Virtual Humanoids: Let Them Be Autonomous without Losing Control.

Angela Caicedo and Daniel Thalmann
Computer Graphics Lab. Swiss Federal Institute of Technology
EPFL, DI-LIG, CH 1015 Lausanne, Switzerland
{caicedo, thalmann}@lig.di.epfl.ch http://ligwww.epfl.ch

Abstract

There have been several efforts to build life-like autonomous creatures in virtual worlds, but only few of them have focused their intentions in presenting human-like autonomous creatures. In this paper we discuss the problem of building autonomous virtual humanoids with goal directed behaviors. We present the decision-making as a process compound of: goal achievement planning, dynamic belief management, evolving goals, internal states and confidence levels. As simulating real humans, we applied to our model theories of *Humans' Trust* to be able to interact with the virtual humanoids and direct them at real time.

Keywords. Agent-oriented approach, goal achievement, dynamic beliefs, decision-making, planning, trust, confidence.

1 Introduction

Virtual environments are becoming more important every day especially in educational and entertainment programs. With this evolution, the user gets more demanding, and the realism of this world is a key point. Users want to control and interact with virtual environments in real time, they want to be immersed in these new worlds. One important contribution to the environment's realism is the human like performance of the humanoids who inhabits those environments. These requirements are the motivation for this work, we endow the virtual humanoids with a reasoning layer allowing them to take their own decisions, to have goal-directed behaviors, to act based on emotional levels and to take advantage of opportunistic situations in dynamic environments.

There have been few developed system where the character behaves in an intelligent and goal-directed way. Improv, (Perlin and Goldberg [7]) is a tool to create actors that respond to users and to each other. This system is completely preprogrammed, and during the simulation actors just follows some scripts, even time between actions are also predefined. This scripted system is very rigid to be able to adapt to changing environments and the definition of actors' actions is made at a very low level (in term of degree of freedom).

Blumberg in [3] showed an interesting architecture to design characters in interactive virtual environments with competent autonomous actions and capable of responding to an external control. In this case, this architecture is more oriented to animal-like simulation, where the dog Silas was one of the applications of the system.

Some works have got closer to human-like characters as the one presented in: Cosmos by Lester et al [5], the PPP person explained by Elisabeth André [1], and Steve by Rickel and Johnson [9]. However these characters have very different goals: they are pedagogical agents in charge of providing advice and explanations to the attendants.

The work presented by Bates in his TOK architecture [2] is very interesting. The architecture is composed of Hap that is in charge of reactivity and goal directed behaviors and Em in charge of emotions and social relationship. The problem in this system is that the Em module only receives the action selected by Hap, this means that the emotions don't take part of the decision making process. Also, their agents don't have emotional reactions to objects that they are not currently sensing.

The second topic of this paper is the directability of these characters. When we talk about virtual humanoids and user interactions, it is worth mentioning the controllability that the user has on these humanoids. Moreover if we talk about agency, we should talk about autonomy, but how can we bind virtual humanoids and agency without loosing the desired controllability of the first one and the autonomy of the second one? We propose a controllability model based on human trust theories.

There have been some works in this area, for example in the controlling approach for Improv [7], the user has direct control through actor's motor skills. Also in Silas (Blumberg [3]) the control is made accessing internal variables of the agent which represent motivations and goals, for example increasing the hunger level ensure that Silas will eat. Badler and Zeltzer have proposed similar decomposition of control. In these approaches, actors are unaware of changes and they can not learn from experience.

In this paper we present some contributions to the creation of virtual creatures:

- We focused on simulating humans-like characters, where special attention is paid to trust models, beliefs about other agents and internal states.
- The process of action-selections is a process which integrate goal achievements efforts, dynamic belief manipulation, internal states rates and beliefs about other agents.
- The model we introduce to control the agent looks forward to achieve a closer human-like feeling of the virtual humanoids. This model allows the agent to be aware of the changes he is accepting, and allow him to learn from them.

The rest of this paper is organized as follows: In the second section, a description of the *Intelligent Virtual Agent* is given. The third section presents the trust model applied to the virtual humanoid to be able to control it. The fourth section presents in depth explanations of the system's intelligent module called *IntelMod*. The fifth section briefly describes a concrete example. Finally we end our exposition with a conclusion of the current work and we present some future extensions.

2 The Intelligent Virtual Agent

To obtain human-like virtual humanoids, we have modelled a reasoning unit called *"Intelligent Virtual Agent" (IVA)*. This unit will be plugged as a *"brain"* to the virtual humans.

IVAs are based on a BDI architecture (Beliefs, desires and intentions), widely described in [8]. This architecture is promising but needs some extensions for achieving our goal: giving to the virtual humanoid the ability to act by himself in a dynamic environment relying on his beliefs, internal states, current state of the surrounded world and the assumption about other agents. It should also allow us to control it in real time. The main additions we have done are:

- Categorize the beliefs, in order to be able to simulate a simple framework of *short term memory* and *long term memory*.
- Inclusion of internal states for the agents in order to simulate basic emotions and mental states like fear, shyness, courage and happiness.
- Reliability on *trust*. Agents should trust to each other, and this reliance will evolve over time. In our model each agent stores some beliefs about other agents. These beliefs are the bases for collaborative work based on *trust*.
- Inclusion of emotions in the plan's structure and in the evaluation plans.

Before going any father lets put a scenario as example, the LIG's lounge (see Figure 12), where agents can go to have a drink, to dance, to meet other agents o just to have a look for a while. Claire who has to decide what to do at any given time of the simulation inhabits the scenario. Claire in this case is one IVA and she has all her knowledge

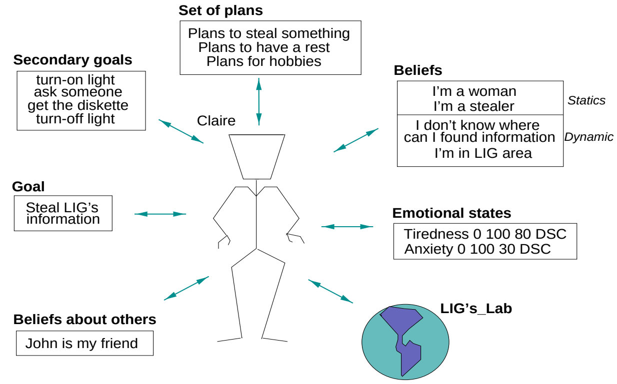

Figure 1: The intelligent virtual agent IVA.

organized in sets, as shown in Figure 1, which are distributed according to their functionality: set of *Beliefs*, set of *Goals*, set of *Competing Plans*, set of *Internal states*, set of *Beliefs About Other*. Based on all his knowledge, the IVA is able to select the correct action to perform, in order to achieve its goal. This process is done by the *Behavioral Engine* which will be explained later in this paper. Figure 2 shows the initial definition of Claire's knowledge.

```
((name Claire)
 (environment LIG's_Lab)
 (internalStates '((tiredness 0 100 80 DSC)
                   (anxiety 0 100 30 DSC)
                   (curiosity 0 100 60 -)))
 (assumptionsAboutOther '((john friend)))
 (mainGoal 'steal LIG's information)
 (secondaryGorals nil)
 (plans *P_Stealer*, *P_Walker*)
 (longTermBeliefs '((I'm a stealer)
                    ((I'm a woman)
                    empty-list)))
 (shortTermBeliefs '((I don't know where to find the information)
                    ((I'm in LIG's area))))
)
```

Figure 2: Agent specification.

2.1 Beliefs

Beliefs represent the knowledge of the IVA, These are a set of statements that the IVA believes to be true. The agent's beliefs are organized in two categories:

Long term beliefs (LTB) are beliefs that will not change during the entire simulation. These beliefs build up the everlasting memories of the agent. i.e.. *I'm a woman*

Short term beliefs (STB) are beliefs that may change. During the simulation some of these beliefs will be removed and some will be added, i.e. At the beginning when Claire arrives to the LIG's Lounge she has a believe: *I have not danced yet*, but when she dance, this statement will be converted into *I have danced*. The IVA remembers things for a given period of time. Adding an *expiration period of time* to each of the short term belief simulates this. Short term beliefs are *forgotten* when the time is over.

The belief's semantic is a *positive close world*, this means that if something is not specified inside the agent's beliefs is consider to be false. i.e. If we want to search into Claire's memories to see if she has been in the bar before, we look for *(I have been in the bar)* inside her beliefs. If this belief is not found, then we assume that *(!(I have been in the bar))* is true.

2.2 Goals

An IVA has one main goal and one or several subgoals. The main goal is the objective that the IVA is trying to achieve at a certain moment. Without a goal the agent is lost, aimless, and no plan will be invoked because there is nothing to fight for. i.e. *I want to drink a beer*.

During the process of achieving a goal, an IVA has to deal with smaller goals on which the outcome of the larger one relies on, that is what we call subgoals. i.e. Looking at the previous main goal's example, some things are needed to get a beer, therefore during this goal achievement there will be subgoals such as *go to the bar, ask if they sell beer, ask for the beer, pay the beer, and finally drink it*.

2.3 Competing plans

An IVA uses a set of competing plans that specify a sequence of actions to be performed in order to reach its main goal. A competing plan $\mathbf{P}_i$ is described as:

$$\mathbf{P}_i = (\mathbf{is}_i, \mathbf{pc}_i, \mathbf{ef}_i)$$

where:

- $\mathbf{is}_i$ is a list of internal states to be checked before the plan can be executed. Each of the internal states has an associated valid value or range. The *Behavioral Engine* is in charge of checking whether the current level of the agent's internal state is adequate (within a given range) for executing the plan. i.e. For Claire to be able to dance, her tiredness level can not be too high.
- $\mathbf{pc}_i$ is a list of preconditions which have to be true before the competing plan can be triggered. The preconditions belong either to the agent's beliefs or to the general knowledge stored in the world.
- $\mathbf{ef}_i$ is a list which contains the effects of a plan execution. When a plan is selected changes at agent or at world level will occur. These changes are consequences of the plan's effects (See Figure 3)

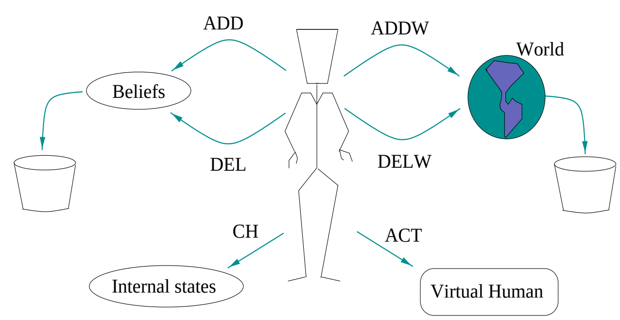

Figure 3: Plans' effects.

ADD used to add new beliefs to the agent, beliefs that will became true when the plan is executed. i.e. Claire wants to take a train, and she has a plan that tells her that a ticket is a requirement for taking a train, also one of her beliefs tell her that she does not have one. The *Behavioral Engine* then activates the plan for going to the counter in order to get it. At the end of the plan execution Claire will have the ticket, and the dynamic beliefs will be accordingly updated, now the statement *I have the ticket* is part of her *Short Term Belief*.

DEL used for deleting beliefs that won't be true any more after the plan's execution. Referring to the previous example, once Claire gets the ticket, it is not true any more that she needs one.

ADDW used to add general knowledge. This is an atomic operation in order to ensure the system reliability. i.e. Claire sit on chair number 1, she will update the world common knowledge by sending *Chair1 being used* as new general knowledge.

DELW used to delete general knowledge that is not true any more. This is also an atomic operation. i.e. When Claire stand up, she has to delete the general knowledge *Chair1 being used*.

ACT used to send an action to be performed by the virtual humanoid. i.e. Claire wants to drink a beer and her plans tell her to go to the bar to get it. In order to execute the plan successfully an ACT action is generated *Claire goes to bar*.

CH reflects some changes in the internal state of the IVA. i.e. When Claire dances, her tiredness level increases.

A competing plan is executable at time *t* when all of its preconditions are observed to be true and when all the internal states have the desired levels in that specific moment. One example of a plan is showed by Figure 4.

Claire will execute this plan when she arrives at the bar, if and only if she is a curious agent. To be a curios agent in this case means that her level of curiosity is bigger than 50. The plan also checks if she is at the bar and if she had never being there before. The conclusion of this plan is to perform the action *Inspect the place* which consist of looking around it. Some updates to the *Short Term Beliefs* are also needed: *She has being in the bar* and *She is inspecting the bar*

Each agent has a set of plans available for choosing which is the next action to perform. Some of

```
(RememberPlan
  (newPlan 'inspect-place
    '((curiosity 50 >))
    '((is at (? place))
      (! (has been is (? place)))
     )
    '((Act (inspect the (? place)))
      (Add (inspecting the (? place)))
      (Add (has been in (? place)))
     ))
   *P_Walker*)
```

Figure 4: Plan example.

the plans' *ADD* effects are at the same time preconditions of another plans, this generate a light connection between plans, as showed by Figure 5. One node of the tree represents a plan. One node is a father *(i.e. node 1)* if it has an *ADD* effect that at the same time is precondition of another plan, in this case called son *(i.e. node 2 is son of node 1)*. The ovals indicate internal values or stimulus and circles represent external events, which are necessary to trigger the plan. The leaves of the plan are the actions to be performed by the virtual humanoid.

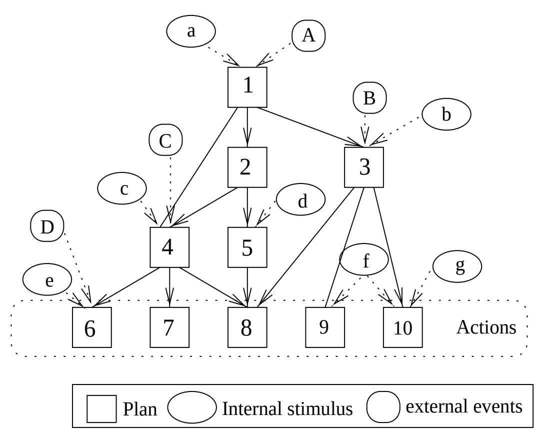

Figure 5: Plans' structure.

2.4 Internal states

The agent has a set of internal states representing physiological or psychological variables of the virtual humanoid, such as level of hunger, fear or boredom. This internal states acts as a stimulus for the agent, i.e. *a high hunger level will stimulate the agent to eat*. An internal state $\mathbf{is}_i$ is described as a tuple:

$$\mathbf{is}_i = (\mathbf{n}_i, \mathbf{min}_i, \mathbf{max}_i, \mathbf{c}_i, \mathbf{cat}_i)$$

for any given internal state *i*:

- $\mathbf{n}_i$ is his name, such as hunger, tiredness, sadness and happiness.
- $\mathbf{min}_i$ is his minimum accepted value.
- $\mathbf{max}_i$ is his maximum accepted value.
- $\mathbf{c}_i$ is his current value.
- $\mathbf{cat}_i$ is his category. (To be explain in section 3.2)

Internal states are constantly being adjusted, as the simulation evolves and plans are adopted. Changes in the internal state are consequences of:

- The *autonomous growth or damping* associated to the internal state. i.e. *The hunger's level increases over time in a specific rate* if no food is given.
- The *side-effects* of an active behavior. i.e. *Eating decreases the level of hunger*

Internal states have two different effects in the plans' executions:

- *Plan Inhibition* the internal states are values that are checked to trigger a plan, some of the plans are inhibited because some levels of the internal states are too high or too low. i.e. In the plan showed before, if Claire is not curious enough (her curiosity level is lower than 50) the plan will not be triggered.
- *Action Intensity Modulation* the internal state influences the strength of the action taken by the virtual humanoid. i.e. The way Claire walks depends on her level of happiness, the higher the happiness the livelier the way she walks.

2.5 The Behavioral Engine (BE)

The paradigm of action selection is not a recent topic, and is not only a problem focused by artificial intelligence researches. Some ethologists such as Tinbergen [11], suggested that the behavior of an animal should be considered as a result of competing behaviors, where each one follows a self-interested goal. In our model the action selection problem is handled by the IVA's *Behavioral Engine (BE)*, who decides which of the competing plans should control the IVA, as shown in the Figure 6.

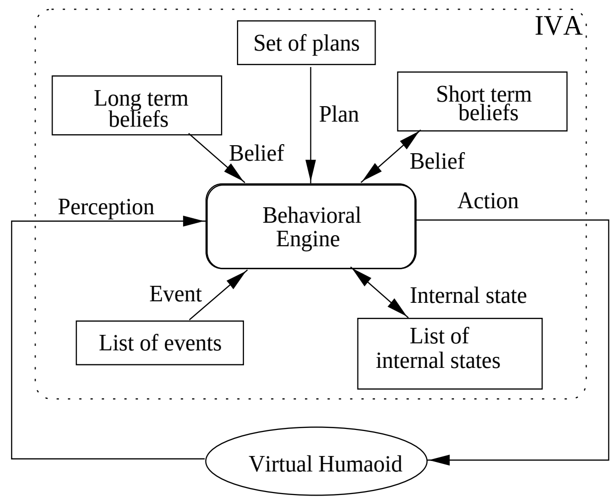

Figure 6: Behavioral engine.

The *Behavioral Engine* first checks in the pending event list for those events that trigger in a specific time slot. The selected events are integrated in the IVA's knowledge, being associated by default with the short-term beliefs, otherwise if it is specified it goes to the long term beliefs. i.e. When triggering the event *Music turned on*, the agent will update his short term belief *Music on*. The BE also checks the perception of the environment to see if this coincide with his beliefs state and makes the necessary updates.

Then the BE chooses the suitable plan for that specific time slot, based on the interplay of IVA's internal and external factors. Each plan has some pre-conditions, some of these concerns about the IVA's beliefs, others concern about the general knowledge stored in the world agent. Also the levels of the internal states has to be suitable to be able to perform a plan, where suitability is defined as having the minimum level specified in the plan's requirements, as mention before.

The BE should go through the agent's plans hierarchy, finding the plan to be executed. More than one plan can be trigger in a given moment, but only one action can take place. i.e. Plan A, B and C are trigger in time $\mathbf{t}_i$, and plan A and C has an *ACT* effect. By default, first checked plan is first to be executed, in this case is plan A and an action $\mathbf{a}_i$ is started. Then plan B is also triggered, in this case there is no action to be performed, just internal updates, so the plan is accepted and also executed. But when the BE tries to trigger plan C, this is not executed because it has an *ACT* effect, and there is already one action being performed. Then the BE store the action of plan C as a pending action.

Some authors like McFarlands [6], pointed out the importance of the interplay between internal and external factors: plans should guide to goal achievement based on the state and knowledge of the agent, but also taking advantage of opportunistic situations in the world. This characteristic is also reflected in our model, because the BE is able to ask for environment's data at any moment. If the Behavioral Engine notice that one of the preconditions is not fulfilled by the agents beliefs, he will check in the environment data [1]. i.e. If Claire wants to take chair number1, one precondition for this is : *Chair1 free*. Claire does not have this information, therefore her BE looks up the world knowledge if *Chair1 free* exist. If it exist the plans will be trigger.

The BE is also in charge of updating the dynamic beliefs and the internal states of the IVA, and updating the world general knowledge.

3 Trusting agents

Many psychologist have studied how humans use trust in every days life, but almost no author in Artificial Life has mention it. We think that to be able to simulate believable intelligent virtual humanoids we need to implement as close as possible the humans' behaviors, and trust can not be excluded.

Before going any further lets assume the definition of trust given by R. Swinth[10]: "Choose to take an ambiguous path that can lead to a beneficial event or a harmful event depending on the behavior of the other person - where the harmful event is more punishing than the beneficial event is rewarding".

In our model each IVA is autonomous, and he can accept or reject an order coming from the user or from another agent. Each IVA has a set of Beliefs about others in which he stores the trust levels associated with them (See Figure 1). An IVA sees the user as another agent, and depending on the user's category he will accept the order or not. The levels of trust will evolve during the simulation, but before explaining how does it works lets discuss a little about human trust. Some researches in psychology had shown that: *"Trust, once established in some degree, is often self-reinforcing because individuals have stronger tendencies to confirm their prior beliefs than to disprove them"* [4]. This characteristic can be explained through a hysteresis curve as shown in Figure 7. Lets imagine a human A who does not trust in B. At the beginning it is very difficult to convince him to trust in B, B must do a lot of effort to make A start trusting in him a little bit. When this happen, agent A will be dramatically easy change his opinion to start trusting in B. Once the level of trust is very high, it gets stuck a little bit and little progress can be see. The same idea is applied to stop trusting in somebody, once agent A trust in B deeply, it is very difficult to make him to change his opinion, but after some small steps, the trust level will drop dramatically, and then just small changes can be noticed.

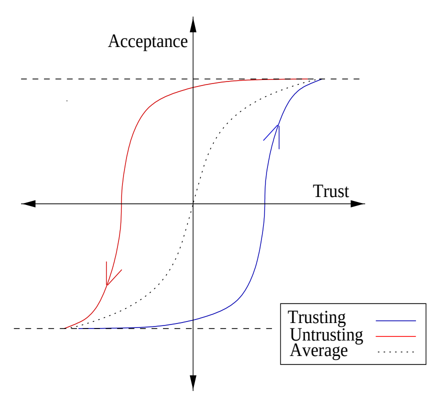

Figure 7: Trusting curve.

To be able to show this behavior we have chosen some categories, from the lowest trusting level to the highest trusting levels: 0-Low, Low, Low-

[1] To be explained later in this paper

Medium, Medium, Medium-High, High, High-Blind, Blindly.

All IVAS storage the name of the other agent and the level of trust associated to it. The value of acceptance for any order coming from a user with certain trusting level can be seen from Figure8, in which the higher the trust level, the higher the possibility of accepting the order, and the lower the trust level, the lower the possibility of accepting it. These values of trust are not fixed and they will evolve during the simulation, as we will explain later in this section.

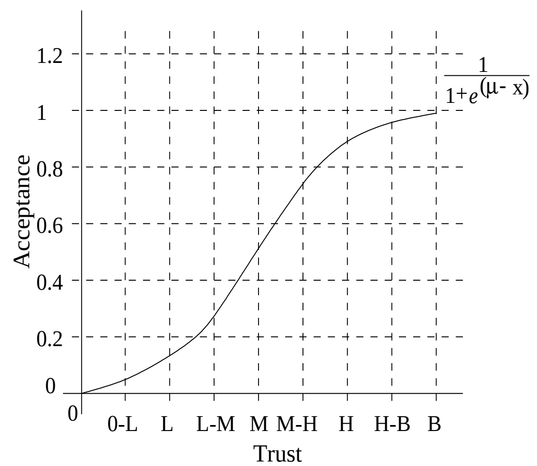

Figure 8: Levels of acceptance.

If the IVA does not have an entry a specific user in his Beliefs about others, then a new entry is created with a default value: Medium Level.

3.1 Controlling the agent at two levels

The user can interact with the IVA through a graphical user interface where he can send some orders to the agent. The user has two types of command that can be sent to any IVA:

Beliefs the user wants the agent to start believing something. For example, the user want that the agent A believes that *"It is raining"*, on other words the user wants that:

(BEL A (It is raining))[2]

This type of command is intended to be used as collaboration paradigm, the user want to help the agent to succeed with his goal. i.e. Claire is lost in the virtual world, she is looking for tickets and she doesn't know where they can be found, and neither the world's common knowledge has that information[3] The user can pass this information at real time through the graphical user interface, *tickets are found at counter*. If Claire accept the belief which is sent, maybe her behavioral engine will trigger some new plans which will solve the situation of finding a ticket.

Orders the user want that the agent perform an action directly. For example the user want the agent A to dance, in our syntax:

$$(ACT(dance))$$

This type of command is sent to control the agent, instead of trying to cooperate with him, we want the agent to do something.

When the IVA receives one of these commands, the *behavioral engine* selects the sender and checks in the set of *beliefs about other* the trusting level of that user, who is seen as another agent. Then the *behavioral engine* applies the adequate acceptance probability to accept or reject this command. But what does it mean to accept a command? For a belief this means to be added to the agent's beliefs, and for an action it means that the agent will perform it in the next slot of time if he is not doing anything at the moment, or when he finish the current action.

3.2 Categorizing the internal states

To be able to implement the dynamic changes on the trusting levels, we categorize the internal states, because these are the based of this trusting model. The IVAs will update the values of the trust based on dramatic changes in his internal state levels.

We wont make any concrete definition of what an emotion means, because this is a topic that still been discuss by psychologist, neither we want to explain the difference between moods, feelings, passions, needs, or sensibilities, because this is not clearly distinguished and it is not our goal to do it so.

[2]Following the Cohen-Levesque multi-modal logic.

[3]The world agent and all the common knowledge will be explained in the next section.

We categorize the internal states as ascendants and descendants, for example, happiness is ascendant, the higher the level the better for the IVA, and hunger is descendant, the lower the level the better. Some internal state won't be categorized as curiosity, because we can not say that a high curiosity level is good or not.

Looking at some internal state we can categorize them as shown in Figure 9.

Emotion	Category	Emotion	Category
Impatience	DSC	Love	ASC
Enthusiasm	ASC	Curiosity	—
Boredom	DSC	Excitement	—

Figure 9: Categorazing the internal states.

3.3 Changing the trust levels

When the IVAs accepts a command coming from the user, the first action that the behavioral engine performs is to check which plans are triggered, and the influence that they have in the internal state. If there are changes bigger than the 20% of the maximum level of the internal state, then, the agent pass to analyses the category of the internal state.

- For an increment bigger than the 20% in ascendant internal state, the trust level of the user will increase by one category, because this means that what the agent accept makes him feel better, and he can trust more in that user.
- For an increment bigger than the 20% in descendant internal state, the trust level of the user will decrease by one category, because this means that what the agent accept makes him feel worse, and he will trust less in the user next time.
- Increments or decrements in non-categorized internal state does not count.

4 The Intelligent Module

The IVAs are not isolated in our system, they are one active element of an intelligent module we have called IntelMod. The IntelMod is composed of five components: Events' interface, Dispatcher Module, World Agent, IVA's Families and the Intelligent Virtual Agent already presented (See Figure 10).

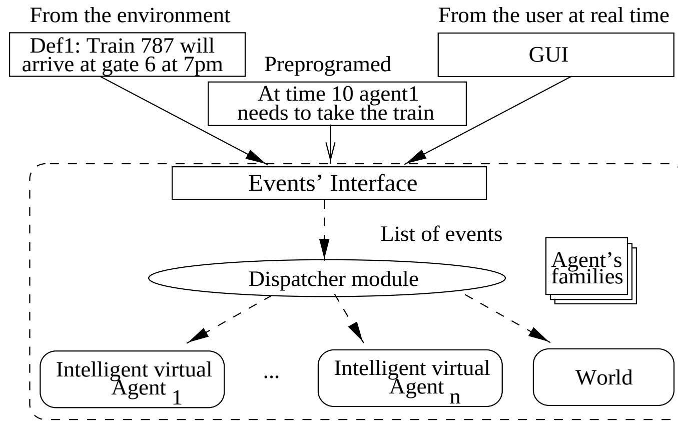

Figure 10: IntelMod.

4.1 Events' Interface

The events' interface is in charge of passing the events sent by the user to the dispatcher module who will decide their final destination.

In order to interact with the IVA, and eventually animate the virtual humanoid, we have defined an event framework. Events are sent to the intelligent virtual agents and the world. An event $\mathbf{E}_i$ has the following structure:

$$\mathbf{E}_i = (\mathbf{name}_i, \mathbf{content}_i, \mathbf{target}_i, \mathbf{timeTrigger}_i, \mathbf{id}_i)$$

An event $\mathbf{E}_i$ is delivered to the IVA specified by $\mathbf{target}_i$ in a given time slot $\mathbf{timeTrigger}_i$, then the IVA is in charge of manipulating the data contained in $\mathbf{content}_i$. The events come from three different sources:

The environment's definition the events coming from the environment's definition belong to a specific world, and they must be specified in advanced by the programmer. (e.g. *The rock music in the bar is turned on at 10pm*).

User's pre-programmed file , the user can define, prior to the simulation, some events to be triggered at a specific time during the simulation. (e.g. *The goal of Claire is to have fun*). In opposition to the events coming from the environment definition, these events belong to the agents, rather than to the world, but they are treated equally in the simulation.

Graphical user interface , during the simulation, the user is offered the possibility to specify agent's goals in real time, to define new events that could change the development of the simulation, or send orders to the agents.

4.2 Dispatcher Module

Once an event has been retrieved from the specific source, the events' interface passes the list of events to the dispatcher module which is in charge of delivering the event to the concerned agent, as shown in Figure 10. Events can have three different kinds of targets:

A specific IVA in this case the dispatcher module uses the target name to find the corresponding agent.

All IVAs these messages are delivered by the dispatcher module to all IVAs available, like a broadcast message.

World agent in this case the message concerns only to the world, and the message is passed to it.

4.3 World Agent

When the IntelMod starts running, it does not have any information about the system, and ignores everything about the environment, the distribution, the position and orientation of the agents and the objects' position.

An instance of a world agent must be already active for an IVA to be able to load and start interacting with the associated virtual humanoid. The world agent manages the general information of the environment, stores the names and the IDs of all active humanoids in the virtual environment. Also each IVA has a reference to the IVA's world which it belongs to, in order to be able to access information about others through it.

The first world agent's action is to connect to the system to get the number of virtual humanoids available in the virtual world and their respective names and IDs. When the world agent has this information, the intelligent agents are able to connect and to start interacting with the virtual humanoids.

The world agent stores common and relevant information for all IVA's. It is organized in two different groups:

Static Common Knowledge where resides all the unchangeable world's information. (e.g. *Cold drinks are found in the bar of the LIG's Lounge.*) This knowledge is loaded into the world at the beginning of the simulation, and only additions are allowed at run-time.

Dynamic Common Knowledge where all the dynamic world's information resides. This information can be changed dynamically during the development of the simulation. (e.g . In a specific moment, the *music is off in the LIG's Lounge*). But later in the simulation, there will be an event that tells the world agent that the music is turned on.

An example of the world specification is showed in Figure 11.

```
((name 'LIG's_Lounge
 (agentsList '(Claire))
 (staticCommonKnowledge
   '((drink is found at bar-place)
   ((dancing can be done at dancing-area)
   ((relax can be done at chair)
   ((time is found at watch)
   empty-list)))))
 (dynamicCommonKnowledge
   '((chair_1 is free bar-chair)
   ((chair_2 is free bar-chair)
   empty-list)))
)
```

Figure 11: World specification.

4.4 IVA's Families

A society role is described by a set of skills, which the agent needs to have in order to fulfil its role requirements. This skills are gruped and specified inside the IVA's families. i.e. client role and waitress role in a bar. All agents belong to one or several agent families. Two agents belonging to the same family have the same abilities (equal

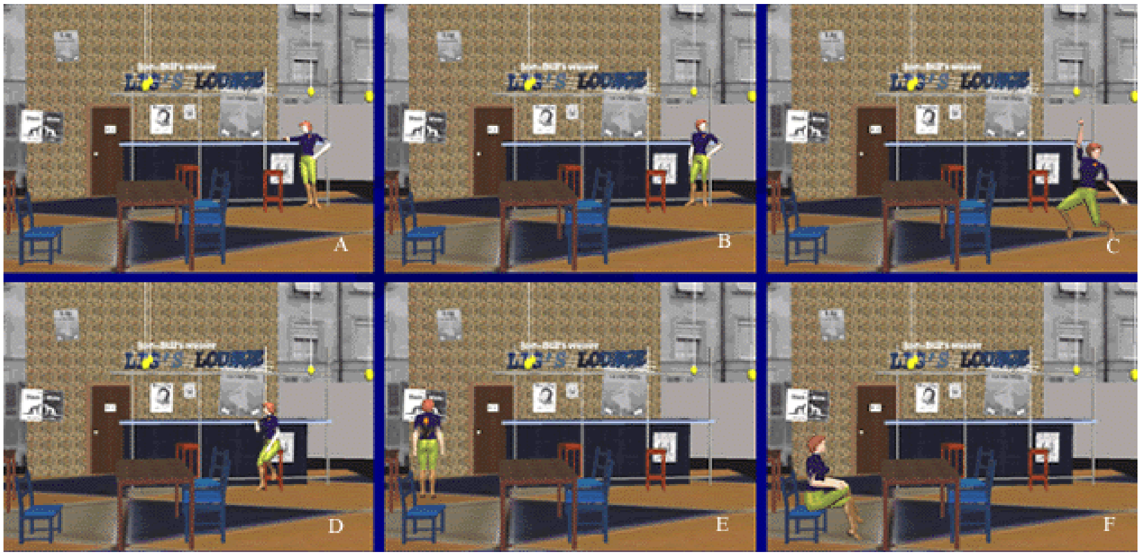

Figure 12: LIG's Lounge's simulation.

set of plans) but their beliefs and emotional states are different at any given moment. e.g. In a virtual LIG's Lounge we can find clients and waitress (Figure 12). All clients are in the LIG's Lounge for the same reason: they would like to have a nice time, but they behave differently because they have different internal states, different beliefs and different assumptions about others. The clients' family has all the plans that allow an IVA to take a drink, take a chair, dance or just meet some people. Waitress does not care about dancing or having fun, they are just working and serving the clients, therefore the waitress' family has another set of plans. A client does not *know* how to serve someone, unless we create a client-waitress' family.

When the user wants to instantiate an IVA, one of the available IVA's families is picked up, and a message is sent to the world agent in order to obtain the control of a virtual humanoid. Once the IVA has the name and the ID of the virtual humanoid, it creates a socket using this name, and starts the bi-directional communication between the virtual humanoid and the IVA.

5 A real example

As we have been mention in this paper, we have developed a simulation of an LIG's Lounge, where the agent can go to have a drink, talk, listen music and dance. In our case we put a client agent called Claire. Claire went to the bar because she wanted to dance and drink something, and because this can be found at a bar. When she arrives she inspects the place because she has never been there before. Then she decides to stay. She is not tired enough, therefore she decided not to sit, but she goes to the serving bar looking for something to drink (See Figure 12a). Nothing is happening... then she starts getting bored (See Figure 12b). A *music on* event occurs, and she goes to dance(See Figure 12c). Then another event occurs *music off*. She stops dancing and she decides now to sit. Her temperature is high, then she sweats, and she hates this, then she cleans her face and takes a seat. She is attracted by a poster, she goes to see it, and then sit again. Her boredom level has increased a lot, reason why she just leaves the bar.

6 Conclusions

In this paper we presented some key points for simulating human-like virtual actors. We presented the action selection paradigm as a process compound of: goal achievement planning, dynamic beliefs management, evolving goals, internal states and confidence levels. We also presented a new approach to control virtual humanoids, based on *Human Trust* theories, where the agent is completely autonomous of accepting or rejecting an order.

Enhancements to be done include improving the memory model, the management of unsolved situations due to lack of suitable plans to apply in that specific moment. The trust model will be applied in virtual humanoids direct communication and collaboration. Our future work will focus on these problems and we will deal with the implementation of verbal communication to be able to add cooperation, and collaborative group behaviors to the virtual environments.

7 Acknowledgments

The authors are grateful to Soraia Musse, Fabien Garat, Elsa Schweiss and Nathalie Farenc for their participation in the development of the ViCrowd, Server and ENVIR which makes possible to test the IntelMod. The research was sponsored by the Swiss National Research Foundation.

References

[1] Elizabeth André. Employing ai methods to control the behavior of animated interface agents. *Applied Artificial Intelligence*, 13(4-5):45–448, June-August 1999.

[2] J. Bates, A. B. Loyall, and W. S. Reilly. An architecture for action, emotion and social behavior. In *Proceedings of the Fourth European Workshop on Modeling Autonomous Agents in a Multi-Agent World*, Italy, 1992.

[3] B. M. Blumberg and T. A. Galyean. Multi-level direction for autonomous creatures for real-time virtual environments. In *Proceedings of SIGGRAPH '95*, pages 47–54, 1995.

[4] Robert Hinde and Jo Groebel. Cooperation and prosocial behaviour. Cambridge University Press, 1991.

[5] J. C. Lester and B. A. Stone. Increasing believability in animated pedagogical agents. In *Proceeding of the First International Conference on Autonomous Agents*, pages 16–21, 1997.

[6] D. McFarland. *Animal Behavior*. Logman Scientific and Technical, Harlow UK, 1993.

[7] K. Perlin and A. Goldberg. Improv: A system for scripting interactive actors in virtual worlds. In *Computer Graphics*, 1996.

[8] A. S. Rao and M. P. Georgeff. Modeling rational agents withing a bdi-architecture. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, San Mateo, CA, 1991. Morgan Kaufmann.

[9] J. Rickel and W. L. Johnson. Animated agents for procedureal trining in virtual reality: Perception, cognition and motor control. *Applied Artificial Intelligence*, 13(4-5):343–382, June-August 1999.

[10] Robert L. Swinth. The establishemant of the trust relationship. *Journal of conflict resolution*, 1967.

[11] N. Tinbergen. *The study of Instinc*. Oxford University Press, New York, 1951.