Attention-based Learning

S. Kasderidis & J. G. Taylor
Department of Mathematics, Kings College London, Strand, WC2R 2LS, UK
stahis@mth.kcl.ac.uk, john.g.taylor@kcl.ac.uk
http://www.kcl.ac.uk/

***Abstract*-We apply an attention-based framework in the creation of an autonomous robot control system. We use a specific task, that of route planning for a robot in a dynamic environment, to present the general control architecture, and finally we show how it can be applied to the problem. Initial results of a simple simulation are presented with focus on the learning aspects.**

I. Introduction

Attention is arguably the highest-level control system in the human brain. It allows a system possessing it to be able to reduce the problems presented by distracters in a noisy environment, and to prioritise tasks to be solved so that less important ones, elicited by distracting cues, can be avoided. An important component of such attention control is the creation of suitable goal structures, regarded as representations in the human frontal lobes. Much research has been recently contacted in attention [1, 2].

A human attention-processing model was based on an engineering control framework ideas [3, 4, 5]. We have also extended the notion of attention processing from a model of the human brain to the more general context of attentive agents [6, 7, 8]. These have both a 'personalised' internal attention control system, as well as an overall attention control system able to handle agent-to-agent interactions. Attention is used in the latter framework in a multitude of ways. It serves as the learning generator, as a local control system in a modality level and as a global competition system among conflicting goals.

In this paper we apply the notion of attention control to a robot in a complex environment. We present the various roles of attention and explain the main ideas. Our focus is on the learning aspects. In section 2 we present a specific task environment we think appropriate to test the architecture. In section 3 the attention architecture is presented. Results are presented in section 4, and a discussion is given in section 5.

II. General problems and a specific task environment

There are a number of problems that must be solved before robust autonomous agents can be achieved. Most of the difficulties arise from the fact that the agents must operate in an open environment. To handle such a case one needs to equip the agent with a number of faculties. On the one hand the agent must operate in an effortless manner when no unexpected conditions arise in its environment, while on the other hand it should be able to learn dynamically in cases where a decreasing performance is realised. The first mode uses the idea of creating and automating appropriate schemata for the solution of the corresponding problems. The second mode is based in the assumption that in a dynamic open environment perfect performance could never be achieved. Erroneous performance will be realised due to novelties, other structural changes in the current environment and the goal oriented behaviour of other agents (limited information). The solution to this problem is to enable the agent to learn dynamically corrections to its poor actions and eventually automate the corrected behaviour. Attention is the ultimate generator of knowledge as attention itself controls the switching from the automated mode to learning modes.

To make these ideas more concrete we consider a specific problem. This is the case of robot route planning. In simple words, we try to find an appropriate path for the robot's movement in an environment where a number of stationary and moving objects exist. Stationary objects typically include household items such as furniture and moving objects include other robots, humans sharing the environment and other less 'intelligent' but mobile equipment.

There are two main factors that generate errors and thus threats for the robot in such an environment. On the one hand novelties could appear and thus suddenly change the conditions of the environment. On the other hand, the assumption of possessing accurate motion models for the other agents does not hold. This is due to the fact that the other agents may also exhibit goal-oriented behaviour and thus introduces difficulty in acquiring their goals and 'motivations' through external observation alone. A degree of uncertainty will always remain. The situation is not uncommon to that of humans walking in a busy street.

Let us now describe a simple scenario that would implement the task environment. Assume that a worker robot is inside a factory floor and its job is to move some item of interest from a point A to a point B. Assume further that these two points are at two distant sides of the floor and that the floor is of rectangular shape and it is divided in a X-Y grid (GridWorld). The robot moves in this grid in a forward direction. When the robot wants to change direction it performs a turn around its z-axis by 45 degrees. The turn could be either left or right (to the local

system of coordinates). The robot carries a number of sensors. They are: a power sensor for detecting the power level, an overall proximity sensor (with a 360 degrees view of the environment), and one position/speed sensor for knowing its own coordinates and speed. In addition an actuator exists which controls the speed and the direction of the robot. The GridWorld is shown in figure 1.

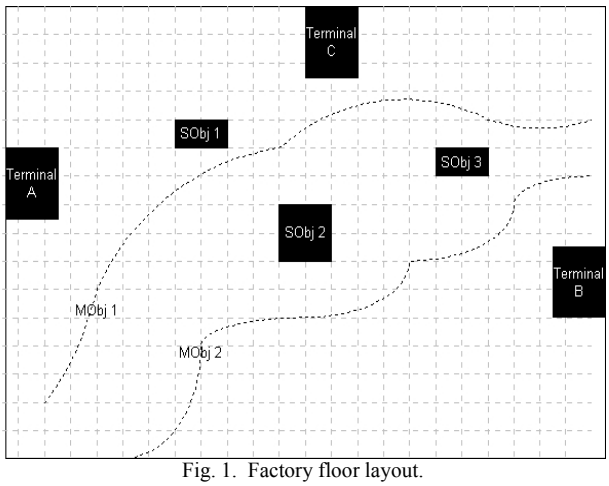

Fig. 1. Factory floor layout.

Terminal C is the recharging station for the robot. Sobj x is the name of a stationary object. Mobj x is the name of a moving object. The figure shows a possible configuration of the GridWorld with three stationary objects and two moving ones. Realisations for the trajectories of the moving objects are also shown with dashed lines.

The problem that we face is how to move from point A to point B and possibly C avoiding collisions both with static and moving objects.

III. Architecture

The architecture we use is based on that extracted from the human brain, in terms of a goal system with an attention control mechanism (the details appear in [7, 8]). An agent possesses a number of goals represented as a tree structure (with higher importance goals positioned nearer the root, less important ones further apart). Each goal is associated with a service procedure (a *schema* that satisfies the goal, given as a template with adaptive elements). The general schema template with *local* attention control is shown in figure 2 (more detail in [8]).

The architecture has four processing levels: level one has sensors and actuators, level two has pre-processing facilities, level three has schemata (services) for atomic goals, and level four has schemata for *composite* goals. Figure 2 represents the service of an atomic goal, which tries to achieve a target state. A composite goal has a different structure than the one shown in figure 2, due to the fact that its service function manages and produces an overall result from a number of children goals. We will not extend our discussion at this level further, but focus on level 3 atomic goals.

In figure 2 seven main modules appear: the *State*, *Observer*, *Goals*, *Monitor*, *Attention Controller*, *Rules* and *Forward Models modules*. The State module collects sensor signals and constructs representations, using three distinct representations, of which only one is used here, the *Native* one, which is the Cartesian product of the incoming signals. The Observer module is a state predictor for estimating the next sensory state and building an "expected state", to be compared, in the Monitor module, with the currently observed state (coming from the State module) to determine if major deviations exist. If a deviation threshold is crossed, an attention event is created and sent to the Attention Controller; it carries an *Attention Index*, a monotonically increasing function of the observed deviation in the Monitor module, bounded in the interval [0, 1]. The Attention Controller module is conceptually a stack/queue that determines the priority with which the dispatched attention events will be handled. In the simplest case the identity function can be used to determine the priority. The Rules module is the action generator component of the schema, using typically a symbolic model, i.e. rules, to produce appropriate responses, with input the current state classification and any attention event present. The Forward Models module produces predictions of the next state of the agent, given the realization of the action proposed by the Rules module. The Goals module serves as a database where a number of adaptive models necessary in the scope of the schema are necessary. It also includes the required target state that must be achieved.

We assume that the robot carries three sensors, and an actuator. In figure 2, the robot uses a number of sensors: the power sensor, the proximity sensor and the position/speed sensor. For simplicity we assume that the proximity sensor provides information for both the current position and speed of the other objects. In addition a single actuator exists which controls the speed and the direction of the robot. Inside the robot only one user defined goal is executed: the *Move Item* goal. The purpose of this goal is to have the robot move an item from a point A to point B. It has two children goals: the *Transport* and *Collision Avoidance* (CA) goals. The transport goal is decomposed in four sub-goals: *Goto A*, *Get Item*, *Goto B*, *Leave Item*. The interesting goal to discuss here is the Goto x. It is further decomposed into the *Route Planning* (RP) and *Move* goals.

The Move goal accepts a newly calculated position from the Route Planner and acts on the actuator. The State module collects information about the current position and speeds of the other agents and itself. It evaluates the current trajectories of the other moving objects through the Observer module. The module uses a motion model for each of the other moving objects,

realised as a suitable neural network. If the observed positions/speeds of other agents differ enough from the expected ones,

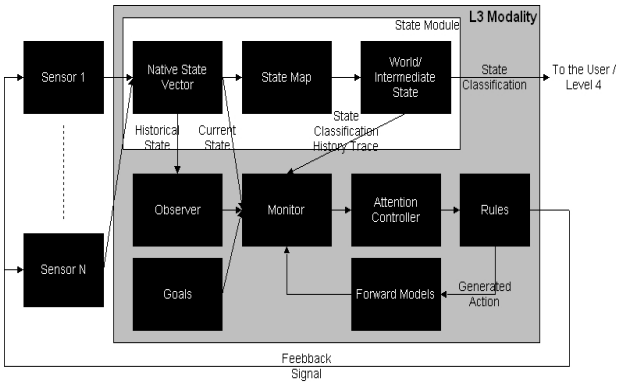

Fig. 2. Level 3 schema architecture.

then an attention event is created. The attention index is calculated as:

$$\text{S-AI-RP}=\text{Step}(||\mathbf{X_{obs}}-\mathbf{X_{exp}}||-T)*(2/[1+\exp(-(||\mathbf{X_{obs}}-\mathbf{X_{exp}}||-T)]-1) \quad (1)$$

(where vectors X_{obs} and X_{exp} represent the observed and expected position of a given moving object respectively). T is a suitable threshold, which measures the proximity of the two locations; Step(•) is the step function; it outputs 1 if the argument is positive, else it is zero. the vector X uses the x and y coordinates of a GridWorld cell. Given the information about the current positions of the various agents, the Rules module suggests a new position closer to the target position, which is then checked against the prediction of the Forward Models so as to make sure that in the next step the other objects and the robot will not collide or that it is not is otherwise occupied. If no problem is detected then the suggested new position is accepted and it is forwarded for execution to the Move goal.

There are two sources of problems. On the one hand, mechanical problems may not allow the proper actuator commands to be performed or alternatively the executed commands may not result in bringing the agent in the desired state that the planner system intended. In both cases we have failures of the Move goal. This goal uses another type of attention, which is called *motor* attention (in contrast with the *sensory* attention described above in (1)). It also observes and compares the expected and the occurred (internal) states of the agent: it makes sure that the agent is brought to the required state so as to conform to the overall environmental state (which includes the other agents) according to the plan of the RP.

During the execution of the planned motor action (i.e. moving to the next cell) we may find that the other agents did not really move according to prediction of the motion model. The Collision Avoidance goal will capture this deviation and create an appropriate attention event, which will result in taking corrective action: simply moving away in the opposite direction to an approaching object. We assume the existence of a safety zone around the agent, with radius R. If the boundary of the safety zone is crossed then an attention event is created with the following attention index:

$$\text{S-AI-CA}=\exp(-||\mathbf{X_{obs}}-\mathbf{X_{agent}}||), \text{ if } ||\mathbf{X_{obs}}-\mathbf{X_{agent}}|| < \mathbf{R}, \quad (2)$$

$$\text{S-AI-CA}=0, \qquad \text{if } ||\mathbf{X_{obs}}-\mathbf{X_{agent}}|| \geq \mathbf{R}$$

The idea behind (2) is that the event is more severe as we enter further into the safety zone. X_{obs} is the observed position of another agent, whereas X_{agent} is the agent's own position. The action that is created as a response to an attention event is twofold: First, it leads to the update of the current motion model for the corresponding other agent. Second, it takes averting action so as to avoid a collision. The structure of the Collision Avoidance goal follows the template of figure 2. The Observer is used for 'following' in runtime the path of the other objects.

It should be made clear at this point that the CA-Observer, RP-Observer and RP-Forward Models modules all use the same underlying motion model for a given observed (other) agent. This is a neural network model that has as inputs the current position, current speed and previous positions (up to some lag time K) of the other agent. The model is updated dynamically as a result of Collision Avoidance's attention control. This is the role of attention as learning initiator. At this point the second role of attention is highlighted: that of a global competition for conflict resolution. We have explained previously that if a collision is detected, corrective action is taken. This is achieved because the CA goal effectively suppresses the sub-tree starting at the Transport goal, which belongs to the same level as itself. The mechanism with which this result is achieved is based on Action Indices (a generalisation of the Attention Index concept). The details are given in [8]. We just provide the relevant formula here:

$$\text{Action Index}=(W+\text{S-AI}+\text{M-AI}+\sum_j \delta(j, \text{contribut.})) / (3+\#\text{ Contributing Children}) \quad (3)$$

Where S-AI and M-AI are the sensory and motor attention indices and W is the goal's intrinsic weight (available through learning of a value system). A child is considering as *contributing* if any kind of attention event is created in its scope (at the given time) or if it posses itself a contributing child. Formula (3) introduces a mechanism, which can be used for comparing 'importance' of execution among conflicting goals. With

appropriate weighting the Collision Avoidance goal can be made to take precedence over the Transport goal. Thus suppression of the Transport sub-tree is achieved and its execution is stopped for the given processing step. The only action that will be generated is this originating from the CA goal. In normal circumstances the Collision Avoidance is suppressed from the Transport goal. Local attention control in figure 2 is achieved by changing sampling rates or otherwise of appropriate sensors. Control is needed in this level in order to minimise the power requirements of enhanced resolution information (in time and space) against the need for higher service times of the robot.

Finally we must observe that the attention indices, (1) and (2), can be used as modulators of the learning rates present in the motion models. In particular a suitable formula is given by (4):

$$r_{t+1}=r_t*AI+\varepsilon \quad \textbf{(4)}$$

Where ε is a permanent small rate, say 10^{-4} , AI is a suitable attention index and r is the learning rate used in the neural network motion model. Learning is assumed to take place in the background continually, using in (4) as attention index formula (2).

IV. Simulation & Results

We use the GridWorld of figure 1 to perform a number of simulations. The size of the grid is 50x50. The stationary objects are located as in the figure. A number of moving objects were used. The motion model was implemented using a neural network approach to the target cell. The network used as inputs the positions from current time up to K steps in the past and the current speed. K was varied from 0 to 5. We assumed also infinite sensing range. The speed of the robot is either set to one or zero. The target of the simulations is to study the effectiveness of the proposed architecture in detecting a danger and learning effective motion models.

The simulation was setup as follows: We have used a varying number of moving objects for checking the agent's effectiveness on avoiding a collision. We used 5, 10, 15, 20 and 25 moving objects. All the other agents were given fixed but unknown trajectories to follow. Randomness in the trajectories was introduced in two ways. On the one hand, the trajectory was formed by selecting random start and end service points for a given moving object. This had the effect of covering the whole floor with intersecting trajectories. On the other hand, small random movements around the main calculated trajectory were introduced at random points. The trajectory was calculated using a linear approach to the target cell and taking corrective steps when approaching a stationary object. When another moving object was approaching, no special action was taken, so collisions were possible. In this way we had a population of simple moving agents, which collided with each other and with our own attention-based agent. The statistic of interest was the number of collisions taking place per step. This was calculated as follows: We assumed a fixed simulation duration of 350 steps; we used 10 different trajectory configurations so as to average the randomness in the experiment's initial configuration and we studied three cases. A baseline case (dummy agent), a predictive agent and an attention-based predictive agent. All of them were using the same basic route-planning algorithm. The difference was introduced in the way that the three types of agents were using the suggestions of the route planning system. The dummy agent simply accepted the suggestions. The predictive agent was using its motion model for forecasting future collisions. If a collision was predicted, then another call to the route planner was made so as to calculate a new position given the new information. The same approach was used by the attention-based agent. The predictive and attentive agents were also updating their motion models (in the next step) if an unsuccessful prediction was made. For these experiments we used a back-propagation method. The resulting network had as inputs the current x and y coordinates of the GridWorld cell, the current heading vector, expressed as a value out of a fixed set of directions and possibly x and y coordinates of previous positions. We used up to K=5 lag vectors of x/y coordinates in the motion models. Typically, the higher the number of the lag steps the better the prediction accuracy of the model. However, all models suffered, naturally, when discontinuities in the path were realised due to a sudden turn due to random variations or due to route planner's calculated trajectory at that point.

The predictive type of agent had a fixed learning rate of r=0.05, whereas the attentive agent used formula (4) for learning rate modulation.

In figure 3 we show a number of series. The labels of *Dummy*, *Smart* and *Smart-Att* correspond to the dummy, predictive and attentive agents respectively. The first sub-graph describes the severity of the collisions while the second one describes the number of collisions suffered. Both figures describe average numbers per step of the trajectory. The results were also averaged over 10 different floor configurations per case of agent, i.e. dummy, predictive and attentive. The calculation of collision was made as follows: We assumed the existence of a safety zone around the agent of interest. The radius of the zone was variable. In figure 3 we used a radius of 6 cells. If during its movement from one cell to another the agent experienced the crossing of its safety zone from another agent, this was counted as a collision, i.e. the route planning system failed to provide a safe enough movement direction. Clearly, the entrance in the safety zone introduces a varying degree of danger, with the highest being an impact by moving on the same grid cell. This could be thought as a hazard function (severity in

the figure). We used a scheme where we assigned 10 'danger' units per cell crossed in the safety zone. For example, assume a radius of 6 cells in every direction and another moving object having entered two cells inside the boundary, i.e. four cells away from our agent, then a value of 20 units was assigned.

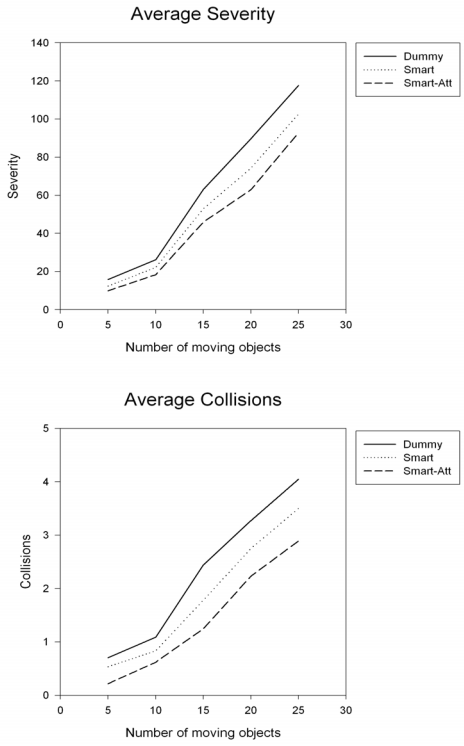

Fig. 3. Simulation results.

The results that are presented in figure 3 show that the predictive agent performed better than the dummy one, and the attentive one better than the predictive one. This observation was confirmed for varying sizes of the safety zone, training time for online model fitting, learning rates (in the case of the predictive agent) and number of training epochs. We used a fixed number (per experiment) of training epochs, typically 150, for updating the motion models. This procedure is equivalent to early stopping so as to avoid over fitting problems. No cross validation or similar approaches were used, as this simulation was intended to study expected "online" performance where not all necessary computational resources are assumed to be available. We used a batch version of the standard gradient descent back-propagation algorithm for this "online" training.

V. Conclusions & Discussion

The simulation results seem to indicate the validity of our assumption that attentive control should speed up the learning of corrections in the motion models. This coupled with the fact that the attentive agent has also used a pure reflexive response when attention levels were high, thus discarding the normal route planning process, indicate that the attention-based framework is a natural way to integrate online learning with normal (i.e. automated) processing when no events are present.

Clearly one can argue that a more sophisticated learning algorithm for the motion model, possibly with modulation of the learning rate parameter, can be applied so as to further increase the performance of the predictive agent. The point that is made here is that the goal-oriented attention-control framework is a natural way to integrate processing and learning aspects in run-time. The attentive type of agent is already a superset of the predictive agent and thus any performance increase in the latter will also benefit the former. In our framework, we do not make any explicit assumptions regarding the nature of the underlying prediction (motion) models. They can be implemented by using many approaches. The Kalman filter approach is a useful tool in many applications, but it is not applicable in systems that do not assume a linear observation function or linear evolution of the underlying dynamical system. In our example, one can express the equations of motion in a Kalman filter formulation, but what remains unclear is the fact if this is a suitable framework in order to model the goals and "intentions" of the other agents, which eventually lead to the lack of information so as to establish a mean path. In other words, it is not immediately clear if the underlying generating mechanism of goals / sub-goal structures follows a linear generation scheme / evolution.

Inclusion of reinforcement type of learning is also possible. This would allow us to further increase the usefulness of the attentive agent by using multiple learning paradigms in an integrated way. These aspects and verification of the current results by using additional configurations and different experimental setups are the next steps to be followed. Variations of the modulating scheme of formula (4) will also be considered.

There is much further work to be done to extend the system beyond the level of simulation described here. We have already mentioned the manner in which training should be extended, as well as consideration needed of the Goals module. The further extensions, already under consideration in the ORESTEIA attention agent project [6, 7, 8], are to develop more general control with greater flexibility still. This could be by using a multi-modal system, so that sensor information in a given modality can be properly assessed at a low level (level 3 in our above discussion) but then combined with other modalities at level 4. This requires careful discussion and development of fused state representatives as well as the

higher-level goal, forward and observed models as well as developing overall rules for the system. Even greater flexibility can be inserted by enabling the system to create its own goals and response rules, most efficiently by reward learning (the rewards arising form the environment), as considered earlier. We hope to develop, and report on, these aspects elsewhere.

Closing this discussion we note the fact that it is possible to use this framework in order to decompose a task in a set of competing goal families; our global attention mode. Local attention control is present in the form of sensor and actuator control. What is not currently present is the implementation of a third mode of attention, which is related with the signal enhancement process, i.e. amplifying suitable features and discarding distracting ones. This can be considered essentially as a feature selection process and it will be integrated in the existing framework in a future stage.

VI. Acknowledgments

The work presented here was originated and developed during the EU IST ORESTEIA project (IST-2000-26091). We kindly acknowledge their support to our work.

References

[1] Miall R. and Walpert D. M. (1996). Forward models for physiological motor control. Neural Networks 9, 1265-1279.

[2] Rushworth M. F. S. et al (1997). The left parietal cortex and motor attention. Neuropsychologia 33, 1261-1273.

[3] Taylor J. G. (2000). Attentional movement: the control basis for Consciousness. Soc. Neuroscience Abstracts 26, 2231, #893.3

[4] Taylor J. G. (2001). Attention as a neural control system. Proc. Int. Joint Conf. Neural Networks, pp 262-276, IJCNN '01.

[5] Taylor J. G. (2003). Paying attention to Consciousness. Progress in Neurobiology 71:305-335

[6] Kasderidis S. & Taylor J.G. (2003). Drawing attention to the dangerous. ICANN 2003 Conference, Istanbul, 27-29 June 2003, Springer.

[7] Kasderidis S. & Taylor J.G. (2003). Attention-based Robot Control. KES 2003 Conference, Oxford, 10-12 September 2003, Springer.

[8] Kasderidis S. & Taylor J.G. (2003). Attentional Agents and Robot Control. KES Journal, IOS Press, 2003. (submitted).