An Adversarial Risk Analysis Model for a Decision Agent facing Multiple Users

Pablo G. Esteban*, Javier G. Rázuri*, David Ríos Insua[§]

*Department of Statistics and Operations Research, Universidad Rey Juan Carlos
Madrid, Spain

* pablo.gomez.esteban@urjc.es

* javierfrancisco.guerrero.razuri@urjc.es

[§]Royal Academy of Sciences

Madrid, Spain

[§]david.rios@urjc.es

Abstract—We provide a model supporting the decision making process of an autonomous synthetic agent which interacts with several users. The approach is decision analytic and incorporates models forecasting the users' behavior. We sketch the implementation of our model with an edutainment robot.

I. INTRODUCTION

We have introduced in [11] Adversarial Risk Analysis (ARA) as a framework to cope with risk analysis cases in which risks stem from deliberate actions of intelligent adversaries. In supporting one of the participants, the problem is viewed through decision analysis, but principled procedures which employ the adversarial structure are used to assess the probabilities of the adversaries' actions. We avoid the standard game theoretic assumption of common knowledge by accommodating as much information as we can within our analysis, through a structure of nested decision models. Depending on the level we climb up in such hierarchy, we talk about 0-level analysis, 1-level analysis and so on, see [1] and its discussion. [1], [10] and [11] have introduced different principles to close the above hierarchy.

In this paper, we explore how the ARA framework may support the decision making of an autonomous agent in its interaction with several users. The model is essentially multi-attribute decision analytic, see [3], but our agent entertains also models forecasting the evolution of its adversaries and the environment surrounding all of them. Our application domain is in robotics, where we aim at supporting the decision making of a bot which interacts with several users.

II. BASIC ELEMENTS

We aim at designing an agent A whose activities we want to regulate and plan. There are r participants or users, $B_1, \dots, B_r \in \mathcal{U}$ which interact with A . An index x will be used to identify the corresponding user. The activities of both A and the B_x 's take place within an environment E . As a motivating example, suppose that we aim at designing a bot A which will interact with a group of three kids, B_1, B_2, B_3 , within a room E .

A makes decisions within a finite set $\mathcal{A} = \{a_1, \dots, a_m\}$, which includes a *do nothing* action. A 's decisions might

affect the users. The B_x 's make decisions within a set $\mathcal{B} = \{b_1, \dots, b_n\}$, which also includes a *do nothing* action. $\mathcal{B}$ will be as complete as possible, while simplifying all feasible results down to a finite number. It might be the case that not all users' actions are known a priori. This set could grow as the agent learns, adding new users' actions, as we discuss below. The environment E changes just with the users' actions. We assume that the environment adopts a state within a set $\mathcal{E}$. The agent faces this changing environment, which affects its own behavior.

A has q sensors which provide readings. Sensory information originating in the external environment plays an important role, which affect the agents' decision-making process. Each sensor reading is attached to a time t , so that the sensor reading vector will be $s_t = (s_t^1, \dots, s_t^q)$. The agent infers the external environmental state e , based on a, possibly probabilistic, transformation function f , so that

$$\hat{e}_t = f(s_t).$$

A also uses the sensor readings to infer which user he is facing, through a probabilistic function h

$$\hat{B}_t = h(s_t).$$

Finally, A employs the sensor readings to infer what such user has done, based on a (possibly probabilistic) function g

$$\hat{b}_t = g(s_t).$$

We design our agent by planning its activities according to the basic loop in Fig. 1, which is open to interventions if an exception occurs.

Fig. 1. Basic Agent Loop

The time the agent spends performing a simple loop like this, is called an *instant*.

III. THE AGENT DECISION MODEL

Essentially, we shall plan our agent's activities over time within the decision analytic framework, see [3]. Note that we could view the problem within the game-theoretic framework, see [5], but we shall avoid the corresponding common knowledge assumptions. Moreover, we focus just on supporting the agent. We describe, in turn, the forecasting model, which incorporates the ARA elements; the preference model; and, finally, the corresponding optimization problem.

A. Forecasting Models

The agent maintains a forecasting model which suggests with which probabilities will the users act and the environment react, given the past history of agent's actions, users' actions and the evolution of the environment. We describe the general structure of such models.

Assume that, for computational reasons, we limit the agent's memory to two instant times, so that we are interested in computing, for each user B_x ,

$$p(e_t, b_t \mid a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}), B_x). \quad (1)$$

Extensions to k instants of memory or forecasting m steps ahead follow a similar path. (1) may be decomposed through

$$p(e_t \mid b_t, a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}), B_x) \times \\ \times p(b_t \mid a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}), B_x).$$

We assume that the environment is fully under control by the users. The motivating example, they control the light, the temperature and other features of the room. Moreover, they may plug in the bot to charge its battery, and so on. In general, only the latest of the users' actions will affect the evolution of the environment. Thus, we shall assume that

$$p(e_t \mid b_t, a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}), B_x) = \\ = p(e_t \mid b_t, e_{t-1}, e_{t-2}).$$

We term this the *environment model*.

Similarly, we shall assume that the users have their own behavior evolution, that might be affected by how they react to the agent actions, thus incorporating the ARA principle, so that

$$p(b_t \mid a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}), B_x) = \\ = p(b_t \mid a_t, b_{t-1}, b_{t-2}, B_x). \quad (2)$$

The agent will maintain two models in connection with (2) for each user. The first one describes the evolution of the users by themselves, assuming that they are in control of the whole environment, and they are not affected by the agent's actions. We call these the *users' models* and describe them through

$$p(b_t \mid b_{t-1}, b_{t-2}, B_x).$$

The other one refers to the users' reactions to the agent's actions. Indeed, it assumes that the users are fully reactive to the agent's actions, which we describe through

$$p(b_t \mid a_t, B_x).$$

We call them the *classical conditioning models*, with the agent possibly conditioning the users.

We combine both models to recover (2). We view the problem as one of model averaging, see [7]. In such case,

$$p(b_t \mid a_t, b_{t-1}, b_{t-2}, B_x) = \\ = [p(M_2 \mid B_x)p(b_t \mid b_{t-1}, b_{t-2}, B_x) + \\ + p(M_1 \mid B_x)p(b_t \mid a_t, B_x)],$$

where $p(M_i \mid B_x)$ denotes the probability that the agent gives to model i , assuming that the user is B_x , with $p(M_1 \mid B_x) + p(M_2 \mid B_x) = 1$, $p(M_i \mid B_x) \geq 0$, which, essentially, capture how reactive to the agent's actions the users are.

Finally, we shall use

$$p(e_t, b_t \mid a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2})) = \\ = \sum_u \left[p(e_t \mid b_t, e_{t-1}, e_{t-2}) \times \right. \\ \left. \times p(b_t \mid a_t, b_{t-1}, b_{t-2}, B_x) \times p(B_x) \right].$$

Learning about various models within our implementation is sketched in Section 4.

B. Preference Model

We describe now the preference model. Assume that the agent faces multiple consequences $c = (c_1, c_2, \dots, c_l)$. At each instant t , these will depend on his action a_t , the users' action b_t and the future state e_t , realized after a_t and b_t . Therefore, the consequences will be of the form

$$c_i(a_t, b_t, e_t), \quad i = 1, \dots, l.$$

We assume that they are evaluated through a multi-attribute utility function, see [3]. Specifically, without much loss of generality see [13], we shall adopt an additive form

$$u(c_1, c_2, \dots, c_l) = \sum_{i=1}^l w_i u_i(c_i),$$

with $w_i \geq 0$, $\sum_{i=1}^l w_i = 1$.

C. Expected Utility

The goal of our agent will be to maximize predictive expected utility, see [4]. Planning several instants ahead requires computing maximum expected utility plans defined through:

$$\max_{(a_t, \dots, a_{t+r})} \psi(a_t, \dots, a_{t+r}) = \\ = \sum_{(b_t, e_t), \dots, (b_{t+r}, e_{t+r})} \left[\left[\sum_{i=0}^r u(a_{t+i}, b_{t+i}, e_{t+i}) \right] \times \right. \\ \left. \times p((b_t, e_t), \dots, (b_{t+r}, e_{t+r}) \mid \right. \\ \left. \mid (a_t, a_{t+1}, \dots, a_{t+r}, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}))) \right].$$

assuming utilities to be additive over time.

This could be solved through dynamic programming. If planning several instants ahead turns out to be very expensive computationally, we could plan just one period ahead. In this case, we would aim at solving

$$\max_{a_t \in \mathcal{A}} \psi(a_t) = \sum_{b_t, e_t} u(a_t, b_t, e_t) \times [p(b_t, e_t \mid a_t, (e_{t-1}, a_{t-1}, b_{t-1}), (e_{t-2}, a_{t-2}, b_{t-2}))].$$

Agents operating in this way may end up being too predictable. We may mitigate such effect by choosing the next action in a randomized way, with probabilities proportional to the predictive expected utilities, that is

$$P(a_t) \propto \psi(a_t), \quad (3)$$

where $P(a_t)$ is the probability of choosing a_t .

IV. IMPLEMENTATION

The above procedures have been implemented within the AISoy1 robot environment (<http://www.aisoy.es>). Some of the details of the model implemented are described next, with code developed in C++ over Linux.

A. Basic elements

The robot's actions in $\mathcal{A}$ include actions for complaining, some ways of calling the users' attention, several options to interact with the users and a *do nothing* action. This totals $m = 14$ alternatives, with $\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}\} = \{\text{cry, alert, warn, ask for help, salute, play, speak, ask for playing, ask for charging, ask for shutting down, tell jokes, tell stories, tell events, do nothing}\}$.

On the users' side, set $\mathcal{B}$, the robot is able to detect several users' actions, some of them in a probabilistic way. Indeed, the robot detects three types of actions: affective actions, aggressive actions, and interacting actions. The robot will also detect whether any of the users made *no action*. This totals $n = 12$ actions with $\mathcal{B} = \{b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9, b_{10}, b_{11}, b_{12}\} = \{\text{recharge, stroke, flatter, attack, offend, move, update, speak, play, order, ignore, do nothing}\}$.

Regarding the environment, set $\mathcal{E}$, the bot may recognize contextual issues such the presence of noise, the level of darkness, the temperature, or its inclination. To do so, the bot has several sensors including a camera to detect objects or persons within a scene, and identify the incumbent user; a microphone used to recognize when the users talk and understand what they say, through an ASR component; some touch sensors to interpret when it has been stroked or attacked; an inclination sensor so as to know whether it is in vertical position or not; and a temperature sensor. The information provided by these sensors is mainly used by the bot to infer the users' actions. Some are based on simple deterministic rules, others are based on more complex probabilistic rules. Here we provide the description of how some of the users' actions are actually detected:

a) Deterministic rules:

- b_4 : attack. Rule: There are changes in the inclination at the following 2 instants or the bot is not in vertical position.
- b_{11} : do nothing. Rule: Detects the presence of the user and the user does not do any of the defined actions and the bot is in vertical position.

b) Probabilistic rules:

- b_6 : offend. Rule: Detection of words in a specific set [insults, threats, etc.] and detects the presence of the user or the name of the bot.
- b_9 : play. Rule: Detects the presence of the user or the name of the bot and the user asks for playing.
- b_{10} : order. Rule: Detects the presence of the user or the name of the bot and the user asks for an action within a given set.

B. Forecasting model

We describe now how we have implemented the relevant forecasting models. D_t will designate the data available until time t .

1) *Modeling $p(b_t \mid a_t, B_x)$* : This model forecasts the users' actions based on the agent action, for each B_x . We shall use a matrix-beta model for such purpose [12]. For each a_t , the prior distribution will be Dirichlet, so that

$$p(b_t \mid a_t = a_j, B_x) \sim \text{Dir}(\beta_{1j}^x, \dots, \beta_{nj}^x), \quad b_t \in \{b_1, b_2, \dots, b_n\}.$$

Now, if h_{ij}^x designates the number of occurrences of user B_x doing b_i , when the bot has made a_j , the posterior distribution will be

$$p(b_t \mid a_t = a_j, D_t, B_x) \sim \text{Dir}(\beta_{1j}^x + h_{1j}^x, \dots, \beta_{nj}^x + h_{nj}^x), \quad b_t \in \{b_1, b_2, \dots, b_n\}.$$

When necessary, we may summarize it through its average

$$\hat{p}_{ij}^x = \frac{\beta_{ij}^x + h_{ij}^x}{\sum_i (\beta_{ij}^x + h_{ij}^x)}, \quad i \in \{1, 2, \dots, n\}, j \in \{1, 2, \dots, m\}.$$

The required data will be stored in a matrix structure, in which the last row accumulates the sum of row's values for each column. There will be one of these structures for each user.

$$\begin{pmatrix} \beta_{11}^t = \beta_{11} + h_{11} & \dots & \beta_{1m}^t = \beta_{1m} + h_{1m} \\ \vdots & \vdots & \vdots \\ \beta_{n1}^t = \beta_{n1} + h_{n1} & \dots & \beta_{nm}^t = \beta_{nm} + h_{nm} \\ \beta_{(n+1)1}^t = \sum_{i=1}^n (\beta_{i1} + h_{i1}) & \dots & \beta_{(n+1)m}^t = \sum_{i=1}^n (\beta_{im} + h_{im}) \end{pmatrix}$$

At each time instant, we shall increment the corresponding ij -th element of the matrix and the corresponding element of the last row. That is, if the sequence was a_j, b_i , we shall update $\beta_{ij}^{u(t+1)} = \beta_{ij}^{ut} + 1$ and $\beta_{(n+1)j}^{u(t+1)} = \beta_{(n+1)j}^{ut} + 1$, with the rest of entries satisfying $\beta_{ij}^{u(t+1)} = \beta_{ij}^{ut}$. Since we expect lots of data, the terms β_{ij}^x will not matter that much after a while. Thus, we shall use the following noninformative prior assessment:

if a pair of actions $a_t = a_j$ and $b_t = b_i$ are compatible, we shall make $\beta_{ij}^x = 1$; otherwise, we shall make $\beta_{ij}^x = 0$.

2) *Modeling* $p(b_t | b_{t-1}, b_{t-2}, B_x)$: We provide now our forecasting model for the current users' action based on what the users have done two time steps before. As above, we use a matrix-beta model for each user (Fig. 2). For $i, j \in \{1, 2, \dots, n\}$, we have a priori

$$p(b_t | b_{t-1} = b_i, b_{t-2} = b_j, B_x) \sim \text{Dir}(\beta_{1ij}^x, \dots, \beta_{nij}^x), \quad b_t \in \{b_1, b_2, \dots, b_n\}.$$

If h_{kij}^x designates the number of occurrences in which the user B_x did b_k after having done b_i and b_j , we have that the posterior is

$$p(b_t | b_{t-1} = i, b_{t-2} = j, D_t, B_x) \sim \text{Dir}(\beta_{1ij}^x + h_{1ij}^x, \dots, \beta_{nij}^x + h_{nij}^x), \quad b_t \in \{b_1, b_2, \dots, b_n\},$$

which we may summarize through

$$\hat{p}_{kij} = \frac{\beta_{kij}^x + h_{kij}^x}{\sum_k (\beta_{kij}^x + h_{kij}^x)}, \quad k \in \{1, 2, \dots, n\}.$$

The data structure used to store the required information will consist of a three-dimensional matrix, and there will be one for each of the users

Fig. 2. Matrix-beta model for each B_x

As before, at each time instant, we update the corresponding kij -th element and the corresponding last row of the cube. The prior β_{kij} 's elements are assessed as before.

3) *Model averaging*: We describe now how model averaging and updating takes place within our model. First, recall that we shall use

$$\begin{aligned} p(b_t | a_t, b_{t-1}, b_{t-2}, D_t, B_x) &= \\ &= p(M_1 | D_t, B_x) p(b_t | a_t, D_t, B_x) + \\ &+ p(M_2 | D_t, B_x) p(b_t | b_{t-1}, b_{t-2}, D_t, B_x), \end{aligned}$$

with

$$p(M_i | D_t, B_x) = \frac{p(D_t | M_i, B_x) p(M_i | B_x)}{\sum_{i=1}^2 p(D_t | M_i, B_x) p(M_i | B_x)}, \quad i = 1, 2.$$

Under the assumption $p(M_1 | B_x) = p(M_2 | B_x) = \frac{1}{2}$,

$$p(M_i | D_t, B_x) = \frac{p(D_t | M_i, B_x)}{\sum_{i=1}^2 p(D_t | M_i, B_x)},$$

with

$$p(D_t | M_i, B_x) = \int p(D_t | \theta_i, M_i, B_x) p(\theta_i | M_i, B_x) d\theta_i$$

We provide now the computations for our models:

- M_1 . We have, for each B_x ,

$$\begin{aligned} p(D_t | M_1, B_x) &= \\ &= \int \dots \int \left(\prod_{i,j} p_{ij}^{h_{ij}^x} \right) k \left(\prod_{i,j} p_{ij}^{\beta_{ij}^x - 1} \right) dp_{ij}, \end{aligned}$$

where k is the corresponding normalization constant. Simple computations lead to

$$\begin{aligned} p(D_t | M_1, B_x) &= \left[\frac{\prod_{i=1}^n \Gamma(\beta_{i1}^x)}{\Gamma(\sum_{i=1}^n \beta_{i1}^x)} \dots \frac{\prod_{i=1}^n \Gamma(\beta_{in}^x)}{\Gamma(\sum_{i=1}^n \beta_{in}^x)} \right] \\ &\left[\frac{\Gamma(\sum_{i=1}^n (\beta_{i1}^x + h_{i1}^x))}{\prod_{i=1}^n \Gamma(\beta_{i1}^x + h_{i1}^x)} \dots \frac{\Gamma(\sum_{i=1}^n (\beta_{in}^x + h_{in}^x))}{\prod_{i=1}^n \Gamma(\beta_{in}^x + h_{in}^x)} \right]. \end{aligned}$$

Now, if we write

$$p(D_t | M_1, B_x) = p_t^{1u},$$

we can see that, if at iteration $t+1$ the bot performed a_j and the user B_x performed b_i , the new model probability is updated to

$$p_{t+1}^{1u} = p_t^{1u} \times \frac{\beta_{(n+1)j}^{ut}}{\beta_{ij}^{ut}}$$

- M_2 . We have

$$\begin{aligned} p(D_t | M_2, B_x) &= \\ &= \int \dots \int \left(\prod_{i,j,k} p_{ijk}^{h_{ijk}^x} \right) k' \left(\prod_{i,j,k} p_{ijk}^{\beta_{ijk}^x - 1} \right) dp_{ijk} \end{aligned}$$

where k' is the appropriate normalisation constant. Simple computations lead to

$$\begin{aligned} p(D_t | M_2, B_x) &= \\ &= \left[\prod_{i=1}^n \frac{\Gamma(\beta_{i11}^x)}{\Gamma(\beta_{i11}^x + h_{i11}^x)} \dots \prod_{i=1}^n \frac{\Gamma(\beta_{inn}^x)}{\Gamma(\beta_{inn}^x + h_{inn}^x)} \right] \\ &\left[\frac{\Gamma(\sum_{i=1}^n (\beta_{i11}^x + h_{i11}^x))}{\Gamma(\sum_{i=1}^n \beta_{i11}^x)} \dots \frac{\Gamma(\sum_{i=1}^n (\beta_{inn}^x + h_{inn}^x))}{\Gamma(\sum_{i=1}^n \beta_{inn}^x)} \right]. \end{aligned}$$

Again, we may write the result recursively as follows. If we designate

$$p(D_t | M_2, B_x) = p_t^{2u},$$

then

$$p_{t+1}^{2u} = p_t^{2u} \times \frac{\beta_{(n+1)jk}^{ut}}{\beta_{ijk}^{ut}},$$

assuming that, at iteration $(t+1)$, the user B_x performed b_k , after having performed b_i and b_j .

4) *Modeling $p(e_t | b_t, e_{t-1}, e_{t-2})$* : We describe now the environment model. For illustrative purposes, we shall actually consider four environmental variables, $e_t = (e_t^1, e_t^2, e_t^3, e_t^4)$, so that:

- e_t^1 , refers to energy level at time t .
- e_t^2 , refers to temperature at time t .
- e_t^3 , refers to inclination at time t .
- e_t^4 , refers to the detection system at time t .

We assume conditional independence for the four environmental variables, so that

$$p(e_t | b_t, e_{t-1}, e_{t-2}) = \prod_{i=1}^4 p(e_t^i | b_t, e_{t-1}^i, e_{t-2}^i).$$

We describe now the models for each environmental variable.

a) **Energy level model**: We shall assume that $p(e_t^1 | b_t, e_{t-1}^1, e_{t-2}^1) = p(e_t^1 | b_t, e_{t-1}^1)$. We just need to know the current energy level and the action of the users (whether they just plugged in or not the bot) to forecast the energy level, or whether the bot is on charge or not. Indeed, we shall assume that

- If $b_t \neq b_1 = \text{recharge}$ and the wire is unplugged, $e_t^1 = e_{t-1}^1 - k_1 \Delta t$, where k_1 is the energy consumption rate.
- If $b_t = b_1 = \text{recharge}$ or the wire is plugged in, $e_t^1 = e_{t-1}^1 + k_2 \Delta t$, where k_2 is the energy recharging rate.

b) **Temperature model**: We shall assume that $p(e_t^2 | b_t, e_{t-1}^2, e_{t-2}^2) = p(e_t^2 | e_{t-1}^2, e_{t-2}^2)$, as we are not able to detect the user's actions concerning temperature changes. We shall assume a simple model such as $e_t^2 = e_{t-1}^2 + (e_{t-1}^2 - e_{t-2}^2) \Delta t$.

c) **Inclination model**: We shall assume the generic model $p(e_t^3 | b_t, e_{t-1}^3)$, being $b_t = \text{attack}$, the relevant user action. The inclination sensor detects only whether (1) or not (0) the bot is in vertical position. Then, we use the evolution matrix shown in Table 1, where, depending on whether the bot was in vertical position or not (e_{t-1}^3) and whether the bot inferred the user action *attack* or another, it will predict the next value of the inclination sensor.

e_{t-1}^3	Attack	Not attack
0	0	0
1	0	1

TABLE I
EVOLUTION OF BEING IN VERTICAL POSITION

d) **Detection system model**: We shall assume generic model $p(e_t^4 | b_t, e_{t-1}^4)$, with $b_t \in \text{interacting actions subgroup}$, the relevant user actions. The detection system shows only whether (1) or not (0) the bot identifies the user's presence. Then, we adopt the evolution matrix shown in Table 2. In this case, p_1 , the probability of detect the user presence when b_t is not in the interacting actions subgroup, follows a *Beta - Binomial* model, see [12],

$$p_1 | \text{data} \sim \text{Beta}(\alpha_1 + x_1, \beta_1 + n_1 - x_1),$$

e_{t-1}^4	$b_t \in \text{interacting actions subgroup}$	$b_t \notin \text{interacting actions subgroup}$
0	1	0
1	1	p_1

TABLE II
EVOLUTION OF THE DETECTION SYSTEM

being n_1 the number of occurrences and x_1 , those where the user has not been detected. If necessary, it may be summarized through

$$\hat{p}_1 = \frac{\alpha_1 + x_1}{\alpha_1 + \beta_1 + n_1}.$$

5) *User Identification $p(B_x)$* : This is based on standard face recognition models using OpenCV libraries, see [6]. We assume that the user is that which maximizes $p(B_x | D_t)$, after obtaining an image of the face of the participant.

C. Multiobjective Preference Model

1) *Basic preference structure*: The bot aims at satisfying five objectives, which, as in [9], are ordered in hierarchical order of importance. They are:

- A primary objective concerning being properly charged.
- A secondary objective concerning being secure.
- A third objective concerning being taken into account by the users.
- A fourth objective concerning being accepted by the users.
- A fifth objective concerning being useful to society at large.

The hierarchy entails that once the bot has attained a sufficient value in a lower level objective it may devote resources to higher level objectives. As an example, until the energy level is not sufficiently high, the bot will tend to stress actions favoring being charged. This is reflected in the weights of the objective functions, on one hand, and the shape of the component utility functions. We describe here the first two objectives.

2) Basic Objectives:

a) *Energy*: The most basic objective pays attention only to the energy level. The bot aims at having a sufficient energy level to perform its activities. A very low energy level is perceived as bad by the bot. A sufficiently high energy level is good for the bot. We represent it through

$$u_1(ene) = \begin{cases} 0, & \text{if } ene \leq lth \\ 1, & \text{if } ene \geq uth \\ \left(\frac{ene - lth}{uth - lth} \right), & \text{otherwise,} \end{cases}$$

with $uth = 0.5$ and $lth = 0.1$

b) *Security*: The second objective refers to security. It essentially takes into account whether the bot is being attacked by any user and it is at an appropriate functioning temperature. It is represented through

$$u_2(\text{attack}, \text{temp}) = w_{21} \times u_{21}(\text{attack}) + w_{22} \times u_{22}(\text{temp}),$$

with $w_{2i} \geq 0$, $\sum_{i=1}^2 w_{2i} = 1$, and weights ordered in importance as follows: $w_{21} > w_{22}$.

The component utility functions are

$$u_{21}(attack) = \begin{cases} 0, & \text{if no attack happened} \\ 1, & \text{otherwise,} \end{cases}$$

$$u_{22}(temp) = \begin{cases} 0, & \text{if } temp < lth \text{ or } temp > uth \\ 1, & \text{if } ltcth < temp \text{ or } < utcth \\ 1 - \left(\frac{(ltcth - temp)}{ltcth} \right), & \text{if } temp < ltcth \\ \left(\frac{uth - temp}{uth - utcth} \right), & \text{if } temp > utcth, \end{cases}$$

with $lth = 0^\circ \text{ C}$, $uth = 35^\circ \text{ C}$, $ltcth$ (lower thermal comfort) = 20° C and $utcth$ (upper thermal comfort) = 25° C .

c) *Global utility function*: Based just on the two lowest level objectives, the global utility function would be

$$w_1 \times u_1(ene) + w_2 \times u_2(attack, temp),$$

with $w_1 \gg w_2 > 0$, reflecting the hierarchical nature of the objectives, and $w_1 + w_2 = 1$.

D. Optimising expected utility

We have implemented a first version of the model in an edutainment bot AISoy1. We started our implementation developing a simulator to prove whether our model worked properly, obtaining coherent results. As a prototype version, we just wanted to prove if it could work with a few amount of user’s and bot’s actions, five and six respectively. In Fig. 3, we find two text boxes accompanied with a screenshot. They are data obtained from the agent through several iterations. The right one shows some sensors’ readings and the actions inferred. On the left side of the figure, we can observe how the agent modifies his behavior depending on being more or less reactive to the user’s actions. The model is implemented in

different timings for different sensors). When relevant events are detected, the basic information processing and decision making loop is shot. However, it is managed by exception in that if exceptions to standard behavior occur, the loop is open to interventions through various threads. We plan only one step ahead and choose the action with probabilities proportional to the computed expected utilities. Memory is limited to the two previous instants.

V. DISCUSSION

We have described a model to control the behavior of an agent in front of several intelligent adversaries. It is multi-attribute decision analytic at its core, but it incorporates forecasting models of the adversaries (Adversarial Risk Analysis). This was motivated by our interest in improving the user's experience interacting with a bot, [2] or [8]. Moreover, we believe though that this model may find many other potential applications in fields like interface design, e-learning, entertainment or therapeutical devices. The model should be extended to a case in which the agent includes some emotional framework to complement his decision making process. It could also be extended to a case in which there are several agents, possibly cooperating or competing, depending on their emotional state. Dealing with the possibility of learning about new user actions, based on repeated readings, and, consequently, augmenting the set $\mathcal{B}$ is another challenging problem. Finally, we have shown what is termed a 0-level ARA analysis. We could try to undertake higher ARA levels in modeling the performance of adversaries.

ACKNOWLEDGMENTS

Research supported by grants from the MICINN project RIESGOS, the RIESGOS-CM project and the INNPACTO project HAUS. We are grateful to discussion with Diego García, from AISoy Robotics, Jesus Ríos and David Banks.

REFERENCES

Fig. 3. A screenshot of our simulator

an asynchronous mode. Sensors are read at fixed times (with

- [1] D. Banks, F. Petralia and S. Wang, *Adversarial risk analysis: Borel games*. Applied Stochastic Models in Business and Industry, **27**: 72-86, 2011.
- [2] C. Breazeal, *Designing Sociable Robots*. The MIT Press, 2002.
- [3] R. T. Clemen and T. Reilly, *Making Hard Decisions with Decision Tools*. Duxbury: Pacific Grove, CA, 2004.
- [4] S. French and D. Ríos Insua, *Statistical Decision Theory*. Kendall, 2000.
- [5] J. C. Harsanyi. 2004. *Games with Incomplete Information Played by "Bayesian" Players*. Management Science, **50**: 1804-1817, 2004.
- [6] R. Hewitt, *Seeing With OpenCV, Part 4: Face Recognition With Eigenface*. SERVO Magazine, April 2007. Retrieved September 16, 2010.
- [7] J. Hoeting, D. Madigan, A. Raftery and C. Volinsky, *Bayesian model averaging: A tutorial*. Statistical Science, **4**, 382-417, 1999.
- [8] R. Kirby, J. Forlizzi and R. Simmons, *Affective social robots*. Robotics and Autonomous Systems, **58** 3:322-332, 2010.
- [9] A. H. Maslow, *A theory of human motivation*. Psychological Review, **50**, 4, 370-96, 1943.
- [10] J. Ríos and D. Ríos Insua, *Adversarial Risk Analysis: Applications to Counterterrorism Modeling*. Risk Analysis, (to appear), 2012.
- [11] D. Ríos Insua, J. Ríos and D. Banks, *Adversarial risk analysis*. Journal of the American Statistical Association, **104**(486):841-854, 2009.
- [12] D. Ríos Insua, F. Ruggeri and M. Wiper, *Bayesian Analysis of Stochastic Process Models*. Wiley, 2012.
- [13] D. von Winterfeldt and W. Edwards, *Decision Analysis and Behavioral Research*. New York: Cambridge University Press, 1986.