Avatar-independent scripting for real-time gesture animation

Richard Kennaway
University of East Anglia

March 2005

Abstract

When animation of a humanoid figure is to be generated at run-time, instead of by replaying pre-composed motion clips, some method is required of specifying the avatar's movements in a form from which the required motion data can be automatically generated. This form must be of a more abstract nature than raw motion data: ideally, it should be independent of the particular avatar's proportions, and both writable by hand and suitable for automatic generation from higher-level descriptions of the required actions.

We describe here the development and implementation of such a scripting language for the particular area of sign languages of the deaf, called SiGML (Signing Gesture Markup Language), based on the existing HamNoSys notation for sign languages.

We conclude by suggesting how this work may be extended to more general animation for interactive virtual reality applications.

1 Introduction

In interactive applications of virtual reality, it is necessary to create animations of avatars and other objects at run-time, according to the demands of the moment. Usually, this is done by replaying motion clips selected from a library of pre-composed motions, created by either manual keyframing or motion capture. This method suffers from several disadvantages.

1. It may require a very large motion library to anticipate every possible action that may be demanded.
2. The same movement will be played in exactly the same way every time it is used, reducing the realism of the scene.
3. The motion data are specific to one avatar.

It is possible to warp motion clips, for example, to adapt a motion for one avatar to another avatar, to have a walk cycle for a character follow a specified path through the world, or to smoothly blend together the end of one clip with the start of another. However, this only gives a limited amount of extra flexibility. Inspection of some current video games indicates that at present little run-time warping or blending is done.

Run-time generation of the motion data from some higher-level description would allow for animation to be generated in a much more flexible way, in order to exactly fit the interactive situation at each moment. In this paper we will describe an avatar-independent scripting language for posture and movement in the particular domain of

sign languages of the deaf, and how we use it to automatically generate animation data in real time for any avatar. A secondary advantage of the method is that the script for a movement is many times smaller than the motion data generated from it, which greatly reduces storage requirements.

The software [14] was developed in the ViSiCAST [6, 34] and eSign [[35]; `http://www.sign-lang.uni-hamburg.de/eSIGN`] projects, as one component of a system which provides signing animations for websites or other interactive multimedia applications. (The software was not deployed until the eSign project; ViSiCAST still relied on motion-captured animations.) A web page using this sytem will include an avatar on the page (implemented as an ActiveX control), and beside each block of text for which signing is available, a button to request that the signing be played. The web page also contains data to drive the avatar when one of these buttons is clicked, not in the form of motion data, but as text in the avatar-independent scripting language. This is translated in real time into frames of motion data (bone rotations and facial morphs). The data frames are passed to the avatar and played at a fixed frame rate. A fuller account of the whole system can be found in [35]. A publicly available web site employing the technology is at `http://www.gebarennet.nl`[1].

The scripting language and its translation to motion data are the subject of the present paper. Our concern is with the concrete geometry of the animation, in contrast to the large number of proposals for scripting languages for embodied conversational agents, which focus on much higher-level concepts. Some of these are briefly surveyed in Section 2.

2 Related work

There have been numerous proposals for scripting languages for embodied conversational agents, including VHML [[19]; `http://www.vhml.org`], HumanML [`http://www.humanmarkup.org`], EMOTE [2], GESTYLE [22], CML [1], AML [1, 25], APML [4], MURML [16], MPML [27], and STEP [11, 12]. A survey of some of these languages and several others can be found in [24], and [28] is a book-length account of various implementations of "social agents". These languages mainly address a higher level of abstraction than we are concerned with here, typical elements or attributes being *angry*, *alert*, *think*, *emphasize*, and *semiote*. The main purpose of most of these languages is to specify the behaviour of such agents, and especially the emotionally expressive facial and bodily movements that they should make to accompany realistic speech and interact in an engaging way with the user. The specification of the concrete movements which perform these high-level concepts is typically seen as a mere implementation detail. Among the proposals that include explicit description of concrete movement, emphasis has been primarily on facial animation, with body animation either ignored (for talking head applications), undeveloped, as in the most recent (2001) VHML specification, or implemented directly in terms of avatar-dependent joint rotations.

The VHD system [32] for directing real-time virtual actors is based on motion capture and predefined postures, and the VHD++ system [26] is a high-level virtual reality engine for integrating lower-level technologies.

[1]Note that the text of the web site is in Dutch, and the signing is in Dutch Sign Language. The animations will play on Internet Explorer running on Windows (2000 or XP), and require installing some software (available from the web site). There is also a video demo on the site for those without the necessary configuration. The live avatar plays rather more smoothly than the video.

Dance notations such as Labanotation [13] have been animated by software such as Don Henderson's LINTER [http://www-staff.mcs.uts.edu.au/~don/pubs/led.html], although that system does not attempt naturalistic animation of detailed avatars. Labanotation and other dance notations present particular challenges to computer animation, as transcriptions written to be read by human dancers rely for their interpretation not only on the dancer's knowledge of dancing, but upon their creative input to flesh out intentionally incomplete transcriptions. Such creative extrapolations are beyond the intended scope of the work we are engaged in.

At the opposite extreme to these are the low-level FAPS (Facial Animation Parameters) and BAPS (Body Animation Parameters) of the MPEG-4 standard [23], and the H-Anim standard [http://www.h-anim.org]. FAPS are a set of basic facial movements expressed in terms of a standard set of feature points of the face. Amounts of movement of these points are expressed in terms of a set of standard measurements of the face (e.g. distance between the eyes). They are thus avatar-independent: the same stream of FAPS data played through different face models should produce animations that are seen as doing the same thing (as demonstrated in [5]). BAPS, on the other hand, consist of joint rotations in the form of Euler angles, and are not avatar-independent. A stream of BAPS data which, played through one avatar, causes it to clap its hands, will fail to do so if played through another avatar which is identical except that its shoulders are a couple of inches further apart: the hands will fail to meet by the same distance. H-Anim is an adjunct to the VRML (Virtual Reality Modelling Language) standard defining a standard skeleton and form of motion data for humanoid animation. Like BAPS it expresses motion data in terms of avatar-dependent joint rotations.

We thus perceive a role for a method of specifying physical posture and movement in a manner which is avatar-independent, human-readable and -writable, and which allows unambiguous calculation of the exact avatar-dependent motion data for any given avatar. Such a method can provide the lower level of implementation which must underlie all the higher-level proposals described above. This is the purpose of the work reported here.

In the area of sign language, Vcom3D [http://www.vcom3d.com] publish a commercial product, SigningAvatar, for performing sign language on web pages. It uses an internal notation for specifying the movements of the avatar, of which details are not available. SignSynth [9] is a signing animation system based on the Stokoe notation, producing output in VRML, and also addressing the issue of translating English text to sign language. The GESSYCA system [17, 18, 8] is concerned with French sign language, and of all the animation systems listed here is closest to the approach of the present paper. It uses a notation system developed for a project called Qualgest (Qualitative Communication Gestures Specification), and uses biomechanical simulation to generate realistic human movements.

3 HamNoSys and SiGML

The language we have developed is called SiGML (Signing Gesture Markup Language), and was developed from HamNoSys, the Hamburg Notation System. HamNoSys [29, 10] is a notation for recording sign language gestures, developed by researchers on sign language at the IDGS (Institut für Deutsche Gebärdensprache) at the University of Hamburg. It differs from other signing notations such as Stokoe [33] in that it is not specific to any one sign language, but is intended to cover all signing gestures in all sign languages.

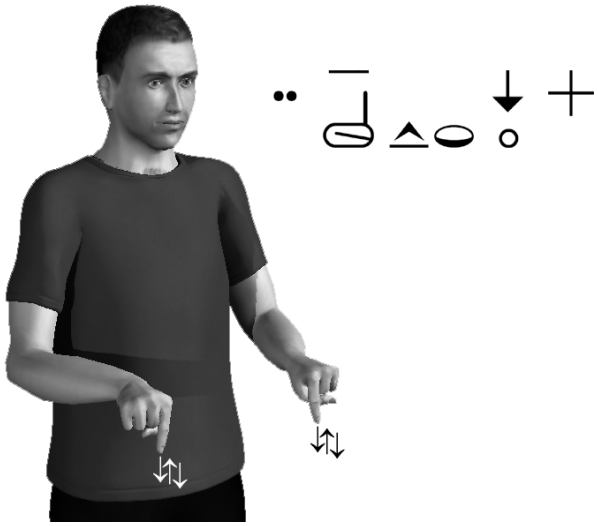


Figure 1: British Sign Language for "here" with HamNoSys transcription

Like most such notations, HamNoSys records signs in terms of hand shape, hand location, and hand movement. It uses a set of special glyphs comprising a HamNoSys font. An example is shown in Figure 1. The symbols mean:

- The hands perform mirror-image actions.
- The hand shape, illustrated in Figure 6, is the *finger2* shape of Figure 5 with the thumb across the fingers and the index finger bent at the base.
- The axis of the hand points forwards.
- The palm faces down.
- The hands move down a small distance.
- Repeat the movement once from the initial posture.

In the most recent version, HamNoSys has been extended to cover facial movement and expression, and non-manual movements of the upper body such as nodding the head, raising the shoulders, etc.

3.1 Principles of HamNoSys

A HamNoSys transcription is independent of the person or avatar performing it. It records only those aspects of the gesture which are significant for the correct performance of a sign. Thus for most signs, it records only what the hands do; the rest of the body is assumed to do whatever is natural in order for the hands to perform the gesture. There are a few signs for which some other body part is significant, such as the BSL (British Sign Language) sign for "Scotland", in which the right elbow is the important part (see Figure 2). In such cases, HamNoSys contains provisions for specifying the significant body part.

HamNoSys also includes some non-manual actions of the torso, shoulders, head, and eyes: shrugging the shoulders, tilting or turning the head, etc. There is also a

Figure 2: BSL sign for "Scotland"

repertoire of facial morphs of two types: visemes (visual equivalents of phonemes, the visible mouth movements used in speech) and other facial movements, such as closing the eyes or puffing the cheeks. These are classified into a number of "tiers", each tier being the set of movements that affects one part of the body: the shoulders, the torso, the head, the eyes, the facial expression, and the mouth. For each tier a sequence of actions can be given, and actions for different tiers being performed simultaneously. There are only limited capabilities for synchronising different tiers with each other or with the signing actions of the arms and hands. Non-manuals are not as fully developed a part of HamNoSys as manual gesturing, and play a correspondingly limited role in our implementation of HamNoSys.

Positions are specified in terms of named locations. These fall into two classes: locations in "signing space", the space generally in front of the signer's upper body, and locations on or close to the surface of the body (see Figures 3 and 4). Signing space is represented in HamNoSys by a three-dimensional grid of discrete points. These are at a number of levels: below the stomach, stomach, chest, shoulders, neck, head, and above the head. From left to right there are five positions: far left, near left, centre, near right, and far right. There are three proximities: near, medium, and far. Points on the body include the palms, the fingertips, the eyebrows, the nose, the chin, and many more; each may be specified with a proximity of medium, near, or touching. In total, there are some hundreds of locations that can be specified. In addition, a position can be specified as the midpoint between two of the above positions.

Hand shapes are classified into the twelve basic types illustrated in Figure 5. Variations such as in Figure 6 can be created by specifying the shapes of individual fingers: straight, rounded, curled, etc. Contacts between fingers can also be specified, such as fingers crossed, or thumb between fingers.

Movements are parallel or sequential combinations of a repertoire of basic movements: straight, curved, and circular. The direction of a movement can be given as a combination of up, down, left, right, forwards, and back. A target location can be given for a movement, instead of, or as well as, the direction of movement.

In many signs, the hands perform mirror image actions. HamNoSys allows transcriptions of such signs to be abbreviated, by specifying the action of the dominant hand, plus a symbol indicating that the nondominant hand mirrors the action. Left-right mirroring is the most common form, but there is also provision for front-back and up-down mirroring, as well as parallel movement of the two hands. The mirroring

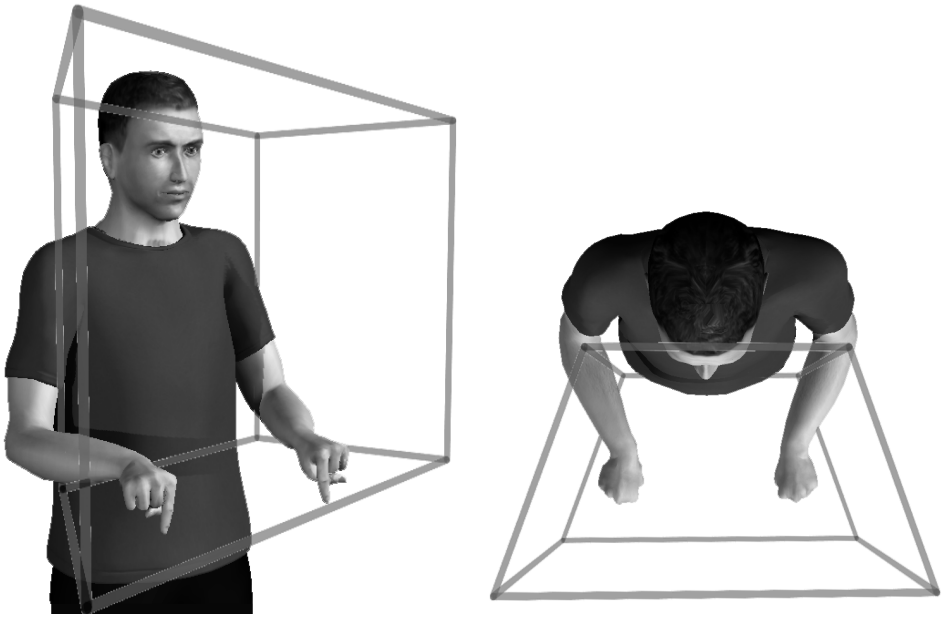


Figure 3: Signing space

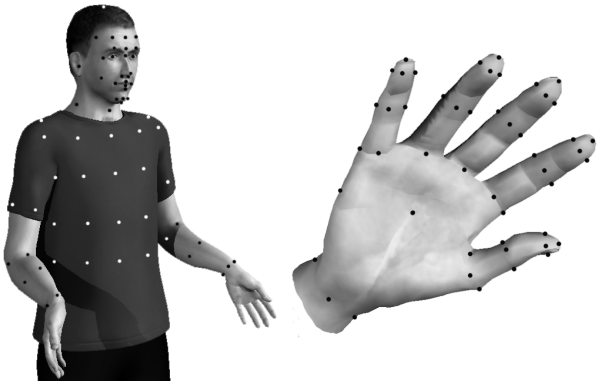


Figure 4: Locations definable in HamNoSys on the surface of the body

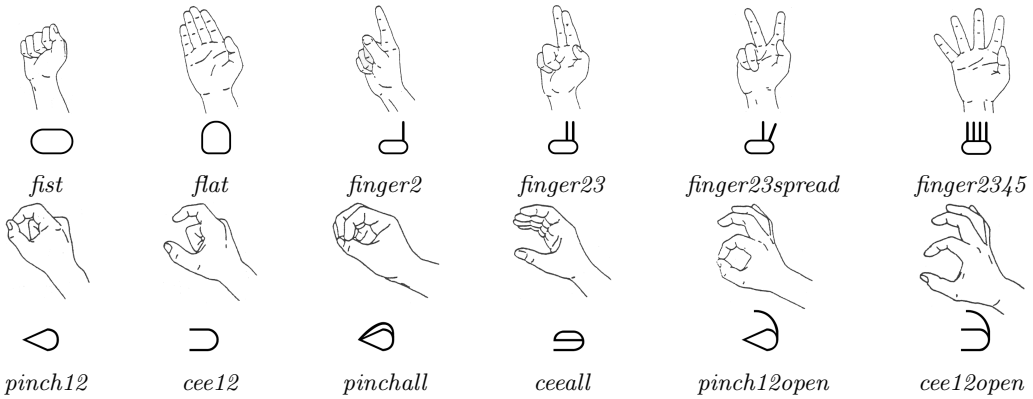


Figure 5: The basic hand shapes, their HamNoSys symbols, and their SiGML names

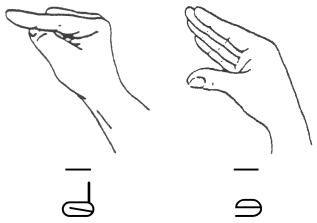


Figure 6: Examples of modified hand shapes

symbols can be regarded as abbreviations which could be eliminated by expanding the transcription.

For someone practiced in the use of HamNoSys, it takes about a minute to write down a transcription of a sign. This compares very favourably with the time it would take to either animate a sign by hand with animation software such as 3D Studio Max, or to record a sign in a motion capture session.

3.2 SiGML

HamNoSys was originally designed as a medium for researchers to communicate with each other, and not with computer processing in mind (other than storing HamNoSys transcriptions in databases). As a result, it had initially no formal syntax, and only a verbal description of the semantics in the form of the HamNoSys manual [29]. To separate out the issues of concrete syntax, we obtained from the IDGS a first draft of a formal grammar, which was then further developed jointly. The initial formal grammar was developed *post hoc*, and in some cases was contradicted by actual practice in the existing corpus of HamNoSys transcriptions. Some transcriptions were indeed erroneous (as judged by the designers of HamNoSys), but others had to be accepted as correct and the grammar modified to allow them.

We then developed an XML application language to represent the grammatical structures of HamNoSys more abstractly, called SiGML: Signing Gesture Markup Language. A parser was written to convert HamNoSys into SiGML, and all subsequent processing is performed on the SiGML, which is read and parsed by the freeware *expat* XML parser. Figure 7 gives the SiGML translation of the HamNoSys of Figure 1.

```
<sign_manual both_hands="true" lr_symm="true">
  <handconfig handshape="finger2" thumbpos="across" mainbend="bent"/>
  <handconfig extfidir="o" palmor="d"/>
  <location_bodyarm location="stomach" side="right_at"/>
  <rpt_motion repetition="fromstart">
    <directedmotion direction="d" size="small"/>
  </rpt_motion>
</sign_manual>
```

Figure 7: SiGML transcription of "here"

4 Animating SiGML

HamNoSys and SiGML are signer- and avatar-independent. To generate motion data for a particular avatar, numerical information about the avatar must be supplied, to be combined with the avatar-independent description of the movement, to produce avatar-specific motion data. The information required divides into two classes, information about the geometry of a specific avatar, and information about body language or signing style, which can vary independently of the avatar.

The geometrical information required about the avatar consists of:

1. The dimensions of the avatar's skeleton: the lengths of its bones, and the positions and orientations of the bones in some standard posture.
2. For each of the non-manuals defined in HamNoSys, the mixture of avatar-specific facial morphs or bone rotations which implements it.
3. The coordinates of every location nameable in HamNoSys, relative to the bone which moves the relevant body part. In principle, a HamNoSys location should be considered as a volume having a certain extent, the choice of which point within that volume to use being dependent on the context, but for implementation we have preferred to model each location by a single fixed point.

The first of these can in principle be read automatically from whatever avatar description file is exported by the modelling software used to create it. The second requires the avatar creator to express each of the HamNoSys facial movements as a combination of the avatar's morphs. The third requires the avatar creator to specify the coordinates of all the HamNoSys locations on the surface of the body. Some of these locations can be discovered automatically. For example, it is fairly easy to determine surface points on the palmar, dorsal, ulnar, and radial sides of each finger, level with the midpoint of the middle phalanx. Other points are best placed manually: it is easier for the modeller to say where the ears are than to determine them algorithmically from analysis of the surface mesh. Specifying the locations of these surface points should be considered a necessary part of the construction of any avatar intended for synthetic animation. Note that no other knowledge is required of the surface mesh itself, which may be arbitrarily dense, subject to the requirement of rendering it at the desired frame rate.

The "body language" class of information consists of the mapping of HamNoSys categories to numbers. It requires:

1. A numerical definition of the size of a "large", "medium", or "small" movement, "near" and "far" proximities, the shape of a "deep" or "shallow" curved arc, and

similarly for all the other geometric categories in HamNoSys. Distances are best given as proportions of various measurements of the avatar: for example, "far" from the chest might be defined as a certain proportion of the length of either arm.

2. A numerical specification of the time required for each type of movement, and of how that time should change when HamNoSys specifies the manner of movement as "fast", "slow", "tense", etc.
3. A numerical specification of how the hands should accelerate and decelerate during movements of different types, performed with different manners.
4. The temporal trajectories that the avatar-specific morphs and bone rotations that implement non-manual movements should follow. This takes the form of attack, sustain, and release times, plus a description of how the elements of the movement ramp up from zero to the full value during the attack, and down again to zero during the release.

Body language information is specific, not to a particular avatar body, but to the personality inhabiting it.

Given all of this information, generating motion data requires solving the following problems.

4.1 Hand location

HamNoSys and SiGML always specify the locations of the hands, but to determine exactly what is meant by a piece of notation that is clear to the human reader can be surprisingly non-trivial.

The hand is an extended object: the whole hand cannot be placed at some point in space, only some particular point on the hand. HamNoSys allows a point on the hand to be specified, but does not require it. If it is omitted, an implementation of HamNoSys must guess which point is intended. Generally, when the hands are not in contact with each other or the body, the centre of the palm is a reasonable choice, but this will usually not be suitable when there are contacts or close proxomities. For example, the BSL sign for "me" uses a pointing hand positioned close to or touching the chest, pointing to the signer. If the centre of the palm is placed there, the index finger will penetrate the chest. The significant location on the hand is here the index fingertip.

There is a trade-off here between the effort required of the program and its author in order to correctly guess the intended meaning, and the effort required of the writer of HamNoSys or SiGML in being explicit about such details.

In one significant case there is no way to express in HamNoSys certain information about hand location. Some HamNoSys transcriptions express the spatial relationship of the two hands to each other, and the relationship of the pair of hands considered as a whole to the torso. For the first part of this information, all that HamNoSys can express is the distance between the two hands, and the locations on the hands that are to be that distance apart. The direction from one location to the other cannot be expressed in HamNoSys. There are some heuristic rules that one can formulate for guessing this information, based on inspection of example transcriptions, but there is no way to be sure that the rules will be correct in all cases, because there is no formal definition of what the correct choice is. For making new transcriptions directly in SiGML, it would be easy to extend SiGML with an additional attribute of the relevant

XML element specifying the direction. However, this would not help in dealing with legacy HamNoSys transcriptions, such as the IDGS corpus of several thousand signs of German Sign Language.

4.2 Arm positioning

Most HamNoSys transcriptions specify only the hand movements. The arms are assumed to move in whatever way is natural to achieve this. The animation software must therefore have inverse kinematic rules to decide how the elbows and shoulders should move in order to place the hands into the required positions and orientations. The rules that we have implemented were derived from informal observation of signers, experimentation, and by obtaining feedback from signers about the perceived quality of the postures and movement.

1. When a hand is in signing space, the plane formed by the upper and lower arm bones should make some angle α with the vertical plane through the shoulder and wrist joints. This angle should vary from zero when the hand is directly to the right of the body, to some angle α_{max} when the hand reaches across to the opposite side.
2. The wrist joint has limited mobility, especially for side-to-side movements. Therefore the angle α chosen by the previous constraint may need to be further modified to prevent excessive wrist bending.
3. When the hand reaches beyond a certain distance from the shoulder, the collarbone should rotate to move the shoulder some distance in the direction of reach.
4. These rules are extended to cover all hand positions and orientations likely to arise in signing, in such as way as to make the resulting arm configuration vary continuously with the hand.

For reaching movements across the body, the animation would be further improved by allowing the upper body to turn and tilt in the direction of reach, but this has not yet been implemented.

4.3 Handshapes

HamNoSys handshape descriptions can be quite complex. To a basic handshape selected from those of Figure 5 may be added any or all of the following:

1. A finger bending to override the bending of all the extended fingers or thumb of the basic handshape. The "extended fingers" are, for example, all of them for the *flat* handshape, the index finger and thumb for the *pinch12* handshape, etc. There are five possible bendings, beside the default value for the handshape, which can nominally be expressed in terms of the amount of bending of the three joints of the finger, as a proportion of the maximum possible. The joints are listed in order going outwards from the palm. *bent* $= (1, 0, 0)$, *round* $= (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, *hooked* $= (0, 1, 1)$, *dblbent* $= (1, 1, 0)$, *dblhooked* $= (1, 1, 1)$. The actual joint angles may vary substantially from these nominal values, and depend on the avatar.
2. A thumb modifier: splayed, opposing the fingers, or folded across the palm. In the case of the three *cee*-type handshapes (*cee12*, *ceeall*, and *cee12open* in Figure 5), a thumb modifier may also specify that the gap between fingertips and thumb tip should be wider or narrower than normal.

3. A specification of which fingers are to be the extended fingers: for example, to specify that the small finger should be extended instead of the index finger in the *finger2* handshape.
4. A specification that a particular finger should be given a specific bending.
5. A specification that one finger should be crossed over another.
6. A specification that the thumb should be placed between two fingers.

Turning a general handshape description into a set of joint angles appropriate to a given avatar would ideally be done by using the dimensions of the avatar's hands to calculate the joint rotations that would produce the correct relationships between all of its parts. Instead of doing this, we calculated lookup tables for the avatars that we were using, by posing the avatar's hands in various shapes and reading the joint rotations. This involved less work, given the limited number of different avatars that we had available to animate, but in the longer run it would be preferable to calculate the tables automatically from the hand geometry.

The lookup tables express the basic handshapes and finger bending modifiers, not in terms of joint rotations directly, but in terms of a parameterisation of each joint according to the number of its degrees of freedom. The middle and end joints of the fingers and thumbs are adequately described as hinge joints, and can be parameterised by a single number. The base joints are more complicated.

The base joint of each finger has two degrees of freedom, which can be labelled "bend" and "splay". In HamNoSys, these are independently specified. Bend is implied by the handshape or explicitly given by a fingerbending modifier. Splay is implied by the handshape: *finger23spread*, *finger2345*, *cee12open*, and *pinch12open* have various degrees of splay of the extended fingers, the other handshapes do not. The problem is then to map any given amounts of bend and splay to the corresponding rotation, taking into account the interaction between the two. The interaction has two effects. Firstly, bend limits splay: when the base joint is fully bent towards the palm, it has almost no mobility about any other axis. So when the value of bend is at its maximum, the rotation should be independent of the value of splay. Secondly, the space of all physiologically possible rotations of the joint is a two-dimensional subset of the three-dimensional space of all mathematically possible rotations. We require a description of that space which makes it easy to calculate the rotation corresponding to a given bend and splay. The third mathematically existing but physiologically infeasible degree of freedom is twist about the axis of the finger.

There are two standard types of mechanical joint having two degrees of rotational freedom: the universal joint and the constant velocity joint[2]. We found that neither of these, with their natural parameterisations, provides an accurate mapping of bend and splay to joint rotation, no matter how the bend and splay axes are selected. All of them require a significant amount of twist to be added for some combinations of bend and splay in order to reproduce the observed orientation of a real finger. To compute this twist one must either construct a lookup table defining twist as a function of bend and splay, deriving a continuous function by interpolation, or find an algorithmic method of computing the required total rotation of the joint. We adopted the latter approach, and require just three extra measurements of each of the avatar's fingers. These could

[2]The set of rotations of a constant velocity joint is the set of all rotations about all axes passing through a given point and lying in a given plane. The name derives from its use in the front half-axles of a front wheel drive vehicle, where it ensures that when the driving side turns at a constant velocity, so does the driven side.

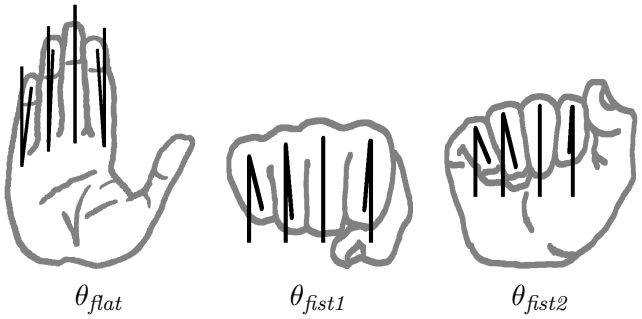


Figure 8: Angles characterising finger base joint articulation

be automatically computed from the locations of some of the definable points on the hands, but at present have been measured manually for each avatar. The measurements are of deviations between the actual directions of the proximal and medial phalanges of the fingers in certain handshapes, and the directions they would have in an ideal simplified hand whose fingers in a flat handshape are straight and parallel to the axis of the hand, and which bend along axes exactly transverse to the fingers. See Figure 8.

1. In the flat hand (in which the fingers are straight and touch each other all along their length), the angle θ_{flat} (a different angle for each finger) between the actual direction of the proximal phalanx of each finger and the ideal direction. Typically, there is some inward tapering of the fingers, depending on their thickness.
2. The same measurement, taken for a fist, giving angles θ_{fist1} for each finger.
3. The same measurement, taken for the medial phalanges in a fist, giving angles θ_{fist2} for each finger.

Let β, σ, and λ be the bend, splay, and longitudinal axes of rotation. These are orthogonal: λ is the axis of the hand, σ is normal to the plane of the hand, and β is perpendicular to both. Write (α, a) for a rotation about axis α of angle a, and define $\mathrm{mid}(a, b, k) = a(1-k) + bk$. Then given angles b and s for bend and splay, and defining $b' = 2b/\pi$, we construct the total rotation as the following composition of four components:

$$(\sigma, \mathrm{mid}(s, 0, b')) \cdot (\beta, b) \cdot (\sigma, \mathrm{mid}(\theta_{flat}, \theta_{fist1}, b')) \cdot (\lambda, \theta_{fist2} b')$$

When $b = 0$, this reduces to $(\sigma, s + \theta_{flat})$ and when $b = \frac{1}{2}\pi$, it reduces to $(\beta, \frac{1}{2}\pi) \cdot (\sigma, \theta_{fist1}) \cdot (\lambda, \theta_{fist2})$, which can be verified to produce the finger orientations exhibited in Figure 8 for the flat and fist handshapes. (Some care is necessary about the choice of directions of the axes and signs of the rotation angles.) In the figure, the three angles are all zero for the middle finger. For θ_{flat}, this is because we define the longitudinal axis of the hand to be the direction of the middle finger in the flat hand. For θ_{fist1} and θ_{fist2}, this is a contingent fact about the particular hand from which the illustrations were made.

Although these angles are typically no more than a few degrees, they make a large difference to the naturalness of the resulting handshapes. Setting them all to zero produces unrealistic claw-like hands. Comparative examples are shown in Figure 9. (The abnormal shortness of the medial phalanx of the ring finger is a defect of the avatar.)

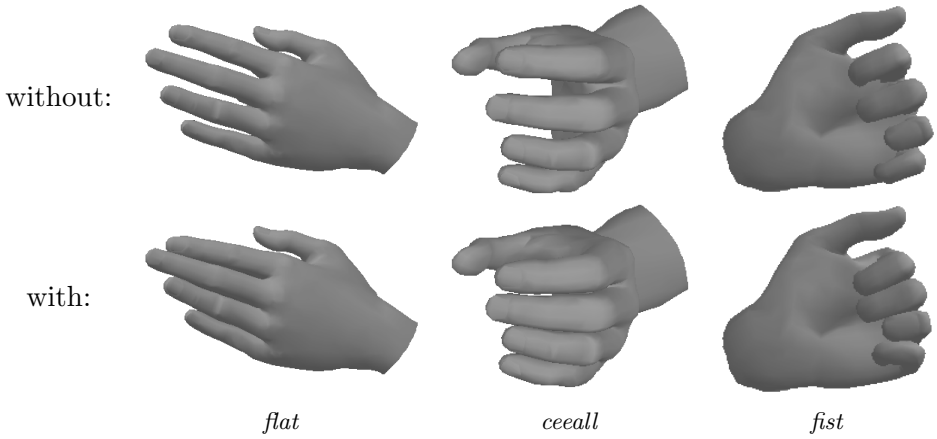


Figure 9: Handshapes implemented with and without the base joint articulation angles

The carpometacarpal joints of the fourth and fifth fingers (joints which lie inside the palm) have some limited amount of mobility which is used when forming a handshape in which the fingers wrap round an object such as a tennis ball. Although one is typically not aware of the movement, it is significant: without that mobility, one would hardly be able to pick up a tennis ball with the whole hand. These joints can be modelled as hinges with their axes lying obliquely across the palm, and which should receive nonzero rotations when the fourth and fifth fingers have nonzero values for both bend and splay. (Many supposedly fully articulated avatars do not include these joints.)

The base joint of the thumb is more complicated. It appears to have two degrees of freedom, which can be very roughly characterised as the angle between the metacarpal bone and the axis of the hand, and the rotation of the thumb about that axis, but a mapping of two real parameters to the space of rotations that produces realistic articulation of all the required handshapes is difficult to synthesize. We eventually discarded the attempt to describe it in terms of bend, splay, and an algorithmically determined twist, and defined the necessary amount of twist for the basic thumb positions in the lookup tables describing the basic handshapes. Intermediate thumb positions are then defined by interpolation of these three parameters.

HamNoSys can also define a handshape as being intermediate between two given handshapes. (An example is shown in Figure 12; the backslash is the betweenness operator.) In this case we linearly interpolate between the representations of the handshapes in terms of joint parameters, before converting to joint rotations.

4.4 Collisions

A certain amount of collision detection and avoidance is required in order to prevent the avatar passing its arms or hands through its body. However, we make the basic assumption that the transcription we are given does not describe something physically impossible, such as a hand movement that begins in contact with the chest and then moves inwards. We have only found it necessary to implement collision avoidance for two situations:

1. When reaching across the body, the upper arm must be prevented from entering the torso, by suitably raising the elbow.

2. Because the front of the torso cannot be assumed to be a vertical plane, a movement that begins in contact with the torso and moves up, down, or across must be modified so as to avoid penetrating the surface of the torso.

To fulfil these constraints, it is not necessary to know the complete surface mesh of the avatar, merely a sufficiency of locations on its surface for interpolation to yield a good enough result. The set of HamNoSys locations is sufficiently dense to achieve this. For this reason, the actual surface mesh of the avatar (which might be arbitrarily complex) does not need to appear in the avatar description used by the synthesis software.

4.5 Trajectories

The route by which a hand moves from one location to another is in general explicitly specified by the transcription, and therefore does not need to be decided by the software. The accelerations and decelerations along that path are not specified in any detailed way. HamNoSys distinguishes five modalities of movement: normal, fast, slow, tense[3], and "sudden stop at the end" (such as in a sign for "punch"). In addition, we distinguish four separate types of "normal" movement in signing: "targetted", "lax", "hard contact", and "linear". These are not explicitly notated, but are implied by context.

Targetted movement is used for meaningful movement, as opposed to lax movement, which is used for non-meaningful movements. There is a marked visual difference between the two. The deceleration at the end of a targetted movement is much greater and shorter than the acceleration at the beginning, giving a velocity profile that peaks after the midpoint of the movement. Lax movement has a broader velocity peak lying more in the middle of the movement, with a sharper initial acceleration and a longer final deceleration than targetted movement. An example of lax movement is in the return stroke of most repeated movements, such as the upwards stroke of the sign for "here" in Figure 1, and the outwards stroke of the elbow in Figure 2. HamNoSys can also notate a form of repetition in which the return stroke is as significant as the forward stroke; for a sign using this repetition, both directions would be performed with the targetted trajectory.

The distinction between lax and targetted movement can also be seen in the transitional movement from the end of one sign to the beginning of the next. When the second sign consists of a single static posture (such as the BSL sign for "me", in which the hand points to the signer), the transition is targetted, but when the second sign contains movement, the movement into its initial posture from the previous sign is lax. (In an early version of the implementation, we used targetted motion for all inter-sign transitions, and received comments from deaf people that those movements looked wrong: they appeared meaningful when they were not. Using lax transitions except before static signs eliminated this criticism.)

Hard contact is targetted movement which ends with the hand contacting a solid object such as the other hand or the torso, and which produces a more sudden stop than a targetted movement ending in mid-air. At the end of the movement, the hand remains in contact with the object for a perceptible time before moving again.

Linear movement, where the distance moved is proportional to elapsed time, is never appropriate for straight line movements in lifelike humanoid animation, but it is

[3] "Tense" here means a slow and steady movement, as if performed with great effort. This is visually quite distinct from a movement which is merely slow, even when the face (which may communicate some of the associated affect) is concealed.

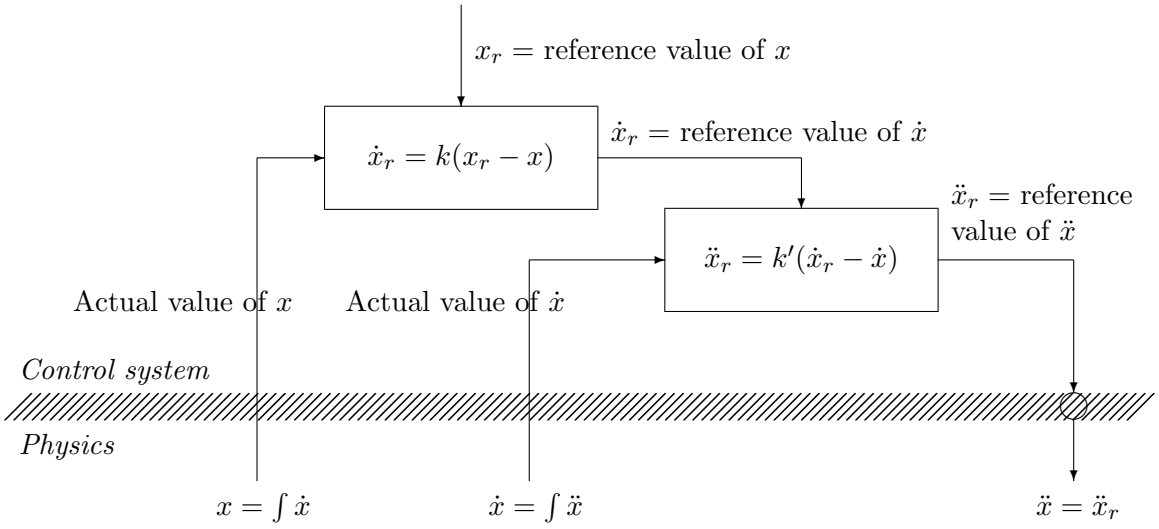


Figure 10: Semi-abstract biocontrol model for synthesizing temporal trajectories

applicable when the hand makes a circular motion. Apart from the transitions into and out of the movement, the angular velocity remains close to constant.

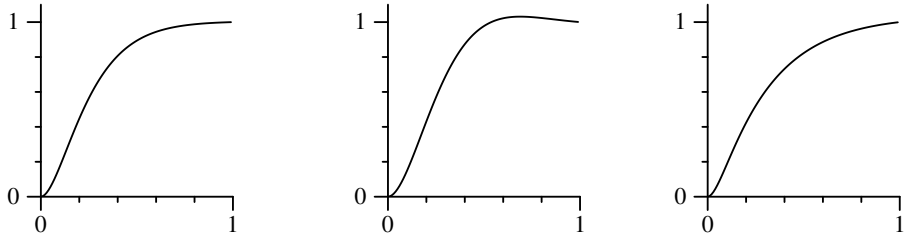
To synthesize suitable temporal trajectories for the different manners, we use the semi-abstract biocontrol model illustrated in Figure 10. This is a two-level cascade control system built from two proportional controllers. The part below the shaded line is the simulated physics. The controlled variable is x, representing the proportion of the distance that has been travelled towards the desired final position, and is initially zero. x_r is the reference or target value, initially 1. The output of the outer controller is $\dot{x}_r = k(x_t - x)$, a demanded rate of change of x which is proportional to the error, the difference between the reference value x_r and the actual value x. This is the reference input to the inner controller, which compares it with the actual rate of change $\dot{x}$, and outputs a demanded acceleration $\ddot{x}_r$ proportional to the error $\dot{x}_t - \dot{x}$. The virtual actuator sets $\ddot{x}$ equal to $\ddot{x}_r$, which determines the time evolution of $\dot{x}$ and x.

The equation of motion of the resulting system is

$$\ddot{x} + k'\dot{x} + kk'x = kk'x_t$$

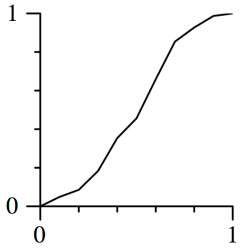
With constant positive gains in both of the proportional controllers, this is mathematically equivalent to damped simple harmonic motion, critically damped when $k'/k = 4$, underdamped for smaller values of k'/k, and overdamped for larger values. The top row of Figure 11 illustrates some trajectories of the system. The horizontal axis is scaled so that the major part of the motion (when it has settled to within 99% of the final value) occurs in one unit of time.

We found that all choices of k'/k gave sluggish behaviour, compared with the performance of signers, especially towards the end of the movement. We modified the system by making the stiffness increase as the target is approached (that is, by increasing k and k' as the error $x_r - x$ decreases, while keeping their ratio constant), so as to model the increase in tension of the muscles as the intended location is reached. This resulted in animations that were judged to be satisfactory, and compared well with trajectories

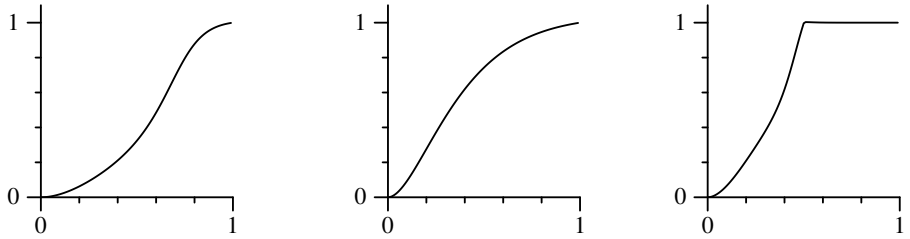


Critically damped ($k'/k = 4$) Underdamped ($k'/k = 2$) Overdamped ($k'/k = 8$)

Trajectories of the basic biocontrol model.



Trajectory of human signer measured from video. Compare with "targetted" below.



Targetted Lax Sudden stop

Trajectories of the modified biocontrol model.

Figure 11: Horizontal axes: time; vertical axes: position.

observed by frame-by-frame stepping through a signing video, of which an example is given in the middle row of Figure 11, for the sign "west" (the flat right hand, facing left, makes a leftwards movement). The movement lasts for 11 frames at 15 frames per second. The "sudden stop" modality uses a greater increase of k and k', sets their ratio to be underdamped, and truncates the trajectory when the final position is reached. Varying the parameters of the system produced suitable trajectories for each of the other HamNoSys modalities, as well as the distinctions between targetted, hard contact, and lax movement. Because HamNoSys manners are a discrete set, we can greatly reduce the computation time by precomputing a trajectory for each one, in the form of a lookup table for a function $f : [0 \ldots 1] \rightarrow [0 \ldots 1]$. When a fraction k of the duration of the movement has elapsed, the hand has moved a fraction $f(k)$ along its path. The bottom row of Figure 11 illustrates some of these trajectories. The "targetted" trajectory is the one that would be used to animate the sign that was measured from video.

In our implementation, the time taken for a movement is independent of the size of the movement. Some researchers have adjusted the timing according to Fitts' law [7], which states that the time for a reaching movement is a linear function of the log of the distance. However, his observations were for reaching movements to a target of definite size and location, in which subjects were asked to favour accuracy over speed. Signing movements are not of this nature. For example, the movements of the hands in the BSL sign for "here" illustrated in Figure 1 have no particular target location of any definite size, and the sign for "Scotland" (Figure 2) does not involve hand movements at all. Observation of signers' movements suggests there is in fact little variation of duration with distance (which, at the level of accuracy and range of distances we are dealing with, is also a reasonable interpretation of the logarithmic relationship expressed by Fitts' law).

The duration can be observed to vary with the shape of the movement: curved arcs are performed more slowly than straight movements, and circular movements take longer still. The durations used by the implementation were arrived at through feedback from signers.

We use the same temporal trajectories for all the other types of movement: change of hand orientation, change of handshape, non-manual movements, and change of facial morphs.

4.6 Interpolation between postures

The trajectories defined in the previous section specify what proportion of the path from one posture to the next the avatar should have moved after any given elapsed time. We now describe the actual interpolation between postures.

The location of a hand at any instant is defined by the spatial path described by the SiGML, the particular point on the hand that must traverse this path, and the proportion of the path that has been traversed. Interpolation is linear with respect to distance travelled along the path.

The path in rotation space from one hand orientation to the next is that defined by spherical linear interpolation.

The location and orientation of the hand at each moment define the location of the wrist joint. From this, the wrist, elbow, shoulder, and clavicle joint rotations are calculated by inverse kinematic routines.

When the hand changes shape, the rotation of each hinge joint of the fingers and thumb is interpolated in the obvious way. The finger base joints interpolate the bend

and splay parameters, and the thumb base joints the bend, splay, and twist parameters described in section 4.3.

Non-manual movements and facial morphs are described as real numbers expressing the amount of each one that is to be present at any instant, and therefore linear interpolation applies.

4.7 Performance

Each frame of animation data consists of about 50 bone rotations, the exact number depending on the avatar: 15 to 17 joints in each hand, 4 in each arm, and several joints in the spine, neck and head. The frame also includes an amount of each of the avatar's facial morphs.

The current system is capable of generating 15,000 frames of animation data per second on a 1.7 GHz desktop PC. This excludes the time taken to read and parse the SiGML input. Although the current system does store SiGML as text, when embedded in a system such as those described in Section 2 it would be feasible to bypass parsing by directly constructing SiGML data structures.

If the animation is to play at 25 frames per second, which gives acceptable legibility for signing[4], this means that the synthesis of motion data from SiGML data structures requires only 0.17% of the time budget, even on what is by current standards a modestly powered machine. At the 60 fps required for fast-action videogames, the fraction required of the time budget is still only 0.4% per avatar.

Synthetic animation from a notation such as SiGML also allows for a great reduction in the amount of data to be stored. The SiGML displayed in Figure 7 represents about 0.8 seconds of animation in about 320 bytes. The corresponding 20 frames of 25 fps animation would (uncompressed) occupy about 12kB, assuming 50 rotations per frame, each represented as a triple of floating point numbers. The latter figure can be substantially reduced by various compression schemes, but it is unlikely that any such scheme could achieve the 37-fold reduction obtained by the SiGML representation. The SiGML itself is susceptible of compression: the HamNoSys in Figure 1 only takes 10 bytes.

5 Automatic generation of SiGML

SiGML is designed to be human-readable and -writable, but for some applications it is necessary to generate SiGML automatically. One example is its original application, sign language.

Signing is often conventionally transcribed as a sequence of "glosses": names of the signs performed, such as *BOOK GIVE*, corresponding to the English sentence "Give me the book". However, this is misleadingly simplified: to animate such a sequence it is not sufficient to merely replace each gloss by its SiGML transcription and generate animation data. This is because many signs take different forms depending on their context. For example, *GIVE* will be made with a handshape indicating the type of object that is being given, and the start and end locations of the movement will indicate who is giving to whom: you to me, him to her, me to multiple people, etc. *GIVE* requires

[4] A commercially produced CDrom for students of signing, from which the example trajectory in Figure 11 was measured, records all video clips at 15 fps.

a set of parameters: the above example is better glossed as *BOOK GIVE*(*you,me,flat-thing*). This variety of instantiations greatly expands the corpus of signs required for interactive discourse beyond what a mere count of the glosses in a dictionary might suggest. For highly inflected signs such as *GIVE*(...), the SiGML transcriptions of its required forms must be generated automatically. Even for signs which might not appear to require parameterisation, such as signs for ordinary concrete objects like *BOOK*, the location at which the sign is performed may express some of the meaning. For example, it may be performed at some arbitrarily chosen location in signing space, in order to be able to refer back to the object later in the discourse by pointing to that location.

In the ViSiCAST project, sign language has been generated from English text via the intermediate form of Discourse Representation Structures and Head-Driven Phrase Structure Grammars. The latter form the input to sign language generation based on the Attribute Logic Engine. This pathway has been described in [30, 31, 20], and will not be further described here. Sign language represented in the parameterised form described above can then be transformed into SiGML, inserting appropriate parameters such as handshape and location.

In virtual reality applications with autonomous interactive avatars, SiGML (or an extension of SiGML covering a wider repertoire of movements) could be generated from the system's internal high-level representation of the avatars' intended actions (e.g. "point to that building"). The SiGML description serves as a low-level, but still avatar-independent, representation of the desired action, which can then be translated into motion data for the particular avatar.

6 Limitations and extensions of HamNoSys

The success of our implementation of SiGML demonstrates the practicality of real-time synthesis of human movement in a certain restricted setting. We are interested in expanding this application to wider repertoires of human movement, such as are required by general interactive virtual reality applications. The experience of implementing SiGML has shown some weaknesses of HamNoSys for this application, and suggests ways in which it might be improved.

6.1 Limitations

6.1.1 Syntax

HamNoSys was originally developed for people to read and write. Its formal grammar was developed in hindsight to fit existing practice. While that practice is easy for those trained in the notation to deal with, the resulting grammar is more complicated than is warranted by the subject matter. This is still true of SiGML. Although it eliminates many of the irrelevant complexities of the concrete syntax of HamNoSys, it is still very much based on HamNoSys. A notation developed specifically for computer animation could be substantially simplified without any loss of expressiveness.

6.1.2 Semantics

HamNoSys depends to a significant extent on human understanding of signing for both parsing and interpretation. A large part of the task of implementing SiGML was taken up by attempts to automatically fill in information that human transcribers typically

omit when transcribing signs, and (often unconsciously) fill in when reading transcriptions.

In the eSign project, among the people who were responsible for creating SiGML transcriptions of the signs required for various applications, those who had less prior experience with HamNoSys sometimes found it easier to produce transcriptions that worked well with the animation system, being less accustomed to depending on human interpretation.

An improved SiGML would have a simpler and more precise semantics, expressed in some form more explicit than merely its implementation.

6.1.3 Handshapes

The twelve basic handshapes of HamNoSys can, in principle, be mapped to (avatar-dependent) tables of finger and thumb joint angles, as can each of the fingerbending modifiers. However, there turns out to be some significant context-dependency. For example, the actual bend angle of a finger that is "bent at the base joint" depends on the basic handshape (compare the base joint angles for the two handshapes in Figure 6, which both use that finger shape modifier). This requires a significant expansion of the per-avatar lookup tables used to translate handshapes to joint rotations. To handle this in a more flexible way, and to cover the less commonly used features of HamNoSys hand descriptions (such as crossed fingers), we augmented the SiGML <`handconfig`> element with the possibility to explicitly specify finger joint angles. This is, however, at the expense of avatar-independence; a better solution would be to calculate the required angles from an avatar-independent specification of the significant properties of the handshape. Such properties would include contacts between different parts of the hand, or opposition of the thumb to specified fingers.

We also note that in the *cee* handshapes, modifications expressing a wider or narrower gap between the fingers and thumb are expressed in HamNoSys by a modifying symbol which describes the thumb position. However, inspection of actual performance shows that changes to the finger-thumb distance are accomplished primarily by varying the bend angle at the base joints of the fingers. The HamNoSys notation for these handshapes does not map in a simple way to physical reality.

6.1.4 Integration of manual and non-manual movements

For historical reasons, HamNoSys makes an artificial distinction between manual and non-manual movements. This separation, enforced by the syntax of HamNoSys and carried over into SiGML, limits the flexibility with which the two classes of movement can be combined and synchronised with each other.

6.2 Extensions

6.2.1 Discrete categories

Several concepts of SiGML are discrete categories: there is a finite set of locations, a finite set of distances, a finite set of directions and sizes of movement, etc. Other applications of scripted animation will require more precision and variability than these provide. For some of these categories HamNoSys allows the specification of a value midway between two built-in values, but this still allows only a limited set of values. Each of these sets of discrete values should be extended to continuous categories by adding

the possibility of specifying fractional interpolation. We have already implemented extensions to SiGML for this purpose.

6.2.2 Timing

We have provided more detailed control of timing than the "fast" and "slow" modalities of HamNoSys, by allowing an absolute duration or a temporal scaling to be applied to any movement or to a whole sign.

6.2.3 Path specification

A SiGML transcription specifies explicitly the spatial paths of the hands — straight line, curved, circular, etc. The software never has to decide what path to follow to move a hand from one place to another. This should remain true for more general animation. The person or automated process that is composing a SiGML transcription may be assumed to be describing physically possible postures and movements. If the author has provided sufficient detail to uniquely determine the posture and movement, the software need not be burdened with the task of guessing or inventing missing detail. If it is desired to ease the animator's task by allowing some details to be omitted and automatically synthesized, algorithms for doing so should be implemented either as a separate SiGML-to-SiGML translation or an external library that can be referenced from SiGML, rather than being built into the implementation of SiGML itself.

6.2.4 Gravitational dynamics

Signing movements are largely independent of gravity. Hands move along their intended paths and the arms do whatever is necessary to achieve that. As a result, sufficiently lifelike accelerations and decelerations can be created from a simple biocontrol model with a small number of parameters, without requiring a detailed physical simulation. In contrast, for lower body movements like walking, or for upper body movements that involve lifting heavy things, gravitational forces are comparable with or greater than muscular forces and play a major role in the dynamics, and hence also the visual appearance. An extension of SiGML to cover these types of movement will require the implementation to be provided with a means of synthesizing them, which will be more complicated than the simple biocontrol model that we have found sufficient for signing animation. Integration with more sophisticated methods such as those surveyed in [21] would be desirable.

6.2.5 Expressive movement

HamNoSys was originally devised for recording signing citation forms, and as such it contains only a discrete range of modalities, those necessary to the correct performance of the signs. Thus SiGML does not have to deal with the entire range of expressive movement. Although signing itself can be very expressive, the "newsreader" type applications in the eSign project do not require it. However, it is an essential aspect of general character animation.

We regard expressiveness as being a concept at a higher level than the intended target of SiGML or any extension of it, and outside the intended scope of this work. SiGML is concerned with the avatar-independent description of concrete posture and movement; it is the responsibility of some higher-level system (such as, for example,

some of the projects briefly surveyed in section 2) to express qualities of expressiveness in concrete terms.

6.3 Principles of posture and movement notation

Some of the principles underlying HamNoSys deserve wider application than they in fact receive in that notation.

6.3.1 Constraint-based posture specification

HamNoSys specifies the positions of the hands either by their relation to each other, and the relationship of the pair of hands considered as a single entity to the body, or by the relationship of each hand to the body separately. This is a useful method of defining posture in an avatar-independent way, but in HamNoSys it is subject to heavy restrictions. The relationship is always one of distance only, not direction (which must be guessed from context). It applies only to position, with hand orientation always defined in absolute terms. We are therefore considering a radically revised language which would specify postures in terms of constraints on the relative positions and orientations of general body parts.

6.3.2 Intention vs. geometry

There is a large number of geometric features of a posture which a transcription notation might record, and there are many different ways of choosing a set of geometric features whose specification will completely determine the posture. For example, any two of the orientation of a hand, the orientation of a straight index finger, and the angle at the base joint of that finger determine the third. Which properties, therefore, should a notation record?

HamNoSys makes a particular choice of features to record. For example, the direction of the hand is expressed in terms of the "extended finger direction" or e.f.d.: the direction the fingers would be pointing in if they were straight, or equivalently, the direction of the metacarpal bone of the middle finger. However, there is evidence [3] that this property of the hand is not phonologically significant for signing. For a handshape that is pointing with the index finger, what is significant is the direction in which the index finger is pointing, which may be identified with the direction of the distal phalanx. So long as that direction is correct, the angle at the base joint of the index finger and the e.f.d. of the hand can both vary considerably without affecting the meaning of the gesture.

This leads to a tendency in some circumstances for HamNoSys transcribers to record an idea of the sign which differs from its actual performance. An example is the sign for "me": an inward pointing index finger. This is commonly but incorrectly transcribed as a *finger2* handshape with an inwards e.f.d. That would be a rather awkward posture to perform, and differs from actual performance. The transcription and actual performance are compared in Figure 12.

We therefore believe, as argued in [15], that a gesture transcription notation should record those properties of a gesture which are essential to its significance: the properties which will remain invariant when the same gesture is performed by avatars of different proportions or different gestural styles. In a slogan, *intention is primary, geometry is secondary.* For a pointing handshape, the direction of pointing is significant; this is

Figure 12: Incorrect and correct transcriptions of “me”

what should be recorded, and the bending of the pointing finger in Figure 12 should be determined from inverse kinematic rules describing the performance of pointing gestures.

7 Conclusions

The present work has demonstrated the practicality of generating gesture animation in real time from an avatar-independent scripting language. The generation of motion data is done not merely in real time, but in a very small fraction of the time budget, each single frame of data taking on average 1/15 of a millisecond to generate on a typical machine. The animations are of a degree of naturalism which has proved acceptable to deaf users, in terms of both legibility and realism. The SiGML representation of a movement is also many times smaller than the motion data which it generates.

Avatars for use with this method must be furnished with additional data beyond the skeleton topology and dimensions, and repertoire of morphs. The additional data consists primarily of the locations of a standard list of named points on the surface of the body.

The work has suggested directions in which SiGML can be improved and extended in order to apply the method to other applications of real-time, naturalistic, humanoid animation.

References

[1] Yasmine Arafa, Kaveh Kamyab, Ebrahim Mamdani, Sumedha Kshirsagar, Nadia Magnenat-Thalmann, Anthony Guye-Vuillème, and Daniel Thalmann. Two approaches to scripting character animation. In *Embodied Conversational Agents - let's specify and evaluate them!*, 2002.

[2] Diane Chi, Monica Costa, Liwei Zhao, and Norman Badler. The EMOTE model for effort and shape. In *SIGGRAPH 2000*, Washington DC, USA, 2000. ACM.

[3] Onno Crasborn. *Phonetic implementation of phonological categories in Sign Language of the Netherlands.* PhD thesis, University of Leiden, 2001.

[4] B. de Carolis, V. Carofiglio, M. Bilvi, and C. Pelachaud. APML, a mark-up language for believable behavior generation. In *Embodied Conversational Agents - let's specify and evaluate them!*, 2002.

[5] P. Eisert and B. Girod. Analyzing facial expressions for virtual conferencing. *IEEE Computer Graphics and Applications*, 18(5):70–78, Sep 1998.

[6] R. Elliott, J. R. W. Glauert, J. R. Kennaway, and I. Marshall. Development of language processing support for the ViSiCAST project. In *ASSETS 2000: 4th International ACM SIGCAPH Conference on Assistive Technologies*, Washington DC, USA, 2000. ACM.

[7] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movements. *J. Experimental Psychology*, 47:381–391, 1954.

[8] S. Gibet and T. Lebourque. High-level specification and animation of communicative gestures. *J. Visual Languages and Computing*, 12:657–687, 2001.

[9] Angus B. Grieve-Smith. SignSynth: a sign language synthesis application using Web3D and Perl. In Ipke Wachsmuth and Timo Sowa, editors, *4th International Workshop on Gesture and Sign Language Based Human-Computer Interaction*, volume 2298 of *Lecture Notes in Artificial Intelligence*, pages 134–145. Springer, 2001.

[10] Thomas Hanke. HamNoSys in a sign language generation context. In Rolf Schulmeister and Heimo Reinitzer, editors, *Progress in sign language research (International Studies on Sign Language and Communication of the Deaf, vol.40)*, pages 249–264, 2002.

[11] Zhisheng Huang, Anton Eliëns, and Cees Visser. STEP: A scripting language for embodied agents. In Helmut Prendinger, editor, *Proceedings of the Workshop on Lifelike Animated Agents*, 2002. `http://www.miv.t.u-tokyo.ac.jp/pricai02-LAA/online-proceedings.html`.

[12] Zhisheng Huang, Anton Eliëns, and Cees Visser. Implementation of a scripting language for VRML/X3D-based embodied agents. In *Proc. Web3D 2003 Symposium*, pages 91–100, 2003.

[13] Ann Hutchinson Guest. *Labanotation: The System of Analyzing and Recording Movement*. Routledge, 1987.

[14] J. Richard Kennaway. Synthetic animation of deaf signing gestures. In Ipke Wachsmuth and Timo Sowa, editors, *4th International Workshop on Gesture and Sign Language Based Human-Computer Interaction*, volume 2298 of *Lecture Notes in Artificial Intelligence*, pages 146–157. Springer, 2001.

[15] J. Richard Kennaway. Experience with and requirements for a gesture description language for synthetic animation. In Antonio Camurri and Gualtieri Volpe, editors, *5th International Workshop on Gesture and Sign Language Based Human-Computer Interaction*, volume 2915 of *Lecture Notes in Artificial Intelligence*, pages 300–311. Springer, 2003.

[16] Alfred Kranstedt, Stefan Kopp, and Ipke Wachsmuth. MURML: A multimodal utterance representation markup language for conversational agents. In *Embodied Conversational Agents - let's specify and evaluate them!*, 2002.

[17] Thierry Lebourque and Sylvie Gibet. A complete system for the specification and the generation of sign language gestures. In A. Braffort, R. Gherbu, S. Gibet,

J. Richardson, and D. Teil, editors, *3rd International Workshop on Gesture-Based Communication in Human-Computer Interaction*, volume 1739 of *Lecture Notes in Artificial Intelligence*, pages 227–238. Springer, 1999.

[18] Thierry Lebourque and Sylvie Gibet. High level specification and control of communication gestures: the GESSYCA system. In *Computer Animation '99*, pages 24–35, 1999.

[19] A. Marriott and J. Stallo. VHML: uncertainties and problems. In *Workshop on Embedded Conversational Agents*, 2002.

[20] I. Marshall and E. Sáfár. Sign language translation in an ALE HPSG. In Stefan Müller, editor, *11th International Conference on Head-Driven Phrase Structure Grammar*, pages 189–201. CSLI Publications, 2004.

[21] Franck Multon, Laure France, Marie-Paule Cani-Gascuel, and Giles Debunne. Computer animation of human walking: a survey. *The Journal of Visualization and Computer Animation*, 10(1):39–54, 1999.

[22] Han Noot and Zsófia Ruttkay. Gesture in style. In Antonio Camurri and Gualtieri Volpe, editors, *5th International Workshop on Gesture and Sign Language Based Human-Computer Interaction*, volume 2915 of *Lecture Notes in Artificial Intelligence*, pages 300–311. Springer, 2003.

[23] Igor S. Pandzic and Robert Forchheimer, editors. *MPEG-4 Facial Animation - The standard, implementations and applications*. John Wiley & Sons, 2002.

[24] C. Pelachaud. Overview of representation languages for ECAs. Technical report, IUT de Montreuil, 2003.

[25] C. Pelachaud and M. Bilvi. Computational model of believable conversational agents. In *Communications in Multi-Agent Systems*, volume 2650 of *LNCS*, pages 300–317, 2003.

[26] M. Ponder, G. Papagiannakis, T. Molet, N. Magnenat-Thalmann, and D. Thalmann. VHD++ development framework: towards extendible, component based VR/AR simulation engine featuring advanced virtual character technologies. In *Computer Graphics International*, pages 96–104, 2003.

[27] Helmut Prendinger, Sylvain Descamps, and Mitsuru Ishizuka. MPML: a markup language for controlling the behavior of life-like characters. *Journal of Visual Languages and Computing*, 15:183–203, 2004.

[28] Helmut Prendinger and Mitsuru Ishizuka, editors. *Life-like Characters: Tools, Affective Functions and Applications*. Cognitive Technologies. Springer, 2004.

[29] S. Prillwitz, R. Leven, H. Zienert, T. Hanke, J. Henning, et al. *HamNoSys Version 2.0: Hamburg Notation System for Sign Languages — An Introductory Guide*. International Studies on Sign Language and the Communication of the Deaf, Volume 5. University of Hamburg, 1989. Version 4.0 is documented on the Web at `http://www.sign-lang.uni-hamburg.de/Projekte/HamNoSys/HNS4.0/HNS4.0de/Inhalt.html`.

[30] E. Sáfár and I. Marshall. The architecture of an English-text-to-sign-languages translation system. In G. Angelova, editor, *Recent Advances in Natural Language Processing (RANLP)*, pages 223–228. Tzigov Chark, Bulgaria, 2001.

[31] E. Sáfár and I. Marshall. Sign language translation via DRT and HPSG. In A. Gelbukh, editor, *Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, LNCS, pages 58–68, Mexico City, Mexico, 2002. Springer-Verlag.

[32] Gaël Sannier, Selim Balcisoy, Nadia Magnenat-Thalmann, and Daniel Thalmann. VHD: a system for directing real-time virtual actors. *The Visual Computer*, 15(7/8):320–329, 1999.

[33] William C. Stokoe, Dorothy Casterline, and Carl Croneberg. *A Dictionary of American Sign Language on Linguistic Principles, rev. ed.* Linstok Press, Silver Spring, Maryland, 1976.

[34] Margriet Verlinden, Corrie Tijsseling, and Han Frowein. Sign language on the WWW. In *Proc. 18th Int. Symposium on Human Factors in Telecommunication (HFT 2001)*, 2001.

[35] I. Zwiterslood, M. Verlinden, J. Ros, and S. van der Schoot. Synthetic signing for the deaf: eSIGN. In *Proceedings of the Conference and Workshop on Assistive Technologies for Vision and Hearing Impairment (CVHI)*, 2004.