HOW TO EASILY DESIGN REUSABLE BEHAVIOURS FOR SITUATED CHARACTERS ?

Tony Dujardin, Jean-Christophe Routier
Laboratoire d'Informatique Fondamentale de Lille,
Université des Sciences et Technologies de Lille, France
tony.dujardin@lifl.fr, jean-christophe.routier@lifl.fr

Keywords: behaviour, action selection, situated agent, video games

Abstract: Since the action selection mechanism chooses the next fired action, we claim that it is the place where the character's behaviour is defined. We propose an ASM wich is able to produce believable and reusable behaviours for situated cognitive characters. It is defined as a combination of several motivations. It is modular and robust to evolutions of the environment, hence the designer task of building behaviour is made easier. The design of NPC in MMORPG can particularly derive benefit from this ASM.

1 INTRODUCTION

The construction of human behaviour is a complex and ambitious application field of AI. The very definition of a realistic or human-level behaviour is complex, the most famous answer is certainly Alan Turing's. Reaching such an AI is not yet at hand. However it is possible to consider simplified instances of this problem and to try to tackle them. Games, since they provide a well defined and bounded context, have always been a good AI target. We join authors in (Laird and van Lent, 2000) to consider that video games are the good target, from experimental or application point of views, for research on believable behaviours. In particular the management of so called "Non Player Characters" (NPC) has to be considered. The construction of believable NPC behaviour enhances the playability of video games, as well as the interest of the players since immersion is increased. It has become a real challenge in video games industry. In this domain, there are a lot of related works. Nevertheless, almost of these works try to find an optimal resolution to build behaviours and these built behaviours are fitted to a particular context and game. They must be rebuilt when some elements in the game change or are added. Therefore they are not suited to game in permanent evolution. This is the case of Massively Multiplayer Online Role Playing Game (MMORPG). In these games, development is incrementally done. New features are often added to the game and the NPC behaviour must then be able to adapt and to take these evolutions into account without further software development.

In this paper, we propose an easy-to-design Action Selection Mechanism (written ASM) dedicated to situated believable characters like video-games NPC are. It permits to easily define several different and cognitively plausible behaviours. It does not aim to solve problems in optimal ways. We are mainly interested in the creation of diversified and realistic behaviours. Our ASM wants to produce reusable behaviours, allows characters to be self-sufficient in every compliant environments, and has explicit settings (it does not require training period).

In section 2, we present the context of this work and its needs. In section 3, we present our solution to build situated character behaviours using our model of ASM. We also define a concrete behaviour which can be obtained with our ASM. We instantiate and experiment it in section 4, before concluding.

2 CONTEXT AND GOAL

Our purpose is to define a mechanism to build believable behaviours for situated characters like MMORPG's NPC. We consider the desired properties for this mechanism according to the targeted application, that is video-games and especially MMORPG.

MMORPG corresponds to a model of “always-running” applications. The software development of these applications is incremental and new elements and abilities can be added in the game. The mechanism to design behaviours must be able to adapt to these evolutions without software development. Another work of the application designer is to distinguish characters, not only in their graphical representations but also in their behaviours. In consequence, the mechanism to design behaviour must be reusable, extensible and must allow several different behaviours. We affirm that it is possible to have a mechanism that is robust with respect to evolutions in the game. This mechanism must be built regardless environment and characters abilities. Finally, it must provide several different behaviours that can be reuse in other applications.

The character must be believable. It is situated, then the mechanism must take into account the character’s environment and even whether the character is cognitive, reactive-like behaviour must be possible. Moreover, to resolve its goals, character must avoid to oscillate between several actions (as Tyrell named the “contiguous action sequences” in (Tyrrell, 1993)). In addition, the character must express individuality, this should be illustrated by different action choices between characters if they are faced to the same situation. Moreover a character must be able to express preferences (attraction or reluctance) on actions that it is induced to execute. In opposition to SOAR (Laird et al., 1987) and ACT-R (Anderson et al., 2004), our proposal is not that our characters solve their goals in an optimal way (with respect to the number of actions for example), but we are interested in the building of reusable and varied behaviours with the same engine. Lastly, the character must take into account the others, since it can compete or cooperate with them. This has a social impact on the behaviour.

In summary, the mechanism to design situated character’s behaviour must be defined regardless from the environment and the character’s abilities. It must provide modular, believable and easy to design behaviours. We assert that our ASM can be this mechanism.

3 BUILDING BEHAVIOURS

In (Seth, 1998) the behaviour is defined as a joint product of character, environment and observer. This definition based on the ethology permits to say that the behaviour results from the set of the actions that a character have performed in an environment. Since an action selection mechanism chooses the next fired action, we claim that it is the place where the character’s behaviour is defined. Then our purpose is to propose an ASM that is able to produce believable behaviours for situated cognitive characters.

In this work, we consider cognitive situated characters performing in an environment supplied with an euclidean space, where neighborhood and distance notions have meanings. A set of *abilities* (or effectors) describes the laws that rule the environment. Each character is provided with its own set $\mathcal{A}$ of some of these abilities and thus is able or not to perform a given action on the environment. It results that characters differ. Known abilities can change too, they can be added or removed.

Each character receives goals that it has to solve. To do it, a character is composed of a perception module, a memory, a planning engine and an ASM. The character’s execution cycle is as follows. First, it perceives and collects new informations from its environment and adds these informations to its memory. Second, according to its abilities and the informations coming from its memory, it builds (or modifies) its plans in order to solve its goals. Third, an action selection mechanism selects the “best” action among the runnable actions emphasized by the plan. Last, the character will carry out the selected action. Then the process starts again. We make the assumption that the planning engine produces the plans and we only address, in this paper, the action selection problem.

While building the plans, the engine exhibits the set $\mathcal{A}^R \subset \mathcal{A}$ of *runnable actions* which are the actions present in the plans and that can be immediately executed by the character. Executing such an action should help to solve at least one of the character’s goals. The purpose of the ASM is to select one among these actions. For this, the ASM assigns a value to each action in $\mathcal{A}^R$ and selects the action with the highest value. The value is computed as the combination of several *criteria* or *evaluator* values, each expressing a behavioural feature. Each evaluator e_i is defined by the function γ_{e_i}:

$$\begin{array}{rccl} \gamma_{e_i}: & \mathcal{A}^R & \rightarrow & \mathbb{R} \\ & a & \mapsto & value \end{array}$$

For each action a in $\mathcal{A}^R$ the final note ϕ uses a combinator function *Comb* to aggregate the n evaluators:

$$\begin{array}{rccl} Comb: & \mathbb{R}^n & \rightarrow & \mathbb{R} \\ \phi: & \mathcal{A}^R & \rightarrow & \mathbb{R} \\ & a & \mapsto & Comb(\gamma_{e_1}(a), \cdots, \gamma_{e_n}(a)) \end{array}$$

Hence, the ASM returns the runnable highest valued action α:

$$\alpha \in \mathcal{A}^R \mid \phi(\alpha) = \max_{a \in \mathcal{A}^R}\{\phi(a)\}$$

Even if functions γ_e and *Comb* are the same for all the characters, different behaviours can be obtained and observed.

3.1 Modular behaviour

As mentioned above, the behaviour results from the set of the actions that a character have performed in an environment. These actions have been selected among the character abilities. We can remark that this action selection can depend on some constraints. For instance, *to eat* a character can prefer "cooking frozen food" instead of "going to the restaurant" because it is tired. In fact, the behaviour is induced (in positive or negative ways) by some *motivations*. We adapt the motivational approach by Jacques Ferber (Ferber, 1999) to build our behavioural engine. According to this, we propose to define the behaviour as actions resulting from the set of motivations suffered by the character. For each motivation, we build an evaluator (and its evaluation function γ) which impacts the action selection. All of these evaluators are combined to obtain the final character choice. Then the ASM can be seen as an aggregator of *motivations*. Each evaluator exerts a constraint on the selection and corresponds to a motivation. This ASM is *modular* and *easy to design* : to add (resp. remove) a motivation requires only to add (resp. remove) the related evaluator. Each evaluator is understandable, because it corresponds to a specific motivation. Moreover, whatever the game or the application, a motivation can be described in the same manner and so the evaluator can be *reused*. The advantage is to be able to give an interpretation for each *motivation* and to be able to define the broad lines of the built behaviour. For example to take into account the feature of "being opportunist" in the agent's behaviour, it requires to define an evaluator to express this motivation. Moreover building evaluators from motivations permits to define them *regardless environment* and *abilities*, and to take them into account to obtain a more believable behaviour.

Since this ASM is modular, reusable and impervious to the game evolutions, it is a robust mechanism to easily design situated character behaviours.

3.2 Believable behaviour

Our preoccupation lies only in the production of behaviours. In this point of view, the purpose is not to produce "the optimal" behaviour but rather to obtain several *believable* behaviours. But, what does "*believable*" mean and how to evaluate it? This is of course a difficult, even impossible, question. In order to try to answer it, we will consider the desired properties for the ASM according to the targeted application, that is MMORPG and more precisely consider the gamer's points of view. To be believable, a situated character must be aware of its environment, its inner state and the other characters. Jacques Ferber (Ferber, 1999) proposes a classification of the motivations in four categories according to their origins : the *personal motivation* groups the motivations due to the character itself, the *environmental motivation* is used to make the character reactive to its percepts and its environment, the *relational motivation* relates to the presence of the other characters and the *social motivation* relates to norms or social rules.

Depending on the desired behaviour, the designer can choose to have zero or several evaluators from each motivations.

Notion of alternative For an ASM, to choose an action among runnable actions amounts to select one path among all the possible ways to solve the character's goals. Indeed, considering the character's knowledge and abilities, the planing engine proposes sequences of actions to solve its goal. For a given goal, several sequences can be possible. We name *alternative* such a sequence. It obviously results that several alternatives can exist for a given action node. When the ASM works, it does not only choose an action but actually one (or several) alternative whose first sequence step is the chosen action. This is important since, to be realistic, a character behaviour should not deviate too much from a goal resolution. Oscillations between several alternatives should be avoided. The action selection mechanism must therefore take into account the alternatives in their whole while performing its choice. Thus, we do not want a character to be involved in a goal resolution and then to change because an action in the alternative is inconsistent with its personality. Moreover, the mechanism takes into account the future actions of the character in long-term predicted by the planing engine, and so must consider actions to be executed in the alternative later. It results that the action selection should not focus on choosing the preferred action at each step, but the preferred way to resolve the goals and therefore the preferred alternative (see figure 1).

3.3 Easy to design behaviour

We build an ASM in following three steps. First, we identify desired behaviour motivations. Second, we define a combination function. Latter, according to the chosen combination function, we define for each behaviour motivation an evaluator and its evaluation function γ.

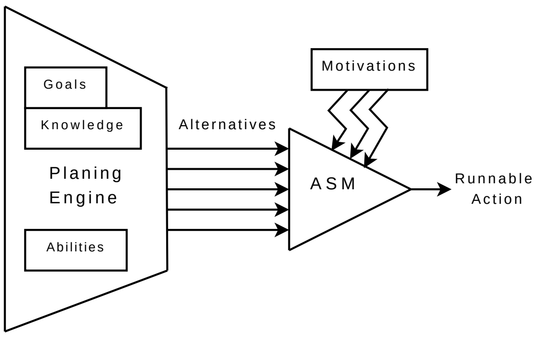

Figure 1: Alternatives are calculated by the planing engine, the best one is chosen by the action selection mechanism. After that, the ASM exhibits the related runnable action.

To achieve an ASM that fits well in obtaining the desired situated character's behaviours, we propose several possible motivations for personal, environmental and social categories of motivations.

Firstly, concerning *personal motivations* :

Goal influence Whose purpose is to take into account that, for a given character, the different goals can have different priorities particular to this character. Goal priority can relate to one inner parameter. This allows to express some behavioural features like, for example, the survival: the lower is the character's energy (an inner parameter value), the more in bad state it is, then the higher the priority of goal "keep the character alive" should be.

Character preferences A character can have several personality traits, for instance it can be "brutal" and "greedy". So this character prefers to break a door and to eat an apple instead of opening the door and practicing some physical exercises. Thus each character has its own preferences on actions. These preferences express how much the character would be inclined to use the action. Then, it should be possible to express neutral feeling, attraction, inhibition or even repulsion for the action. This feature is a mean to express character's personality considering that the personality expresses through the executed actions.

Achievement in time Each action can take more or less time to be executed. This cost can be taken into consideration for promoting the alternative that takes the less time to be resolved.

Multi-goal revalorization If the same runnable action allows to solve several agent goals, then this action makes progress quicker towards the goal achievement and must be favoured.

Inertia When a agent is involved in a goal resolution (in an alternative), inertia expresses the character's trend to continue in the same alternative.

Secondly, considering the *environmental motivations*:

Opportunism Behaviour must benefit from the surroundings. Because the character is situated, it can evaluate distances between it and other entities. Thus, it is possible to take into account whether or not the target of an action is close to the character. Therefore the opportunism evaluator influences the ASM in order to favour an action whose target is nearby. This feature introduces reactive-like behaviours. The main effect of this feature is to bring the character to be temporarily diverted from a goal because of its situation. This feature expresses the opportunist behaviour.

Achievement in space This evaluator promotes alternatives issued from goals that can be accomplished in a few steps, hence the importance of localization. Thus achievement in space will push the character to be diverted from a goal to achieve actions of another goal which can be solved in few actions.

Finally, we consider the *relational motivations* :

Altruism This feature expresses how much it is inclined to help others. It balances the importance of the other's goals with its owns.

Reputation To decide whether or not it helps another, a character can refer to the reputation of the other. This reputation can result from social exchanges and history or from relative social status of both characters. For example, a character will be strongly inclined to execute goals (orders) given by a higher ranked character.

In our ASM proposition, we are akin to the *persistence*, *activations proportional to current offsets*, *balanced competition*, *contiguous action sequences*, *interrupts if necessary*, *opportunism*, *combination of preferences*, *flexible combination of stimuli* and *compromise candidates* criteria listed by Tyrrell in (Tyrrell, 1993).

Combination function The evaluator values are aggregated using the *Comb* function. This function plays a similar role to the arbitrator of DAMN (Rosenblatt, 1995). Therefore it has an important influence in the interpretation of the evaluations that obviously depends on the used function. Thus, depending on the chosen function, the value returned by an evaluator can promote or penalize the action. Hence, it is obvious that the used combination function must be known while defining the evaluator's functions. In the same way the range of the values must be defined. It is possible for a combination function to let the possibility to the evaluator to be neutral or to express attraction, repulsion or even inhibition. Then for an evaluator, returning 1 is interpreted as neutral, a value greater than one means that the feature promotes the action, a value between 0 and 1 penalizes, and 0 implies an inhibition that annihilates all the other feature's motivations. We present here a solution to easily design an ASM for MMORPG.

4 EXPERIMENT

We will present here one of our experiments leaded to validate our ASM and the various motivations. This experiment's ASM uses a *Comb* function which is the multiplication operator composed with a multi-goal revalorization. All the inner and environment motivations are implemented but the experiment does not consider the social and relational motivations. We do not have enough space here to detail all the used operators. We focus on two of them : *character preferences* and *opportunism*.

Character preferences. The whole alternative is considered, gathered and combined to produce this motivation value. The evaluator, γ_{pref}, is defined as follows:

$$\forall\alpha \in \mathcal{A}^R, \gamma_{pref}(\alpha) = Pref(alternative(\alpha))$$

where *alternative* is the function which gathers the preferences of all the actions in the alternative of α and $Pref$ is the combination function, here, the *harmonic mean*.

Opportunism. Outside the opportunism range θ_{opp}, targets have no influence and actions receive the neutral value. Inside, the closer a target is, the most favoured the corresponding runnable action is. The evaluator, γ_{opp}, of the opportunism is defined as follows:

$$\forall\alpha \in \mathcal{A}^R, \gamma_{opp}(\alpha) = \max\left(1, 1+\log_{\theta_{opp}}\left(\frac{\theta_{opp}}{dist(c,target(\alpha))}\right)\right)$$

where c is the acting character and *target* the function that returns the closest known target for action α.

In the experiment, we consider a situated cognitive character c distinguished by a radius vision and an inner attribute representing its *energy*. The value of this attribute decreases at each step. c's abilities are *move*, *take*, *eat*, *break*, *unlock*, *open*, *explore*[1]. Preferences are assigned to these properties and since we decide that we want a character which leans to be brutal, we give a preference value of 1.5 (attraction) to *break* and give 0.7 (reluctance) to *unlock*. Other abilities receive a neutral value (i.e. 1). These choices should lead c to prefer to break door rather than to unlock them (since doors will be the only possible targets for these actions).

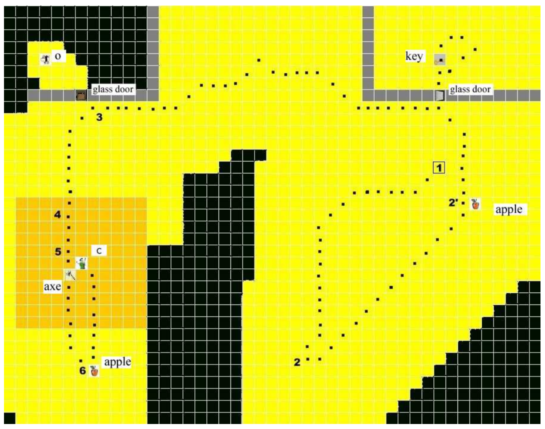

Figure 2: The environment.

Once built (and only then), c is situated in an environment, for example the one of Fig 2. Then we assign goals to c. In our particular case, its first goal, g_1, is to *take the object o* in the upper-left room, locked by a glass door whose key is in the upper-right room. Its second goal, g_2, is to maintain its *energy level above some value v*. g_1 receives a constant priority. Priority of g_2 depends on the energy attribute's value. Two apples and one axe are present in the environment, eating an apple gives energy and the axe can be used to break a door. c does not have a priori knowledge on the environment and must explore it. Unknown places appear in black in figure 2 where the vision radius can also be perceived. At each step c executes the action chosen by the ASM in order to solve the goals. The trajectory is showed with black dots.

Let us consider the run of c. At the beginning, c is located at point **1**, it perceives only the right apple. Since its starting energy level is above v. The only active goal is g_1 (i.e. because g_2 is satisfied, its priority is $-\infty$). Since c does not know the environment, c explores it to find o. Figure 3 shows the different values given to runnable actions by the ASM. At the beginning, the sole runnable action is *explore*, hence it is chosen. While exploring, c loses energy, reaching point **2**. Its energy level falls under v, then g_2 receives a priority depending on the energy value. It implies that the *move to apple* (to eat it) action becomes runnable. As shown in the chart, its value is the greatest, then c choses to move towards the apple. Since its energy decreases during these moves, the priority of

[1]The action *to explore* allows our agent to scout for an unknown target.

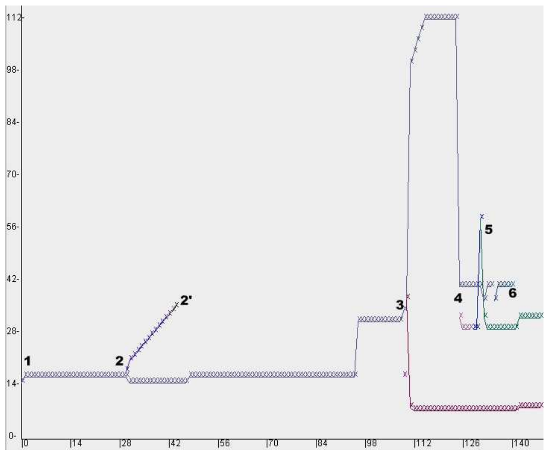

Figure 3: The runnable actions ASM values curves (the ordonate) during the steps on the simulation (the abscissa).

this action increases too. In point **2'**, *c* eats the apple, receives energy and g_2 becomes inactive again and *c* explores again to find *o*. Reaching point 3, *c* perceives *o*, trying to *open* door, *c* "learns" that it is locked. The plan proposes then 2 possibilities to pass through the door: to *unlock* or to *break* it. Then two runnable actions arise corresponding to both alternatives: first, *move* towards the previously perceived key, second *explore* to find something to break the door. Since *c*'s personality leans to be brutal and then *c* prefers to *break* rather to *unlock*, this explains why the alternative including the *explore* action is favoured in comparison to the one with *move*. The latter corresponds to the lowest curve starting from **3** and the first to the uppermost curve. The latter is especially high since, by coincidence, at the same time, g_2 becomes active again, then *explore* is again runnable in order to find some food, and *multigoal revalorization* promotes *explore*. Exploring, *c* goes "down" and perceives the axe once at **4**. Then runnable action, for *break* is no more *explore*, but *take* axe, which corresponds to the new curve in the middle. And *explore* loses multigoal revalorization. This explains why the uppermost curve weakens. But it still remains the most prioritary. Then reaching **5**, because of *opportunism* since *c* is close to the axe, the runnable action *take axe* is favoured and becomes the most prioritary one. The peak at **5** is then due to *opportunism*. The corresponding small collapse of *explore* is due to the temporary lose of *inertia*. Once axe is taken, *opportunism* motivation disappears and *explore* becomes again the most prioritary, then *c* finds and eats the second apple in **6**. *break* the door is the action selected by the ASM. *c* moves "up" towards the door, breaks it and takes *o* (not shown).

This small experiment illustrates the ASM's work and the various motivations: *opportunism*, *goals*, *preferences*, *inertia*, *multigoal revalorization*. *Achievements in time* and *space* are not hightlighted here but have been evaluated in other experiments.

5 CONCLUSION

"Always-running" applications are a very constraint context to behaviour designers. We propose a model of action selection mechanism defined as a combination of several motivations. This ASM allows to define modular, believable and easy to design behaviours. Since it is robust to evolutions of the environment and motivations are understandable, the designer task of building behaviours is made easier. Such an ASM can be used to design the behaviour of believable cognitive situated characters like NPC in video games. Characters can be easily distinguished and various personalities can be obtained. A concrete proposition has been done and experiments have been made to validate it.

Forthcoming works concern the implementation of relational evaluators and the carrying out of other experiments. Simultaneously a collaboration is in progress with a MMORPG company to use this ASM. Other motivations are investigated too, for instance emotional feature.

REFERENCES

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111(4).

Ferber, J. (1999). *Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence*. Addison Wesley.

Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence*.

Laird, J. E. and van Lent, M. (2000). Human-level AI's Killer Application: Interactive Computer Games. In *the 17th Natl Conf. on Artificial Intelligence*.

Rosenblatt, J. K. (1995). DAMN: A distributed architecture for mobile navigation. In *the AAAI Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents*.

Seth, A. (1998). Evolving action selection and selective attention without actions. In *the 5th International Conference on Simulation of Adaptive Behavior*.

Tyrrell, T. (1993). *Computational Mechanisms for Action Selection*. PhD thesis, University of Edinburgh.