Drawing Attention to the Dangerous

Stathis Kasderidis[1], John G. Taylor[1], Nicolas Tsapatsoulis[2] and Dario Malchiodi[3]

[1] Department of Mathematics, King's College London, Strand, WC2R2LS, UK
stathis@mth.kcl.ac.uk - john.g.taylor@kcl.ac.uk
[2] School of Electrical and Computer Engineering,
National Technical University of Athens,
9, Iroon Polytechniou Str., 15773 Athens, Greece
ntsap@image.ntua.gr
[3] Department of Computer Science, University of Milan,
Via Comelico 39/41, 20135 Milan, Italy
malchiodi@dsi.unimi.it

Abstract. In this paper we present an architecture of attention-based control for artificial agents. The agent is responsible for monitoring adaptively the user in order to detect context switches in his state. Assuming a successful detection appropriate action will be taken. Simulation results based on a simple scenario show that Attention is an appropriate mechanism for implementing context switch detector systems.

1 Introduction

Attention is a very important attribute possessed by many animals. It becomes increasingly under voluntary control and less reflexive as the evolutionary tree is ascended. In this paper we consider the application of this facility to artificial agents. We consider how attention can be introduced into such agents, specifically those involved in the guidance of humans in tasks involving 'wearables'. Seen as a natural part of the *'Disappearing Computer'* project [1], it is necessary to solve the task of fusing what might be a considerable amount of data about the human wearer, leading to possibly crucial decisions as to his/her welfare.
J.G.Taylor has elsewhere [2],[3],[4] introduced attention in engineering control terms; this uses both inverse and forward models [5] in order to optimise information processing used in decision-making. We implement a control architecture, which includes the simplest components: state maps and attention control generator. A simplified scenario is used in simulations to show the case of User Monitoring by the response of the attention system, in particular by adaptively changing the sensor resolution, as well as taking into account the user profile.

2 The User Monitoring Problem

In many circumstances intelligent artefacts would have to monitor their users and decide on behalf of them for their welfare. To facilitate intelligent decision-making the artefacts should be able to adapt their monitoring strategy according

to the context of the user. Due to (mainly) power and communication limitations sensors could not be continually polled in their highest sampling rate. This leads to the problem of limited information for effective decision-making. The controlling artefacts then have to adaptively monitor their users by increasing or decreasing the sampling rates of the various sensors involved. This is the User Monitoring Problem. It is an optimization problem in the sense that one has to balance energy and communication overhead against higher resolution information about the user and his environment. A possible strategy for achieving such a balance is attention control.

There are two principle ways in which a state transition could take place. It is either a slow, gradual process (*'adiabatic'*) or a *'sudden'* jump to the new state. According to this we propose a solution to the User Monitoring problem, which consists of two elements. On one hand one uses attention control to capture the fast changes while classifier systems could capture the departure from one context to the other in the slow timescale.

We present now a health monitoring scenario that would make more concrete the User Monitoring Problem. Here we have a user with heart problems, that needs to regularly monitor his heart condition. There is a Heart Rate (HR) sensor attached to the user while a second artefact (his PDA) polls the sensor, controls its sampling rate and informs the user. For simplicity we assume that all the available information is the Heart Rate signal (instead of the ECG). The device acts as the user interface and is able to call the health monitoring service provider in case of an emergency. The software agent resides in the PDA device. A number of events can take place, which induce a change in the state of the user. We use the variable of *Alert Level* to distinguish the user states. The resting state has the *Normal* classification. Additionally two more states are considered. An *Attention Seeking* and a *Dangerous* one. The goal of the agent is to detect successfully the states and take appropriate actions. If an event of interest takes place (either a Heart Attack, or another major stress) we assume that the following cycle of events takes place:

(1) *Rest state* 1 - Initial Steady State, (2) $Pre-cursor\ state$ - indicating the onset of a Heart Attack, (3) $Onset\ of\ Heart\ Attack$ - say with duration of 1 min, (4) *Rest state* 2 - Final Steady State.

The first Rest State is the current equilibrium state where the system uses a default *monitoring level* for the user. We use as default rate 1 sample/min. The second state is a pre-cursor signal which in many phenomena is present and for this reason useful to be exploited for expectation of system change. We make the assumption here that indeed this signal exists in order to simplify the presentation. The existence of a pre-cursor could be exploited by using appropriate rules for the sampling rate setting of the sensor. In a typical setting the sampling rate would be increased on expectation of the main event. In our case the sampling rate is increased to 10 samples/min. The onset of a Heart Attack is manifested by the increase in the mean level of the Heart Rate. At the end of the event the rate returns to another resting state eventually (Rest State 2). During the main event the sampling rate increases to 1 sample/second.

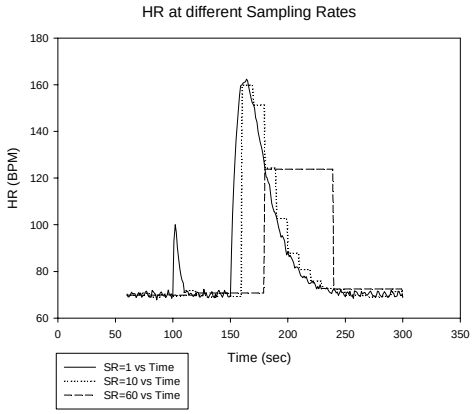

Fig. 1. Heart Rate signal in three different sampling rates: $\frac{1}{60}$ Hz, $\frac{1}{10}$ Hz and 1 Hz.

A typical time course of the Heart Rate series is shown in Figure 1. There we plot the series with the three sampling rates present in our scenario. It is clear that only by using the highest sampling rate (SR=1) one can accurately follow the sequence of events. In the figure (and for the rest of the paper) we use the values of 60 (1 sample/60 secs), 10 (1 sample/10 secs) and 1 (1 sample/sec) to indicate the lowest, medium and highest rate.

3 Architecture

The architecture we use is based on that extracted from the human brain, and is shown in Figure 2. Attention is, in the human brain, involved in two forms of control: sensory and motor response [6]. We have made that division explicit, since there is both the need for sensory feedback, to lower level, i.e. to input sensors so as to change sampling rate, as well as in control of motor response for guidance to higher-level systems.
Let us explain now in detail the process that takes place in Figure 2. The user generates a Heart Rate time series, which is sampled at an appropriate rate by the sensor. The output from the sensor is the current instantaneous value of the Heart Rate signal measured in BPM, $HR(t)$. Obviously if one increases the sampling rate he will observe more of the micro-structure of the series (see Figure 1). This raw sensor data is then forwarded to the user's PDA. This has a number of modules that take part in the processing. The first module collects the data and builds an appropriate state representation for the time series; it is called State Vector. The next module in the sequence is the State Classifier system. It uses as input the State Vector and outputs a classification for the Alert Level (State Classification Vector). The classifier system is implemented by some appropriate method [7], [8]. In this way known knowledge can be used for classification purposes.

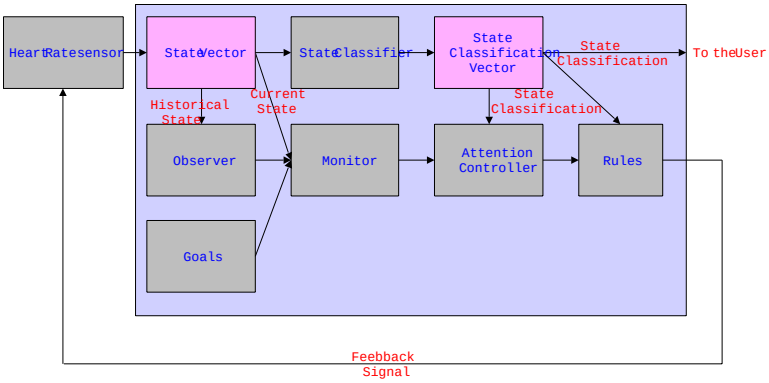

Fig. 2. Proposed Attention Architecture for an agent.

The State Vector we use has a 'Takens' embedding representation [9]. The vector is used in the Observer module as well. This module includes a model for predicting the current state, i.e. $HR(t)$, given a history up to some lag time T. In other words the Observer implements the following mapping:

$$HR(t) = f(HR(t-1), ..., HR(t-T)) \tag{1}$$

This is achieved by using, for example, an appropriate trained neural network. We also assume that the history of the series up to some lag time T is stored in a *Historical State* vector. However, for simplicity this is not present in Figure 2. At the next step the Observer prediction and the actual state (coming from the State Vector) are compared in the Monitor module. There, a measure of similarity (or closeness), E, is calculated and this is compared against a given threshold. The threshold is a user specific value and it is given by the *User Profile*. In our case the Monitor module calculates the $\|\cdot\|_1$ metric. Other possibilities exist. If the error measure calculated is higher than the threshold value this indicates the existence of an event of interest that needs attending to. The above concept is represented by an Attention Index (AI) given by:

$$AI = 1 - Pr(E > \delta) \tag{2}$$

where E is the Monitor error measure and δ is the aforementioned threshold. Pr is the probability that the error is larger than the threshold. We assume that a suitable error distribution has been selected. This definition implies that the more 'improbable' is an event the higher is the index that it creates. This is consistent with the fact that unexpected changes in the user state imply in turn a rather large error deviation between the prediction and the current state. As a result higher priority should be given to such events. The Attention Index created together with the State Classification Vector are then forwarded to the Attention Controller. The threshold value δ is provided as a piece of information coming from another module. This is the Goals module. Its main function is to

indicate to the system the type of goals we try to achieve as well as help in the creation and processing of new goals. The various properties of the User profile (such as thresholds) are generally fuzzy concepts. For simplicity we assume here that they are crisp numbers. We will not expand further at this point because in our simple scenario the Goals module is not involved any further.
The Attention Controller receives the Attention Index from the Monitor module for further processing. It implements conceptually two distinct functions:

(a) It has a *priority policy* for inducing an order in the space of Attention Events,
(b) It has a *dispatch policy* for deciding which jobs will be dispatched in the next processing stage.

The above structure is a general template for building appropriate Attention Controllers. An example of a priority policy is a competition mechanism. In this paper we adopt a very simple solution: the priority queue model. The ordering in this queue is already defined by the value of the Attention Index. The motivation of definition given in eq. 2 should be now clear: The highest priority event should make use of the system resources for its own processing. This principle is borrowed from the human attention system and it provides effective guidance for control purposes, as we will see in the next section.
Finally, the implementation of actions that are necessary to achieve the desired system behavior, after an Attention Event is dispatched, is left in the Rules module. There, a set of rules, guides the system actions. In our case it is the change of the monitoring level according to the given recognized state. The appropriate rules are as follows (CAL=Current Alert Level, PAL=Previous Alert Level):

If PAL=*Normal* and CAL =*Attention-Seeking* increase rate.
If PAL=*Attention-Seeking* and CAL =*Dangerous* increase rate.
If PAL=*Attention-Seeking* and CAL =*Normal* decrease rate.
If PAL=*Dangerous* and CAL =*Attention-Seeking* decrease rate.

The above rules enable the Rules module to set the appropriate sampling rate in the sensor by means of a feedback signal.
The above description does not include a crucial element of the Architecture, that of learning. There are a number of circumstances where the system will fail. In these cases a reinforcement error learning process takes place using the Error measure calculated in the Monitor for this purpose. Moreover, there is the case of the external user feedback to the system for adaptation purposes. This second process is one of the mechanisms that are provided in the Architecture for the modification of the User Profile set. This set provides actually the various specific thresholds and the model parameters that implement the various models present in the architecture. The first adaptation mechanism present is the off-line training of the models in a supervised way. When errors occur the system stores the data for which the performance was not satisfactory. Supervised training is conducted then for a new determination of the User Profile.

4 Simulation and Results

Using the above architecture we implement the scenario of Section 2 in a simulation. The following assumptions are made:

1. A User Profile exists that guides the system for Heart Attack detection, i.e. the thresholds for the specific user are given. We distinguish three cases: (i) *Normal* [60-80], (ii) *AttentionSeeking* [80-100], (iii) *Dangerous* [>100].
2. One time series alone is used; that of the Heart Rate. No data fusion is necessary due to other sources.
3. A User model (State Classifier) has been implemented using an appropriate technique [7],[8] that distinguishes the classes of events described above.
4. We use a synthetic Heart Rate signal.
5. The Heart Attack pattern is described by an appropriate functional form, e.g. $f(x) = a_1 x e^{-a_2 x}$. The parameters are sampled from Normal distributions for the various events. We assume knowledge of the mean and variances of these distributions for training a neural network prediction model for the Observer.
6. A number of threshold values are used. Results in Figure 3 are generated by using the value of δ.

We train a back-propagation 3-5-1 fully connected model to learn the 'profiles' of a heart attack. The two-parameter family of the pulses mentioned above represents the Heart Attacks profiles. A standard pre-cursor signal is present having a similar form with the main event signal but with less amplitude. The simulation is run for a period of 300 seconds. Two events take place. One pre-cursor event at 100 secs and the main event (Heart Attack) at 150 secs. Typical results are presented in Figure 3. We use only the simplistic rules, mentioned in the previous section, for controlling the sampling rate. Obviously more complete strategies can be used.
The first diagram shows the Heart Rate signal and the prediction of the Observer's model. The solid line represents the HR signal sampled with a variable sampling rate according to the rules applied at the given moment. This is what the user would see. The dotted line represents the prediction of the Observer model. Compare the perceived signal from the part of the user at variable rate with the same signal at the highest sampling rate at Figure 1. The second diagram shows the sensor sampling rate with time. The third diagram indicates the system activity. This has two components. The solid line represents the state classification. The dotted line is the activity of the Attention Controller. The Alert Level is shown as 0 for Normal, 1 for Attention Seeking and 2 for Dangerous. The creation of an attention event is indicated by the value of 3.

The simulation starts with the highest sampling rate of 1 Hz (1 sample/sec). Due to no other activity, no rule is fired so this sampling rate remains unchanged. However, the default monitoring policy is at 1 sample/min. The former rate was chosen so as to show in the first part, 0-100 secs, the close agreement between the prediction of the Observer with the HR series. At 100 sec the pre-cursor signal arrives. A few seconds later the deviation of the Observer prediction and the actual HR series is larger than the threshold. An attention event is created.

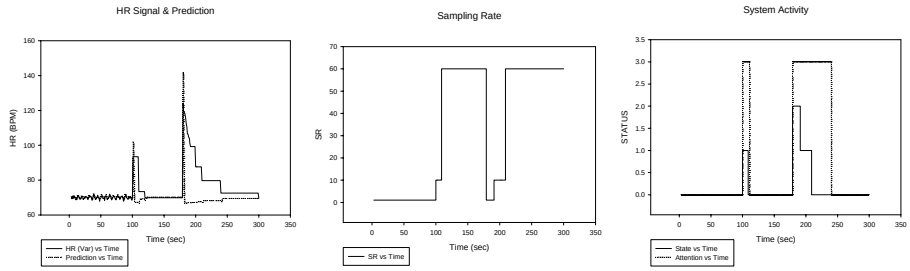

Fig. 3. Simulation results: (a) HR signal, (b) Sampling Rate and (c) System Activity.

This in turn fires the rule that sets the new sampling rate. The new rate is 0.1 Hz. At the same time the HR signal rises from the Normal Alert Level to the Attention Seeking one. This is shown in the System Activity diagram. Eventually the pre-cursor signal levels off and the Observer prediction follows again the HR series time course. At this point the firing of attention events stops. At the same time the rule for setting the sampling rate to the default level, $\frac{1}{60}$ Hz, is activated. This is represented in the Sampling Rate diagram as the value 60.
A period of inactivity exists until the main event arrives at 150 secs. Again the Observer's prediction differs from the HR series, so an attention event is fired. This changes both the sampling rate and the Alert Level classification. The sampling rate goes to 1Hz and the Alert Level takes the value of Dangerous. As the event progresses in time, the Alert Level drops to Attention-Seeking. As a consequence the sampling rate drops to 0.1Hz (10 in Sampling Rate diagram). Eventually the HR signal levels off and the sampling rate returns to the default level of $\frac{1}{60}$ Hz, while no other attention activity is generated.
Some comments are now in due. The first one is related to activation of the rules. The rules are used after an attention event is created. However, there are two ways in which such an event can come to existence. The first one is the deviation of the Observer's prediction from the actual signal. This is intended to capture sudden changes in the signal and indicate fast transitions from one equilibrium state to another. The second way in which attention is caught is by comparing the previous time step's (or steps') State Classification with the current one. An alteration indicates a (typically) slow transition process. These two ways implement the necessary schemes that detect user context switches. See also discussion in Section 2. The second point concerns the prediction ability of the Observer model. One conceptually estimates this model in a stationary regime and then uses it to discover deviations from the actual state. There are two possibilities for such deviations to occur: (a) the model is correct in its prediction and indeed the actual state is different from what was expected, or (b) the model prediction is simply wrong and nothing out of the ordinary has happened. The first is distinguished from the second by using the classification history trace of the State Classifier. If the history indicates that we do have a change in the state we probably deal with the former case. Otherwise the Observer has failed and needs

further estimation. This incremental improvement can be implemented as either a supervised learning scheme (off-line) or as a reinforcement scheme (on-line) using the Monitor Error level.

5 Conclusion and Discussion

The attention-based control scheme described in Section 3 is part of the ORESTEIA Architecture for attention-based agents [10]. There, an agent is composed by a number of artefacts partitioned in four levels. At the first level the sensors and actuators exist. In the second one pre-processing facilities are provided. The third level involves local decision makers and the fourth one has global decision makers. Attention control is used in both the third and fourth level
The data fusion problem due to a number of sensors is, currently, under development. The problem here lies in the creation of a proper state representation that captures effectively the environment and user states. In some cases, sensors of different modalities contribute correlated but distinct information about the occurrence of an external event. All the sensor measurements are different aspects of the same stimulus applied either on the environment or directly on the user. Building such a representation is a context capturing problem.
Additionally, learning needs to be present so as the system would be able to adapt better to its environment. Adaptation is achieved by improving the classification power of the State Classifiers and the prediction ability of the Observer model. The latter can be used in order to predict further into the future so as to enable the system to have a more proactive behavior than it currently has.

References

1. Disappearing Computer Project. http://www.disappearingcomputer.org
2. Taylor J. G.: Attentional movement: the control basis for Consciousness. Soc. Neuroscience Abstracts **26**, item 893.3, (2000), 2231.
3. Taylor J. G.: Attention as a neural control system. Proc. Int. Joint Conf. Neural Networks (IJCNN'01), (2001), 262-276.
4. Taylor J. G.: Paying attention to Consciousness. Trends in Cognitive Sciences **6**, (2002), 206-210.
5. Miall R. and Walpert D. M.: Forward models for physiological motor control. Neural Networks **9**, (1996), 1265-1279.
6. Rushworth M. F. S. et al.: The left parietal cortex and motor attention. Neuropsychologia **33**, (1997), 1261-1273.
7. Tsapatsoulis N., Wallace M., and Kasderidis S.: Improving the Performance of Resource Allocation Networks through Hierarchical Clustering of High Dimensional Data. Int. Conf. of Art. Neural Networks (ICANN'03).
8. Apolloni B. et al.: Learning rule representations from boolean data. Int. Conf. of Art. Neural Networks (ICANN'03).
9. Abarbanel H.: Analysis of Observed Chaotic Data. Springer Verlag (1996).
10. FET DC ORESTEIA Project. http://www.image.ntua.gr/oresteia.