Cursive: Controlling Expressive Avatar Gesture Using Pen Gesture

Francesca A. Barrientos and John F. Canny
University of California, Berkeley
EECS Computer Science Department
387 Soda Hall # 1776
Berkeley CA 94720-1776, USA
{fbarr,jfc}@cs.berkeley.edu

ABSTRACT

We describe an interaction technique for expressive nonverbal communication through an avatar. The input consists of letters written with expressive modulation, by hand, using a pen tablet input device. This technique, which we call Cursive, translates an alphabetic letter into a body gesture. Our design is inspired by the observation that handwriting, like some kinds of human gesture, informs through the symbols it depicts, and through the manner in which the symbols are produced. The technique is especially suited for desktop and portable virtual reality applications—applications that do not use special rooms or equipment but only two-dimensional displays and simple pointing devices for interaction with the virtual environment. Our system solves the difficult problem of controlling the movements of a highly articulated, 3D avatar using a common input device within the context of an office or other real world environment.

Categories and Subject Descriptors

H.4.3 **[Information Systems Applications]**: Communications Applications - *Computer conferencing, teleconferencing, and videoconferencing*.

H.5.2 **[Information Interfaces and Presentation]**: User Interfaces - *Input devices and strategies, Interaction styles.*

I.3.6 **[Computer Graphics]**: Methodology and Techniques - *Interaction techniques*.

General Terms

Design, Human Factors.

Keywords

Avatars, computer-mediated communication, virtual environments, gesture, novel interaction technique, pen gesture, nonverbal communication.

CVE'02, September 30-October 2, 2002, Bonn, Germany.

1. INTRODUCTION

Desktop virtual reality applications involving avatars used to be relegated to games and graphical chat worlds, applications in which participation in the virtual world is the focus of the user's attention. In these worlds, avatars provide the embodiment of the user within that virtual world. In the near future, however, avatars may start appearing in many other types of visual applications whose primary purpose is not entertainment but communication.Though interpersonal communication applications usually focus on sending verbal messages, the additions of avatars will enable these applications to transmit nonverbal bodily encoded messages as well.

Avatars may be used instead of video teleconferencing to allow geographically distant co-workers to discuss projects. As phones, including mobile phones, become more technologically sophisticated, user's will be sending visual content in addition to their voices over the air waves. And even when video phones are commonplace, users may not want to send their real images to each other. A phone call at an inopportune moment may require that the video camera be turned off. In those situations avatars will provide a way to transmit nonverbal communication signals. Even in asynchronous communication such as email, avatars may be sent along with text in order to add messages that mere words are too clumsy for.

One of the challenges for the user interface designers of these new applications is in designing a means for controlling avatar gesture that gives users the ability to communicate fluidly using the mix of verbal and nonverbal signals that are used in face-to-face conversation. The challenge is made all the more difficult by the fact that these applications must be used with very simple input devices.

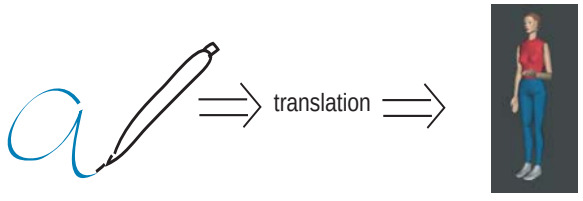

Figure 1: A written letter is translated into an avatar body gesture.

We have developed an interaction technique that we believe meets these challenges The idea is to use gesture as input, but pen gesture rather than body gesture. A user invokes an avatar gesture by writing a single letter on a pen tablet device, as shown schematically in Figure 1. The pen gesture is recognized and used to select

a particular avatar gesture. In addition, stylistic features of the writing, such as speed or size are computed. Then the gesture symbol plus these style features are used to specify an avatar gesture whose style is modulated by the writing style features.

The advantages of this approach are:

- Discrete and continuous input derived from a single action;
- People easily write while talking and often produce expressive writing spontaneously (e.g. doodling while on the phone);
- No mode switches between gestures or, between discrete and continuous control; and
- Pen movement is minimal/compact.

We have implemented a nonverbal communication application using this technique. A description of this application will serve to illustrate how this technique works.

2. DEALING WITH SIMPLE INPUT DEVICES

Desktop virtual reality is a virtual reality experience that is constrained by the technologies that one would normally use at an office desk. Graphical interaction with the virtual world takes place through a computer monitor, the representation of a three dimensional world being projected onto a two dimensional plane. The user interacts with the world using the staples of desktop computing, a keyboard and a pointing device.

A central difficulty in controlling avatar body movement, including gesture, is that humanoid avatars have many joints whose motions needs to be coordinated all at once. Though capture (e.g. using computer vision) and reproduction of a user's own body motion in the avatar seems the most direct solution, it may not be the most desirable solution. First, the user's body is situated in a completely different environment from the avatars. The avatar may be "walking" all around a virtual world, but the human is sitting or standing in front of a computer. The user may be wearing jeans and a T-shirt while the avatar is dressed in a navy business suit. The user's body movements are constrained by physics and their own physical abilities, but the avatar's body may be whole and have full mobility. And the avatar may be addressing a voluble group of other avatars while the user is ensconced in an office with coworkers who are engaged in their own activities.

Because of the myriad differences in context between the user's physical and virtual self, avatar control in which the user must physically act out body movements may be undesirable and inappropriate. We are left then with controlling our avatar using only the keyboard and the pointing device.

Gestures derive their meaning both from their specific form (in conjunction with verbal communication) and from their spatial and dynamic qualities. Though the analogy is inexact, we might consider breaking down the communicative value of a gesture into its symbolic and expressive components. For instance, drawing the arm across the body in a horizontal line may symbolically refer to a distance between two places or two ideas. The length of the gesture may indicate the magnitude of the distance while the speed at which the gesture is performed may indicate the relative importance of the distance within the discourse.

Using this division, we simplify the problem of controlling avatar gesture by devising a technique in which a user chooses from a set of predefined gestures. In effect these are the symbols. To produce gestures that can communicate more subtle layers of meaning, stylistic differences in the user's input are used to determine the spatial and dynamic qualities of the gesture.

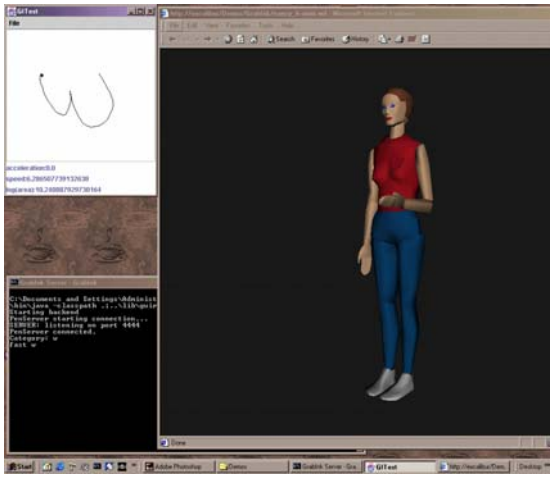

Figure 2: Screen shot of cursive implementation

3. PEN GESTURES INDEX AND MODULATE

Pen gestures, like human gestures, can also be resolved into symbolic and qualitative parts. In most pen gesture user interfaces, mainly the symbolic portion of the gesture is used. A pen gesture is recognized, and its identity is used like a like a button event, triggering some action in the interface. However, pen gestures also have spatial extent and dynamic variations. A gesture can be drawn quickly or slowly, or varyingly along its path. Further, if a pen tablet input device is pressure sensitive, pressure variation can be part of the gesture's quality. This suggests that writing may be a natural interaction technique for controlling avatar gesture.

We map these complementary types of gesture (pen and body) together in our technique. The user invokes a particular gesture by writing a particular alphabetic letter. The interface identifies the letter and additionally extracts other stylistic feature values from the way the letter was written. When the gesture is performed on the avatar, its style is modulated by the way the letter was written by the user.

4. AVATAR WORLD APPLICATION

Our interaction technique, which we call *Cursive*, can be used within an avatar world, that is, multiuser virtual world where users interact with each other through their avatars. Browsers for such multiuser worlds typically have controls for managing navigation in the world, keeping track of other avatars, and transmitting verbal communication. The verbal communication can be text chat or spoken words.

Cursive extends the capabilities of avatars within VRML based virtual worlds. It provides an interface for controlling the avatar body motions that is used alongside the existing virtual world browser. The graphical element of the interface consists of a window that captures ink strokes from the user. A screen shot of our application is show in Figure 2. The avatar gesture that is generated is sent to all copies of the user's avatar that are visible in any browsers

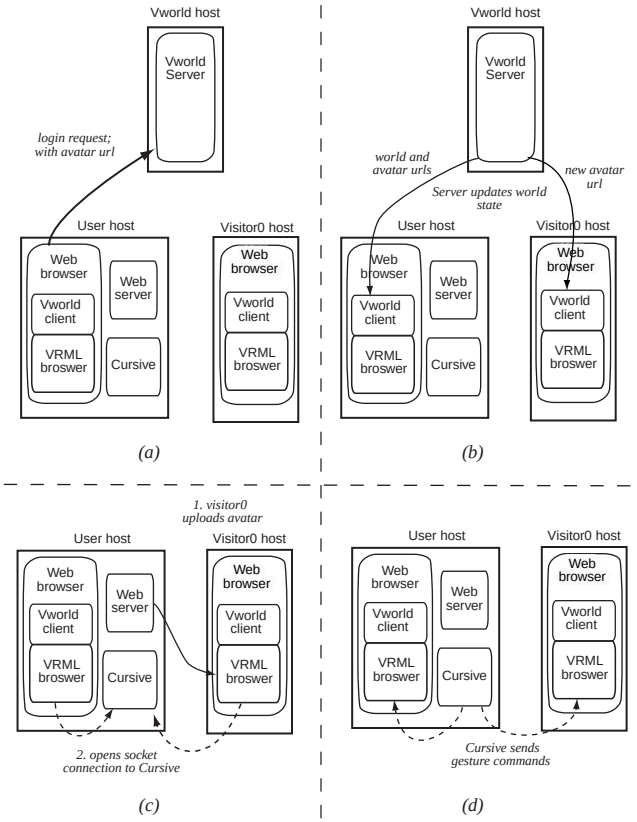

Figure 3: Communication among Cursive, virtual world manager and clients.

4.1 System Architecture

Our application is written in Java and consists of two main parts. A frontend provides the graphical user interface capturing the user's pen gestures and translating these into animation commands. The backend takes the animation commands and controls the animation of the avatar.

An avatar is a VRML file in which script nodes control the animation. The Cursive backend communicates with the avatar through a Java interface in the script nodes. The Java code receives the gesture encoding via a socket connection with the Cursive frontend on the user's host.

Our approach is designed so that we can "drop" expressively animated avatars into existing avatar worlds. The application should, in principle, work with virtual worlds that allow user-defined avatars and require only a standard Web browser with a VRML plug-in that supports the Java language. We have used the *Cortona* VRML client from Parallel Graphics [2].

Here we describe how this application works along with a standard virtual world manager. Figure 3 shows the processes that are involved. The virtual world manager handles logging in and logging out visitors and storing the url's for the world and any avatar vrml files. It also keeps track of world state such as the location of each avatar. Visitors view and interact with the world using a virtual world browser that runs inside of a web browser. In addition, running the Cursive application requires the user to be running a web server on their machine.

When a user logs into the virtual world manager, they also supply the url for their avatar as shown in Figure 3a. The vworld manager updates the user's browser with url's for the world and for any other visiting avatars. All other visitors are informed of the state change and they receive the url for the new avatar. This updating is shown in Figure 3b. The vrml browsers retrieve the actual vrml files from the hosts where they reside. In Figure 3c, Visitor0 uploads the user's avatar from the user's host.

When the avatar is uploaded, it runs the Cursive backend which is attached to the user's avatar. The backend opens a socket connection to the Cursive frontend. Note that the user's own avatar in their browser also connects to the Cursive frontend via a socket. When the Cursive frontend receives a pen gesture from the user, is sends the animation control data to any connected backends as in Figure 3d.

4.2 Cursive Frontend

To use Cursive with their avatar, the user runs the Cursive frontend on their host. As mentioned earlier, the GUI for Cursive appears, and presents a window to the user for capturing pen gestures. The frontend opens a socket to the network that it uses to send animation commands to the backend.

When a user writes a pen gesture into the gui, the ink is sent to a character re cognizer. The recognizer classifies the pen gesture (i.e. recognizes it.) As it is currently implemented, Cursive only recognizes single stroke letters in the script alphabet.

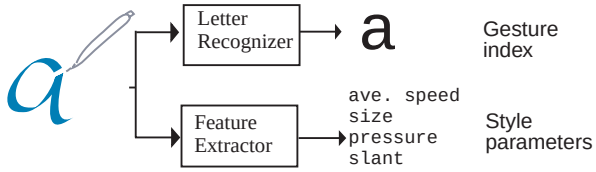

Figure 4: Cursive extracts both letter identity and writing features from the pen gesture.

Cursive also extracts values for a number of features of the pen gesture as shown in Figure 4. We use a feature based recognizer developed by Rubine [7], so the features are extracted as a by product of the recognition algorithm. However the feature extraction could be done independently of the gesture recognition. The particular features that we use in this implementation are *speed* and *size*.

The recognized letter selects the appropriate avatar gesture *type* from the gesture library. Then the feature vector extracted from the pen gesture is used to produce a specific *gesture instance*. This avatar gesture instance consists of an array of curves, one for each avatar joint, which represent the joint rotations over time. The interaction of the letter identity and letter features is shown in Figure 5.

As in traditional animation, we encode the gesture animation as a series of *key frames*. Each key frame is made up of the values of the joint rotations for every joint in the avatar at a particular instant in time. As they are sent over the network key frames are timestamped at the time they should appear during the gesture animation sequence.

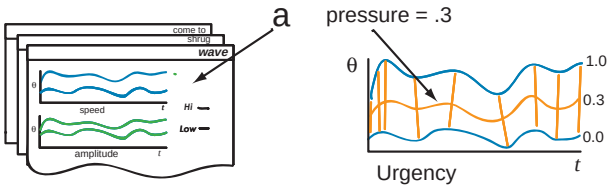

Figure 5: The letter identity selects a particular gesture while each feature modulates the joint motions.

4.3 Backend

The client application interfaces with the avatar through a vrml script node.Within Java enabled VRML browsers, script nodes provide hooks for Java code to pass data to VRML geometry nodes. Avatars built using the h-anim standard allow the backend to address specific avatar joints.

Once the avatar and Java code are loaded, the backend opens a network socket connection to the Cursive frontend and waits for key frames. The key frame values are used to animate the avatar. Because the backend needs to update the animation at a steady frame rate, it buffers the key frames as they come in and uses the timestamps to determine the timing of the gesture.

When the backend needs to update the animation at a time that falls between key frames, it calculates *between frames*. The joint rotations are represented using quaternions, so quaternion interpolation is used to calculate the between frames. Our implementation uses spherical linear interpolation.This type of interpolation finds points along the shortest arc between two quaternions on the unit hypersphere as shown in Figure 6.

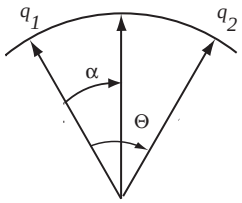

Figure 6: Spherical interpolation.

The *slerp* function performs a spherical linear interpolation between two quaternions q_1 and q_2 by an amount α which varies from *0* to *1*.

$$\text{slerp}(q_1, q_2, \alpha) = q_1 \frac{\sin((1-\alpha)\Theta)}{\sin\Theta} + q_2 \frac{\sin(\alpha\Theta)}{\sin\Theta}$$

5. SYNTHESIZING EXPRESSIVE VARIATION

The style parameters from the pen gesture are mapped to meaningful parameters for the body gesture. An example of one such mapping is shown in Table 1. (This mapping need not be the same for all gestures.)

The gesture that the client uses to animate the avatar is a specific body *gesture instance*. Cursive computes the gesture instance on the fly. To compute this instance, we start with a set of gesture prototypes. Then the pen gesture feature values are used to blend the prototypes into a single modulated avatar gesture.

Table 1: Mapping pen gesture features to avatar gesture parameters.

pen gesture feature	avatar gesture parameter
bounding box area	size
average speed	speed

Within Cursive gestures are stored as a set of joint trajectories. All of these trajectories, when played on an avatar, are recognizable as the same avatar gesture *type.* Qualitatively they represent extremal forms of the same gesture. For each avatar gesture parameter, there is a minimal and maximal gesture trajectory.

5.1 Blending Gesture Forms

To create a specific instance of a gesture type, Cursive blends the trajectories of prototypes for the gesture. For example, say a gesture type *x* is parameterized by the four parameters. Let each parameter have values ranging from *0* to *1*. We can denote the joint trajectory for this gesture type *x* as $\boldsymbol{J}_x(t)$. In fact, $\boldsymbol{J}_x(t)$ is a vector whose size is the same as the number of avatar joints. The $\boldsymbol{J}_x(t)$ trajectory is parameterized by our four parameters $[p_1\, p_2\, p_3\, p_4]$. We present the type by a family of prototypes, each of which is made of up a gesture instance with extremal values for the four parameters. That is, for four parameters, we use the sixteen prototypes:

$$\boldsymbol{J}_x[0\ 0\ 0\ 0](t)$$
$$\boldsymbol{J}_x[0\ 0\ 0\ 1](t)$$
$$\ldots$$
$$\boldsymbol{J}_x[1\ 1\ 1\ 1](t)$$

A particular gesture instance is just one that has specific parameter values which are not all extremal.

We compute a given gesture instance from the pen gesture feature vector using multilinear interpolation. In multilinear interpolation we interpolate along only one parameter at a time while holding the other parameters constant. Then we recursively interpolate along each successive parameter.

We can most easily illustrate this procedure for the case of two parameters p_1 and p_2. This procedure is illustrated in Figure 7.We

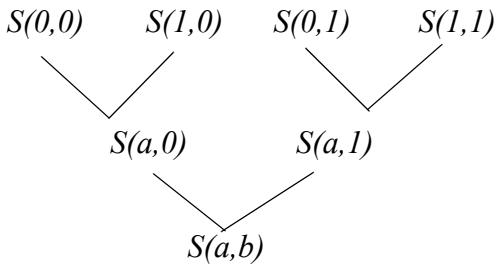

Figure 7: Multilinear interpolation along two dimensions.

wish to produce a new motion for $p_1 = a$ and $p_2 = b$ using the sample motions *S(0,0), S(0,1), S(1,0), S(1,1)* . In the first round of interpolations we hold p_2 constant and interpolate along p_1. The interpolation between *S(0,0)* and *S(1,0)* yields *S(a,0)*. Interpolating between the other pair *S(0,1)* and *S(1,1)* results in the motion *S(a,1)*. Then we interpolate between resultant motions *S(a,0)* and *S(a,1)* to arrive at the desired motion *S(a,b)*. The cases of more than two parameters proceed similarly.

5.2 Interpolating Speed

Speed is different from the other parameters because it requires interpolating between trajectories with different time ranges. A fast version of a gesture takes less time than a slow version of the same gesture. Before interpolating, we take the extra step of compressing or dilating the prototype curves. This is done by resampling the

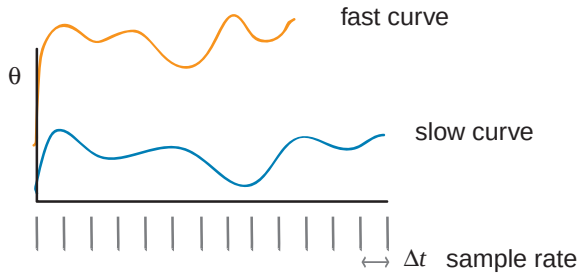

Figure 8: The slow and fast prototypes are sampled at the same rate.

prototype curves. Originally the curves are sampled at the same rate as shown in Figure 8

The first step is to determine the duration of the interpolated curve. Let d_0 be the duration of the slower speed and d_1 be the duration of the faster. Then the duration d of the interpolated curve is

$$d = \alpha d_1 + (1 - \alpha) d_0$$

To compress the slower curve to the new duration, we resample the curve with a new sample rate given by the equation

$$\Delta t' = \frac{d}{d_0} \Delta t$$

where Δt is a sample rate on the prototype trajectories. Similarly, the faster curve is expanded by a new sampling rate.

$$\Delta t'' = \frac{d}{d_1} \Delta t$$

The interpolation parameter α is determined by the following equation

$$\alpha = \frac{d - d_0}{d_1 - d_0}$$

These new sample points are aligned and linear interpolation takes place as usual as shown in Figure 9.

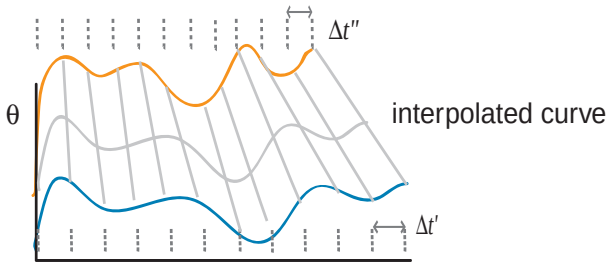

Figure 9: The prototypes are resampled, and a new curve is interpolated as usual.

6. GESTURE LIBRARY CREATION

The library of gesture prototypes used by Cursive is created offline. Because gestures and their variations must have expressive capability, creation of the library requires careful design. The first step in creating the library is deciding on a repertoire of gestures. These can be selected any number of ways depending on the avatar's application. For a particular person they may be idiosyncratic gestures. For a theatrical character avatar, they may be gestures associated with the character. One might also design gestures to suit a particular function such as use during discussion of an architectural design. We do not have to create a repertoire of gestures with specific semantics since the real meaning of a human gesture is created at the time the gesture is invoked and within the context of cotemporal verbal communication.

After the repertoire has been selected, style variations for each gesture are designed. The style variations should ideally be independent of each other, and should blend well in a meaningful way. Of course, this is a subjective judgement.

Before recording, the gesture model choreographs how the style variations will be performed. These trajectories are recorded from a human using a standard performance capture technique. We have built our own performance capture system which uses the Flock of Birds position orientation sensors built by Ascension Corporation. The specific number of sensors used depends on the particular gesture being recorded. A single arm gesture can be created minimally with trajectories for the shoulder, elbow and wrist joints. Four sensors are used and attached to the torso, upper arm, lower arm and hand of the model. Even for a single arm motion, additional sensors can be added to make the motion of the avatar's whole body more "lifelike."

7. MAPPING PEN GESTURE FEATURES TO INTERPOLATION PARAMETERS

Our implementation maps the physical features of the writing to the analogous features in the gesture. That is, increasing the size of the letter increases the size of the gesture and increasing the speed of writing increases the speed of the movement. To do this, the normalized values for the writing features are used as the values for the interpolation parameters of size and speed.

Though some mappings such a as pen gesture size to avatar gesture size are obvious, others may depend on personal preference. Ideally, the gesture model and the online user of the avatar would be the same person.That will not always be possible, so it will be use-

ful to have several libraries available from different models so that a user can find one that best matches the gestures they would like to use.

8. EVALUATION

Using our implementation in the lab we have been able to evaluate the feasibility and possibilities of our technique. Our experimentation in the lab consisted of the researchers producing one or two modulated gestures. We found that individually modulating the speed and the size of the gesture was very straightforward. Modulating the two together is a little more difficult since writing a letter smaller often results in writing with increasing speed and vice versa. With additional practice and appropriate tuning of the modulation parameters, simultaneous control of the two becomes easier.

Similarly, we found that the ease of talking while controlling the avatar depended very much on the amount of practice one had with using the system. Initial use required concentrating a lot on the feedback offered by watching the avatar gesture. It is likely some visual feedback will always be necessary though not necessarily the fully rendering of the whole avatar. With practice it became easier to drive while talking. The experience may be likened to talking while driving a car: it can be done but it depends on the complexity of the conversation and the conditions of the road.

During demonstrations we found that the technique was readily understood by the audience. A few people questioned whether simultaneously specifying the symbolic and expressive aspects of gesture was desirable. An alternative is to choose the gesture with one action, and then to invoke and modulate the gesture with a second action. We agree that determining the desirability simultaneous control needs to be further investigated.

9. RELATED WORK

Many avatar chat worlds provide a means for users to select emotional expressions (smiling, frowning, etc.) or communicative actions (such as winking or waving) for their avatar. The Palace is an avatar world with avatars represented by two dimensional graphic images [1]. The user interface supplies a button panel for the user to select different avatar images which show the avatar with difference expressions or postures.

A more sophisticated gesture control interface was developed for ComicChat, a 2D chat environment in which avatars appear as comic strip characters, and the virtual world displays as evolving comic strip frames [5]. Users have the ability to select bodily expressions using an *emotion wheel*. The different directions around the wheel select different expressions: *coy*, *happy*, *laughing*, *shouting*, *angry*, *sad*, *scared* and *bored*. The distance from the center of the wheel selects the magnitude of the expression. The very center of the wheel selects the *neutral* expression. Gestures are also generated through analyzing the chat text. Certain keywords or phrase types trigger particular expressions and actions by the avatar.

A nonverbal communication application was developed for VLNET by Guye-Vuillème [4]. VLNET comprises several research projects to develop the technologies required for various tasks in very large networked 3D virtual environments with realistic looking avatars. The interface consists of three panels: a panel for selecting affect display, a panel for selecting body and facial gesture, and a panel for selecting actions. A separate slider adjusts the speed of the avatar actions.

Slater et al. developed a non-verbal communication interface for avatar expressions in order to study the feasibility of using virtual reality to rehearse for a live dramatic performance [8]. User's control their viewpoints, and subsequently the position of the avatar's head by moving their mice. Minute movements of the mouse allow the actors to generate head nods, wags and glances toward and away from their interlocutors. The arms can be moved up and down either separately or independently using sliders. Simple full body actions such as standing and sitting were are predefined.

This project also developed a novel drawing interface for controlling facial expression. The users create expressions by drawing a stylized mouth and eyebrows onto a generic smiley face. Mouths and eyebrows are drawn with the mouse using $\vee$ and $\wedge$ shaped strokes, respectively. The intensity of the expression is modified by adjusting the slopes of the angles on the input gesture.

Our research is most closely related to these projects in which the designers were trying to provide a way for the user to directly control the avatar's body. With the exception of [8], these systems do not have an interface which allows a user to specify a body gesture along with multiple style performance parameters using just a single input action. The *drawing* interface in [8] is geared toward facial expression, but was not extended to body gestures.

Improv, developed by Perlin and others at the Media Research Laboratory at NYU is a system for authoring behavior based animated characters [6]. Authors define the animation for the character at many levels. At the lowest level authors define animations for a set of actions, such as wave or sit. Specifying an action means defining how the avatar's parts will move—or its mesh will deform—over time for the duration of the action. These animation specifications include not only the path of the parts over time, but also a level of noise which will be added to the motion in order to make the motion look more lifelike. At a higher level, authors define a set of scripts for the actor. Scripts describe higher level behaviors such as dancing, or joking. They are made up of sequences of actions and other scripts where each action or script may have a triggering event which initiates it. Authors are also able to create user interface widgets to give users interactive control over the character. Using the widgets, the user can trigger particular actions or scripts. The system in general is designed for creating autonomous characters.

The BodyChat system developed by Vilhjálmsson at MIT treats avatars as communicative agents that act on behalf of their users to fulfill communicative aims [9]. This project is concerned with the nonverbal signals used in conversation regulation. Users specify high level goals such as initiating, rejecting or ending conversations. The avatars automatically handle the nods, glances and other gestures required to negotiate the conversation. Like ComicChat, BodyChat also uses natural language processing to infer when certain gestures are appropriate.

The BodyChat system was part of research lead by Cassell at the MIT media lab. In [3] Cassell and Vilhjálmsson propose that the design of an avatar conversation system should be based on a theory of conversational agents.The theory, they suggest, should

account not only for the various functions of conversational bodily behaviors, but also the user's preference for controlled versus autonomous avatar animations

In their work they point out many of the problems of using graphical user interfaces to activate communicative animations. Partly for this reason, they have moved towards treating avatars as agents. Further, their work focuses on a single important aspect of communicative behaviors: conversation regulation. However, the range of communicative behaviors spans a wider range of functions than this, and avatars may afford new ways to accomplish these functions. We are trying to develop a more general approach. Our work tries to address the deficiencies in standard interface elements such as buttons and pull-down menus by providing an alternative means of input.

10. SUMMARY

This paper presents a novel interaction technique for applying pen gestures to avatar gestures in order to produce expressive nonverbal communication. We have argued for the desirability of controlling avatar gesture through pen gesture, and suggested it as feasible for both desktop and portable virtual reality applications. We have described a technique for using both the symbolic and qualitative data extracted from pen gesture in controlling avatar gesture with expressive modulation. We described an implementation of this technique that is designed to be used within certain types of existing multi-user virtual world systems. And we have related our experiences in designing a gesture library for use with this technique.

11. CONCLUSIONS AND FUTURE WORK

We have devised and implemented a novel interaction scheme for controlling avatar animation for nonverbal communication in a virtual environment. At this time we use letters because they are a natural pen gesture for most people. In the future we would like to study the use of other forms of pen gesturing to control avatar expression. We also plan to conduct user studies of our technique.

12. ACKNOWLEDGMENTS

This research was supported in part by National Science Foundation award #EIA-0122599: "ITR/SI: Societal Scale Information Systems: Technologies, Design and Applications."

Erin Dare was our gesture model provided valuable input on the design of gesture movement variations

13. REFERENCES

[1] The Palace, in www.digitalspace.com, Currently supported by DigitalSpace Corporation.

[2] "Cortona," in http://www.parallelgraphics.com: Parallel Graphics Corporation.

[3] Cassell, J. and H. H. Vilhjálmsson, "Fully Embodied Conversational Avatars: Making Communicative Behaviors Autonomous," *Autonomous Agents and Multi-Agent Systems*, 2, no. 1, pp. 45-64, 1999.

[4] Guye-Vuillème, A., T. Capin, I. Pandzic, N. Magnenat-Thalmann, and D. Thalmann, "Non-Verbal Communication Interface for Collaborative Virtual Environments," *in Proc. CVE 98*, June 1998, Manchester, 1998.

[5] Kurlander, D., T. Skelly, and D. Salesin, "Comic Chat," in *SIGGRAPH 1996*, New Orleans, LA, pp. 225 - 236, 1996.

[6] Perlin, K. and A. Goldberg, "Improv: A System for Scripting Interactive Actors in Virtual Worlds," in *Proc. SIGGRAPH 96*, pp. 205-216, 1996.

[7] Rubine, D., "Specifying gestures by example," in *SIGGRAPH 91*, pp. 329-337, 1991.

[8] Slater, M., J. Howell, A. Steed, D. P. Pertaub, M. Gaurau, and S. Springel, "Acting in Virtual Reality," in *Collaborative Virtual Environments 2000*, pp. 103-110, 2000.

[9] Vilhjálmsson, H. H. and J. Cassell, "BodyChat: Autonomous Communicative Behaviors in Avatars," in *Proceedings of the Second International Conference on Autonomous Agents*, May 9-13, Minneapolis, pp. 269-276, 1998.