Synthetic Animation of Deaf Signing Gestures

Richard Kennaway

School of Information Systems, University of East Anglia, Norwich, NR4 7TJ, U.K.
jrk@sys.uea.ac.uk

Abstract

We describe a method for automatically synthesizing deaf signing animations from a high-level description of signs in terms of the HamNoSys transcription system. Lifelike movement is achieved by combining a simple control model of hand movement with inverse kinematic calculations for placement of the arms. The realism can be further enhanced by mixing the synthesized animation with motion capture data for the spine and neck, to add natural "ambient motion".

1 Introduction

1.1 Background

The object of the ViSiCAST project is to facilitate access by deaf citizens to information and services expressed in their preferred medium of sign language, using computer-generated virtual humans, or avatars. ViSiCAST addresses three distinct application areas: broadcasting, face-to-face transactions, and the World-Wide Web (WWW). For an account of the whole project see [2].

The task of signing textual content divides into a sequence of transformations:

1. From text to semantic representation.
2. From semantic representation to a sign-language specific morphological representation.
3. From morphology to a signing gesture notation.
4. From signing gesture notation to avatar animation.

This paper deals with the last of these steps, and describes an initial attempt to create synthetic animations from a gesture notation, to supplement or replace our current use of motion capture.

Related work by others in this area includes the GeSsyCa system [3, 10] and the commercial SigningAvatar [17].

1.2 Motion capture vs. synthetic animation

ViSiCAST has developed from two previous projects, one in broadcasting, *SignAnim* (also known as *Simon-the-Signer*) [12, 13, 19], and one in Post Office transactions, *Tessa* [11]. Both these applications use a signing avatar system based

on *motion capture*: before a text can be signed by the avatar, the appropriate lexicon of signs must be constructed in advance, each sign being represented by a data file recording motion parameters for the body, arms, hands, and face of a real human signer. For a given text, the corresponding sequence of such motion data files can be used to animate the skeleton of the computer-generated avatar. The great merit of this system is its almost uncanny authenticity: even when captured motion data is "played back" through an avatar with physical characteristics very different from those of the original human signer, the original signer (if already known to the audience) is nevertheless immediately recognizable.

On the other hand, motion capture is not without drawbacks.

- There is a substantial amount of work involved in setting up and calibrating the equipment, and in recording the large number of signs required for a complete lexicon.
- It is a non-trivial task to modify captured motions.

There are several ways in which we would wish to modify the raw motion capture data. Data captured from one signer might be played back through an avatar of different body proportions. One might wish to change the point in signing space at which a sign is performed, rather than recording a separate version for every place at which it might be performed. Signs recorded separately, and perhaps by different signers, need to be blended into a continuous animation. Much research exists on algorithmically modifying captured data, though to our knowledge none is concerned specifically with signing, a typical application being modification of a walking character's gait to conform to an uneven terrain. An example is Witkin and Popović's "motion warping" [14, 20]. We are therefore also interested in synthetic animation: generation of the required movements of an avatar from a more abstract description of the gestures that it is to perform, together with the geometry of the avatar in use. The animation must be feasible to generate in real time, within the processing power available in the computers or set-top boxes that will display signing, and the transmitted data must fit within the available bandwidth.

Traditional animation (i.e. without computers) is a highly laborious process, to which computers were first introduced to perform in-betweening, creating all the frames between the hand-drawn keyframes. Even when animations are entirely produced on computer, keyframes are still typically designed by hand, but using 3D modelling software to construct and pose the characters. In recent years, physics-based modelling has come into use, primarily for animating inanimate objects such as racing cars or stacks of boxes acted on by gravity. Synthetic animation of living organisms poses a further set of problems, as it must deal not only with gravity and the forces exerted by muscles, but also with the biological control systems that operate the muscles. It is only in the last few years that implementation techniques and fast hardware have begun to make synthetic real-time animation of living movement practical. The tutorial courses [4, 7] give an overview of this history and the current state of the art.

We present here a simplified biomechanical model dealing with the particular application of signing and the constraints of real-time animation.

2 Description of signs

We start from the HamNoSys [16] notation for transcribing signing gestures. This was developed by our partners at IDGS, University of Hamburg, for the purpose of providing a way of recording signs in any sign language. It is not specialised to any individual sign language. This makes it well suited to the ViSiCAST project, which includes work on British, Dutch, and German Sign Languages. The IDGS already has a substantial corpus of HamNoSys transcriptions of over 3000 German signs.

HamNoSys describes the physical action required to produce a sign, not the sign's meaning. It breaks each sign down into components such as hand position, hand orientation, hand shape, motion, etc. HamNoSys up to version 3 only records manual components of signs. The current version 4 also describes facial components, and we are beginning to extend the work reported here to include them.

We will not attempt to describe HamNoSys in detail (see the cited manual), but indicate its main features and the issues they raise for synthetic animation. A typical HamNoSys transcription of a single sign is displayed in Figure 1. This

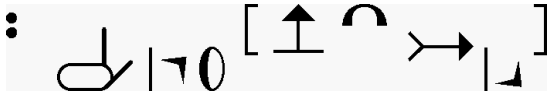

Fig. 1. HamNoSys transcription of the DGS sign for GOING-TO

is the DGS (German Sign Language) sign for "GOING-TO". The colon sign specifies that the two hands mirror each other. The next symbol specifies the handshape: the finger and thumb are extended with the other fingers curled. The next (the vertical line and the bat-wing shape) specifies that the hand points upwards and forwards. The oval symbol indicates that the palm of the right hand faces left (so by mirroring, the palm of the left hand faces right). The part in square brackets indicates the motion: forwards, in an arc curved in the vertical plane, while changing the orientation of the hand so as to point forwards and down.

Note that HamNoSys describes signs in terms of basic concepts which are not themselves given a precise meaning: "upwards and forwards", "curved", "fast", "slow", etc. Other aspects are not recorded at all, and are assumed to take on some "default" value. For example, HamNoSys records the positions of the hands, but usually not of the rest of the arms. For signs which do not, as part of their meaning, require any particular arm movements beyond those physically necessary to place the hands, no notation of the arms is made. It is a basic principle of HamNoSys that it records only those parts of the action which are required to correctly form the sign, and only with enough precision as is necessary. People learning to sign learn from example which parts of the action

are significant, and how much precision is required for good signing. To synthesise an animation from a HamNoSys description requires these choices to be made by the animation algorithms.

HamNoSys was originally developed to meet the needs of humans recording signs. As can be seen from the example, it uses a specialised font; in addition, a grammar was only developed for it *post facto* to formalise the practice of HamNoSys transcribers. As these features make it awkward to process by computer, we have developed a version of HamNoSys encoded in XML [1], called SiGML (Signing Gesture Markup Language). Abstractly, the SiGML representation is identical to the HamNoSys, but represents it as ordinary text without special fonts, and like all XML languages, it can be parsed by standard and widely available XML software. In SiGML, the GOING-TO sign is represented thus:

```
<sign_manual both_hands="true" lr_symm="true">
        <handconfig extfidir="uo" palmor="l"
                handshape="finger2" thumbpos="out"/>
        <tgt_motion>
                <directed_motion direction="o" curve="u"/>
                <handconfig extfidir="do"/>
        </tgt_motion>
</sign_manual>
```

3 Synthesis of static gestural elements

3.1 Disambiguation

We describe by examples how we have approached the task of making precise the fuzzy definitions of HamNoSys concepts.

HamNoSys defines a set of 60 positions in the space in front of the signer. These are arranged in a grid of four levels from top to bottom, five from left to right, and three from near to far. The four vertical levels are indicated by the glyphs (shoulder level), (chest level), (abdomen level), and (below abdomen level). For any given avatar, we define these to be respectively at heights s, $(s+e)/2$, e, and $(e+w)/2$, where s, e, and w are the heights of the shoulder, elbow, and wrist joints of the standing avatar when both arms hang vertically. We have defined these heights in terms of the arms rather than the torso, because these measurements are guaranteed to be present for any avatar used for signing.

HamNoSys indicates left-to-right location by a modification to these glyphs. The five locations at chest level are represented by the glyphs , , , , and . We define their left-to-right coordinates in terms of the positions of the shoulders: centre is midway between the shoulders, and the points left and right are regularly spaced with gaps of 0.4 times the distance between the shoulders.

The three distances from the avatar are notated in HamNoSys by (near), no explicit notation for neutral, and (far); we define these in terms of the shoulder coordinates and the length of the forearm.

An important feature of this method of determining numerical values is that it is done in terms of measurements of the avatar's body, and can be applied automatically to any humanoid avatar. The precise choice of coordinates for these and other points must be judged by the quality of the signing that results. The animation system is currently still under development, and we have not yet brought it to the point of testing.

Hand orientations are described by HamNoSys in terms which are already precise: the possibilities are all of the 26 non-zero vectors (a, b, c), where each of a, b, and c is -1, 0, or 1. When more precision is required, HamNoSys also allows the representation of any direction midway between two such vectors. These directions are the directions the fingers point (or would point if the fingers were extended straight); the orientation of the palm around this axis takes 8 possible values at 45 degree intervals.

To complete the specification of HamNoSys positions requires definitions along similar lines of all the positions that are significant to HamNoSys. In addition to these points in space, HamNoSys defines many contact points on the body, such as positions at, above, below, left, or right of each facial element (eyes, nose, cheeks, etc.), and several positions on each finger and along the arms and torso. The total number of positions nameable in HamNoSys comes to some hundreds. Any avatar for use in Hamnosys-based signing must include, as part of its definition, not only the geometry of its skeleton and surface, and its visual appearance, but also the locations of all of these "significant sites".

3.2 Inverse Kinematics

Given a definition of the numerical coordinates of all the hand positions described by HamNoSys, we must determine angles of the arm joints which will place the hand in the desired position and orientation. This is a problem in "inverse kinematics" (forward kinematics being the opposite and easier problem, of computing hand position and orientation from the arm joint angles).

The problem can mostly be solved by direct application of trigonometric equations. Two factors complicate the computation: firstly, the arm joint angles are not fully determined by the hand, and secondly, care must be taken to ensure that physiologically impossible solutions are avoided.

If the shoulder joint remains fixed in space, and the hand position and orientation are known, then one degree of freedom remains undetermined: the arm can be rotated about the line from the shoulder to the wrist joint. If the sign being performed requires the elbow to be unusually elevated, this will be notated in the HamNoSys description; otherwise, the animator must choose a natural position for the elbow. The solution we adopted requires the plane containing the upper arm and forearm to have a certain small angle to the vertical, pointing the elbows slightly outwards. The angle is increased when the arm reaches into signing space on the opposite side of the body, to prevent the upper arm passing into the torso. In addition, for such reaches, and for reaches into the "far" part of signing space, greater realism is obtained by using the sternoclavicular joint to let the shoulder move some distance towards the target point. A repertoire

of inverse kinematic constraints and formulae for arm positioning can be found in [18].

For positions around the head, care must be taken to avoid the hand penetrating the head. It should be noted that HamNoSys itself does not attempt to syntactically exclude the description of physiologically impossible signs. One can, for example, specify a hand position midway between the ears. This is not a problem; the real signs that we must animate are by definition possible to perform. This implies that a synthetic animation system for signing does not have to solve general collision problems (which are computationally expensive), but only a few special cases, such as the elbow positioning described above.

3.3 Contacts

Besides specifying a hand position as a point in space or on the body, HamNoSys can also notate contacts between parts of both hands. The BSL two-handed spelling signs are an example of these. The inverse kinematic problem of calculating the arm angles necessary to bring the two hand-parts into contact can be reduced to two one-arm problems, by determining for each arm separately the joint angles required to bring the specified part of the hand to the location in space at which the contact is required to happen.

This is an area in which motion capture has difficulty, due to the accuracy with which some contacts must be made. A contact of two fingertips, for example, may appear on playback of the raw data to pass the fingers through each other, or to miss the contact altogether. This is due to basic limitations on the accuracy of capture equipment. When using motion capture data we deal with this problem by editing the calibration data for the equipment in order to generate the correct motion.

3.4 Handshapes

HamNoSys distinguishes around 200 handshapes. Each of these must be represented ultimately as the set of rotations required in each finger and thumb joint to form the handshape. We split this task into two parts. First, we specify the required joint angles as a proportion of the maximum values, independently of the avatar. Secondly, we specify for each avatar the maximum bendings of its finger and thumb joints. For those handshapes which require a contact to be made (e.g. the "pinch" shape requiring the tips of the thumb and index finger to meet), the precise angles that will produce that contact should be calculated from the geometry of the avatar's hands.

4 Motion synthesis

We have so far discussed how to synthesise a static gesture. Most signs in BSL include motion as a semantic component, and even when a sign does not, the

signer must still move to that sign from the preceding sign, and then move to the next.

If we calculate the joint angles required for each static gesture, and then linearly interpolate over time, the effect is robotic and unnatural. Artificial smooth trajectories can be synthesised (e.g. sine curve, polynomial, Bezier, etc.), but we take a more biologically based approach and model each joint as a control system.

For the particular application of signing, the modelling problem is somewhat easier than for general body motion, in that accurate physics-based modelling is largely unnecessary. Physics plays a major role in motions of the lower body, which are primarily concerned with balancing and locomotion against gravity. This is also true of those motions of the upper body which involve exertions such as grasping or pushing. Signing only requires movement of upper body parts in space, without interaction with external objects or forces. The effect of gravity in shaping the motion is negligible, as the muscles automatically compensate.

We therefore adopt a simplified model for each joint. The distal side of the joint is represented as a virtual mass or moment of inertia. The muscles are assumed to exert a force or torque that depends on the difference between the current joint angle and the joint angle required to perform the current sign, the computation being described in detail below.

Simplifications such as these are essential if the avatar is to be animated in real time.

4.1 A brief introduction to control systems

This brief summary of control theory is based on [15]. We do not require any mathematics.

A *control system* is any arrangement designed to maintain some variable at or near a desired value, independently of other forces that may be acting on it.

In general, a control system consists of the following parts:

1. The *controlled variable*, the property which the controller is intended to control.
2. A *perception* of the current value of the controlled variable.
3. A *reference signal* specifying the desired value of the controlled variable.
4. An *error signal*, the difference between reference and perception.
5. The *output function*, which computes from the error signal (and possibly its derivatives or integrals) the *output signal*, which has some physical effect.

The effect of the output signal in a functioning control system is to bring the value of the controlled variable closer to the reference value. The designer of a real (i.e. non-virtual) control system must choose an output function which will have this effect. For the present application, our task is to choose an output function such that, when the reference angle for a joint is set to a new value, the joint angle changes to reach that value in a realistic manner.

4.2 Hinge joints

Our application of this to avatar animation places a controller in each joint. For a hinge joint such as the elbow, the controlled variable is the angle of the joint. The reference value is the angle which is required in order to produce some gesture. The perception is the current angle. The output is a virtual force acting on the distal side of the joint. The latter is modelled as a mass, whose acceleration is proportional to the force. We also assume a certain amount of damping, that is, a force on the mass proportional to and in the opposite direction to its velocity.

The mass, the force, and the damping are fictitious, and not intended as an accurate physical model; their values are tuned to provide a realistic-looking response to changes in the reference value.

Figure 2 illustrates the response of this system to a series of sudden changes in the reference value. For a sequence of static gestures, the reference value will change in this manner, taking on a new value at the time when the next gesture is to be begun. Parameters of the control system can be adjusted to give whatever speed of response is required. This can be different for different signs: Hamnosys can notate various tempos such as fast, slow, sudden stop, etc., and these will be implemented by translating them into properties of the controller.

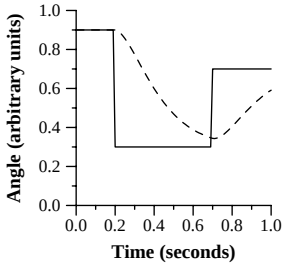

Fig. 2. Solid line: reference value. Dashed line: current value.

4.3 Higher degree joints

A turret or universal joint has two degrees of freedom. An example is the joint at the base of each finger, which can move the finger up and down, or left and right, but cannot rotate the finger about its own axis. This can be modelled as a pair of hinge joints at right angles, and the method of the preceding section applied to each hinge. This will not be accurate if the rotation of either hinge approaches a right angle, but the universal joints in the upper body all have sufficiently limited mobility that this is not a problem.

A ball and socket joint such as the shoulder has three degrees of freedom. In principle, it can be modelled by three hinge joints in series.[1] However, there is no

[1] The angles of the hinges are known as an *Euler angle* representation of the rotation.

obvious way to choose axes for the hinges that corresponds to the actual articulation of the muscles. Mathematically, there are singularities in the representation, which correspond to the physical phenomenon of "gimbal lock", which does not occur in a real shoulder joint. Aesthetically, animation of the three hinge angles separately tends to give wild and unnatural movements of the arm.

Instead, we reduce it to a single one-dimensional system. If the current rotation of the shoulder joint is q, and the required rotation is q', we determine a one-dimensional trajectory between these two points in the space of three-dimensional rotations.[2] The trajectory is calibrated by real numbers from 0 to 1, and this one-dimensional calibration used as the controlled variable. When the reference value is next changed, a new trajectory is computed and calibrated. Thus we use interpolation to define the trajectory, and a controller to determine acceleration and velocity along that trajectory.

4.4 Moving signs

Some moving signs are represented in HamNoSys as a succession of static postures. They may also be described as motion in a particular direction by a particular amount, with or without specifying a target location. These can be animated by considering them as successive static postures, in the latter case calculating a target location from the direction and size of the movement.

An explicit route from one posture to the next may be indicated (straight, curved, circular, zigzag, etc.). Since the method described above of animating the joints tends to produce straight movements of the hands, explicitly straight movements can be handled in the same way. More complex movements are handled by computing the desired trajectory of the hands, computing trajectories for the joint rotations by inverse kinematics, and using those as moving targets for the joint controllers.

HamNoSys can also specify the tempo of the motion. If the tempo is unmarked, it is performed in the "default" manner, which is what we have attempted to capture by our control model of motion. It may also be fast or slow (relative to the general speed of the signer), or modulated in various ways such as "sudden stop" or "tense" (as if performed with great effort). These concepts are easily understood from example by people learning to sign. Expressing them in terms of synthesised trajectories is a subject of our current work.

4.5 Sequences of signs

Given a sequence of signs described in HamNoSys or SiGML, and the times at which they are to be performed, we can generate a continuous signing animation by determining the joint angles required by each sign, and setting the reference values for the joint controllers accordingly at the corresponding times (or slightly in advance of those times to account for the fixed lag introduced by

[2] We use spherical linear interpolation (also known as *slerp*), a standard method of interpolating between rotations which is free of the singularities of Euler angles.[6]

the controllers). The blending of motion from each sign to the next is performed automatically, without requiring the avatar to go to the neutral position between signs.

4.6 Ambient motion

Our current avatar has the ability to blend signing animation data with "ambient" motion — small, random movements, mainly of the torso, head, and eyes — in order to make it appear more natural. This can be used even when the animation data come from motion capture. It is particularly important for synthetic animation, since we only synthesise movements of those joints which play a part in creating the sign. If the rest of the body does not move at all — and there are few signs which require any torso motion as part of their definition — the result will look unnaturally stiff. Motion capture data can be blended with synthesized data; a possible future approach is to generate these small random movements algorithmically.

5 Target avatars

These ideas were initially prototyped in VRML 97, the Virtual Reality Modelling Language [8]. This is a textual description language for 3D worlds with animation and user interaction. Associated with VRML is a standard called H-Anim [5], which specifies a standard hierarchy of joints for a humanoid skeleton, and a standard method of representing them in VRML. We have written software to convert a description of the geometry of an avatar into an H-Anim-compliant ball-and-stick model, and to convert a sequence of gestures described in SiGML into animation data for the avatar.

A ball-and-stick avatar is not suitable for production-quality signing, but there are two reasons for using one in prototyping. It can be drawn faster, and thus allows more frames per second and smoother animation, which is important for closely judging the quality of motion. More importantly, a ball-and-stick model gives the animator a clearer view of the motion. (It is quite illuminating to play motion capture data back through such an avatar. The motion appears every bit as lifelike.) A fully modelled avatar can obscure the finer details of the movement, although it is of course more realistic for the end user.

The same software can generate animations for any H-anim compliant avatar, such as that shown in Figure 3(b), and with minor changes, for any other avatar based on a similar hierarchically structured skeleton. The avatar currently in use by ViSiCAST is called Visia, and was developed by Televirtual Ltd., one of the ViSiCAST partners. Besides having a body structure with fully articulated hands, it is also capable of performing facial animation. It has a seamless surface mesh, that is, its surface deforms continuously as the underlying bone skeleton is animated. Visia is illustrated in Figure 3(c). Under consideration as a possible target is BAPs (Body Animation Parameters), a part of the MPEG-4 standard concerned with animation of humanoid figures [9], and closely connected with H-anim.

(a) Ball-and-stick avatar (b) Bob (c) Visia

Fig. 3. Signing avatars

6 Conclusions

We have described above the design and initial implementation of a method of automatic synthesis of manual signing gestures from their transcriptions in the HamNoSys/SiGML notations. We are confident that the approach described here will produce results that compare favourably with existing alternatives. Perhaps the most interesting comparison will be with the system based on motion capture, as already used in ViSiCAST. As our synthetic signing can target the same avatar model as the motion capture system, this provides the opportunity to undertake two kinds of comparison. Firstly, we will be able to do a meaningful comparison of user reaction to our synthetic signing with reaction to signing based on motion capture. In addition, as our synthesis process drives the avatar via a stream of data whose form is identical to that produced from motion-capture, we are also in a position to perform quantitative comparisons between the two methods.

7 Acknowledgements

We acknowledge funding from the EU under the Framework V IST Programme (Grant IST-1999-10500). The "Bob" avatar of Figure 3(b) is available at *http://ligwww.epfl.ch/~babski/StandardBody/*, and is due to Christian Babski and Daniel Thalmann of the Computer Graphics Lab at the Swiss Federal Institute of Technology. Body design by Mireille Clavien.

References

1. D. Connolly. *Extensible Markup Language (XML).* World Wide Web Consortium, 2000.
2. R. Elliott, J.R.W. Glauert, J.R. Kennaway, and I. Marshall. The development of language processing support for the ViSiCAST project. In *ASSETS 2000 - Proc. 4th International ACM Conference on Assistive Technologies, November 2000, Arlington, Virginia*, pages 101–108, 2000.

3. S. Gibet and T. Lebourque. High-level specification and animation of communicative gestures. *J. Visual Languages and Computing*, 12:657–687, 2001. On-line at `http://www.idealibrary.com`.
4. A. Glassner. Introduction to animation. In *SIGGRAPH '2000 Course Notes*. Assoc. Comp. Mach., 2000.
5. Humanoid Animation Working Group. *Specification for a Standard Humanoid, version 1.1*. 1999. `http://h-anim.org/Specifications/H-Anim1.1/`.
6. A. J. Hanson. Visualizing quaternions. In *SIGGRAPH '2000 Course Notes*. Assoc. Comp. Mach., 2000.
7. J. Hodgins and Z. Popović. Animating humans by combining simulation and motion capture. In *SIGGRAPH '2000 Course Notes*. Assoc. Comp. Mach., 2000.
8. The VRML Consortium Incorporated. *The Virtual Reality Modeling Language: International Standard ISO/IEC 14772-1:1997*. 1997. `http://www.web3d.org/Specifications/VRML97/`.
9. R. Koenen. *Overview of the MPEG-4 Standard*. ISO/IEC JTC1/SC29/WG11 N2725, 1999. `http://www.cselt.it/mpeg/standards/mpeg-4/mpeg-4.htm`.
10. T. Lebourque and S. Gibet. A complete system for the specification and the generation of sign language gestures. In *Gesture-based Communication in Human-Computer Interaction*, Lecture Notes in Artificial Intelligence vol.1739. Springer, 1999.
11. M. Lincoln, S.J. Cox, and M. Nakisa. The development and evaluation of a speech to sign translation system to assist transactions. In *Int. Journal of Human-computer Studies*, 2001. In preparation.
12. I. Marshall, F. Pezeshkpour, J.A. Bangham, M. Wells, and R. Hughes. On the real time elision of text. In *RIFRA 98 - Proc. Int. Workshop on Extraction, Filtering and Automatic Summarization, Tunisia*. CNRS, November 1998.
13. F. Pezeshkpour, I. Marshall, R. Elliott, and J. A. Bangham. Development of a legible deaf-signing virtual human. In *Proc. IEEE Conf. on Multi-Media, Florence*, volume 1, pages 333–338, 1999.
14. Z. Popović and A. Witkin. Physically based motion transformation. In *Proc. SIGGRAPH '99*, pages 11–20. Assoc. Comp. Mach., 1999.
15. W. T. Powers. *Behavior: The Control of Perception*. Aldine de Gruyter, 1973.
16. 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 3.0 is documented on the Web at `http://www.sign-lang.uni-hamburg.de/Projects/HamNoSys.html`.
17. SigningAvatar. `http://www.signingavatar.com`.
18. D. Tolani and N. I. Badler. Real-time inverse kinematics of the human arm. *Presence*, 5(4):393–401, 1996.
19. M. Wells, F. Pezeshkpour, I. Marshall, M. Tutt, and J. A. Bangham. Simon: an innovative approach to signing on television. In *Proc. Int. Broadcasting Convention*, 1999.
20. A. Witkin and Z. Popović. Motion warping. In *Proc. SIGGRAPH '95*, 1995.