DogMate: Intent Recognition through Anticipation (Extended Abstract)

Eurico Doirado
INESC-ID and Instituto Superior Técnico
Av. Prof. Cavaco Silva, TagusPark
2780-990 Porto Salvo, Portugal
eurico.doirado@ist.utl.pt

Carlos Martinho
INESC-ID and Instituto Superior Técnico
Av. Prof. Cavaco Silva, TagusPark
2780-990 Porto Salvo, Portugal
carlos.martinho@ist.utl.pt

ABSTRACT

Autonomous virtual agents generally lack the support to understand a fundamental character in their world — the user's avatar. This has a negative impact on their behavioural believability. In this paper, we present an approach to detect the intent underlying certain actions of the user. We begin by introducing the relation between intention and action, then proceed on elaborating a framework that can interpret the intent of an action based on the matching and mismatching of anticipated behaviour. We then present a test-case in which our framework is used to create an agent architecture controlling a virtual dog that interacts with the user within a virtual world. Finally, we discuss three aspects of our evaluation: the user's intent recognition, its interpretation and the solution's efficiency. Our results suggest that our solution can be used to detect certain intents and, in such cases, perform similarly to a human observer, with no impact on computational performance.

1. INTRODUCTION

In an interactive virtual world, it is but a common scenario for virtual agents and the avatar controlled by the user to interact with each other. Because of their role in the virtual world, some virtual agents (e.g. sidekicks [4] in computer and video games), interact with the user's avatar during long periods of time. As time passes, the user progressively creates expectations regarding the virtual agent's behaviour based on the experience they share together. The virtual agent, on the other hand, usually remains the same independently of its past interactions with the user. The virtual agent lacks the capacity to generally understand the user and as such is unable create expectations about her behaviour. This lack of adaptation becomes critical as it may be enough to break the user's suspension of disbelief [11].

Researchers in Artificial Intelligence have been tackling the problem of believability for characters in an interactive environment for nearly two decades. Back in 1992, Bates defined a believable agent as an interactive character that does not disrupt the user's suspension of disbelief [1]. He argued that a character does not need to have a complex behaviour to appear believable and Loyall, arguing in the same direction, stated that it is not necessary for a synthetic character to be fully realistic. If the character, at some point, is able to display specific and context-aware behaviour, it could be enough for the viewer to portrait it as having a life of its own [5]. An example is the work of Martinho and Paiva [7], in which a virtual agent tries to anticipate what the user will do and reacts emotionally whether it is disappointed or excited about what happens in the virtual world based on its expectations. Although anticipation has rarely been considered when creating believable agents, Martinho [6] highlights it as an essential part of the creation of believable behaviour and shows how powerful it can be in producing context-aware behaviour virtual agents need to feel alive.

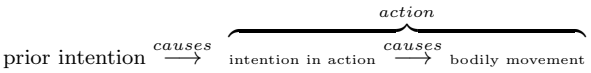
Anticipating what the user's avatar will do in the virtual world is, however, just a first step. The reason why the user did a certain action is just as important as the action itself, as it might change her understanding of her surroundings. Blumberg [2] stated that understanding the intentionality underlying a synthetic character's behaviour is an important aspect of believability. Should not it be just as important for the character to understand the user herself? Tomlinson et al. [11] argued that one of the key-aspects to create believable behaviour is the ability to react according to the user's actions and the intention underlying them. This includes, first, understanding the user's intention and, then, enabling the character to express an adequate behaviour in context. In this work, we focus on the problem of understanding the user's intention. Recognizing the user's intentions will provide virtual agents interacting with her the ability to express behaviour that matches the situation. Because our work is aimed at real-time interaction in resource intensive applications (e.g. current generation computer and video games), computational cost is a fundamental constraint. Our solution will have to be lightweight to be usable.

The remainder of this paper is divided as follows. First, we model the relation between intention and action, and discuss how action can be divided into movement, target and intent. We then discuss how each concept can be applied in the context of virtual worlds. Afterwards, we propose an anticipation-based framework (DogMate) to detect, from a virtual agent perspective, the intent underlying some of the user's actions. The following section describes our test-case (K9) that exemplifies how our framework can be used to control the behaviour of a virtual agent (in this case, Rusty, the user's dog companion). The evaluation of our test-case follows, focused on three aspects: intent recognition, intent interpretation and solution's efficiency. Finally, we conclude and present a few directions to further pursue this work.

Paper submitted as: I Mean It! Detecting User Intentions to create Believable Behaviour for Virtual Agents in Games, Eurico Doirado and Carlos Martinho, *for the AAMAS2010 conference.*

2. MODEL OF INTENTION

In his essay on intentions, Searle [10] distinguishes two level of intentions and relates them to actions as follows:



Searle states that while a *prior intention* (a notion similar to premeditation) may lead to an *action*, an *action* may exist without a *prior intention*, but has to contain both an *intention in action* and a *bodily movement*. Consider an example in which someone wants to enter a building (*prior intention*). First, he needs to open the door (*action*). The action of opening the door has an *intention in action* (the desire to open the door) and a *bodily movement* (the actual opening of the door). The relation between intention and action is important as, in interactive virtual worlds, every change caused by the user to the application is a result of an action of the avatar the user controls.

The Belief-Desire-Intention model (BDI) [3] introduced the concept of intention in agent architectures [9]. In the BDI model, intention refers to the "volition to accomplish a plan of actions", a definition similar to Searle's *prior intention*. In the BDI model, similarly to Searle, an intention is fulfilled through action, and the decision process generally relies on having access to the agent internal state. When considering the user of an interactive virtual world, however, the application can only access the sequence of inputs that the user introduces through the input devices (rather than her internal state). All the information is information about her avatar, limited by the rules defined by the interactive virtual world. As such, the intentions we are trying to recognize are those of the user's virtual representation, which might differ from her real intentions.

Guided by the previously described approaches, in this work, we model an *intention* as leading either to other intentions or actions (fig.1). The *action* itself is composed by three components: (1) *movement* (e.g. getting closer to, using an item); (2) *target* (e.g. another agent, a static entity); and (3) *intent* (e.g. attacking, picking up, talking).

Figure 1: Model of Intention.

For instance, consider that the user moves her avatar closer to a virtual agent, known to be her enemy. This action could be defined has having "getting closer" as the movement component, the "virtual enemy agent" as the target component, and (most probably) "attack" as the intent component. Now consider another example, in which the user moves her avatar towards the door of a building (an entity that cannot move by itself in the virtual world). This action also has the same "getting closer" movement component, but now the target component is the "door" and the intent component is (probably) "opening the door".

As a first step in detecting user's intentions, in this work, we focus on the detection of the *intention in action* underlying the user's actions (henceforth designated as *intent*), and leave *prior intentions* for future work.

3. COMPONENTS OF AN ACTION

The logic underlying the rules of a virtual world is inherently constrained by the graphical structure in which the action takes place. In the game literature, what happens in the virtual world can often be reduced to two fundamental components: *movement* and *collision detection*. Therefore, the concept of *distance* is fundamental to understand what is happening in the virtual world. As an example, consider a virtual dog agent trying to bite another virtual agent. To be able to catch its opponent, the virtual dog will need to reduce the distance between them until it is close enough to perform the action of biting the other agent.

As a first step, we perceive all that is happening in the virtual world by reducing everything to distance variations between entities. If the distance between two entities is below a certain threshold, then something happens. These entities can be *intentional* or *static*. Intentional entities are able to move and act on their own and, as such, have inherently an intent associated with that movement, while static entities can only be acted upon, and have no intentional state attributed, although displaying certain stimulus-response compatibility (affordances): if a virtual agent approaches a closed door, it probably intends to open it.

Because we reduce everything to distance variations, detecting the user's intention is, in our approach, to understand why the user's avatar is moving towards or away from certain entities. Because distance varies generally in a continuous manner, expectations can be created regarding distance variation. Such expectations, when violated, can be used as a temporal signals that new intentions may have been generated by unpredicted change in the action. The next sections discuss how the three components of the action — movement, target, and intent — are computed in our model.

3.1 Movement

A first problem in the detection of the user's avatar intent is to detect when a new action begins. For that effect, we use movement. Consider that the user's avatar is moving in the direction of a certain entity (a door) in a predictable manner. At a certain point, it unexpectedly changes direction and moves towards another entity (an enemy). While all this happens, Rusty, our virtual dog, is observing. Figure 2 depicts such a situation.

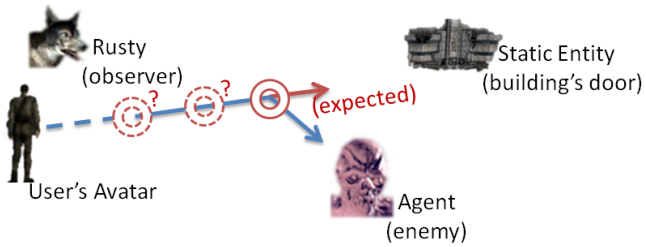


Figure 2: When the user's avatar makes an unexpected movement, Rusty detects one action for the intentional entity (the user's avatar) and another action for the static entity.

We consider this event marks the moment the user's avatar decided to interact with the new entity. While unable to determine the start of any action until that point, Rusty is

now confident that the user's avatar started a new action: to go toward the enemy. It is important to note that another action also started: getting away from the door.

In practice, we monitor the distance variation between pairs of entities. When an unexpected variation occurs, it marks the start of a new action: if the first derivative of the distance between entities is negative, they are moving closer; if positive, they are moving away from each other.

3.2 Target

When monitoring the distance between an intentional (e.g. user's avatar) and a static entity, we know that when the distance changes, the intentional (not the static) entity is moving. When both entities are intentional, however, other measurements are needed to disambiguate between possibilities for agency. Consider, for instance, that distance between the user's avatar and another virtual enemy agent has decreased more than expected. In this case, there are three possible scenarios (fig.3): the user's avatar moved closer to the enemy; the enemy moved closer to the user's avatar; both moved closer to each other.

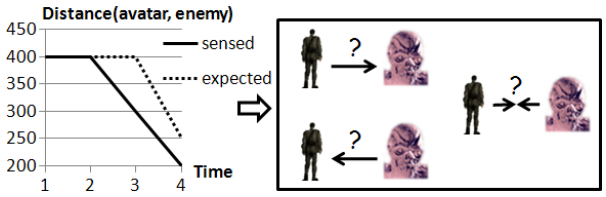


Figure 3: Distance between intentional entities alone is not enough to determine agency.

To determine agency, we monitor the entities relative velocities. When the velocity of an entity changes unexpectedly, we assume it may also influence distance between entities in an unexpected manner. As such, if we monitor both an unexpected distance variation as well as an unexpected velocity variation of an entity, the entity is considered responsible for the movement, and the other entity is marked as the target of the movement. In such cases, four combinations are possible: the user's avatar is getting {closer to, further away from} the agent; the agent is getting {closer to, further away from} the user's avatar.

3.3 Intent

The last component of an action is its intent. The intent results from a natural combination of movement and target. As discussed previously, the rules underlying virtual worlds are mostly based on distance variation and collision detection: if the user's avatar wants to interact with an entity, it first needs to reach within activation range of the entity. By detecting the movement of "getting closer", we can devise the intent based on the affordances of the target, i.e. the type of interactions allowed by the target entity. The same reasoning can be applied for "getting away from".

Consider an example in which Rusty notices the user's avatar moving closer to (movement) an enemy (target). From the affordances given by the enemy entity, Rusty assumes it must be to attack it (intent). If the user's avatar was to move toward the building's door, Rusty would assume it would be to open it (the only thing one can do to a closed door in the virtual world). If she would move her avatar away from an enemy, Rusty would assume she wanted to avoid him.

4. DOGMATE FRAMEWORK

The previous section described how the components of an action are created by confronting expectations to the virtual agent's perceptions. When the perception is outside of the prediction range, our model assumes that a new action (movement, target, intent) was detected. As such, one primary concern when using our model is the ability to make predictions. Additionally, each time a virtual agent is considered to have an unexpected movement, several actions may be detected simultaneously (the virtual agent gets closer or further away from other entities with every movement). Thus, we need to be able to classify and compare the relevance of each action. Finally, we want our virtual agents to interpret the virtual world according to their own point of view.

In this section, we present DogMate, a framework for generating intents in a virtual world from an agent perspective. We start by addressing the three enunciated concerns, then discuss the flow of execution within an agent architecture.

4.1 Predictors

To fully identify an action, we make use of three predictors between pairs of entities. One estimates the distance variation between the entities (to determine movement) and two estimate each entity's relative velocity (to determine agency). We currently use the euclidean distance as it represents a good trade-off between estimation accuracy and computational cost. Our predictors are implemented using the following equations:

$$\hat{d}_t = d_{(t-1)} + \hat{\dot{d}}_t \tag{1}$$

$$\hat{\dot{d}}_t = \dot{d}_{(t-1)} + \hat{\ddot{d}}_t \tag{2}$$

The expected distance at time t ($\hat{d}_t$) depends on the expected velocity $\hat{\dot{d}}_t$ (eq. 1) which, in turn, is based on the expected acceleration $\hat{\ddot{d}}_t$ (eq. 2). To estimate the acceleration, we use the following equation:

$$\hat{\ddot{d}}_t = \frac{\sum_{i=1}^{n} \dot{d}_{(t-i)} - \dot{d}_{(t-(i+1))}}{n} \tag{3}$$

We predict the acceleration (eq. 3) by applying a moving average [8] to the last n sensed accelerations. Empirically, we found that a moving window of $n = 3$ gives adequate results, as the window is small enough to adapt quickly to acceleration change, while large enough to mitigate small acceleration variation due to the noise present in the original signal.

4.2 Relevance

In figure 2, Rusty observes an unexpected movement of the user's avatar which generates two actions: "getting closer to the enemy" and "getting away from the door". However, the user probably only intended to make one of them. While both may contain relevant information for decision-making, it is important to be able to quantify and classify their relative importance for the virtual agent.

To express the relevance of the action for the virtual agent at time t (R_t), we use the following equation, representing the degree of unexpectedness of the sensed value:

$$R_t^{ent} = \frac{PredictionError_t^{ent}}{ExpectedError_t^{ent}} = \frac{err_t^{ent}}{e\hat{r}r_t^{ent}} = \frac{|x_t^{ent} - \hat{x}_t^{ent}|}{e\hat{r}r_t^{ent}} \quad (4)$$

In equation 4, x_t and $\hat{x}_t$ represent the sensed and expected value, respectively. By computing relevance as a ratio, we make the prediction independent from the metric it measures (or even its range). When $R_t^{ent} > 1$, we say that the signal is salient and that an expectation violation occurred.

The expected error ($e\hat{r}r_t$) gives us a margin in which a sensed value might differ from the expected value and still be considered as an expectation confirmation. We compute it using the following equation:

$$e\hat{r}r_t^{ent} = k \times err_{(t-1)}^{ent} + (1-k) \times e\hat{r}r_{(t-1)}^{ent} \quad (5)$$

The variable k takes values within the $[0, 1]$ interval. A high value for k allows the estimation to adapt quickly to huge variations, while a low value makes the estimation more conservative. When an unexpected event occurs, the acceleration predictors need a certain time-frame to re-adapt to the signal and emit correct predictions once again. Within this time, it is important to avoid detecting a false unexpected event as a result of adaptation. After adapting to the new signal, prediction error (and consequently the expected prediction error) will decrease.

Let us exemplify how prediction and relevance work using the scenario from figure 2. The user initially moves her avatar simultaneously toward an intentional entity (enemy) and toward a static entity (door). Assume that, when the unexpected movement occurs, the user is 700 units away from the door and 400 units away from the enemy. Additionally, because the user's avatar was walking toward the door, we expect the distance to the enemy ($\hat{x}_t$) to remain around 400 units, while the distance to the door is expected to decrease to 650 units. Finally, because our predictions have been accurate this far, we expect a low prediction error in both cases: 10 units. Now, the user broke our expectations and is (x_t) closer to the enemy (300 units away from the enemy) while further away from the door (the distance to the door increased to 750 units):

$$R^{door} = \frac{|750 - 700|}{10} = 5$$
$$R^{enemy} = \frac{|300 - 400|}{10} = 10$$

When comparing both actions resulting from the distance variation to the door and to the enemy, we can say that the action to go toward the enemy is two times more relevant (unexpected) than the one to go further away from the door. We assume that the more unexpected an action is, the more likely the user meant to do that action intentionally (in this case the action to go toward the enemy).

With this model, we are not only able to distinguish actions and order them by relevance. If we combine relevant changes in both inter-entity distance and relative velocity, we can decide what entity is responsible for the action (the other entity being considered the target of the action). In this case, if the virtual enemy agent had a relevant change in relative velocity, it would mean the enemy started to attack the user's avatar, while if the relevant change occurred with the relative velocity of the avatar, the user's avatar would have initiated combat.

4.3 Affective Appraisal

To model the intent interpretation from an agent's perspective, we appraise the intent inspiring ourselves in the emotivector's sensation model [6]. First, we set desired (and undesired) values for distance, depicting where the virtual agent would like the distance between two entities to be (or move away from). Then, when a relevant change occurs in the sensed distance between the two entities, we compare the sensed value, the expected value and the desired/undesired value to produce an affective state. If the sensed value is closer to the desired value, it is considered a positive sensation. If it was expected to be even closer it is a "positive but worst than expected", if not it is a "positive and even better than expected". The same applies if the sensed value is diverging from the desired value, but in such case it is considered a negative sensation. The reasoning for getting closer or away from undesired values is similar. As a result of this affective appraisal, one of four affective states is generated for each unexpected change. The affective states will influence the virtual agent's decision making.

As an example, consider that Rusty sees the user's avatar is hurt. Rusty desperately wants the user to avoid any threat, by setting an undesired value of 0 for the distance between the user's avatar and any virtual enemy agent. Consequently, if it detects the user has an intent to attack an enemy, he will view that intent as bad for the user (the sensation will be a negative one). Based on this personal interpretation, Rusty will assume a context-specific strategy and display an adequate context-aware behaviour, totally different from the one resulting from appraising the event as positive.

4.4 DogMate's Flow

We now have described all elements needed to recognise actions and their underlying intents, to appraise and classify them. DogMate (represented in fig. 4) is the name we gave to this framework and we now review its flow during one update cycle. Let us, again, turn to the example depicted in figure 2 from Rusty's perspective. Let us also consider that Rusty knows that the user's avatar has an objective: to break through the building to invade it. Rusty also knows that the player-character is healthy enough to handle a few enemies. These are his current beliefs and we use them to set the desired value of predictors (1 in fig. 4) such that reducing the distance toward enemies or the door is considered as positive within this context. Rusty's beliefs also contain information about the position of entities that Rusty can sense (i.e. within a certain radius). This information is fed to the predictors at each cycle.

To represent the possible actions between the user and an enemy, we use one set of three predictors (one distance-based and two velocity-based). To represent the possible actions between the user and the building's door, we use a set with one predictor (distance-based) (2). The user is currently moving her avatar through the world and is reducing her distance to both the door and the enemy. Suddenly, something unexpected occurred: the avatar reduced drastically its distance to the enemy and the distance to the door increased. At this time, the distance-based predictor in both sets became salient (3). In the set of the door (a static entity), we only have one predictor so we know that the user is getting further away (movement) from the door (target). In the second set, the predictor that monitors the

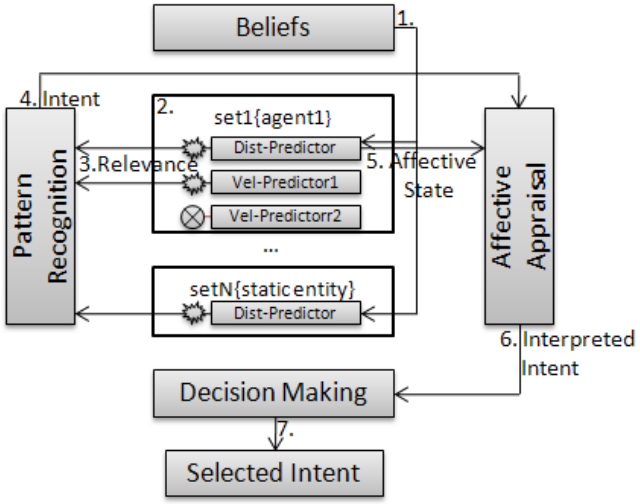


Figure 4: (1-2) predictor parametrisation; (3-4) intent detection; (5-6) affective appraisal; (7) action selection.

relative velocity from the avatar to the enemy is also salient. This pattern means that an action was initiated by the user, whose target is the enemy, and her movement is to get closer to it. Two intents are produced based on these recognised attributes (4).

Because Rusty knows that the user's avatar can handle another enemy, the distance-based predictor leads to the emission of a positive sensation (5) which makes Rusty interpret that distance reduction (and the corresponding intent to attack) as beneficial for the user (6). On the other hand, because Rusty wants the user to complete his objective, he interprets the increased distance to the door as negative and consequently views with bad eyes the disinterest of the user in completing her task.

Finally, these intents are compared to each others in the decision making module to select the decision that Rusty will adopt. Because the predictor that monitors the player-character's relative velocity to the enemy is the most salient of them all, we choose to select that intent as the one that will make Rusty react (7). As such, Rusty encourages the user's action through verbal and non-verbal behaviour.

In the next section, we describe how we integrated this framework within K9, our test-case, to control Rusty acting as the user's dog sidekick within a computer game.

5. TEST-CASE: K9

K9 ("Canine") is a small role-playing game environment used as a test-case for our framework. Not only K9 but the whole implementation of this work was built based on the modding framework of Fallout 3[1] (FO3) and takes full use of its game mechanics. K9 story takes place as a prequel to the original story. The player-character (the user's avatar) ends up being kidnapped and used as an experiment in genetic tests. The result is that he finds himself linked to his own dog, Rusty, and is not only able to feel what Rusty feels but also to understand him.

[1]Bethesda Game Studios, Fallout 3 (2008): http://fallout.bethsoft.com/.

In this section, we first present the interaction between FO3 and Rusty's Brain, the component which includes DogMate, then we describe how intents were used to control the behaviour of Rusty in the game.

5.1 Interacting with DogMate

To implement DogMate as an independent component, we used Fallout Script Extender[2] (FOSE). This system gave us an indirection which made possible to invoke functions from a dynamic library (dll) outside of the original engine. This component (Rusty's brain in fig.5) includes DogMate and a world interface which contains sensors and effectors.

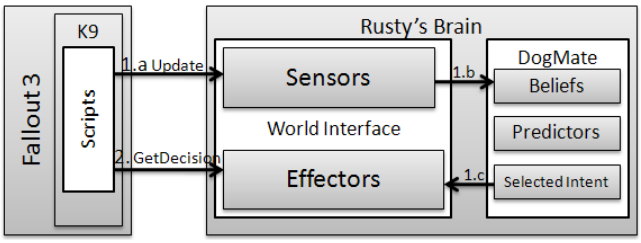


Figure 5: Interaction between K9 and Rusty's brain.

First, let us outline the general flow that corresponds to the brain being updated (fig.5). It starts from within K9's scripts (1a). At this point, information about the world is sensed so that we only consider detected enemies and relevant entities (1b) from Rusty's perspective. This information is stored in DogMate's beliefs. DogMate performs its update cycle, eventually detecting the intents of relevant virtual agents. As the result of its cycle, DogMate sends primitives corresponding to the selected behaviour for Rusty. We store this information in a buffer (1c), the effectors, and in the end of the K9 cycle, the script checks which was the decision (2). Within the FO3 engine, this information is used to execute verbal and non-verbal behaviour for Rusty. Rusty's behaviour consists of an animation coupled with its corresponding sound and a subtitle that serves two purposes, to reinforce Rusty's barks and express his thoughts (in K9, the player-character is able to understand Rusty).

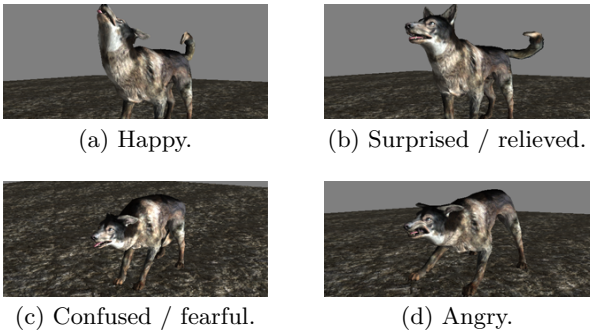
The main purpose of the *world interface* is to act as an interface between the engine and the rest of brain. It is responsible to keep an up to date cache that replicates the game engine information deemed relevant for our component. This information is then culled using Rusty's sensors and transformed into its own beliefs. This information typically either represents the perception needed to update a predictor or its desired value. The world interface is also composed of effectors which consist in a buffer that stores the decision. Once the decision is received through the "Get-Decision" invocation (2 in fig.5), the engine processes and makes Rusty behave in the intended manner.

5.2 Rusty's Behaviour

Rusty's "natural" behaviour is to accompany the player-character and help him along the way. However, whenever his brain selects an intent, Rusty expresses its appraisal of the intent based on several animations, each tied to a

[2]I. Patterson, S. Abel and P. Connelly, Fallout Script Extender (2008): http://fose.silverlock.org/.

unique affective state. The affective states are the sensations elicited when a predictor is salient.



(a) Happy. (b) Surprised / relieved.

(c) Confused / fearful. (d) Angry.

Figure 6: Rusty can express four affective states.

We associated sounds like barking or panting to these animations. In the "happy" animation (fig.6(a)), Rusty raises his tails as he barks happily, to the sky, five times. When he is surprised (fig.6(b)), with a touch of relief, he pants a few times while waving his tails in a joyful manner. In contrast, when he is confused (fig.6(c)), with a touch of fear, he lowers his head, whimpers and hide his tails in-between his legs. Finally, when he is angry (fig.6(d)), he assumes a serious and threatening pose in which he growls ferociously. Along the animations, we reinforce Rusty's expression by using on-screen subtitles, that reflect Rusty's own thoughts as understood by the player-character.

Once a decision is retrieved from Rusty's brain, the engine executes the corresponding set of animation, sound and subtitle. In table 1 we show a few examples of the available possibilities when the user displays the intent to attack or avoid an agent while having having either a good or bad physical condition to handle the confrontation.

Intent	Condition	Appraisal	Thought / Counsel
Attack	Good	Happy	*"Oh yeah! Let's get him!"*
Attack	Bad	Angry	*"An enemy now? That would be bad..."*
Avoid	Good	Angry	*"A little fight wouldn't be that bad!"*
Avoid	Bad	Happy	*"That's it! Let's avoid that one."*

Table 1: Examples of Rusty's thoughts.

6. EVALUATION

In this work, we pursued several objectives. First, we wanted to create a framework to support the creation of believable behaviour for virtual agents in general, by allowing the detection of some of the virtual agents and user's avatar intentions. We were particularly interested in applying such an approach to a particular role: virtual sidekick agents for computer games, and this interest guided our development to K9. To evaluate our approach, we focused on analysing whether the framework could correctly detect the user's intentions. We were also interested in understanding how well the recognition of intentions was performing when compared to an external human observer. Finally, we wanted to be aware of the performance. Virtual worlds are real-time applications in which the greatest emphasis is almost always given to the visual component. Other components, such as the application's physics or artificial intelligence, are left with minimal resources. With such restrictions in mind, we wanted to verify the computational cost (in terms of CPU consumption) of our solution. In the next subsections, we review our experiment and discuss the results.

6.1 Experiment

To evaluate our test-case, we performed an experiment in which we asked subjects to play the game. The experiment took place at a public venue in which most participants were university students whose age varied from 19 to 28. Before asking them to start the experiment, we made sure that they had a minimal knowledge of how to play the game. Once ready, we asked them to play a session of K9's main level, in which their objective was to invade the enemy's headquarters. Each interaction lasted for two to three minutes.

Each game session was recorded through a screen capture software. Immediately after finishing a session, we asked the participant to annotate the recognized intents. For the annotation, participants could choose one of five available options: whether they wanted to attack an enemy or flee from it; whether they wanted to open the door or were avoiding it; the fifth option was to be used if their intent did not match any of the previous four options. From this evaluation, we gathered a total of 54 samples (instants in which Rusty expressed itself based on the user's intent) from 9 different participants.

Because human observers watching the game would also fail at recognizing certain intents, we wanted to compare the performance of our approach with that of a human observer. Our motivation is that if the mismatch between the real intent and the recognized intent can be understood as natural by the user, it might appear as believable. However, for the user to understand Rusty's reactions, she has to interpret the situation from Rusty's point of view. With this idea in mind, we randomly selected 30 samples from the 54 available. We then published them on a video streaming website. Although we lost some video quality, it gave us the possibility to pursue the evaluation virtually. Our aim was to reach a larger population this time and each participant was asked to review the thirty samples and to classify them using the same options previously available to the participants. With this approach we collected 820 valid samples from 30 observers. We now discuss our results.

6.2 Intent Recognition

We gathered a set of 54 samples from the venue's participants and we compared each of them against their respective annotation. We found that the participant's intent was recognized 61% of the time (see table 2).

Rusty	samples	matches	%
user opens the door	17	13	76.47%
user attacks enemy	25	17	68.00%
"avoid" intents	12	3	25.00%
total	54	33	61.11%

Table 2: Matching Rusty's recognized intent and user's intent.

During our tests, we found out that participants rarely fled

from an enemy (although being positively detected several times). They also had a certain difficulty to express negative intents such as "avoiding an entity" because, in reality, when they did so, it was mostly due to the fact that they started to do something else (i.e. "to seek" another entity). Comparatively, we achieved a recognition rate of 71% for intents related to moving toward an entity (first two rows in table 2). These results support our assumption that if the user moves her avatar toward an entity, she probably has the intent to interact with it.

The gathered data also suggests that some refinements could help increase the recognition performance. At the conceptual level, it may be important to take the virtual agent's field of view into account (rather than only using a perception radius around the virtual agent, as implemented in K9). Even if the user could be focusing her attention on something she cannot directly see (e.g. she wants to ambush an enemy by making a detour around an obstacle and get in his back), such information could be useful to confirm specific situations and prune others. At the implementation level, the notion of distance could be implemented differently. In K9, we chose to use euclidean distance as a compromise between accuracy and performance. However, in a world with a high number of obstacles (e.g. buildings, rocks, interiors), this value may be inadequate. A more realistic value for distance could be computed using path-finding algorithms. However it is a costly process in virtual worlds of a dynamic nature, and the added value for intent recognition remains to be measured.

6.3 Intent Interpretation

We do not need realistic behaviour to achieve believability, only to display specific and context-ware behaviour [5]. When Rusty fails to recognize the user's intent, we do not want it to break the user's suspension of disbelief by displaying inadequate behaviour based on its failed intent recognition. If we ask a human observer to comment about the possible intentions of his peer, he too might get some intentions wrong. However, every time the observer recognizes an intention, he can justify it based on his observation. Then, if the user uses some theory of mind, and puts herself in the shoes of the observer, limiting her own knowledge to the information the observer had, she might agree with the observer that the recognized intention could indeed be valid from a certain point of view. As such, we wanted to verify whether Rusty succeeds or fails at recognizing the same intents a human observer would succeed or fail to recognize.

We classified each sample according to whether Rusty and/or the external observer had recognized correctly the user's intent (see table 3) and applied a Person's chi-square test, $\chi^2(1, N = 820) = 18.31$ ($\rho < 0.001$). Results suggest that, generally, Rusty would recognize the same user's intents the human observer would, and fail to recognize the same user's intents the external observer would fail to recognize. As such, Rusty and external human observers seem to perform similarly when it comes to recognizing intents, leading us to believe that Rusty's intent reaction may be perceived as believable by the user.

The 820 gathered samples also reinforce our previous statement about negative intents. Subjects assuming the role of observers also had difficulties in identifying such intents as only 6.14% were correctly recognized (compared to 62% positive intent recognition — see table 4). The small variation between Rusty's performance and the lower human observer's performance may be related to the fact that Rusty's has access to information that is not always on the game screen. This is desirable for virtual sidekick agents, as the user will value the added understanding of the surrounding virtual world, without feeling that the sidekick has access to information the user simply cannot access.

Rusty ≈ Observer χ^2 ($\rho < 0.001$)		External observer		total
		recognized	not recog.	
Rusty	recognized	377	272	649
	not recog.	68	103	171
total		445	375	820

Table 3: Matching external observer's interpreted intent and Rusty's recognized intent.

Observer	samples	matches	%
user opens the door	325	217	66.77%
user attacks enemy	381	221	57.40%
"avoid" intents	114	7	6.14%
total	820	33	54.27%

Table 4: Matching observer's recognized intent and user's intent.

6.4 Limitations

The experiment helped us identifying several limitations in our approach. First and foremost, not all actions' intent can be equally identified. Our results support that "moving toward an entity" is correlated with the user intending to interact with the said entity. However, our results also show that "moving away from an entity" is not correlated with the intent to avoid the same entity. As referred earlier, a possible explanation is that users explained their actions by stating they wanted to do something new and not by stating they did not want to do an action anymore: if the user stops attacking and moves away from a virtual enemy agent, she might not be fleeing from it, she might just be looking for something that suddenly appeared on the floor or attacking another virtual enemy agent.

The second limitation is that our approach emits intents that have no past history, as if each one was the first and only detected intent. This results in duplicated and correlated intents emitted disregarding their possible cause. If the user gets surprisingly closer to an enemy and, seconds later, her distance decrease surprisingly once again, two different intents are emitted while referring to a same intent: to attack the virtual enemy agent. Also, attacking an enemy might produce an intent that suggests that the user is fleeing from another one. This lack of history and correlation between intents led to the emission of some false intents.

6.5 Computational Impact

A graphical application usually requires most of the hardware resources to be dedicated to the graphics engine. When each CPU cycle is crucial, the most important concern is to sustain the graphical frame rate, as it is one of the most noticeable aspects of the interaction for the end-user. While the game logic also needs its share of resources to be pro-

cessed, in computer and video games, the approach is that the lower the better. With this in mind, we aimed for a system with low resources requirements: if it is to be added on top of current generation game technology, then it has to be hardly noticeable.

Our first concern was the framework's update rate. An empirical test showed that 2 to 4 hertz tend to present good results. Lower than 2 hertz seems unnatural as it takes too much time between the user's action and the intent's detection. Any frequency higher than 4 hertz seems to produce the same unnatural behaviour, as if Rusty was reacting immediately. In a special testing environment, we used 300 predictors divided into 100 sets, to solely monitor virtual enemy agents. This number is exaggerated, but should cover most of the user's potential centre of focus in a virtual world. Even if we use more entities, those which are not relevant for the user's actual situation can be culled. Profiling this instantiation of the framework did not show any significant overhead (0.18%) at 4 hertz.

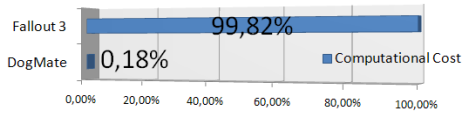


Figure 7: DogMate only used 0.18% of CPU resources to monitor 100 virtual agents.

Real-time constraint are mandatory in applications such as computer and video games. While our solution only recognize a subset of all possible intents, we would argue that, with such a low impact on CPU cycles, it does recognize them almost for free. As such, we defend this approach could be used to help in creating believable behaviour for synthetic entities inhabiting the virtual world.

7. CONCLUSION

Autonomous virtual agents generally lack the support to understand a fundamental character in their world — the user's avatar. This has a negative impact on their behavioural believability, as the virtual agents are unable to provide adequate context-ware behaviour to the user. In this paper, we provided a framework for recognizing some of the intentions underlying the actions of other virtual agents sharing the same virtual world. By understanding intentions in particular situations, we allow virtual agents to display specific context-aware behavioural reactions and improve their perceived believability.

Based on Searle's definition of intention, we started by modelling the relation between intention and action, and discussed how an action can be divided into three components: movement, target and intent. We then proceeded to detail how the three components of an action can be detected in a virtual world, based on the matching and mismatching of anticipated behaviour related to distance change between entities. We detailed how unexpected events are triggers for new intents, which type depends on unexpected events that become relevant simultaneously, and how an affective appraisal provides the agents with a personal view on the situation. The whole process constitutes our framework (DogMate) that we implemented as an agent architecture.

A test-case (K9) was presented, showing how our framework can be connected to a current generation game engine. In K9, DogMate controlled the behaviour of a virtual dog sidekick, Rusty, that interacted with the user within a virtual world, and advised her based on the intents it would interpret from the actions of her avatar. The test-case evaluation suggests that our framework can be used to identify some of the user's intents and, at least in some tasks, performs comparably to a human observer, an encouraging result as we were mostly focused on the generation of believable behaviour. The results also show that while the framework only recognizes a subset of possible intents, it has a low computational impact, and as such is suited for integration.

We believe the limitations identified during evaluation could be addressed by introducing a higher-level layer that would create a history of the intents to filter duplicates and related intents. Additionally, the framework should also be able to detect intentions in expected movement (e.g. while exploring), by introducing prior intentions to the framework. These will undoubtedly require new components to recognize the user's plans, their purpose and if an action is part of it. All these are possible directions we are considering for future work.

8. REFERENCES

[1] J. Bates. The nature of characters in interactive worlds and the oz project. Technical report, Carnegie Mellon University, 1992.

[2] B. M. Blumberg. *Old tricks, new dogs: ethology and interactive creatures*. PhD thesis, MIT, The Media Lab, 1997.

[3] M. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, 1987.

[4] K. Isbister. *Better Game Characters by Design: A Psychological Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.

[5] A. B. Loyall and J. Bates. Personality-rich believable agents that use language. In *Proceedings of the First International Conference on Autonomous Agents*, pages 106–113. ACM Press, 1997.

[6] C. Martinho. *Emotivector: Affective Anticipatory Mechanism for Synthetic Characters*. PhD thesis, Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal, September 2007.

[7] C. Martinho and A. Paiva. Using anticipation to create believable behaviour. In *Proceedings of the AAAI 2006*, pages 175–180. AAAI Press, 2006.

[8] NIST/SEMATECH e-Handbook of Statistical Methods. http://www.itl.nist.gov/div898/handbook/pmc-/section4/pmc4.htm, October 2009.

[9] A. S. Rao and M. P. Georgeff. Bdi agents: From theory to practice. In *Proceedings of the first international conference on multi-agent systems (ICMAS-95)*, pages 312–319, 1995.

[10] J. R. Searle. *Intentionality, an essay in the philosophy of mind*. Cambridge University Press, Cambridge, NY, USA, 1983.

[11] B. Tomlinson, M. Downie, and B. Blumberg. Multiple conceptions of character-based interactive installations. In *ITU Recommendation I371*, pages 5–11, 2001.