 1, 2, 3, 4 support a more general belief about "the 'goodness' of a mean" chosen (or not yet chosen) for achieving the intended result *p*. But the Belief Network becomes even more complex after that the agent has selected the action for achieving the final intended result. Beliefs in Belief-set 5 support beliefs in Belief-sets 1, 3, 4 and indirectly give a support to the general belief about "the 'goodness' of the mean". Indeed whenever an agent believes with high strength that "the result will be successfully achieved through the intended action *a1*", the belief about "the ability to perform *a1*" and the belief about "the degree of instrumentality of *a1* with respect *p*" are directly strengthened and so the general belief about "the 'goodness' of *a1*" is indirectly strengthened.
The organization of the Belief Network shows that different kinds of beliefs both support the belief about "the success in the achievement of the final result" and the belief about "the 'goodness' of the mean".

Curiosity-driven epistemic actions (the second general functional category of EpAs) are simply driven by the meta-goal of acquiring new information, of getting new knowledge.

2.3 Possible application: discovery learning

Let us cross the modal typology with the functional typology. At this point it is relevant to notice that parasite epistemic actions can be planned in order to:
1. merely achieve an epistemic goal (parasite epistemic actions driven by curiosity);
2. achieve an epistemic sub-goal that is a necessary condition for achieving a practical goal (parasite epistemic actions driven by practical goals).
Our double dimension of EpAs (functional and modal) can be applied for describing schemas of *discovery learning* (see [1] for further analysis). Discovery learning is in fact the intentional version of the functional learning that was already instantiated in TOTE cybernetic system [12] and that is embedded in BDI cognitive architectures [15]. In BDI models and in TOTE at each round after the execution of an action the agent tests automatically whether the action *a1* has been successful in the achievement of a given result *p*. If the action *a1* has been successful then the belief that *p* and the belief about instrumentality *a1* → *p*, the beliefs about ability etc… are strengthened (viceversa in case of failure). That test is not intentional as well as the learning associated with it: the test is automatic and the learning is merely functional. On the contrary in *discovery learning* the agent has the explicit epistemic goal *G3* of discovering (learning) a good procedure to achieve a pragmatic intended result *p* at time *t1* and as a sub-goal of *G3* the agent has the explicit goal of
testing whether a given procedure a1 is a good mean for achieving result p at time t1.
The verification of the general belief about "the 'goodness' of the mean *a1*" is inserted in a general plan of intentional learning for achieving the final intended result *p*. Indeed the agent could reason as follows. In order to achieve *p* at time *t1* I must discover whether a procedure *a1* is effectively good for achieving *p* at *t1*. In order to discover that, I have to test the actual hypothesis about instrumentality of *a1* with respect to *p* at *t1* and about the ability of doing *a1* at *t1*. In fact (assuming that the

agent has access to real relations of support among his own beliefs) beliefs in Belief-sets 1, 3 support general beliefs about "the 'goodness' of the mean" (see previous analysis of Belief Network in section 2.2). Let us assume that *the agent believes that if the procedure a1 allows to achieve p at t1-n then the procedure a1 will allow to achieve p at t1* (Assumption*). Since beliefs in Belief-set 5 (about success) support beliefs in Belief-sets 1 (about instrumentality) and 3 (about ability) and support indirectly general beliefs about "the 'goodness' of the mean", it is much more economical to perform procedure *a1* at *t1-n* and check whether *a1* is successful in achieving *p* at *t1-n* in order to test (given Assumption*) the belief in Belief-set 5 about "the success in the achievement of *p* at *t1*" and so (first step) in order to test the beliefs in Belief-sets 1 and 3 about "the ability to do *a1* at *t1*", about "the degree of instrumentality of *a1* with respect to the achievement of *p* at *t1*" and (second step) in order to test the general belief about "the 'goodness of *a1* with respect to the achievement of *p* at *t1*".

In order to test his hypothesis the agent can plan to "try" to do at time *t1-n* the procedure *a1*. *TRYING a1 p at time t1-n* means here: *execute the procedure a1 + test whether procedure a1 allows to achieve p at t1-n (test whether the procedure is successful in the achievement of p at t1-n)*.

The present schema of discovery learning is an interesting example of pragmatic action that is executed in combination with a pure epistemic action **in order to** achieve first an epistemic goal (i.e. to discover a good procedure for achieving *p* at *t1*) that is a necessary condition for the achievement of the final intended result *p* at *t1*.

3 Epistemic control and epistemic monitoring

Expectations as mental states are not enough well described by a static belief about future events or a static combination of belief + goal; they have a more active aspect and they elicit a specific behavior that characterizes the "activity" of waiting for, and expecting. Important components of any expectation are either the *Epistemic Control* (EpC) or the *Epistemic Monitoring* (EpM). The category Epistemic Control (EpC) and the category Epistemic Monitoring are for us sub-categories of the Epistemic Action category.

We define **Epistemic Control on result p** as follows.

Any intentional Epistemic Action: 1) aimed at testing the epistemic component of the (positive or negative) expectation that p will hold at a time t1 in the future; 2) executed before that the agent has the present-directed intention to do an action with the further intention to achieve p at t1.

The previous definition can be formalized as follows[5].

POS-EXPECT x p = (BEL x (LATER p)) ∧ (INTEND x p) ∧
((¬∃ a INTEND x a | (INTEND x p)) v ((∃ a INTEND x a | (INTEND x p)) ∧ **(1)**

[5] We use again the logic given in [7] in order to formalize our notion of Epistemic Control. Moreover, the notion of relativized intention is used to express the mean-end relation in the decomposition of the plan (action *a* is intended by the agent relative to an intended result *p* i.e. the agent is committed to do action *a* and if he discover that he does not intend that *p* then he drops the commitment with the action *a*).

(¬ ∃ a (INTEND x a | (INTEND x p))?;a))).
NEG-EXPECT x p = (BEL x (LATER p)) ∧ (INTEND x ¬ p) ∧
((¬∃ a INTEND x a | (INTEND x ¬ p)) v ((∃ a INTEND x a | (INTEND x ¬ p)) ∧
(¬ ∃ a (INTEND x a | (INTEND x ¬ p))?;a))).
EPISTEMIC-CONTROL x p = ((POS-EXPECT x p) v (NEG-EXPECT x p)) ∧
(DOES x **TEST** (LATER p))

An Epistemic Control is an Epistemic Action that can be only executed in Macro-phase 1. In the previous formalism (BEL x (LATER p)) in the definition of the positive and the negative expectation represents a belief about possible future success (in Belief-set 5); (DOES x **TEST** (LATER p)) represents the fact that a given test (EpA) on proposition LATER p is executed. We do not develop here the formal theory of the different kinds of test that can be executed for verifying the truth value of a given proposition. We want only to remark that different kinds (**TEST** (α)) should be specified by referring to the modal dimension given in section 2.1 (Pure versus Parasite EpAs, EpAs on inferential sources versus EpAs on perceptive sources)[6]. Moreover, we have formalized the fact that the agent intends to achieve the result *p* (and believes that *p* will be achieved) and the fact that two situations are possible: 1) the agent does not have any intention to execute an action relative to the intention to achieve the intended result *p* (Macro-phase 1, before having choosen the action); 2) the agent has an intention to execute an action relative to the intention to achieve the intended result *p* (Macro-phase 1, after having chosen the action). In this second situation a constraint is given in the formula. It is excluded the case in which the agent is committed to "believing he is about to do the intended action (as a mean for achieving intended result *p*) then doing it" and finally he does it. Given the constraint it is impossible that the agent executes an Epistemic Control in Macro-phase 2. This is as close as we can come to exclude from our definition of Epistemic Control the case in which the agent executes an Epistemic Action on the belief about success in the achievement of the final intended result "having the present-directed intention to do an action with the further intention to achieve that result" (or "doing intentionally an action with the further intention to achieve that result")[7].

We define **Epistemic Monitoring on result p** as follows.

Any intentional Epistemic Action: 1) aimed at testing the epistemic component of the (positive or negative) expectation that p (where the expectation that p is composed by the intention that p will be true at a given moment t1 in the future and the belief

[6] One of the most powerful logics of test actions is the one given in [18]. In that dynamic logic a test action on an arbitrary proposition *p* is simply a procedure that: 1) establishes the truth of either proposition *p* or proposition *¬p* (property of informativeness); 2) performs an update of the belief-base given either the true proposition *p* or the true proposition *¬p*. That formal theory does not provide an extensive analysis of the different ways in which an EpA can be performed.

[7] The distinction between *Present-directed Intentions (*or *Intention in Action)* and *Future-directed Intentions (*or *Prior Intentions)* has been introduced in Philosophy of Action. There are several converging and diverging definitions and analysis of the previous notions. In the present analysis we agree with the definition of Bratman [3] who assigns to the intention (as a mental state) the feature of *persistence* (if an agent intends to do an action then the agent is committed to do that action and he will give up his intention if and if only if particular conditions hold). According to Bratman a Present-directed Intention is the intention to do something *now* whereas a Future-directed Intention is the intention to do something *later.*

that p will be true at a given moment t1 in the future); 2) executed when the agent "has the present-directed intention to do an action"[8] *(or "does intentionally an action") with the further intention to achieve p at t1.*

We include in our definition of Epistemic Monitoring only the case in which the agent executes an Epistemic Action on the belief about success in the achievement of the final intended result "having the present-directed intention to do an action with the further intention to achieve that result" (or "doing intentionally an action with the further intention to achieve that result").

Given the previous definition it follows that an Epistemic Monitoring can only be executed in Macro-phase 2 at different stages of the execution of the action aimed at *p* or just after that the action aimed at *p* has been completely executed. Epistemic Monitoring is a test on BEL-FUT-SUCC x a1 p in Belief-set 5 (see section 2.1) that is executed during the execution of the action aimed at *p*[9]

At this point we want to argue that the following statement is valid.

THESIS 1. An Epistemic Monitoring is always performed in Macro-phase 2.

Indeed as soon as an action has been executed for achieving an intended result *p*, there is at least an automatic not intentional perceptive test on the success of the action (see the brief discussion in section 2.3 about TOTE and BDI models). The thesis becomes even stronger in realistic open worlds where there is always some degree of uncertainty and intentional tests on the success of the action are generally performed even during the execution of the action.

4 Conclusion

From the previous analysis it seems that Expectations should be considered as unitary subjective experiences, typical and recognizable mental states that have a global character. Although made up of (more) atomic components Expectations form a *gestalt*. How can we account for this gestalt property in our analytic, symbolic, (de)composition framework? We have implicitly pointed out some possible solution to this problem. For example, a higher level predicate exists (like EXPECT) and one

[8] The notion of "Intending to do something" (the "Pure Intending") and the notion of "Doing something intentionally" have been clearly distinguished in [8]. The Simple View assumes that "an agent does something intentionally if and only if "the agent intends to do that" and has been strongly criticized in [3]. According to Bratman an agent does intentionally also what he believes will be a consequence of his intended action, moreover he does intentionally all spontaneous actions that are done during the execution of the intended action. This position is slightly different from the position presented in [16]. According to Searle an agent can have the "Intention in Action to do a certain action" without the need of a previous "Prior Intention to do that action" (all intentional actions have intentions in action but not all intentional actions have prior intentions). This seems to apply to automatic and spontaneous *micro-actions* that where not planned by the agent (they were not object of Prior Intentions) but that are executed during the execution of the *macro-action* that was the object of the Prior Intention (Bratman would say that those spontaneous micro-actions are done intentionally but they are not object of Present-directed Intentions).

[9] We refer to Epistemic Monitoring also for EpAs on *belief about success or failure in intermediate results achievement* (see section 4.1).

can assume that although decomposable in and implying specific beliefs, goals and intentions this molecular predicate is used in mental operations and rules. One might assume that the left part of a given rule for the activation of a specific Epistemic Goal is just the combined pattern: belief + intention. One might assume that we "recognize" our own mental state (thanks to this complex predicate or some complex rule) and that this "awareness" is part of the mental state: since we have a complex category or pattern of "expectation". This would create some sort of "molecular" causal level: an Expectation concerning a proposition that determines the execution of the Epistemic Action aimed at verifying the truth value of that proposition.

Reference

1. Anzai, Y., & Simon, H. A. (1979). *The theory of learning by doing*. Psychological Review, 86, pp. 124-140.
2. Audi, R. (2001). The Architecture of Reason: the structure and substance of rationality. Oxford University Press.
3. Bratman, M. E. (1987). *Intentions, plans, and practical reason*, Cambridge, MA: Harvard University Press.
4. Butz, M. V., Goldberg, D. E., Stolzmann, W. (2002) The anticipatory classifier system and genetic generalization. *Natural Computing*, 1, pp. 427-467.
5. Castelfranchi, C. (1996). Reasons: Belief Support and Goal Dynamics. *Mathware & Soft Computing*, 3. pp. 233-47.
6. Castelfranchi, C., Lorini, E. (2003). Cognitive Anatomy and Functions of Expectations. *IJCAI '03 Workshop on Cognitive modeling of agents and multi-agent interaction,* Acapulco, Mexico.
7. Cohen, P. R. , Levesque, H. J. (1990). Intention is choice with commitment. *ArtificialIntelligence*, 42, pp. 213-261.
8. Davidson, D. (1980). *Essays on Actions and Events*. Oxford University Press.
9. Davidsson, P. (1993). A framework for organization and representation of concept knowledge in autonomous agents. In *Scandinavian Conference of Artificial Intelligence.*
10. Herzig, A., Lang, J., Polacsek, T. (2000). A modal logic for epistemic tests. In *Proceedings of European Conference on Artificial Intelligence (ECAI'2000)*, Berlin, August.
11. Miceli M., Castelfranchi C. (2002). The Mind and theFuture: The (Negative) Power of Expectations. *Theory and Psychology*, Vol. 12(3), pp. 335-366.
12. Miller, G., Galanter, E., Pribram, K. H. (1960). *Plans and the structure of the behavior.* Rinehart & Winston, New York.
13. Minsky, M. (1975). A framework for representing knowledge. In P.H. Winston (Ed.), *The Psychology of Computer Vision*, pp. 211-277. McGraw-Hill.
14. Pashler, H. E. (1999). *The Psychology of Attention*. Cambridge, MA: MIT Press.
15. Rao, A. S., Georgeff, M. P. (1992). An abstract architecture for rational agents. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91),* Morgan Kaufmann Publishers, Sidney, Australia.
16. Searle, J. (1983). *Intentionality*. Cambridge University Press.
17 Van Linder, B., van der Hoek, W., Meyer, J.-J. Ch. (1998). Formalising abilities and opportunities. *Fundamenta Informaticae*, 34, pag. 53-101.
18. B., Van Linder, W., van der Hoek, and J.-J.Ch., Meyer (1994). Tests as Epistemic Updates. In A.G. Cohn (Eds.) *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI'94)*, pp. 331-335, Wiley, Chicester.

Internal simulation of perception in minimal neuro-robotic models

Tom Ziemke

University of Skövde, School of Humanities and Informatics,
PO Box 408, 54128 Skövde, Sweden
tom@ida.his.se

Abstract. An increasing number of cognitive scientists are getting interested in simulation or emulation theories of cognitive function in general, and representation in particular. These replace the amodal, typically symbolic representations of traditional theories with agent-internal simulations of perception and action that use at least partly the same neural substrate as real, overt sensorimotor processes [1, 2, 5]. Our own experimental work on minimal neuro-robotic models, mostly based on Hesslow's theory [2], has initially focused on providing simple robotic agents with the capacity to act blindly after some exploration of the environment, guided by their own internal simulation of perception, based on the repeated use of forward models predicting the sensory consequences of actions [3, 7]. Current work also addresses simulations at higher levels of abstraction, using hierarchical architectures: on the one hand, a combination of unsupervised categorization and prediction (cf. [4]) and, on the other hand, a two-level mixture-of-experts architecture (cf. [6]).

References

1. Grush, R. (in press). The emulation theory of representation: Motor control, imagery, and perception. *Behavioral and Brain Sciences*, to appear.
2. Hesslow, G. (2002). Conscious thought as simulation of behaviour and perception. *Trends in Cognitive Sciences, 6*, 242-24.
3. Jirenhed, D.-A.; Hesslow, G. & Ziemke, T. (2001). Exploring Internal Simulation of Perception in Mobile Robots. In: Arras et al. (eds.) *2001 Fourth European Workshop on Advanced Mobile Robotics - Proceedings* (pp. 107-113). Lund University Cognitive Studies, vol. 86.
4. Nolfi S. & Tani J. (1999). Extracting regularities in space and time through a cascade of prediction networks. *Connection Science*, 11(2), 129-152.
5. Svensson, H. & Ziemke, T. (in press). Making sense of embodiment: Simulation theories and the sharing of neural circuitry between sensorimotor and cognitive processes In: *Proc. of the 26th Annual Meeting of the Cognitive Sci. Society*. Mahwah, NJ: Lawrence Erlbaum, to appear.
6. Tani J. & Nolfi S., (1999). Learning to perceive the world as articulated: An approach for hierarchical learning in sensory-motor systems. *Neural Networks*, 12, 1131-1141.
7. Ziemke, T., Jirenhed, D.-A. & Hesslow, G. (in press). Internal simulation of perception: A minimal neuro-robotic model. *Neurocomputing*, to appear.