Gillies, Marco and Ballin, Daniel

Integrating autonomous behavior and user control for believable agents

Integrating Autonomous Behavior and User Control for Believable Agents

Marco Gillies[*]
University College London Adastral Park Campus
Ross Building pp1, Adastral Park
Ipswich IP5 3RE, UK
m.gillies@ucl.ac.uk

Daniel Ballin
Radical Multimedia Lab, BT Exact
Ross Building pp4, Adastral Park
Ipswich IP5 3RE, UK
daniel.ballin@bt.com

Abstract

Autonomous agents can help users by taking on a substantial workload, and performing tasks that are too complex for a human. However, in some systems complete autonomy is undesirable as it removes control from the user. It is therefore important to include some user control while maintaining the reduced workload associated with an autonomous system. This is particularly true of user avatars in virtual worlds, appropriate non-verbal communication is too complex to be directed explicitly and should therefore be controlled by an agent. However, the non-verbal communication should express the feelings of the user, so a degree of human input is needed. This paper presents Demeanour, an autonomous system for generating body language in avatars, which integrates user input based on a three level methodology of customization and control.

1. Introduction

Recent years have seen considerable interest in multi-user graphical virtual environments, both in academia and in commercial applications, particularly in multi-player computer games. In these environments users are represented as graphical “bodies”, called avatars. Avatars have traditionally been passive graphical objects that only act through direct commands of the user. They are very different from the autonomous agents that also inhabit virtual environments, which are computer controlled, behave pro-actively and respond autonomously to users’ actions. Recently, however, researchers are starting to realize that it is important to endow avatars with some of this autonomous behavior, turning them into ”semi-autonomous” avatars (e.g. Vilhjálmsson and Cassell[21], Sengers, Perry and Smith[19] and Gillies, Ballin and Dodgson[11]). Adding autonomy to an avatar can allow it to exhibit complex behavior without requiring the user to perform time-consuming control sequences, which may well distract from other tasks.

One area in which autonomous behavior is important is expressive behavior. People are constantly in motion, making often very subtle gestures, posture shifts and changes of facial expression. We do not always consciously notice making these movements and neither do we always consciously notice others making them. However, they are vital to reading other people’s feeling and to our evaluation of others. Vilhjálmsson and Cassell[21] point out that this type of behavior is difficult for a user to directly control, as in real life we are producing it constantly and subconsciously. The effort of constant control is very time-consuming and distracting from other tasks. As the behavior is subconscious we often do not even know which behavior is appropriate at a given time. Expression is therefore an area where it can be very useful to add autonomy to an avatar. However, it also demonstrates that autonomy is not sufficient on its own. It is not sufficient for the system to produce behavior if it is unrelated to the emotion or attitude that the user is attempting to express to other users. The meaning of the expression, if not the details, should be under the control of the user. It is therefore important to have an autonomous system that is integrated with a user interface. This interface must be unobtrusive and not distract the user excessively from other tasks, if the advantages of autonomy are to be retained. In this paper we present a methodology for integrating autonomy and user control, and then go on to describe an implementation.

2. Methodology

The aim of this research is to find ways in which end-users can influence the behavior of semi-autonomous avatars without causing an excessive overhead on their other activities. One method would be to measure the affective state of the user, e.g. using computer vision based anal-

* We would like to thank BT Exact for funding this work, and Amanda Oldroyd for supplying the graphical models.

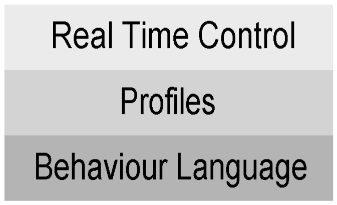


Figure 1. The different levels of control within the Demeanour framework.

ysis of facial expression, and mapping it onto the avatar. Unfortunately such methods are currently difficult or unreliable. Also users often want to hide their real feelings. Therefore our aim is to user interfaces based on explicit control but make them as unintrusive as possible. There are two types of explicit control. Users can give commands to the agent in real time while using the virtual world and interacting with other agents, we call this real time control. It is also possible to perform customizations on the agent's behavior before using the virtual world or between sessions, we call this off-line customization In order to minimize overhead while using the virtual world it is desirable to have as much control as possible provided through off-line customization. There are two types of person who might want to customize behavior. Firstly unskilled end-users, who will possibly be using the virtual world for entertainment, should not be required to learn complex tools but should be able to adapt agents to their requirements. The virtual worlds themselves are created by professional designers who should be provided with powerful tools to design the behavior of the agent in the world. This is important as the behavior of agents is likely to vary considerably between different worlds used for different purposes (e.g. between a business conferencing environment and an adventure game).[1] We therefore propose a methodology for user control of agents that divides control and customization into three levels shown in figure 1:

Behavior Language. The behavior of the agent should be entirely definable by the designers of a virtual world, not controlled by a fixed program. This capability is provided via a definition language. As they are not expert programmers, this should be easy to learn, analogous to HTML. It is likely that a controller will be created once per environment and shared by all agents in that environment. Customization of individual agents' behaviors can be done via profiles.

1 World designers can also customize the behavior of the avatars by changing content on which the animation is based (for example, bvh motion files) based this is not the focus of this paper.

Profiles. It has been shown that users of on-line worlds are very keen to customize the graphical appearance of their avatar[8] and it is likely that they will also be happy to perform similar customization on the avatar's behavior, if easy to use tools are provided. We therefore propose a set of such tools that can be used to define a profile for their agent. This profile defines the unique aspects of its behavior.

Real-time Control. It is also necessary for a user to provide some control of their agent while interacting on-line. It is important that this is unobtrusive as possible and well integrated with the other tasks to be performed in the world.

3. Related Work

Our work builds on a body of work on autonomous characters for virtual environments, for example, Blumberg and Galyean[5]; Badler, Phillips and Webber[2]; Tu and Terzopoulos[20], and Rickel and Johnson[17]. There has been extensive research on autonomously producing expressive behavior a number of types including facial expression(Pelachaud and Poggi[15]), eye gaze(Cassel et. al. [6], Rickel and Johnson[17] and Gillies and Dodgson[12]), gesture (Cassell et. al.[6]), style of motion (Chi et. al.[9]) and, like our current implementation, posture (Cassell, Nakano and Bickmore[7]), Bécheiraz and Thalmann[4]).

Vilhjálmsson and Cassell[21] introduced the idea of adding autonomous behavior to user avatars, which was developed by Sengers, Perry and Smith[19]. Though Vilhjálmsson and Cassell[21] do provide some user control it is very limited. The TEATRIX system by Paiva, Machado and Prada[14] combines direct control of an autonomous character with a more reflective level of control which takes users (in this case school children) out of the virtual world allowing users to update the internal state of their character, while reasoning about their role in the story. However, this form of control is not suitable to all applications, so a method of control that is more integrated with the main interface is needed. This work also builds on a longer tradition of methods for directing autonomous characters including Blumberg and Galyean's multi-level control[5] or Perlin and Goldberg's Improv system[16]. Similar issues are also important for tele-operated robotic systems[22] where the requirements of an operator must be applied to an autonomous robot. In particular our three-layer control model is partially inspired by the work of Scerri, Ydrén and Reed[18].

4. An Example Behavior

The following description of the Demeanour framework will use as an example a behavior network we have developed. It models the way people relate to each other or their attitude to each other and is based on the work of Argyle[1].

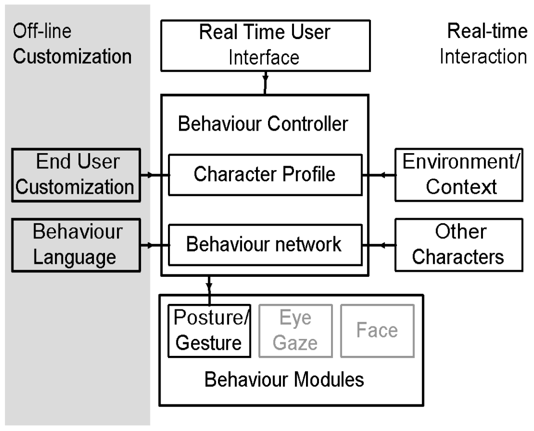


Figure 2. An overview of the Demeanour Framework

In our model the attitude of one person to another is expressed through posture and, to a more limited degree, gesture. It is discussed in more detail in [10].

Though there is an enormous variety in the way that people can relate to each other Argyle identifies two fundamental dimensions that can account for a majority of non-verbal behavior, affiliation and status. Affiliation can be broadly characterized as liking or wanting a close relationship. It is associated with close postures; either physically close such as leaning forward or other close interaction such as a direct orientation. Low affiliation or dislike is shown by more distant postures, including postures that present some sort of barrier to interaction, such as crossed arms. Status is the social superiority (dominance) or inferiority (submission) of one person relative to another, we will not discuss it directly in our examples.

It is also very important that agents are able to react to each others' behavior. The relationship between the attitude behavior of two agents can take two forms, compensation and reciprocation. Argyle presents a model in which people have a comfortable level of affiliation with another person and will attempt to maintain it by compensating for the behavior of the other, for example, if the other person adopts a closer posture they will adopt a more distant one. Conversely there are times where more affiliation generates liking and is therefore reciprocated, or where dominance is viewed as a challenge and so met with another dominant posture. Argyle suggests that reciprocation of affiliation occurs in early stages of a relationship.

5. The Demeanour Architecture

We have developed a non-verbal communication framework, Demeanour, which embodies three-level control. Figure 2 shows an overview. The main section of Demeanour is a behavioral controller that determines the type of behavior to be performed at a given time. This contains a behavior network that gives the structure of the controller and is defined using the behavior language. The details of the behavior can vary between agents and is determined by an agent profile. This profile is defined by end-users during an off-line customization step. Sections of profile can also be loaded and unloaded during real-time interaction, enabling different behavior depending on context. Agents react to the behavior of other agents and the end-user can also control the agent's behavior using a text-chat based interface. Finally the agent is actually animated by behavior modules that interface with the underlying graphics API[2]. Currently only body animation, covering posture and gesture is supported but we are planning to add eye gaze[3] and facial animation.

As shown in figure 2 the Demeanour framework combines a number of factors such as user input; context, and the behavior of other agents and from the result generates appropriate expressive behavior. Internally all these factors are represented as values of one of two types, continuous and discrete. Continuous values are floating point numbers. Discrete parameters are members of a discrete set of values, or enumeration, by analogy with C++. Enumerations can be user defined or the built-in boolean enumeration consisting of [true, false]. The outputs of the architecture are a number of parameters passed to behavior generating modules. These parameters are, again, either continuous or discrete.

How the input factors are mapped to outputs is defined by the user via the behavior language described in section 6.1. The mapping consists of a number of terms which are intermediary values calculated from other terms, including input factors. The following section describes terms and how they combine.

5.1. Terms

Terms are the fundamental components of the Demeanour framework. They are either continuous or discrete values calculated either from inputs or other terms. Some provide outputs to the behavior modules and some to other agents. As the value of a term can be the input of another term they can be described as a directed graph as shown in figure 3.

There are a number of different types of term depending on how they are calculated, as described in the next section. Section 5.3 describes an example of a behavior network built from a number of terms.

2 The current implementation uses TARA, BT Exact's scene-graph based graphics API

5.1.1. Types of term

Parameters are the simplest type of term. They have one single value and do not depend on other terms. The value of a parameter can be set in a number of ways. A default value is given when the parameter is defined in the behavior language. Parameters are the primary method of adapting the behavior of an agent either via profiles (section 6.2) or in real time (section 6.3). They are also used for input from other agents (section5.2). An example of a parameter is 'friendliness' which controls how generally affiliative the agent is towards other agents.

Sum of product terms are the basic way of combining other terms. As the name suggest the values of other terms are combined by addition and multiplication:

$$t = t_1 t_2 t_3 + t_4 t_5 + t_6 t_7 t_{8S} + \cdots$$

Sum of products are mostly restricted to continuous terms but boolean discrete terms can be combined using AND and OR rather than multiplication and addition.

An example of the use of sum of product terms is the calculation of affiliation. Affiliation depends of factors from the agent's profile, how friendly the agent is and how much it likes the other agent. It also depends on the behavior of the other agent, on how close or distant its body language is, among other factors. For convenience the calculation is split into two steps (and therefore two terms). The first is the calculation of the factors depending solely on the agent itself, called the desired affiliation. This is a sum of the various factors, each of which is multiplied by a weighting factor that can be altered in the agent profile. The affiliation itself is calculated by a weighted sum of the factors depending on the other agent's behavior added to the desired affiliation. This process is shown in figure 3.

Switch terms are analogous to switch statements in C++. The value of the term is chosen from the value of a number of other terms depending on the value of choice term (which is discrete):

$$\begin{aligned} t &= t_1 \quad \text{if} \quad t_c = a \\ &= t_2 \qquad\quad t_c = b \\ &= t_3 \qquad\quad t_c = c \end{aligned}$$

Switch terms are used when some discrete (often contextual) factor radically changes the meaning or use of a type of behavior. For example, head nodding shows approval when listening but is rarely used otherwise. Therefore the head nod behavior is either set to be proportional to affiliation or to zero depending on a boolean flag which specifies whether the other agent is talking.

Random group terms provide a way of producing a variety of possible outputs from a single term. For example, a distant posture might be expressed in a number of ways, leaning backward, crossing arms or looking away from the other person. A random group is a group of terms all of which take a single input term (distance in the example). The values of the members of the group vary randomly over time but are constrained always to sum to the value of the input, if all the members are continuous. Some members can also be boolean discrete terms. If so these have a probability of being true proportional to the input term. Each of these boolean terms has a numerical value attached that is subtracted from the sum of the continuous terms.

5.2. Input from other agents

An important part of autonomous behavior for avatars is the ability to react appropriately to the behavior of other agents. Behavioral controllers can therefore take into account the behavior of other agents. This is done in such a way that these reactions can be specified using the behavior language and customized using profiles, as with other aspects of the controller (see section 6).

This is implemented by allowing an agent access to terms in the behavior controller of other agents. Each network specifies a number of terms that are exported and therefore accessible to other agents. It also defines a number of parameters as being imported, i.e. corresponding to a term belonging to the other agent. When two agents start to interact the imports of one are matched, by name, to the exports of the other. For the rest of the interaction the values of the exported terms are used as the values of the imported parameters.

As shown in figure 3 the affiliation of an agent is calculated as the sum of a number of factors including the affiliative behavior of the other agent. Each of these factors is given a weighting factor whose sign determines whether the agent compensates or reciprocates a particular attitude. These weighting factors can be set in a profile and so can be made to depend on the agent itself or on context. In fact rather more complex behavior is possible, for example, an agent may reciprocate dominant behavior expressed as space filling postures but react with affiliation if it is expressed as relaxation.

5.3. Example network

Figure 3 shows in diagrammatic form a fragment of the attitude network that deals with affiliation (status is calculated in a similar way). At the top of the diagram the actual value for affiliation is calculated as a weighted sum of a number of factors (for the sake of clarity not all the factors used are actually shown). This is done in two stages; firstly the factors depending on the agent itself are calculated. These factors are represented as parameters (here 'liking of other' and 'friendliness' are shown). Then factors depending on the other agent's behavior ('close' and 'dis-

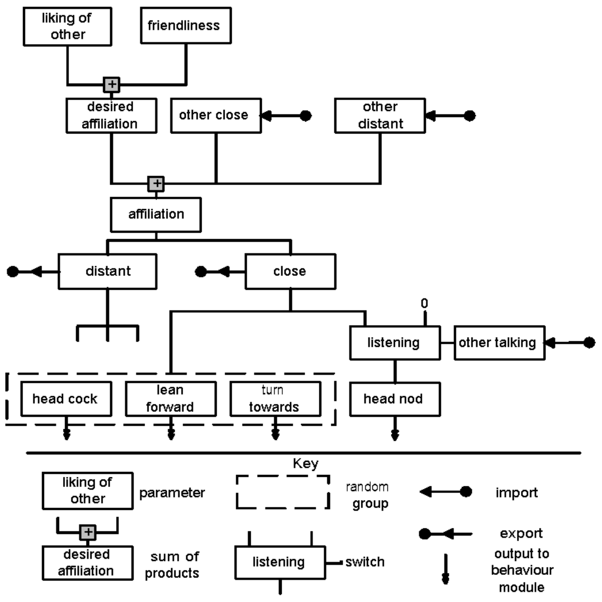


Figure 3. A section of a behavioral controller.

tant') are added in. These are import terms and are therefore taken directly from the controller of the other agent. As the behaviors associated with positive and negative affiliation are very different it is split into two terms, 'close' which is equal to the affiliation and 'distant' which is its negation. Both of these terms are constrained to be greater than 0. The 'close' term is then mapped into actual behavior (as is 'distant' but it is not shown in the diagram). In order to vary the behavior produced a random group is used. At semi-regular intervals a new combination of the various behaviors ('head cock', 'lean forward' and 'turn towards') is produced, this combination is always proportional to the value of 'close'. These behavior types are output terms and are passed as parameters to the underlying animation system. Another affiliative behavior is head-nodding, but this is only shown when the other person is talking. This behavior is controlled by a switch node ('listening'), based on a boolean import term which specifies if the other agent is talking. If 'other talking' is true then 'head nod' is proportional to 'close' otherwise it is zero. Each of the postures and gestures are given weights by the behavioral controller. If two that use the same body part are given non-zero weights they are either blended together if they are compatible (e.g. "head cock" and "head nod") or the one with the greater weight is chosen if they cannot be performed simultaneously (e.g. "arms crossed" and "hand in pocket").

6. The Different levels of Control

As discussed in section 2 the Demeanour framework provides multiple levels at which an agent's behavior can be customized and controlled. The first, the creating behavior controllers is aimed at expert, professional users. As this paper is mostly concerned with end-user interaction these first two methods will only be described briefly and the main focus will be on the methods aimed at end-user: off-line customization with profiles and real-time control.

6.1. Behavior Language

A simple declarative language is provided for defining the set of terms that make up the controller, an important customization task for virtual world designers. The declarative nature of the language mirrors the structure of the behavioral controllers, which consists of relationships and mappings between terms and is naturally statically defined. This simplified, declarative language makes defining the controller far simpler than it would be with a more general scripting language.

6.2. Profiles

Demeanour provides a system of agent profiles for off-line customization by end-users or world designers. A profile is a set of data that determines the unique behavior of an agent, how it differs from other agents. In Demeanour the behavior language determines the structure of the behavior network which controls the agent's actions. Customization is possible by altering the values of the parameter terms in the network e.g the weighting for how the closeness behavior of other agents affects an agent's affiliation. A profile can set this weighting to a positive value to achieve reciprocating behavior, negative for compensation, and a low or zero value for indifference to the other's status. Thus a profile consists of a number of values for parameters of the network (including weighting factors in sum of product terms), which are stored in an XML-based format separate from the behavior network definition.

6.2.1. Profiles and context Each agent has a main profile that determines their behavior in general. However, in real life people behave differently in different situations (e.g. a night club or business meeting), it is therefore important that the behavior of the agent can be adapted to different situation without explicit user control. The importance of this type of adaptation is brought out in work by MacNamee, Dobbyn, Cunningham and O'Sullivan[13]. Thus the profile also contains a number of sub-profiles that can be loaded and unloaded in different circumstances. The behavioral controller maintains a stack of these sub-profiles. The base of the stack is the main profile that is always loaded, as new sub-profiles are loaded they are pushed onto the stack overriding settings from sub-profiles lower on the stack. The sub-profiles themselves can be designed by end-user, as with the main profile, or by world designers, thus allowing them to adapt agents' behavior to situations that are not foreseen by end-users. Sub-profiles are divided into three types based on how and when they are loaded:

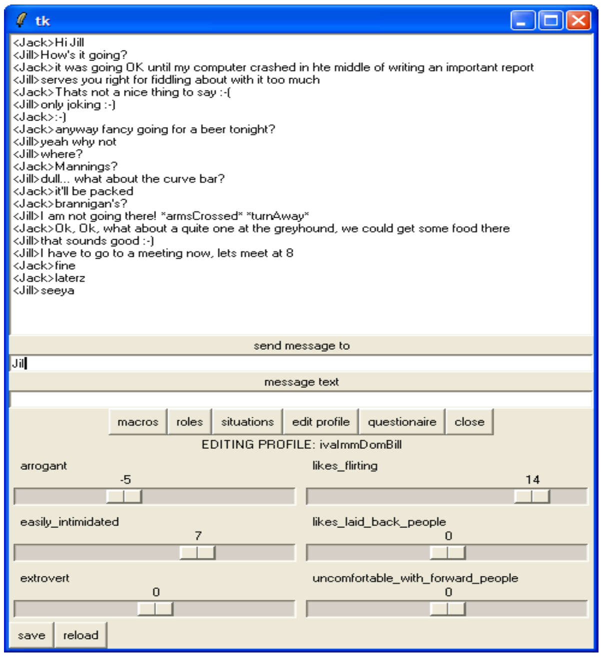
Person sub-profiles are loaded when starting interaction with a new person, and are specific to a particular individual. They thus represent the attitude to that person. The parameter 'Liking of other' in figure 3 is an example of a parameter that represents an attitude to a particular person and is suited to inclusion in a person sub-profile.

Role sub-profiles represent the behavior when the agent is performing a particular role, for example, an agent acting in a customer service role might have all low affiliation behavior disabled. These roles can be loaded by the end-user or automatically loaded in a given situation.

Situation sub-profiles are specific to a particular environment or context in the world, for example, flirting behavior might be disabled in an office environment but re-enabled in an office party context. These sub-profiles are loaded automatically when a situation is entered and apply to all agents in that situation.

6.2.2. Profile design tools Profiles are a means of customization for end-users and as such it is important that there are easy to use tools with which to design them. The most direct method is to assign values to parameters directly whether by hand editing files or via a user interface. However, parameters are often closely linked to the internal workings of the behavioral controller and not necessarily intuitive to end-users, so this method should generally be confined to world designers and advanced users.

We propose the use of "adjectives". These are names in plain language that describe a particular character trait or group of traits that is understandable to end-users. These adjectives are mapped onto actual settings of the internal parameters, each adjective affecting a number of parameters. For example, 'extrovert' might combine dominance with high affiliation while 'easily intimidated' might indicate compensation behavior to dominance (i.e. responding submissively to dominant behavior). Each adjective is a fixed set of parameter values and therefore is itself a self contained profile. They can be designed at the same time as the behavior network, through direct profile authoring tools as above. An end-user designs their profile as a combination of the adjectives. They are presented with series of sliders each labelled with an adjective name (as shown in the bottom half of the window in figure 4), the values of the sliders represent the proportions of the various adjectives. The values contained in the adjectives are multiplied by the slider values and summed to obtain the final profile. This provides


tk

```
<Jack>Hi Jill
<Jill>How's it going?
<Jack>it was going OK until my computer crashed in hte middle of writing an important report
<Jill>serves you right for fiddling about with it too much
<Jack>Thats not a nice thing to say :-(
<Jill>only joking :-)
<Jack>:-)
<Jack>anyway fancy going for a beer tonight?
<Jill>yeah why not
<Jill>where?
<Jack>Mannings?
<Jill>dull... what about the curve bar?
<Jack>it'll be packed
<Jack>brannigan's?
<Jill>I am not going there! *armsCrossed* *turnAway*
<Jack>Ok, Ok, what about a quite one at the greyhound, we could get some food there
<Jill>that sounds good :-)
<Jill>I have to go to a meeting now, lets meet at 8
<Jack>fine
<Jack>laterz
<Jill>seeya
```

send message to: Jil

message text

macros | roles | situations | edit profile | questionaire | close

EDITING PROFILE: ivalmmDomBill

arrogant: -5
likes_flirting: 14
easily_intimidated: 7
likes_laid_back_people: 0
extrovert: 0
uncomfortable_with_forward_people: 0

save | reload

Figure 4. The text chat interface for Demeanour. (The 3D avatars are currently shown in a different window).

a customization tool that is easy to use, abstracts from the internal workings of the controller, and is itself easily customizable by world designers.

Profiles can also be made to evolve via real time control. After an interaction the user may have altered the parameter values of the agent's network using the methods described in the next section. They then have the option of saving these changes as a profile, either to the agent's main profile; to a person sub-profile for the person they interacting with, or as a new role. This allows the user to adapt behavior while using the system, thus refining the agent's profile without repeated customization steps.

6.3. Real time control

The real time control interface to an agent should be well integrated with the main user interface of the world. We have chosen to demonstrate Demeanour using a text chat interface that is very common in 3D multi-user virtual worlds and more generally on the internet. Users have a form of conversation consisting of short messages that are sent to the other users on pressing the return key. We have implemented such an interface, as shown in figure 4.

As with profiles, the control occurs through changing the values of the parameters of the controller. The most direct way of doing this is to edit their values via a set of sliders. Certain parameters may be exposed to the user interface in the behavior language. Such parameters are displayed on a set of sliders that the user can use to edit the parameters. This is an important interface component as it gives the user direct control of the agent's internal state. The user may also use profiles to change behavior in real time, adopting a role (see section 6.2.1) can change a variety of aspects of behavior at once. However, these two interfaces are still rather intrusive and are not well integrated with text chat.

The main interface directly uses the conventions of text chat. Textual communication on the internet already has its own vocabulary to express emotion and attitude, namely emoticons or smilies :-). In our example a smiling emoticon :-) will increase the 'friendliness' parameter while a frownie :-(will reduce it. This provides a very natural interface that does not intrude on a conversation, and is already well understood by internet users.

The behavior language can define textual commands that are parsed in the text chat interface. These can be emoticons ;-) or arbitrary text strings *bow* (we use the convention that textual commands are enclosed in asterisks to distinguish them from normal text). The commands have a sequence of actions attached. These actions can consist of altering a parameter value but they can also be a direct request for a type of behavior. For example, a particular posture or gesture such as *arms crossed*. This allows a more direct control over the body language that allows the user to create more exact effects. The two can be combined, if a posture such as crossed arms is know to be associated with low affiliation the command to request it can also reduce an associated parameter such as 'friendliness'. This allows for a more implicit control of the state of the agent. The user is able to control the affective state both directly and by demonstrating associated behavior. This allows users to have a very exact affective control (both the attitude and its exact expression) and to be able to control an agent without having to understand exactly how the internal parameters work. These three types of control, parameter, action and combined control, promise to provide a flexible, user friendly and unobtrusive user interface for a semi-autonomous avatar.

7. Example Interaction

This section will describe an example interaction between two agents, "Jack" and "Jill" that both use the behavioral controller described in section 5.3. When the two characters are initialized they each have their own individual profile loaded. For example, Jill's profile contain entries that give a high negative weight to other characters' closeness behavior when calculating affiliation (giving compensatory behavior) and a high positive value to relaxation, she also has a low positive value for the "friendliness" parameter, give a a moderately high equilibrium value of affiliation. When the two character start an interaction they load sub-profiles for each other, Jill has no profile for Jack, but Jack has a low positive value for the "other liking" parameter for Jill. These parameter setting result in moderate affiliative behavior for both character such as that shown in figure 5 top left (Jill left, Jack right). The conversation starts with the user controlling Jack typing "Hi Jill :-)", the smilie results in an increase in the "friendliness" parameter and therefore greater affiliation. This makes Jill's compensatory behavior dominate resulting in more distant behavior. The fact that Jack's user has sent a message results in gesturing behavior accompanying the attitude behaviour (top center) which eventually wears off (top right). If a later interaction occurs in a different environment a different situation profile can be loaded, for example if the situation is one in which Jack is more confident (e.g. his home) a high value for the confidence parameter will be loaded resulting in more relaxed, which Jill will react to with more affiliation (bottom left). Figure 5 also shows examples of status behaviour, with Jack either reciprocating or compensating for Jill's dominant behavior (bottom center and right).

8. Conclusion and Further work

In this presented the Demeanour framework for generating expressive behavior, and describe how it blends autonomous behavior with user control. The core framework is largely complete and our initial results are promising. We aim to extend the framework by adding new modalities of expression, we are working on an eye gaze module and are planning one for facial expression. Evaluating this type of system requires applying them to real situations. We would like to test Demeanour with a number of different applications. Most importantly we are currently planning user trials in order to provide a formal evaluation.

References

[1] M. Argyle, *Bodily communication*, Routledge, 1975.

[2] N. Badler, C. Philips, and B. Webber (eds.), *Simulating humans: Computer graphics, animation and control*, Oxford University Press, 1993.

[3] D. Ballin, M. Gillies, and B. Crabtree, A framework for interpersonal attitude and non-verbal communication in improvisational visual media production, *1st European Conference on Visual Media Production (CVMP)* (London, Uk), March 2004.

[4] P. Bécheiraz and D. Thalmann, A model of nonverbal communication and interpersonal relationship between virtual actors, *Proceedings of the Computer Animation '96*, IEEE Computer Society Press, June 1996, pp. 58–67.

[5] B. Blumberg and T. Galyean, Multi-level direction of autonomous creatures for real-time virtual environments, *ACM SIGGRAPH*, 1995, pp. 47–54.

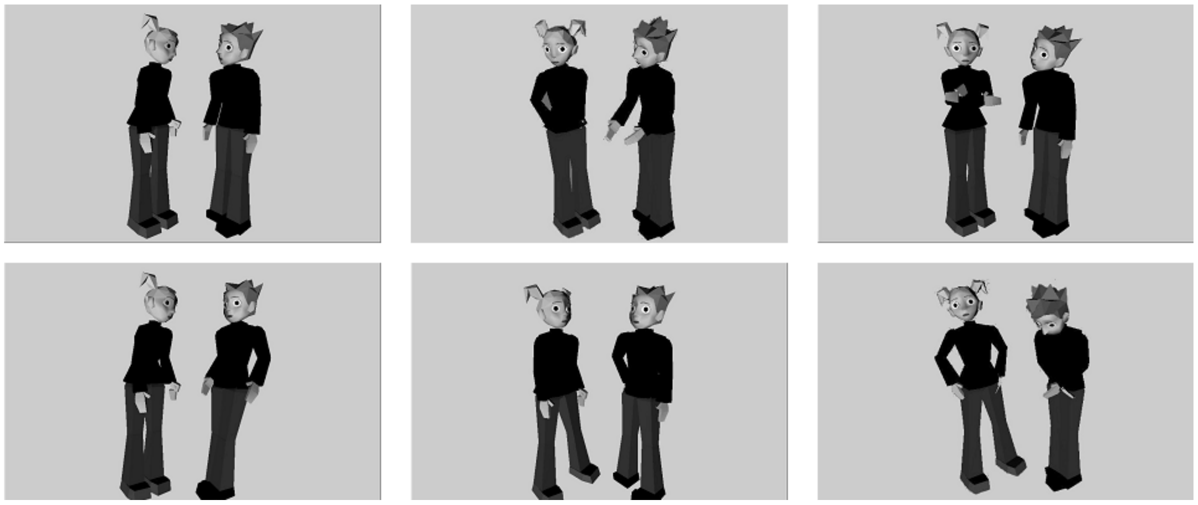


Figure 5. Examples of postures generated by the Demeanour framework in an interaction between two agents, Jill (left) and Jack (right), as described in section 7.

[6] J. Cassell, T. Bickmore, L. Campbell, K. Chang, H. Vilhjálmsson, and H. Yan, Embodiment in conversational interfaces: Rea, *ACM SIGCHI*, ACM Press, 1999, pp. 520–527.

[7] J. Cassell, Y. Nakano, T. Bickmore, C. Sidner, and C. Rich, Non-verbal cues for discourse structure., *41st Annual Meeting of the Association of Computational Linguistics* (Toulouse, France), 2001, pp. 106–115.

[8] L. Cheng, S. Farnham, and L. Stone, Lessons learned: Building and deploying virtual environments, *The Social Life of Avatars, Presence and Interaction in Shared Virtual Worlds* (Ralph Schroeder, ed.), Computer Supported Cooperative work, Springer, 2002.

[9] D. Chi, M. Costa, L. Zhao, and N. Badler, The emote model for effort and shape, *ACM SIGGRAPH*, ACM Press/Addison-Wesley Publishing Co., 2000, pp. 173–182.

[10] M. Gillies and D. Ballin, A model of interpersonal attitude and posture generation, *Fourth Workshop on Intelligent Virtual Agents* (Kloster Irsee, Germany) (Thomas Rist, Ruth Aylett, Daniel Ballin, and Jeff Rickel, eds.), September 2003.

[11] M. Gillies, N. Dodgson, and D. Ballin, Autonomous secondary gaze behaviours, *AISB workshop on Animating Expressive Characters for Social Interactions* (Imperial College London) (Ruth Aylett and Lola Cañamero, eds.), April 2002.

[12] M. Gillies and N. Dodgson, Eye movements and attention for behavioural animation, *Journal of Visualization and Computer Animation* **13** (2002), 287–300.

[13] B. MacNamee, S. Dobbyn, P. Cunningham, and C. O'Sullivan, Men behaving appropriately: Integrating the role passing technique into the aloha system, *AISB workshop on Animating Expressive Characters for Social Interactions* (Imperial College London) (Ruth Aylett and Lola Cañamero, eds.), April 2002.

[14] A. Paiva, I. Machado, and R. Prada, The child behind the character, *IEEE Transactions on systems, man and cybernetics: Part A: Systems and Humans* **31** (2001), no. 5, 361–368.

[15] C. Pelachaud and I. Poggi, Subtleties of facial expressions in embodied agents, *Journal of Visualization and Computer Animation.* **13** (2002), 287–300.

[16] K. Perlin and A. Goldberg, Improv: A system for scripting interactive actors in virtual worlds, *Proceedings of SIGGRAPH 96* (New Orleans, Louisiana), Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH / Addison Wesley, August 1996, pp. 205–216.

[17] J. Rickel and W. L. Johnson, Animated agents for procedural training in virtual reality: Perception, cognition, and motor control, *Applied Artificial Intelligence* **13** (1999), 343–382.

[18] P. Scerri, J. Ydrén, and N. E. Reed, Layered specification of intelligent agents, *PRICAI 2000*, August 2000.

[19] P. Sengers, S. Perry, and J. Smith, Traces: Semi-autonomous avatars, *current available at http://www-2.cs.cmu.edu/ phoebe/work/publications.html*, 2000.

[20] X. Tu and D. Terzopoulos, Artificial fishes: Physics, locomotion, perception, behavior, *ACM SIGGRAPH*, 1994, pp. 43–49.

[21] H. H. Vilhjálmsson and J. Cassell, Bodychat: Autonomous communicative behaviors in avatars, *second ACM international conference on autonomous agents*, 1998.

[22] M. Wilson and M. Neal, Diminishing returns of engineering effort in telerobotic systems, *IEEE Transactions on systems, man and cybernetics: Part A: Systems and Humans* **31** (2001), no. 5, 459–465.