An opportunist action selection mechanism which has taste [1]

Tony Dujardin
Philippe Mathieu
Jean-Christophe Routier

LIFL - CNRS UMR8022
Universite des Sciences et Technologies de Lille
59655 Villeneuve d'Ascq Cédex - FRANCE
E-mail: {dujardin,mathieu,routier}@lifl.fr

Abstract

We propose an action selection mechanism (ASM) for situated and cognitive multi-agent systems authorizing a large variety of behaviours. Our action selection allows, first, the management of several goals with dynamic priorities. Second, it enables the possibility for the agent to be temporarily diverted from its main goal in order to achieve a secondary goal promoted by the environment. Third, it directs the choice of the runnable action according to the preferences of the agent. In that purpose, our action selection mechanism takes into account five factors: dynamic priority goals, opportunism, agent taste, multi-goal revalorisation and inertia.

1 Introduction

Our research context is the simulation of cognitive agents that are geographically situated in their environment and are able to achieve goals according to their knowledge. These agents have therefore to compute and to execute their plans in their environments. We consider that agents classically represent their plans by a tree. This context is adapted to video games and particularly to non-player-characters, whose behaviour must be believable in order to not harm the game recreation. We are not interested here in planning tree construction, we suppose that the root is a goal to solve and leaves are runnable actions (whose conditions are satisfied). Our proposal is not that our agents solve their goals in an optimal way (with respect to the number of actions for example), but that their resolution is realistic and coherent with respect to the desired agent behaviour.

Realistic behaviours simulation [3] brings several problems, among which are : knowledge representation, planning that aims to solve agent goals (in the form of plans built from agent knowledge) and the design of an action selection mechanism (or ASM for short), which chooses the action that the agent will carry out among the runnable actions proposed by its plans. Our proposal concerns this last point. An ASM makes it possible by the choice of the action to carry out different more or less realistic behaviours. These behaviours are parametrized by five factors, which are goal priorities, environment influences (including opportunism), agent taste, multi-goal revalorisation and inertia. Each factor enables to control a well defined part of the agent behaviour, allowing to have an easiest understanding of parameters than in other action selection mechanisms (like number of hidden layers or number of neurons by layer in a neuron network).

Goal priorities make it possible to choose the action to be carried out according to the importance of its goal. **Environment influence** allows an agent to be temporarily diverted from its current goal, in order to carry out an action promoted by the environment but corresponding to a lesser priority goal. **Agent taste** is a factor independent from a given simulation. It takes into account the preferences that the agent gives to actions. Thus, an agent is able to choose

[1]This work is cofinanced by the CPER TAC Nord-Pas de Calais and the European funds of FEDER.

between two actions that would not be distinguishable through goals priorities or the environment. **Multi-goal revalorisation** is used to promote actions that makes progress more quickly towards the goal achievement. **Inertia** allows to obtain contiguous action sequence execution.

In section 2, we define the basic concepts used in this article. In section 3, we present our action selection mechanism taking into account the tree factors mentioned above. In section 4, we compare our proposition to related works, before concluding.

2 Preliminary definitions

We define here some elements of the planning trees which is used to represent agent beliefs and intentions.Each **action** has a set of conditions necessary to its execution and a set of effects resulting from its execution. For example,

Action:	to open the door.
Conditions:	the agent has the key, the door is locked.
Effects:	the door is open.

The planning tree used is an **And/Or tree**, which is a tree structure alternating **And-node** and **Or-node**. An And-node is labeled by an action, its children are the conditions that must all (AND) be solved for the action to be runnable. An Or-node is labeled by a condition, it proposes the alternative (OR) of its children which are the actions that allows to satisfy this condition (a "sub-goal"). Therefore, tree leaves are the actions that are runnable (their conditions are satisfied) and are called **runnable action** ($Run.Act_a$).

$$\forall a \in Agent, \forall g \in Goal_a, \quad Run.Act_a : \begin{array}{ccc} Goal_a & \rightarrow & Leaf \\ g & \mapsto & x \end{array}$$

$$Run.Act_a(g) \in \{x \mid x = Leaf(Plan_a(g))\}$$

The And/Or tree root is an **agent goal** which is composed by a set of conditions that must be satisfied. For each goal g of the agent a corresponds a planning tree (noted $Plan_a(g)$).

$$\forall a \in Agent, \forall g \in Goal_a, \quad Plan_a : \begin{array}{ccc} Goal_a & \rightarrow & Tree \\ g & \mapsto & Plan_a(g) \end{array}$$

We call **alternative**, an action sequence that enables a goal achievement. Within an **And/Or-tree**, an alternative is composed by all children from **And-nodes** and one choice among all the children (actions) from each **Or-node**. Two alternatives are equivalent, if they are built for the same goal, and the same leaf and if they contain the same actions in different order. In our ASM, we do not distinguish two equivalent alternatives.

$$Alternative_a : \begin{array}{cccc} (Tree & \times & Run.Act_a) & \rightarrow & \{Action\}^* \\ (Plan_a(g) & \times & leaf) & \mapsto & \{x\}^* \end{array}$$

Each planning tree contains all alternatives (different continuations of possible actions) that can be used to solve the goal. Once planning trees are built, the action selection mechanism must choose the action to be carried out among all the runnable actions.

$$\forall a \in Agent, \forall g \in Goal_a, \quad Selection : \begin{array}{ccc} Agent & \rightarrow & Action \\ a & \mapsto & x \in Run.Act_a(g) \end{array}$$

To select an action amounts to select one of the alternative (with this action as leaf) and to promote it. This choice can be called into question at each selection. The ASM values each runnable action (by ϕ_a) and retains the one with the greatest value.

$$\forall a \in Agent, \forall g \in Goal_a, \quad \phi_a : \begin{array}{ccc} Run.Act_a(g) & \rightarrow & \mathbb{R} \\ x & \mapsto & ActionValue \end{array}$$

Next action : $x \in Run.Act_a(g) \mid \phi_a(x) = \max_{y \in Run.Act_a(g)}\{\phi_a(y)\}$

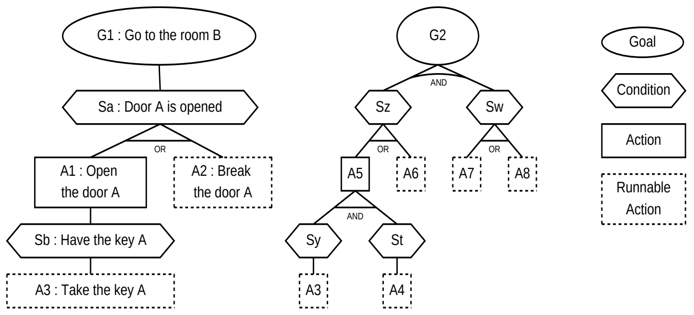

Figure 1: Two planning trees, the first one relates to the goal $G1$ (action names and conditions are given to show a concrete And/Or tree example), the second one relates to the goal $G2$. $G1$ and $G2$ goals have respectively the priorities $goal_a(G1)$ and $goal_a(G2)$. If $goal_a(G1) < goal_a(G2)$ runnable action priorities {A2, A3} are less than the one of runnable actions {A3, A4, A6, A7, A8}. A3 is present as runnable action in two goals, it will receive the priority given by the goal with greater one, here $G2$. Up to now, we do not have other criteria to distinguish actions from the same goal. Thus the evaluation of a runnable action corresponds to the priority of the goal : $\phi_a(A2) = goal_a(G1)$ and $\phi_a(A3) = \phi_a(A4) = \phi_a(A6) = \phi_a(A7) = \phi_a(A8) = goal_a(G2)$.

Classically, this value corresponds to the action evaluation according to the ASM factors. Thus, the evaluation function is the way to obtain various behaviours. Choice of factors taken into account to compute ϕ_a is thus of primary importance to obtain an action selection mechanism that allows different and realistic behaviours.

3 Our action selection mechanism

Our action selection mechanism combines five factors : goal priorities, environment influence, agent taste, multi-goal revalorization and inertia. In this part we incrementally detail our action selection mechanism. Firstly, we take into account only goals with dynamic priorities (noted *goal*). Secondly, we will add environment influence (with opportunism noted *opp* and achievement noted *ach*). Thirdly, we take into account agent taste (noted *taste*). Finally, we present multi-goal revalorization and inertia (noted *inert*). We also present the way of combining these factors in order to obtain variable and realistic behaviours. This combination, for an agent a gives the value (ϕ) of each runnable action x, by the formula : $\phi(x) = goal_a(x) * opp_a(x) * ach_a(x) * taste_a(x) * inert_a(x)$.

3.1 Goal influence

The importance of a given goal is intrinsically variable. A goal allowing the agent survival has certainly a higher priority than others. Moreover, goal importance is sometimes related to one parameter, for example the weariness : The higher is my energy, the less tired I am. In order to realize this importance, we assign to each goal a priority denoted *goal*. It is characterized by a function which can be a constant function for a static priority or variable for an evolving priority.

$$\forall a \in Agent, Goal_a : \begin{array}{ccc} Goal_a & \to & \mathbb{R} \\ g & \mapsto & GoalValue \end{array} \left| \begin{array}{l} goal_a(g) : \forall a \in Agent, \forall g \in Goal_a, \\ 0 \leq goal_a(g) \leq MAX_a, \end{array} \right.$$

$$\forall x \in Run.Act_a(g), \phi_a^1(x) = goal_a(g).$$

This priority influences the ASM which will favor actions whose root goals have the most important priority (see FIG.1). Since the goal priority value can evolve during time, the choice of the goals to be realized is then called into question. This allows to take into account new goals or the dynamics of existing goals and can lead to a modification in the order of goal realization. However this influence does not take into account the environment. Thus an agent adopts a behaviour close to the video game Sims characters, where goals are given to characters by the user and are organized in a queue (FIFO). To illustrate the need of taking the environment into account, we will treat a well-known example in this game : the newspaper and the mail. Let us consider a Sims character

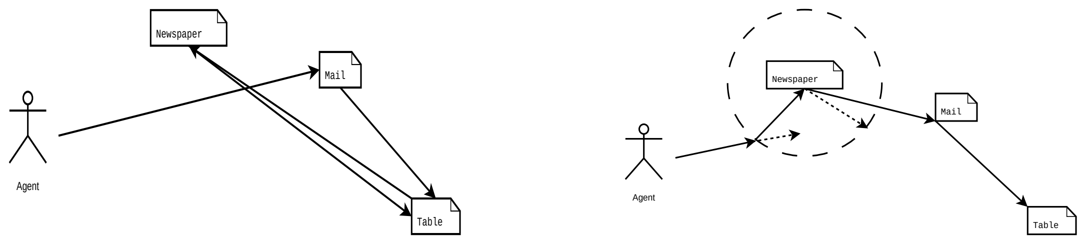

Figure 2: Influence of opportunism on action selection. In this example, the agent a has two goals, G_p the goal to put the newspaper down on the table, G_m the goal to put mail down on the table, with $goal_a(G_p) < goal_a(G_m)$. To achieve these goals the agent must take these objects (actions T_p and T_m) and then put them down on the table (actions D_p and D_m). **On the left,** ASM does not take into account opportunism ($\phi_a(D_m) = \phi_a(T_m) = oal_a(G_m)$ and $\phi_a(D_p) = \phi_a(T_p) = goal_a(G_p)$) thus agent carries out actions in this order : T_m, D_m, T_p and D_p, this execution is very expensive in moves. **On the right,** our ASM taking into account environment : at the beginning, the computed action sequence is the same (T_m, D_m, T_p and D_p). But while passing close to the newspaper (at distance lower than $ThresholdOpp_a$ symbolized by a dashed circle) the agent is diverted from mail in order to take the newspaper *by opportunism*, the proposed action sequence becomes T_p, T_m, D_m, and D_p. Once the newspaper is taken the agent will not execute D_p, because the action T_m is the most favored. The agent thus ends up taking the mail and putting them down on the table carrying out the sequence (T_p, T_m, D_m and D_p).

that has two goals: to take its mail and newspaper outside the house, considering the mail being of greater importance than the newspaper. In this case, the Sims behaviour is very expensive in moves (mail-house-newspaper-house) and not very realistic. A more believable behaviour would have been to take both the mail and newspaper. To obtain this kind of behaviour, it is necessary to allow the ASM to take into account the environment and to carry out action with weaker priorities at low cost. For that purpose, we set up environment influence on agent behaviour.

3.2 Environment influence : opportunism

To be temporarily diverted from a goal (to put the mail) in order to realize a closer goal (to take the newspaper), an agent must take into account its environment. Because our agents are situated, they can evaluate distances between them and environment objects. Thus, it is possible to take into account whether or not the target of an action is close to the agent. Therefore the ASM can be modified in order to favor an action because its target is nearby, even if this temporarily diverts the resolution of the highest goal priority. Environment influence on agent behaviour a is called *opportunism*. We denote it opp_a.

$$\forall a \in Agent, opp_a : \begin{array}{rcl} Action & \rightarrow & \mathbb{R} \\ x & \mapsto & OppValue \end{array} \left| \; target : \begin{array}{rcl} Action & \rightarrow & Agent \\ x & \mapsto & t \end{array} \right.$$

$ThresholdOpp_a$ is the agent a opportunism threshold,

$d_a(target(x))$ is the distance between agent a and target (t) of action x.

$opp_a(x) = \max(1, \frac{ThresholdOpp_a}{d_a(target(x))})$,

$\forall a \in Agent, \forall g \in Goal_a, \forall x \in Run.Act_a(g), \phi_a^2(x) = \phi_a^1(x) * opp_a(x) = goal_a(g) * opp_a(x)$

Remarks. According to $ThresholdOpp_a$ value, we can obtain agents which express more or less opportunism. Our choice to use the product to combine $goal_a$ and opp_a allows to obtain :

- If $d_a(target((x))) \geq ThresholdOpp_a$, opp_a value is 1, with the product, opportunism has no influence (see FIG. 2 on the left).
- If $d_a(target(x)) < ThresholdOpp_a$, opp_a value is higher than 1 and increases with the reduction of the distance agent-target, thus increasing the influence of opportunism (see FIG. 2 on the right).

3.3 Environment influence : achievement

Our agent chooses the actions to be carried out according to goal priority and the distance between agent and action target. We have introduced a second environment influence, which we called *achievement* feature. We denote it ach_a for agent a. This parameter promotes alternatives issued from goals which can be quickly accomplished. Thus, like opportunism, achievement will push the agent to be diverted from a goal to achieve actions of another goal which can be solved in few actions. The more an agent has an important value for this feature, the more it will estimate that the fact of finishing a goal is important rather than considering another goal. Even if this one is intrinsically more important but needs more actions to be achieved. For example, for the goal G1 in the figure FIG.1, the agent will favor the action A2 which achieves the goal rather than the action A3 that needs two actions to achieve the goal (with $ThresholdAch_a \leq 2$).

$$\forall a \in Agent, \forall g \in Goal_a, \quad ach_a : \begin{array}{ccc} Action & \rightarrow & \mathbb{R} \\ x & \mapsto & AchValue \end{array}$$

$ThresholdAch_a$ is the threshold of the agent's feature of achievement a,

$nbA_a(x)$ is the number of actions (displacement included) of action x alternative.

$\forall x \in Run.Act_a(g), nbA_a(x) = \|Alternative_a(Plan_a(g), x)\|$

$ach_a(x) = \max(1, \frac{ThresholdAch_a}{nbA_a(x)})$

$\forall x \in Run.Act_a(g), \phi_a^3(x) = \phi_a^2(x) * ach_a(x) = goal_a(g) * opp_a(x) * ach_a(x)$

Remarks. $ThresholdAch_a$ value makes it possible to refine agent achievement feature. This feature is different from opportunism because it is based on the number of actions (moves included) to carry out between the runnable action and the goal. It does not consider the distance between agent and action target. Our choice to use the product to combine $goal_a$, opp_a and ach_a allows to have an influence only when the number of actions is lower than the threshold and whose importance increases according to the reduction of the number of actions in the alternative.

Thus, at this point our action selection mechanism takes into account goal priorities and environment influences (considering both the distance between agent and target, and goal achievement), but does not allow to distinguish different actions in the same environment. Let us take for example, an agent that must go in a room closed by a door, it has two possibilities of action: to `open` or to `break` the door. It is thus necessary to set up a third factor making it possible to distinguish two alternatives depending on actions they are built of. We call this factor, the agent taste.

3.4 Agent taste

Each agent has its own preferences on actions which are simulation undepend. These preferences are values attributed to the actions which make it possible to express that an agent can have a repulsion towards an action. This **repulsion** must depreciate the alternative where the action is. An agent can also like to make an action, this **attraction** must favor alternatives where this action is. Finally, an agent may never want to perform an action, this **inhibition** must be taken into account like an inhibition of the alternative (an absolute repulsion). Agent taste (denoted *taste*) represents the impact of preferences on alternative choice (and thus selection of its runnable action). To value an alternative, we are interested in action set belonging to this alternative. It is only the presence within the alternative that is important, not its position within the alternative.

To compose agent taste (*taste*) with influences from goals and environment, we must select an interval for preference value on the actions in order to obtain the *taste* value. After having compared several compositions on various intervals, we choose to retain the product in the interval $[0, N]$. This satisfied our expectations on absence of influence ($taste = 1$), attraction ($taste \in]1, N]$), repulsion ($taste \in]0, 1[$) and inhibition (by taking $taste = 0$, in this case $\phi_a = 0$). This choice allows us to write that:

$\forall x \in Run.Act_a(g), \phi_a^4(x) = \phi_a^3(x) * taste_a(x) = goal_a(g) * opp_a(x) * ach_a(x) * taste_a(x)$

The *taste* value represents the combination of action preferences within an alternative. We studied four possible combinations (allowing the realization of our definition of the agent taste) in order to obtain *taste*. These four combinations are the **minimum**, the **N minima mean**, the **geometric mean** and the **harmonic mean**. To calculate the *taste* values within the And/Or tree built by the agent, we must first find all alternatives contained in this tree.

In an And/Or tree for a goal and a leaf, we have several equivalent alternatives. Taste value is obtained by actions contained in an alternative without taking into account their order. In addition, the higher the taste value of an alternative is, the higher value of the runnable action will be.

So we can reduce the alternative search, by pruning our course. Thus, we start (given a goal) with a leaf and then systematically consider better alternatives for the agent plan. The selection between each alternative is made according to the composition that has obtained the highest taste value. Since all suggested composition functions satisfy the following additive relation: $\max(Composition(B \cup C \cup D), Composition(B \cup F \cup G)) = note(B) + \max(Composition(C \cup D), Composition(F \cup G))$.

Taste represents the agent taste value combining all actions within the action x alternative,

$$\begin{array}{llll} taste: & Run.Act_a & \rightarrow & \mathbb{R} \\ & x & \mapsto & Comb(Alternative_a(x)) \end{array}$$

Comb is a function used to combine different node preferences.

$$\begin{array}{llll} Comb: & \{\text{Actions}\}^* & \rightarrow & \mathbb{R} \\ & \{x\}^* & \mapsto & TasteValue \end{array}$$

In our approach, we can notice that factors taken into account by the ASM to solve a goal, are represented by the totality of the goal planning tree. Indeed, the agent taste stays in each Action-node, the goal influence is present at the roots, the achievement is built according to the alternative size and opportunism is calculated from the leaves (runnable actions).

3.5 Multi-goal revalorization and Inertia

Agent behaviour are shaped by goal and environment influences, and tastes associated to each action. These various factors can be used in many different kind of simulations. Nevertheless, to obtain a more realistic behaviour, we decide to add two other factors that are multi-goal revalorization and inertia.

Among agent planning trees, it is possible that an action appears several times as runnable action (example of A3 in figure 1). This action thus makes one or more goals "progress" more quickly, this is why it must be favored by the **multi-goal revalorization**. The principle is to combine the values received by all occurrences of an action x in order to obtain a reinforced value. We realize this with the probabilistic sum of these values after having normalized them.

$ThresholdAch_a$: the upper limit for ach_a
$ThresholdOpp_a$: the upper limit for opp_a
$Limit = MAX_a * ThresholdOpp_a * ThresholdAch_a * N_a$

MAX_a: the upper limit for $goal_a$
N_a: the upper limit for $taste_a$

$$\forall a \in Agent, \forall g \in Goal_a, \forall x \in Run.Act_a(g), E(x) = \{x' | \exists g' \in Goal_a, x' \in Run.Act_a(g'), x = x'\}$$

$$\phi_a^5(x) = Limit * PS_{E(x)}(\frac{\phi^4(x)}{Limit}).$$

Where $E(x)$ represents all runnable actions similar to x and $PS_{E(x)}$ denotes the probabilistic sum of values in $E(x)$ and $\frac{\phi^4(x)}{Limit}$ performs normalization.

By using probabilistic sum, we obtain : $\forall y \in E(x),\ Limit \geq (\phi_a^5(x) = \phi_a^5(y)) \geq \phi_a^4(y)$.

In order to avoid oscillation phenomena between various alternatives and thus to ensure a certain persistency in behaviour, the value $\phi_a(x)$ of the action x selected at the moment T will be reinforced by a percentage, noted $inert_a$ corresponding to the action **inertia** for the agent a. Thus, for an action y to be chosen at step $T + 1$ instead of the action x, $\phi_a(y)$ will have to be higher than $\phi_a(x) * inert_a(x)$. If it is the case, the value $\phi_a(y)$ will be also developed by same percentage $inert_a$. Once the action carried out, we reflect inertia on the action father in order to persist with the same alternative. Finally, we obtain $\forall x \in Run.Act_a(g), \phi_a(x) = \phi_a^5(x) * inert_a(x)$.

PECS	Our proposal
Determines the new values of the internal state variables.	Goal priorities are dynamic
Calculates the corresponding intensity of each motive.	Calculates the intensity of influences
Compares the various competing motives and select the one with the highest intensity as the action guiding one.	Selects the action having the best value ϕ
Performs the action which is demanded by the action guided motive.	Performs the selected action

Figure 3: Comparison between the approach PECS and our proposal

Tyrrell criteria	Our proposal
Persistence	The inertia of the action in progress until its achievement, including during a move necessary to its execution.
Activations proportional to current offsets	the dynamic goal influence can be used like homeostatic variables.
Balanced competition	No difference between runnable actions completing only one goal, but revalorization of the multi-goal runnable actions.
Contiguous action sequences	Inertia is transmitted to the father of the carried out action, allowing persistence on the alternative.
Interrupts if necessary	Our inertia allows the interruptibility. Environment influences can develop actions, making it possible to stop current alternative.
Opportunism	Same definition as our opportunism.
Combination of preferences	Taking into account all alternative actions and goal influence in the selection among runnable actions.
Flexible combination of stimuli	Action Preferences, several possible function to combine influences and the taste.
Compromise candidates	Multi-goal revalorisation.

Figure 4: Tyrrell's criteria fulfilled by our approach

4 Related works

Action notation is an important problem, many research works [8, 2] were made in this field. Among them we can quote Pattie Maes [7] and Alejandro Guerra-Hernández [5]. They build a network connecting actions according to preconditions and action effects. These networks operate similarly to artificial neural networks, where each neuron transmit energy to its successors, starting from a threshold value of energy received and the reception of negative energy called inhibition. Other approaches containing rules as SOAR such [6] obtain behaviours allowing the optimal resolutions of the agent goals. These behaviours are given mainly by these rules which can evolve (for example by training phases). In SOAR, the action selection is realized by operators. An operator is an abstract type and its concretization remains to be realized and adapted according to the desired behaviour. We propose a concrete action selection mechanism which allows obtaining different behaviours. Thus agents having the same planning trees can solve their goals in different ways. For that, our agents have preferences over each action, these preferences constitute the agent taste. From these preferences (simulation independent) and their combinations with influences (related to simulation), our ASM allows this differentiation of behaviours while keeping coherence in the choice of the action to be carried out. In this direction, we meet the work of Bernd Schmidt [9] on the model PECS (Physical conditions Emotional state Cognitive capabilities Social status) which functions in four stages (see FIG.3). Nevertheless, we bring in addition to the concept of influence, the concept of taste which makes it possible to have preferences, repulsions or inhibitions on certain actions. Moreover an ASM must allow, at the same time opportunism while keeping a certain persistence (to avoid oscillations between the actions), like the proposal of Blumberg [1]. Toby Tyrrell [10] describes the problems which must be taken into account to evaluate a good action selection mechanism. We satisfy the main criteria (see FIG.4).

5 Conclusion and prospects

A realistic simulation requires to respect a wholeset of constraints [10] allowing a great variability of behaviours. We propose a concrete action selection mechanism allowing to take into account these constraints in a cognitive approach. For that in a system where agents are cognitive and situated, we use an action selection mechanism based on five main factors: goal influence, environment influence, agent taste, multi-goal revalorization and inertia. Each factor enables to control a well defined part of the agent behaviour, allowing to have a obvious comprehension of these parameters. These factors allow the management of several dynamic goal priorities, temporary diversion of the alternative execution solving a goal in order to carry out another action developed by the environment or the agent taste. Moreover at each step the choice is questioned.

Currently, we experienced our proposal on the CoCoA platform [4] in order to validate it. The basic modules of knowledge, planning and action selection mechanisms are currently taken into account for action notation. The perception module is the only one which does not influence the action value, taking this into account would make it possible to obtain an new influence type: " when I see an objective I want more to carry it out than when I don't see it". We could also consider action order in the notation: a very remote action would have a less influence compared to a closer one. Finally, it would be interesting to use our "mono-agent" action selection mechanism, to allow the cooperation between agents. An agent would leave what it must do, in order to help another agent, because its action is less important than the other agent action. For that, it is necessary to be able to evaluate the importance of an action, i.e. to select the most important action among actions that the agent can make and assistance that it can bring to other agents.

References

[1] Bruce Blumberg. Action selection in hamsterdam: Lessons from ethology. In *Proceedings of the 3rd int. conf. on the simulation of Adaptive behaviour*, 1994.

[2] Joanna J. Bryson. Action selection and individuation in agent based modelling. In *Agent 2003: Challenges of Social Simulation*, 2003.

[3] Damien Devigne, Philippe Mathieu, and Jean-Christophe Routier. Interaction-based approach for game agents. In *19th European Conference on Modelling and Simulation(ECMS'05)*, 2005.

[4] Damien Devigne, Philippe Mathieu, and Jean-Christophe Routier. Team of cognitive agents with leader : how to let them acquire autonomy. In *Proceedings of IEEE Symposium on Computational Intelligence and Games(CIG'O5)*, 2005.

[5] Alejandro Guerra-Hernández. Modeling behavior: an action selection approach. In *Workshop on Distributed Simulation, Artificial Intelligence, and Virtual Environments*, 1998.

[6] John E. Laird, Allen Newell, and Paul S. Rosenbloom. Soar: An architecture for general intelligence. *Artificial Intelligence*, pages 1–64, 1987.

[7] Pattie Maes. How to do the right thing. *Connection Science Journal, Special Issue on Hybrid Systems*, 1990.

[8] P. Pirjanian. An overview of system architecture for action selection in mobile robotics, 1997.

[9] Bernd Schmidt. Human factors in complex systems : The modelling of human behaviour. In *19th European Conference on Modelling and Simulation(ECMS'05)*, 2005.

[10] Toby Tyrrell. *Computational Mechanisms for Action Selection.* PhD thesis, University of Edinburgh, 1993.