An Adaptive Human-Aware Software Agent Supporting Attention-Demanding Tasks

Tibor Bosse[1], Zulfiqar A. Memon[1,2],
Jan Treur[1], and Muhammad Umair[1,3]

VU University Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV Amsterdam
Email: {tbosse, zamemon, treur, mumair }@few.vu.nl
URL: http://www.few.vu.nl/~{tbosse, zamemon, treur, mumair}

[2]Sukkur Institute of Business Administration (Sukkur IBA)
Air Port Road Sukkur, Sindh, Pakistan

[3] COMSATS Institute of Information Technology, Department of Computer Science
Lahore, Pakistan

Abstract. This paper presents a human-aware software agent to support a human performing a task that demands substantial amounts of attention. The agent obtains human awareness in an adaptive manner by use of a dynamical model of human attention which is parameterised for specific characteristics of the human. The agent uses a built-in adaptation model to adapt on the fly the values of these parameters to the personal characteristics of the human. The software agent has been implemented in a component-based manner within the Adobe® Flex® environment.

Keywords: software agent, human-aware, adaptive, attention

1 Introduction

Software agents can provide more dedicated support when they have a certain level of human-awareness. This may require awareness not only of personal characteristics such as preferences, but also of (dynamically changing) states of the human. Examples of such states are emotion and stress, fatigue and exhaustion, goals and intentions, and attention states. Acquiring awareness of such states is a nontrivial challenge. Sensors may be used by the agent to get information about the human's current state, such as face readers, eye-trackers or heart rate sensors, but sensor information can rarely be directly related in a one-to-one manner to the human's states that are of concern.

A more general situation is that such sensor information can be used in a more indirect manner in dynamical models that express temporal relationships between a number of variables including these human's states and the sensor information; e.g., [17]. As humans may show substantial individual differences in characteristics in cognitive functioning, such a dynamical model usually includes a number of parameters for a number of specific characteristics of the human. Therefore they only can be used in a dedicated manner when sufficiently accurate estimations can be made for the values of these parameters as representations for the characteristics of the human considered. For applications in software agents this implies that such an agent does not only need a dynamical model of the human's processes, but also an adaptation model describing how the parameter values of the former model can be adapted over time to the characteristics of the human.

This paper presents a software agent that makes use of such a parameterised dynamical model of the human, and in addition possesses an adaptation model that on the fly tunes the parameter values to the human characteristics. A case study was undertaken to test it for a human's attention states. The human-attention-awareness for this case study uses three ingredients: (1) sensor information on the human's gaze, (2) information on features of objects in the environment, and (3) a dynamical (differential equations) model of the dynamics of attention levels over time integrating the instantaneous information from (1) and (2), and using partial persistency of attention over time.

The agent's adaptation model was designed and implemented in a generic manner, but for the case study was applied to the dynamical model for attention, thereby addressing, among others, parameters for weights of different features of objects, the effect of distance between an object and gaze, and an attention persistency factor.

The software agent and the further ambient support software environment has been implemented within the Adobe® Flex® development environment according to a component-based design. For the gaze sensoring it integrates the Tobii® eye-tracker.

In this paper Section 2 presents the assessment of the human's attention state for the case study and the parameters involved in this model, depending on personal characteristics of the human. Section 3 presents the background ideas of the parameter adaptation approach developed, and in Section 4 the overall adaptation model is described. In Section 5 some more details of the developed software and some results for the case study are discussed. In Section 6 a verification of the model is presented. Section 7 is a discussion.

2 The Dynamical Model for the Human's Attentional State

In this section the dynamical model for attentional states related to a number of objects used is briefly described, it was adopted from [7]. As a first step for each object its potential for attracting attention is determined. The *attention-attracting potential* of an object O is expressed by a weighted sum $\Sigma_i\, w_i * V_i$ where V_i is the value of object feature i, and the *object feature weight factors* w_i (normalised so that their sum is 1) are parameters that may depend on the human. Examples of features are: brightness, colour, and size. The values for these variables are expressed by numbers in the interval [0, 1], with 0 no attraction potential and 1 highest attraction potential. To take gaze into account the potential of the object for attraction human's attention is divided by a function depending on the human's gaze location:

$$V(O) = (\Sigma_i\, w_i * V_i\,)\ /\,(1 + \alpha * d(O, G)^2\,)$$

Here $d(O, G)$ is the Euclidean distance between the location of the object O and the human's gaze G. The parameter α (represented by a real number ≥ 0) is a *gaze distance effect rate*; it affects how fast an object loses the human's attention as the gaze moves further away from the object. The higher α is, the lower the attention for objects distant from the human's focus.

It is assumed that a person can have a fixed total amount of attention A distributed over all available objects; for convenience the attention scale is set in such a way that $A = 1$. The *attention level* $AV(O)$ of an object O expresses the amount of the human's attention directed at the object O as a fraction of the human's total attention. If there are n objects in total and the attention value $V(O_j)$ of the object O_j is indicated by V_j ($1 \leq j \leq n$), then the attention level $AV(O_i)$ of the object O_i (in the context of the other objects) is determined in a normalised form as

$$AV(O_i)\ =\ V(O_i)\,/\,\Sigma_j\, V(O_j)$$

Note that due to the normalisation $AV(O_i)$ also depends on the other objects. It is assumed that attention for a certain object persists over shorter time periods. To model this the attention values for object O can be modelled as a *persistent attention value* $PAV(O)$ using the following difference equation:

$$PAV(O)(t+\Delta t)\ =\ PAV(O)(t)\ +\ \beta * (AV(O)(t)\ -\ PAV(O)(t))\Delta t$$

Here β is an *attention flexibility rate* with a positive value between 0 and 1, and time step Δt with $0 \leq \Delta t \leq 1$; a high value of β results in fast changes and a low value in a high persistence of the old value. Written in a more concise differential equation format the dynamical model is as follows:

$$dPAV(O)\,/\,dt\ =\ \beta * (AV(O)\ -\ PAV(O))$$

This represents a system of n differential equations for all of the n objects involved, which via $AV(O)$ integrates the gaze information and information about the object features over time.

For the case study also for objects that need attention, a model was used to estimate the level of urgency of objects. This is done based on a number of *urgency-indication factors* by a weighted sum

$$UV(O) = \Sigma_i\ \ r_i * UV_i$$

Here UV_i is the value of the i-th urgency aspect and r_i the *urgency aspect weight factor* for this aspect (with total sum 1). The values for these variables are expressed by numbers in the interval [$0,\ 1$], with 0 no urgency and 1 highest urgency. As for attention values, also for these urgency values a certain extent of persistence is applied, thus obtaining $PUV(O)$:

$$PUV(O)(t+\Delta t)\ =\ PUV(O)(t)\ +\ \beta' * (UV(O)(t)\ -\ PUV(O)(t))\Delta t$$

or

$$dPUV(O)\,/\,dt\ =\ \beta' * (UV(O)\ -\ PUV(O))$$

Here β' is the *urgency flexibility rate*, which determiones how fast the *PUV* values are changing upon new input.

To determine whether there is enough attention for objects that demand attention some comparison has to be made. The attention levels that are estimated are considered as offered attention utilisation, and the urgencies can be considered as demands. However, in principle these quantities are not expressed according to a measure such that they are comparable. Therefore some rescaling has been made in the comparison, in the following discrepancy assessment:

$$D(O) = s_1 * PUV(O) - s_2 * PAV(O)$$

with s_i *comparison weight factors*. The interpretation is that $D(O) = 0$ represents sufficient attention for the object, $D(O) < 0$ more than sufficient, and $D(O) > 0$ insufficient attention. The resulting discrepancy assessments can be used as input to determine appropriate intervention actions; how this can be done will not be discussed in this paper; however, see [14].

As shown, the overall dynamical model involves a number of parameters some of which relate to characteristics of the human and some others to characteristics of the environment, or to a combination. As a summary an overview is given in Table 1.

characteristics	*parameter*	*symbol*	*range*
human characteristics	object feature weight factors	w_i	$0 \leq w_i \leq 1$ & $\Sigma\, w_i = 1$
	gaze distance effect rate	α	$0 \leq \alpha$
	attention flexibility rate	β	$0 < \beta \leq 1$
environmental characteristics	urgency aspect weight factors	r_i	$0 \leq r_i \leq 1$ & $\Sigma\, r_i = 1$
combined characteristics	comparison weight factors	s_i	$0 \leq s_i \leq 1$

Table 1. Overview of the parameters in the dynamical model

3 The Adaptation Approach

When using a dynamical model to assess a human's states in the context of a task and environment, the software agent has to maintain beliefs about characteristics of the human, used as parameters in the model. As often it is not possible to determine accurate values at forehand, this section describes a method by which the agent adapts its beliefs concerning human characteristics to the real characteristics. The main idea is as follows. The agent initially receives rough estimations of the values for these human characteristics, and maintains them as beliefs. Using the dynamical model with parameter values as represented by these initial beliefs, the agent predicts the human state, up to a certain time point. When at that time point, for example by observation, information is obtained that can be related to the real value of one or more state variables of the model, this can be used as input for the adaptation process. The agent then tries to minimize the difference between predicted and real value by adjusting the beliefs on the human characteristics (i.e., the parameter values which were initially assumed). This process of adaptation is kept going on until the difference is low enough, i.e., until the agent has a sufficiently accurate set of beliefs about the human's characteristics. To be able to make reasonable adjustments it is needed to obtain information on how a change in a parameter value affects the difference between predicted and real value of the variable that is considered; this is called the sensitivity of the variable value for the parameter value. In more detail the adaptation process is described as follows.

3.1 Sensitivities: how they are determined

Within this adaptation process sensitivities of state variables for changes in parameter values for human characteristics play an important role. The *sensitivity S* of variable X for parameter P is the number such that a change ΔP in the value of P of parameter P will lead to a change ΔX in X which is (approximately) proportional to ΔP with proportion factor S:

$$\Delta X = S\, \Delta P$$

This is an approximation which is more accurate when the Δ's are taken small; in fact the sensitivity is the partial derivative $\partial X/\partial P$. To determine a sensitivity S (so determining the partial derivative $\partial X/\partial P$) in principle both analytical and experimental/heuristic methods or a combination of them can be used. As systems of differential equations encountered usually cannot be solved analytically, a purely analytical approach to determine sensitivities is often not feasible. As an approximation method the following can be done. A small change ΔP in the parameter is tried to make an additional prediction for X, and based on the resulting change ΔX found in the two predicted values for X, by

$$S_{X,P} = \Delta X/ \Delta P$$

the sensitivity S can be estimated. The idea is that this is done for each of the parameters, one by one.

3.2 Sensitivities: how they are used

Given that a sensitivity $S_{X,P}$ of variable X for parameter P is known it can be used in the following manner. First it can be noticed that sensitivities for some parameters P can be 0 or almost 0. Apparently, such parameters do not have any serious effect on the outcome of the value of variable X. Therefore, changing them based on available values for X does not make much sense: a deviation in value of X cannot be attributed to them, due to their low influence. Based on the estimation of a sensitivity which has distance at least τ_S to 0 (where τ_S is a small threshold value), in principle a better guess for the value of P can be determined by taking

$$\Delta P = -\lambda * \Delta X / S_{X,P}$$

where ΔX is the deviation found between observed and predicted value of X; so, for example, when $\Delta X = 0.25$ and $\lambda = 0.3$, then for $S_{X,P} = 0.75$ this obtains $\Delta P = -0.3*0.25 /0.75 = -0.1$. However, when the sensitity $S_{X,P}$ is a bit smaller, it could be possible that the adjustment of the value of P based on the formula above would exceed the maximum or minimum value of its range. For example, when $\Delta X = 0.25$, $\lambda = 0.3$, and $S_{X,P} = 0.025$ it results in $\Delta P = -0.3*0.25 /0.025 = -3$. To avoid such problems, a kind of threshold function can be applied that maps, for example for a parameter with values in the interval [$0, 1$], the proposed adjustment on a [$0, 1$] interval (which can still be multiplied by a factor for other intervals):

$$\Delta P = \lambda * th(\sigma, \tau, \Delta X / S_{X,P})*(1-W) \qquad \text{when } \Delta X / S_{X,P} \geq 0$$
$$\Delta P = -\lambda * th(\sigma, \tau, -\Delta X / S_{X,P})*W \qquad \text{when } \Delta X / S_{X,P} < 0$$

Here the threshold function with steepness σ and theshold value τ is defined by

$$th(\sigma, \tau, V) = 1/(1+e^{-\sigma(V-\tau)})$$

or, to allow for lower steepnes values by

$$\begin{aligned} th(\sigma, \tau, V) &= [\ 1/(1+e^{-\sigma(V-\tau)}) - 1/(1+e^{\sigma\tau})\ ] / [\ 1 - 1/(1+e^{\sigma\tau})\ ] \\ &= [\ 1/(1+e^{-\sigma(V-\tau)}) - 1/(1+e^{\sigma\tau})\ ] / [\ e^{\sigma\tau}/(1+e^{\sigma\tau})\ ] \\ &= [\ 1/(1+e^{-\sigma(V-\tau)}) - 1/(1+e^{\sigma\tau})\ ] * [1+e^{-\sigma\tau}] \end{aligned}$$

When for more than one variable X information about its real value is obtained, the adjustment ΔP for parameter P is taken as the average of all calculated adjustments based on the different variables X such that the sensitivity $S_{X,P}$ is not close to 0.

4 The Overall Adaptation Process

Based on the adaptation approach explained in Section 3 above, the overall adaptation process was modelled as follows:

Initialisation

1. Take VF the *focus set of variables* X for which information about its real value can be obtained and take a time point t for which information about the real value of all X in F is to be obtained.
2. Take G the subset of parameters P for which adaptation is desired; the other parameters are kept constant.
3. Assume initial values for all of the parameters P.
4. Choose a value for adaptation speed λ.

Sensitivity Determination

5. By simulation determine predicted values V_X at time point t for each X in VF, using the assumed values of the parameters.
6. For each parameter P in G, by simulation determine predicted values CV_X at time point t for each X in VF, using only for P a value changed by some chosen ΔP and the unchanged assumed values for the other parameters.
7. For each parameter P in G and each variable X in VF, determine the sensitivity $S_{X,P}$ of X for P at time point t by dividing the difference between values for X found in 5. and 6. by ΔP:
$$S_{X,P} = (CV_X - V_X) / \Delta P$$

8. Take $VF(P)$ the *focus set of variables X for P* of those variable X for which the sensitivity $S_{X,P}$ has distance to 0 of at least the sensitivity threshold τ_S:
$$VF(P) = \{ X\in VF \mid |S_{X,P}| \geq \tau_S \}$$
9. Take PF the *focus set of parameters* for which the sensitivity for at least one $X\in VF$ has distance to 0 of at least the sensitivity threshold τ_S:
$$PF = \{ P\in G \mid \exists X\in VF\ |S_{X,P}| \geq \tau_S \} = \{ P\in G \mid VF(P) \neq \emptyset\}$$

Adjustment Determination

10. For each variable $X \in VF$ determine the deviation ΔX of the predicted value of X at t from information that is obtained about the real value of X at t.
11. For each parameter $P\in PF$ determine the change ΔP as
$$\Delta P = \lambda * th(\sigma, \tau, (\Sigma_{X\in VF(P)}\ \Delta X / S_{X,P}) / \#(VF(P)))*(1-W) \quad \text{when } (\Sigma_{X\in VF(P)}\ \Delta X / S_{X,P}) \geq 0$$
$$\Delta P = - \lambda * th(\sigma, \tau, -(\Sigma_{X\in VF(P)}\ \Delta X / S_{X,P}) / \#(VF(P)))*W \quad \text{when } (\Sigma_{X\in VF(P)}\ \Delta X / S_{X,P}) < 0$$
12. For each parameter $P\in PF$ adjust its value by ΔP.

By repeating this process a number of times both for each time point considered and for different time points, over time an approximation is obtained.

5 Further Details of the Implementation

In this section some details of the the ambient support environment for attention-demanding tasks are described. The software environment has been implemented within the Adobe® Flex® development environment. The software was implemented according to a component-based design. Between the different components event-driven interaction takes place, implemented using ActionScript. Moreover, for the gaze sensoring a specific sensoring component was included that connects the Tobii® eye-tracker used.

To address the possibilities for human-aware ambient support of attention-demanding tasks a (simplified) simulated environment to perform such a type of task has been developed; for more details, see [14]. A person has to (1) inspect visually displayed moving objects in the environment, and identify whether such an object is dangerous (enemy) or not (ally), and (2) for each of such objects, depending on the identification perform some actions. The player's task consists in classifying each object as ally or enemy, and shooting down the enemies while letting allies land safely. Identification of an object is a cognitive task, in a simplified form represented by an arithmetical calculation. The player uses a cannon at the bottom of the screen to shoot down (hostile) objects.

The main idea for an ambient agent supporting this tasks is that it has awareness about whether or not the human pays enough attention to urgent situations. Urgency is determined as a weighted sum of a number of factors, which is multiplied by an identification factor which is *0* for friends and *1* for enemies; so for friends the urgency will always be *0*. Examples of factors used are speed, direction and time and distance to the ground. Using this urgency model the agent can determine how urgent the situations concerning certain objects are, and compare it to the amount of attention, thus obtaining the discrepancies for each of the objects (according to the model described in Section 2). These discrepancies play the role of the variables indicated by *X* in Section 4.

For the adaptation model a crucial aspect is the way in which a deviation between the discrepancy value estimated by the agent and in reality can be determined. In this case, as an approximation the mouseclicks of the person have been used as an indication of discrepancy; see Table 2.

enemy/ friend	*mouse click*	*indicated discrepancy*	*current discrepancy*	*deviation based on current minus indicated discrepancy*
enemy	+	negative	positive	current discrepancy + *0.2*
enemy	+	negative	negative	*0*
enemy	-	positive	positive	*0*
enemy	-	positive	negative	current discrepancy - *0.2*
friend	+	negative	positive	current discrepancy + *0.2*
friend	+	negative	negative	*0*
friend	-	negative	positive	current discrepancy + *0.2*
friend	-	negative	negative	*0*

Table 2. Determination of deviations

For example, if at some time point a mouseclick occurs at a certain position of an enemy, it is assumed that this indicates sufficient attention for that position at that point in time which implies an indication that the discrepancy should be negative. Similarly, when for a certain duration no mouse click occurs at a friendly position, this is considered a sign for negative discrepancy. If the current discrepancy is positive and the indicated negative, the deviation is determined as the current discrepancy estimation plus *0.2*.

Fig. 1 shows for the example scenario how the three objects move compared to gaze positions. Object 1 (upper graphs), representing an enemy, moves almost vertically, and almost all the time the gaze position has some distance to this object, except maybe around time points 800 and 950, here the gaze is a bit closer. In contrast, the positions of the friendly object 2, depicted in the second row of graphs, often coincide with the gaze positions. The lowest part shows that the positions of object 3 coincide with gaze positions only around time points 250 and 950. Tables 3 and 4 describe information about the settings in the example scenario. Note that the person considered performs far from perfect as the gaze is often at (friendly) object 2 and not at object 1 which is the enemy. Therefore an appropriate estimation of discrepancy between urgency and attention would show a rather positive value for object 1 and a rather negative value for object 2.

	Brightness	*Size*	*Identification*
Object 1	*0.9*	*0.2*	enemy
Object 2	*0.1*	*0.9*	friend
Object 3	*0.25*	*0.3*	friend

Table 3. Some of the object features

α	β	*Identification weight*	*Urgency weight*	*Attention weight*	λ	*Urgency flexibility*
1	*0.2*	*1*	*0.3*	*0.9*	*0.02*	*0.02*

Table 4. Parameter values

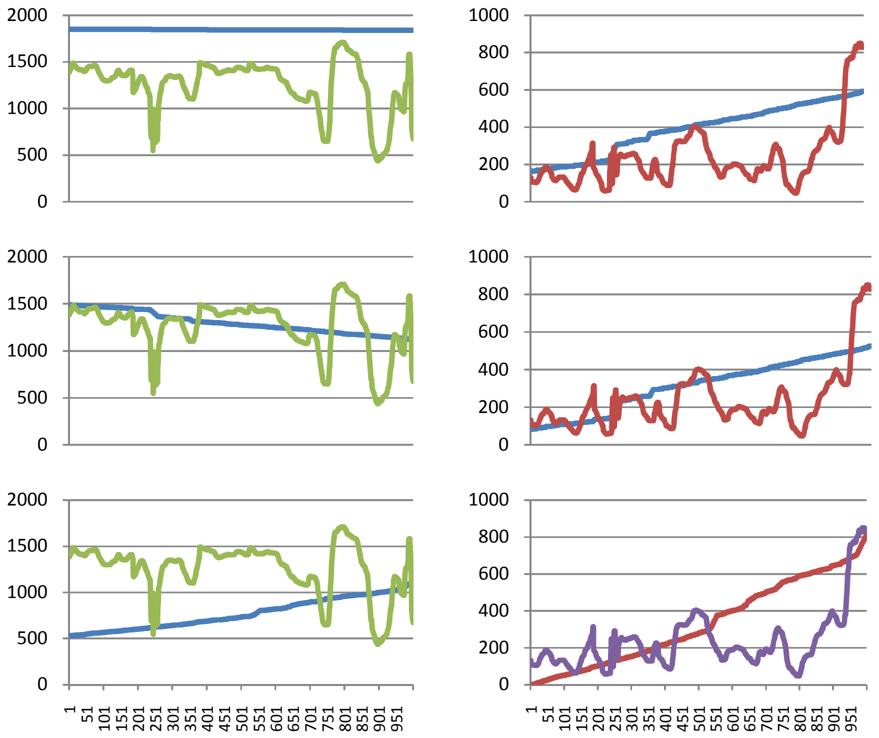

Fig. 1. Positions of three objects over time compared to gaze positions in the example scenario: left hand sides show x-coordinates, right hand sides y-coordinates. It shows that from time point 0 to 200 the gaze is near object 2, around 250 near object 3, from 300 to 500 near object 2 again, around 800 most close to object 1, and after 900 near object 3 again.

In Fig. 2 it is shown how the discrepancies develop over time for each of the objects. It shows that initially the discrepancies are not estimated well. For example, for object 1 discrepancy starts negative. This incorrect estimation of the discrepancy can be explained by the parameter setting for weight factors for brightness and size: the brightness weight factor is initially *0.9*, and object 1 happens to have a high brightness. The more or less converse situation is shown for object 2, which has a low brightness and high value for size. However, for the estimation of discrepancy the high size is not counted much as initially the weight factor parameter for size is only *0.2*.

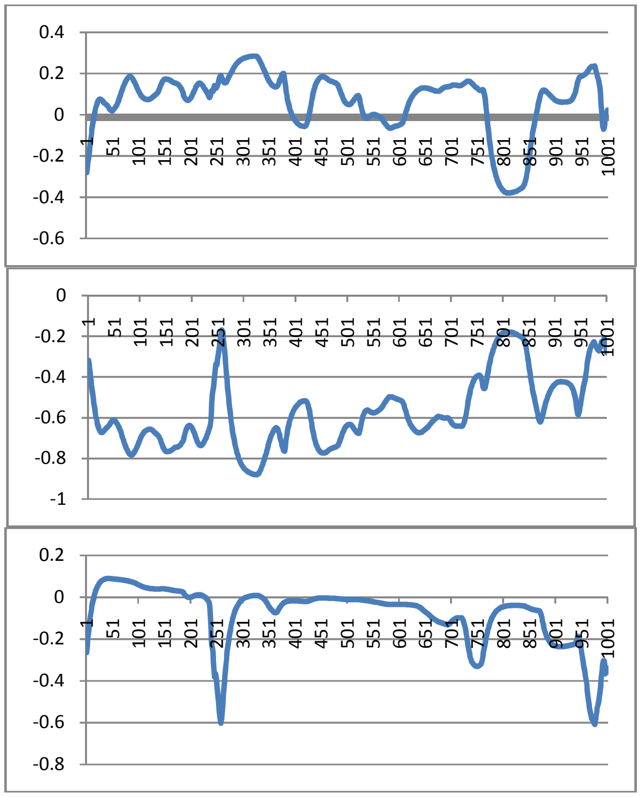

Fig. 2. Estimated discrepancies between urgency and attention for the three objects

Due to these apparantly inadequate parameter settings, the ambient software agent initially has a false awareness of the person’s attention; for the agent it is as if the person pays enough attention to the enemy object but this is not the case in reality. By the adaptation process the initially inadequate parameter values for the brightness and size weight factors are changing to more appropriate values: from *0.2* vs *0.9* to around *0.8* vs *0.2*, as shown in Fig. 3. As a result of this adaptation, the discrepancies shown in Fig. 2 show a more faithful representation of the situation after, say time point 100 or 150: the discrepancy for object 1 is estimated as rather high positive, and for object 2 strongly negative. This gives the ambient software agent an appropriate awareness of the attention of this person who is almost all the time looking at the wrong (friendly) object, and does not seem to notice the enemy object.

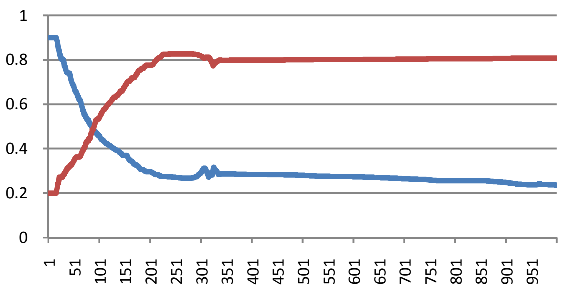

Fig. 3. Adaptation of the parameter values for the brightness weight factor (from *0.9* to around *0.2*) and size weight factor (from *0.2* to around *0.8*)

6 Evaluation

In order to evaluate whether the agent functions as expected, an automated analysis of dynamic properties has been performed. In this analysis, a number of dynamic statements that were expected to hold for the agent have been formalised in the language TTL [4], and have been automatically verified against simulation traces (using Matlab). The predicate logical language TTL supports formal specification and analysis of dynamic properties, covering both qualitative and quantitative aspects. TTL is built on atoms referring to *states* of the world, *time points* and *traces*, i.e. trajectories of states over time. In addition, *dynamic properties* are temporal statements that can be formulated with respect to traces based on the state ontology Ont in the following manner. Given a trace γ over state ontology Ont, the state in γ at time point t is denoted by state(γ, t). These states can be related to state properties via the formally defined satisfaction relation denoted by the infix predicate |=, comparable to the Holds-predicate in the Situation Calculus: state(γ, t) |= p denotes that state property p holds in trace γ at time t. Based on these statements, dynamic properties can be formulated in a formal manner in a sorted first-order predicate logic, using quantifiers over time and traces and the usual first-order logical connectives such as ¬, ∧, ∨, ⇒, ∀, ∃. A first dynamic property that was expected to hold for the agent (expressed as P1 below) addresses the adaptation of parameter values:

P1 - Smaller range of adaptation over time
For all traces γ, and all parameters p
the difference between the highest and the lowest value of p is larger in the first part of γ than in the second part.

P1 ≡ ∀γ:TRACE ∀p:PARAMETER ∀x1,x2,y1,y2:REAL
max(x1, p, γ, 0, end_time/2) & min(x2, p, γ, 0, end_time/2) &
max(y1, p, γ, end_time/2, end_time) & min(y2, p, γ, end_time/2, end_time)
⇒ x1 - x2 > y1 - y2

Here, end_time denotes the last time point of the simulation trace, and max and min are defined as follows:

```
max(x:REAL, p:PARAMETER, γ:TRACE, tb:TIME, te:TIME) ≡
∃t:TIME
  state(γ, t) |= has_value(p, x) & tb ≤ t ≤ te  &
  ¬ [ ∃t':TIME ∃x':REAL    state(γ, t') |= has_value(p, x') & tb ≤ t' ≤ te  & x' > x ]

min(x:REAL, p:PARAMETER, γ:TRACE, tb:TIME, te:TIME) ≡
∃t:TIME
  state(γ, t) |= has_value(p, x) & tb ≤ t ≤ te  &
  ¬ [ ∃t':TIME ∃x':REAL    state(γ, t') |= has_value(p, x') & tb ≤ t' ≤ te  & x' < x ]
```

Automated checks pointed out that this property indeed holds for all generated traces. For example, for the first half of the trace shown in Figure 3, it turns out that the parameter 'brightness' varies between *0.18* and *0.90* (a difference of *0.72*), whereas for the second half of the trace it only varies between *0.20* and *0.17* (a difference of *0.03*). Similarly, the parameter 'size' varies between *0.20* and *0.84* in the first half of that trace (a difference of *0.64*), and between *0.79* and *0.80* in the second half (a difference of *0.01*). In addition to checking whether the parameter adaptation is performed correctly, it is interesting to check whether the accuracy of the model improves over time due to these parameter adaptations. To this end, one basically needs to check whether the estimated attention corresponds more to the actual attention over time. Since no information about the actual attention is available, the estimated attention is compared to the mouse clicks of the participant. In particular, the amount of *hits*, *misses*, *false alarms*, and *correct rejections* are counted, similar to signal detection approaches [13]. For the current purposes, these notions are defined as follows:

hit - The model estimates a *high* attention level for contact c at time point t, and indeed the participant *does click* on this contact within d time points

miss - The model estimates a *low* attention level for contact c at time point t, whilst the participant *does click* on this contact within d time points

false alarm - The model estimates a *high* attention level for contact c at time point t, whilst the participant *does not click* on this contact within d time points

correct rejection - The model estimates a *low* attention level for contact c at time point t, and indeed the participant *does not click* on this contact within d time points

These cases have been summed over all time points and all contacts. Within TTL, the following properties have been formalised to count the above cases (where th is a threshold to estimate high attention, and D is a constant to represent response time, which was taken 500 msec):

hit

```
hit(γ:TRACE, t:TIME, x:CONTACT) ≡
    ∃t':TIME ∃i:real
      state(γ,t) |= belief(has_value(av_for(x), i)) & i > th &
      state(γ,t') |= belief(clicked_on(x)) & t < t' ≤ t+D
```

miss

```
miss(γ:TRACE, t:TIME, x:CONTACT) ≡
    ∃t':TIME ∃i:real      state(γ,t) |= belief(has_value(av_for(x), i)) & i ≤ th &
      state(γ,t') |= belief(clicked_on(x)) & t < t' ≤ t+D
```

false alarm false_alarm(γ:TRACE, t:TIME, x:CONTACT) ≡
∃i:real state(γ,t) |= belief(has_value(av_for(x), i)) & i > th &
¬∃t':TIME [state(γ,t') |= belief(clicked_on(x)) & t < t' ≤ t+D]

correct rejection correct_rejection(γ:TRACE, t:TIME, x:CONTACT) ≡
∃i:real state(γ,t) |= belief(has_value(av_for(x), i)) & i ≤ th &
¬∃t':TIME [state(γ,t') |= belief(clicked_on(x)) & t < t' ≤ t+D]

By counting these occurrences, one can calculate the sensitivity and specificity of the model as follows:

sensitivity = hits/(hits+misses)
specificity = correctrejections/(correctrejections+falsealarms)

In principle, an accurate model has a high sensitivity and a high specificity. However, these values also depend on the choice of the threshold th. An extremely low threshold (of *0*) always results in a sensitivity of *1* and a specificity of *0*, whereas an extremely high threshold (of *1*) always results in a sensitivity of *0* and a specificity of *1*. Therefore, the accuracy should be determined with the threshold as a variable. To this end, an ROC (Relative Operative Characteristic) analysis has been performed [10].

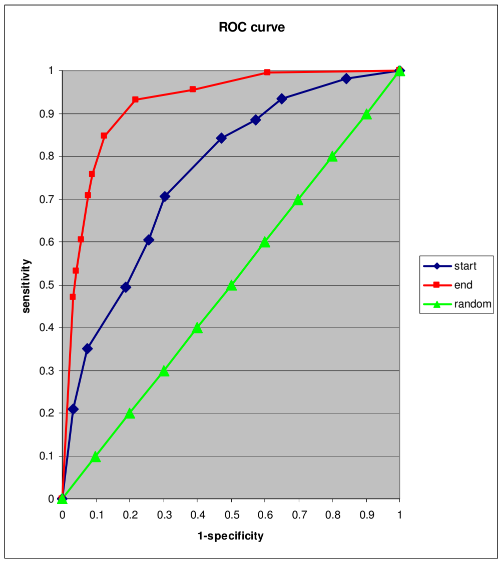

Fig. 4. ROC curves for the model in two different stages, and a random estimation.

The results of this analysis are shown in Figure 4. This figure shows the ROC curves for an example situation in which our agent was applied. The blue curve indicates the start of the scenario, in which the parameters were not yet tuned (in particular, the first 12000 entries of the log file), whereas the red curve indicates the end of the scenario, in which the parameters were tuned (the last 12000 entries). The green curve indicates the results of a baseline agent which makes random estimations with respect to attention. Each curve contains 11 data points (representing 11 different values for th) As shown in Figure 4, the model produces much better after parameter adaptation results than before.

7 Discussion

In order to function in a knowledgeable manner, intelligent support agents need awareness of the characteristics and states of the human(s) they are supporting. In fact to acquire such human-awareness they need to perform some form of mindreading (e.g., [9], [11], [12]). In order to obtain the capability of mindreading the presented agent was assumed to be equipped with a dynamical model of the human (cf. [7], [15], [17]). In addition to the variables for which values are calculated over time, typically such a dynamical model involves a number of parameters that represent characteristics of the human and the environment. As often individual differences between humans exist, a major challenge here is how the agent can acquire appropriate beliefs on values for parameters representing human characteristics that indeed reflect the specific person involved. Sometimes personal characteristics of a human can be determined at forehand by means of questionnaires and/or interviews. However, such methods do not guarantee appropriate outcomes, as what a person says he or she does is not always the same as what a person actually does. Therefore the option to estimate such parameters (e.g., [16], [18]) may be a better direction to explore. If such parameter estimation is performed by the agent at runtime, this results in an adaptive agent that over times gets more accurate beliefs on the characteristics of the human.

The adaptive software agent presented here possesses a parameter adaptation model that enables it to perform mindreading in particular concerning a human's attention states (e.g., [7], [15]). This attention-reading capability is based on information acquired by sensoring of the gaze, features of the relevant objects in the environment, and a dynamical model in the form of a system of differential equations integrating all this information. For the latter the dynamical model described in [7] was adopted. As a case study for the parameter adaptation model it was applied to the parameters in this attention-reading model. By a formal verification process it was shown for this case study that the adaptive software agent satisfies a number of expected properties.

The software environment was implemented according to a component-based design within the Adobe® Flex® development environment, which makes it easy to adapt. Interaction between components was implemented using ActionScript, in an event-driven manner. A sensoring component was included for the gaze sensoring which connects to the Tobii® eye-tracker.

The software agent with a capability to adapt to personal human characteristics as presented here may be the basis for applications in Ambient Intelligence or Pervasive Computing (e.g., [1], [2], [3]), where it is assumed that computing takes place in the background without addressing the human by an explicit interaction. When such agents can perform some forms of mindreading based on observations and dynamical models they possess, they can indeed act in the background without directly addressing the human. Some steps in this direction for the case of attention-reading were explored in [5] and [6]; however these approaches are not adaptive: parameter values need to be given initially. Another approach, not on attention-reading but on emotion-reading can be found in [8]. Although there parameter adaptation takes place, this is only for a simple case where only one parameter is adapted and where an analytical approach for parameter adjustment was used as the differential equation could be solved analytically to obtain an explicit formula for the sensitivity.

References

1. Aarts, E.; Collier, R.; Loenen, E. van; Ruyter, B. de (eds.) (2003). Ambient Intelligence. Proc. of the First European Symposium, EUSAI 2003. Lecture Notes in Computer Science, vol. 2875. Springer Verlag, 2003, pp. 432.
2. Aarts, E., Harwig, R., and Schuurmans, M. (2001). Ambient Intelligence. In: P. Denning (ed.), The Invisible Future, McGraw Hill, New York, pp. 235-250.
3. Bosse, T., Hoogendoorn, M., Klein, M., and Treur, J., A Component-Based Ambient Agent Model for Assessment of Driving Behaviour. In: F.E. Sandnes et al. (eds), Proc. of the 5th International Conference on Ubiquitous Intelligence and Computing, UIC'08. Lecture Notes in Computer Science, vol. 5061. Springer Verlag, 2008, pp. 229-243.
4. Bosse, T., Jonker, C.M., Meij, L. van der, Sharpanskykh, A., and Treur, J. (2009). Specification and Verification of Dynamics in Cognitive Agent Models. International Journal of Cooperative Information Systems, vol. 18, 2009, pp. 167-193
5. Bosse, T., Lambalgen, R. van, Maanen, P.P. van, and Treur, J., Attention Manipulation for Naval Tactical Picture Compilation. In: Proc. of the 9th Conference on Intelligent Agent Technology, IAT'09. IEEE Computer Society Press, 2009, to appear.
6. Bosse, T., Lambalgen, R. van, Maanen, P.-P. van, Treur, J., Automated Visual Attention Manipulation. In: L. Paletta, Tsotsos, J.K. (eds.), Attention in Cognitive Systems, Proc. of the Fifth International Workshop on Attention in Cognitive Systems, WAPCV'08. Lecture Notes in Artificial Intelligence, vol. 5395. Springer Verlag, 2009, pp. 257-272.
7. Bosse, T., Maanen, P.-P. van, Treur, J., Simulation and Formal Analysis of Visual Attention, Web Intelligence and Agent Systems Journal, vol. 7, 2009, pp. 89-105.
8. Bosse, T., Memon, Z.A., and Treur, J., Adaptive Estimation of Emotion Generation for an Ambient Agent Model. In: Aarts, E., Crowley, J.L., Ruyter, B. de, Gerhäuser, H., Pflaum, A., Schmidt, J., Wichert, R. (eds.), Ambient Intelligence, Proceedings of the Second European Conference on Ambient Intelligence, AmI'08. Lecture Notes in Computer Science, vol. 5355. Springer Verlag, 2008, pp. 141–156.
9. Bosse, T., Memon, Z.A., and Treur, J., A Two-Level BDI-Agent Model for Theory of Mind and its Use in Social Manipulation. In: Proceedings of the AISB 2007 Workshop on Mindful Environments, 2007, pp 335-342.
10. Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27, 861-874.

11. Gärdenfors, P. (2003). How Homo Became Sapiens: On The Evolution Of Thinking. Oxford University Press, 2003.
12. Goldman, A.I. (2006). Simulating Minds: the Philosophy, Psychology and Neuroscience of Mindreading. Oxford University Press.
13. Green, D.M., Swets J.A. (1966) Signal Detection Theory and Psychophysics. NY: Wiley.
14. Memon, Z.A., Oorburg, R., Treur, J., Umair, M., and Vos, M. de, A Software Environment for Human-Aware Ambient Support of Attention-Demanding Tasks. Technical Report, VU University, Department of AI.
15. Itti, L. and Koch, C., Computational Modeling of Visual Attention, Nature Reviews Neuroscience, vol. 2, no. 3, 2001, pp. 194-203.
16. Pearson, C.E., 1986. Numerical Methods in Engineering and Science. CRC Press.
17. Port, R.F., and Gelder, T. van (eds.), 1995. Mind as Motion: Explorations in the Dynamics of Cognition. MIT Press, Cambridge, Mass.
18. Sorenson, H.W. (1980). Parameter estimation: principles and problems. Marcel Dekker, Inc., New York