Requirements For A Signing Avatar

Vince Jennings, Ralph Elliott, Richard Kennaway, John Glauert
School of Computing Sciences, University of East Anglia, Norwich NR4 7TJ, UK.
{V.Jennings, R.Elliott, R. Kennaway, J.Glauert}@uea.ac.uk

Abstract

We present the technical specification for an avatar that is compliant with Animgen, the synthetic signing engine used at the University of East Anglia for generating deaf signing animations. The specification will include both the basic definition required for any standard animating avatar, and the additional parameters that Animgen requires to generate signing. Avatars compatible with Animgen are created using the ARPToolkit, an application developed at UEA that has a plug-in architecture for tools that are used for rigging an avatar mesh for animation. The toolkit also generates the additional data needed by Animgen for each avatar.

1. Introduction

For any avatar to be animated there is a standard set of requirements that must be met in the avatar file, which must include a mesh, a skeleton, a texture, and, if facial animation is also required, a set of morph targets. The mesh represents the visible shape of the avatar, and, together with the texture, defines the avatar's appearance. The skeleton, a mathematical construct in software which is not visible, has its bones linked to the vertices in the mesh, so that changing the rotation of any bone in the skeleton results in the movement of the mesh vertices linked to it. The morph targets, also referred to as blend shapes, each represent a deformation of the static mesh to both the area around the mouth and jaw for speech synchronisation, and to the cheeks, eyelids, eyebrows, and forehead for facial expressions.

JASigning is a synthetic animation system for deaf signing, written in Java, that has been developed at UEA, taking as input avatar-independent Gestural SiGML (Signing Gesture Markup Language) (Elliott *et al*, 2004, 2007) and producing as output motion data for any avatar. SiGML is an XML form of HamNoSys (Hamburg Notation System) (Prillwitz *et al*, 1989; Hanke, 2004) that is used by Animgen (Kennaway, Glauert, Zwitserlood, 2007) to generate signing animation.To achieve this JASigning requires additional information that cannot be obtained from the standard information above, and must be provided in separate files. To demonstrate the need for the extra data an example would be where a sign requires that the tip of the index finger on the right hand touches the tip of the nose. These locations cannot be obtained from the standard specification, but are provided in the extra files.

The ARPToolkit [ARP] was developed at UEA to provide a unified application for creating avatars that not only met the standard requirements for animation but also have the additional data needed for deaf signing. Additionally, the tools developed in the toolkit were designed to automate some of the tasks of avatar rigging, and to provide simple interfaces for some of the more complex tasks, such as morph target creation, making the toolkit accessible to users who lack the technical skills needed for the majority of commercial software that would otherwise have to be employed.

For the purposes of the JASigning software, each ARP signing avatar is effectively defined by a set of four avatar definition files.
The first of these contains binary data, the other three are XML:

- Main Avatar Definition
- ASD, Avatar Standard Description
- Animgen Configuration Data
- Nonmanuals

2. Main avatar definition file

The main avatar definition file, **avatardef.arp**, contains only the data needed for an avatar to perfom standard animations, and has none of the extra data that Animgen needs for the generation of deaf signing. Its major components are:

2.1 Vertex List

A list of vertices that represents the mesh defining the shape of the avatar, with texture coordinates and vertex normals for each. Each vertex and normal is defined relative to its linked bone(s), with the bone initially aligned along the X axis. Meshes for the eyes, teeth, and tongue must be present, but not contiguous with the rest of the mesh. The hair and ears can also be modelled separately from the main mesh, but all other parts of the avatar mesh must be a single contiguous mesh. To allow realtime animation (at 25fps or more) on an average specification machine the vertex count of the mesh should not exceed 10,000.

2.2 Texture Map

The texture map, which may optionally be held in a separate file or embedded in the file in a standard format such as PNG, defines the appearance of the avatar. All texture should be contained in a single file. For good quality a minimum size of 1024 X 1024 pixels is suggested.

2.3 Skeleton

The skeleton structure fits within the mesh, and includes bones for animating the eyes, which must be child nodes of the head bone. It must include all bone names used by Animgen (see asd.xml below), but can include additional bones (e.g. metacarpals), although these will be ignored by Animgen. Bone names are all 4 character (4cc) codes.
The bone hierarchy, as specified in the asd.xml file, must be adhered to, but is compatible with other standard hierarchies such as H-Anim and BVH. Translations and rotations for each bone are in the parent's coordinate space, with the transform for each bone being multiplied by its parent

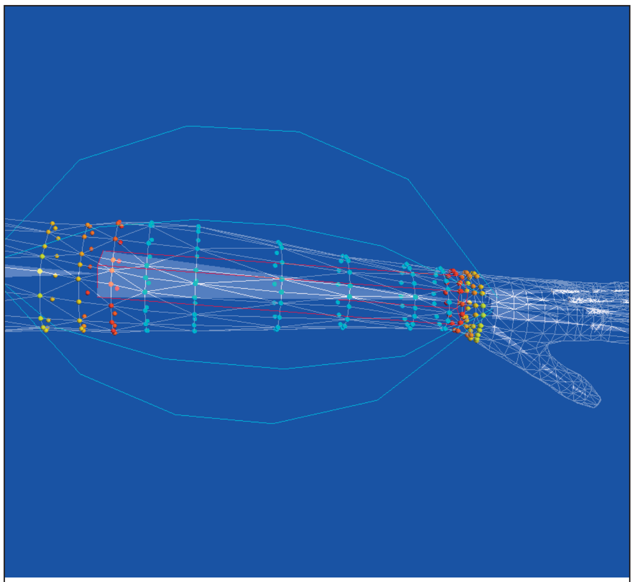
Figure 1. Mesh to skeleton attachment

bone's transform. Zero length bones may be included on all leaf nodes in the bone hierarchy if desired, again with unique 4 character names. These are sometimes required in other animation applications, and will also be ignored by Animgen. The ARPToolkit has tools for creating skeletons and for adjusting them to fit the avatar mesh.

2.4 Mesh-to-Skeleton Attachment Data

This data is a list of links between vertices in the mesh and the bone(s) that will animate them, with a weight for the influence of each bone. A maximum of 4 links per vertex is permitted, with a preferred maximum of 3. The weights of all links to a vertex must sum to 1.0. This follows standard industry practice for this type of data as more than 4 links to a vertex makes weight calculations very complex. Vertex weights are calculated in the ARPToolkit during construction of the skeleton, and their weights subsequently altered to produce good deformation at the joints by ‘painting’ the weights for each vertex using the mouse.

Figure 1 shows the linkage between the vertices of the arm and the bone. The envelope determines which vertices are assigned to the bone, and the colour of each vertex shows the weight (between 0 and 1) that this bone applies to the transformation for this vertex. These are typically 1 across the centre of the bone, reducing at the joints where adjacent bones also apply their weights.

2.5 Morph Targets

The list of morph targets contained in a standard non-signing avatar file would consist of the visemes necessary for lip synchronisation to speech, and for facial expressions.

Each morph target represents a deformation of the mesh to produce facial animation. A morph target includes a list of indices for vertices in the mesh, a deformation vector for the full displacement of the vertex (1.0), and a normal for the fully displaced vertex. Negative amounts for morphs are not supported, e.g. for moving eyebrows down instead of up, so morphs for all movement directions must be provided.

For a deaf signing avatar additional morph targets are

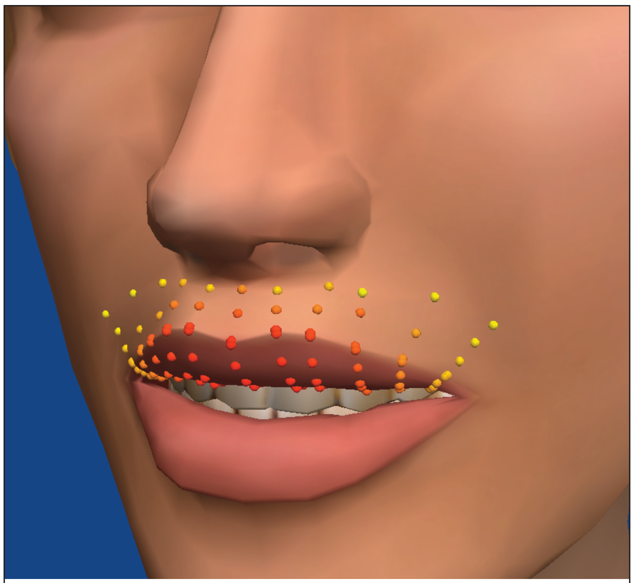
Figure 2. Upper lip morph target

needed, particularly for the cheeks and the tongue, which are used for a wide range of facial gestures.

Morph names are arbitrary, and can be matched to those used by Animgen, the synthetic signing engine used in JASigning, by editing the avatar's nonmanuals.xml file.

Morph targets are created in the ARPToolkit, where a library of primitive morphs are first defined for movements of the jaw, lips, tongue, cheeks, nose, eyebrows, and eyelids. Selections of these primitives are then combined to produce the morphs for phonemes and signing mouth gestures and uploaded into the avatar.

Figure 2 shows the vertex selection for the morph primitive for the upper lip, with the colour coding indicating the weighting of the transform that moves the vertices vertically, with red indicating a heavier weight falling off to yellow for a low weight.

3. The ASD File Format

The purpose of the Avatar Standard Description (ASD) file, **asd.xml**, is to define all the avatar-related data needed by Animgen. It defines the skeleton, with the bone names and hierarchy used by Animgen, in a reference pose that enables Animgen to establish the correct rotation axis for elbows and thumbs.

The ASD file also defines a set of approximately 380 feature points on the surface of the mesh of the upper body, arms, hands, and head, which Animgen may use as reference points when it needs to determine locations in signing space. On the arms and hands these points are defined on 2 axes at each joint and again midway between each joint. Each of these points is assigned a unique identity code recognised by Animgen.

To simplify the task of defining the feature points, which would otherwise have to be defined individually by hand, tools have been developed in the ARPToolkit to carry out ray tracing from the bones of the skeleton to intersect with the mesh at the desired locations. This process is automatic for the upper body, arms, and hands, with a secondary as-

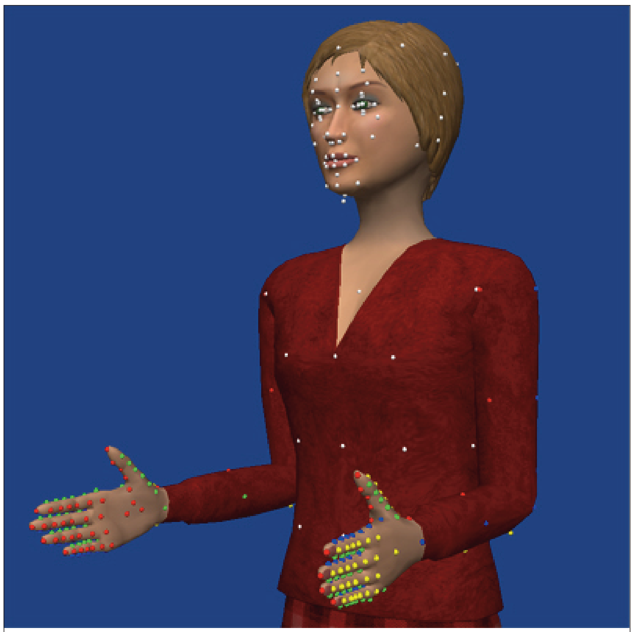
Figure 3. Feature points with reference pose

sisted manual process for the head locations.
In the section of an file asd.xml shown below, the skeleton hierarchy is shown by the joint relationship “ROOT”, “SPI1”, “SPI2”, etc, with feature points being listed under their “owner” bones. Positions of each point are relative to their “owner”. For example T-LS is “torso front at left shoulder”, and its “owner” is “ROOT”.

```
<?xml version="1.0" standalone="yes"?>
<avatarStaticData version="1.0">
  <avatar name="arp-anna" version="1.0">
    <skeleton scale="0.04445039">
      <joint name="ROOT" position="0.000000 0.000000 0.000000" rota-
tion="0.000000 0.000000 0.707330 0.706883">
        <feature name="T-LS" position="9.430 -4.333 1.636" />
        <feature name="T-CS" position="9.433 0.000 1.932" />
        <feature name="T-RS" position="9.436 4.333 1.638" />
        <feature name="T-LC" position="7.265 -2.168 3.171" />
        <feature name="T-CC" position="7.266 0.000 3.284" />
        <feature name="T-RC" position="7.268 2.165 3.175" />
        <feature name="T-LA" position="4.269 -2.166 2.485" />
          ...
        <joint name="SPI1" .... >
          <joint name="SPI2" .... >
            <joint name="SPI3" .... >
              <joint name="LCLR" .... >
                ....
              </joint>
            </joint>
          </joint>
        </joint>
          ....
      </joint>
    </skeleton>
  </avatar>
</avatarStaticData>
```

4. Animgen Configuration Data File Format

The **config.xml** files contain the settings for controlling many of the aspects of the synthetic signing generated by Animgen, and is loaded by Animgen when processing a SiGML file to produce animation. The file defines timings, signing space, constraints, trajectories, hand shapes, constants, repetitions, and rest poses.

For example, the following code defines a handshape for a fist with the index finger extended.

```
<handshapes>
  <finger2
      specialbends="0000"
      ordinarybends="4440"
      extendedfingers="2"
      class="fist"
  />
</handshapes>
```

Each finger bending consists of 4 numbers, representing respectively the bends at the first, second, and third joints, and the splay angle. For each of these, 0 represents the value when the joint is not bent and 4 is its maximum bending. Each handshape has two different finger bendings:"specialbends" is the bending of the extended fingers (e.g. the index finger for the finger2 handshape) and "ordinarybends" for the other fingers. The thumb is not described here."extendedfingers" is the set of extended fingers (which includes the thumb for some handshapes).
The ARPToolkit provides facilities to interactively set values for hand shapes in the config.xml file, reloading the modified file and displaying the changed handshape in real time.
Signing space for the avatar is defined in terms of the avatar’s dimensions such as arm lengths and feature points on the torso.

```
<signingspace
            horiz_spacing = "0.8"
            vert_spacing = "0.25"
            inout_spacing = "0.15"
            signspacesitesize = "1.2"
            fan = "0.6" curve = "1"
            nearbelly = "0.10"
            torsositesize = "0.10"
            neckheight = "0.02"
/>
```

Before processing a SiGML file Animgen will first load a config.xml file from a directory common to all avatars. It will then load an avatar specific config.xml file that may contain alternative settings that will override those in the common file. A typical example of this would be settings for hand shapes, where variations in bone sizes between skeletons may affect hand shapes.

5. Nonmanuals File Format

The purpose of the **nonmanuals.xml** file is to define how each SiGML/HNS nonmanual feature is implemented using the avatar’s morph targets. It maps the standard names of nonmanuals used in SiGML/HNS to the names of the morph targets in the main avatar definition file, or to parallel and sequential sets of these morph targets. The mapping also includes durations and trajectories (timings) for these nonmanuals. The file also includes mappings from Sampa.

5.1 Examples.

A mapping from a SiGML name for a mouth gesture to an avatar's morph names:

```
<mouth_gesture sigmlName="D01">
   <parmorph>
      <morph name="eee" amount="0.6" timing="x m t s m l x"/>
      <morph name="ulpr" amount="0.2" timing="x m t s m l x"/>
      <morph name="ulpl" amount="0.2" timing="x m t s m l x"/>
   </parmorph>
</mouth_gesture>
```

For the SiGML mouth gesture D01 the gesture comprises the morphs "eee" with an amount of 0.6, "ulpr" with an amount of 0.2, and "ulpl" with an amount of 0.2. Enclosing all three in the <parmorph> </parmorph>element indicates that these should be combined in parallel. All parallel combinations of morphs must have the same timing, with the optional "x", in this case, at each end indicating that this gesture should be adjusted to last the same length of time as the manual gesture that it accompanies.

A mapping from Sampa to an avatar's morph names:

```
<sampa phonemes="O_I:">
   <morph name="ooo" timing="m t - m t"/>
   <morph name="eee" timing="m t m m t"/>
</sampa>
```

Here "O_I:" represents a diphthong which is mapped to two morphs, "ooo" and "eee", performed in sequence, each with a different timing.

Non-facial nonmanuals. These are animations of the head, spine, and shoulders that are expressed as "pseudomorphs" in SiGML, but are processed by Animgen into bone animations.

```
<head_movement sigmlName="NO">
   <morph name="HTLF" amount="0.03" timing="m t - f l"/>
   <morph name="HTLF" amount="-0.03" timing="m t - f l"/>
   <morph name="HTLF" amount="0.03" timing="m t - f l"/>
   <morph name="HTLF" amount="-0.03" timing="m t - f l"/>
   <morph name="HTLF" amount="0.03" timing="m t - f l"/>
   <morph name="HTLF" amount="-0.03" timing="m t - f l"/>
</head_movement>
```

This produces a set of sequential bone movements of the head from left to right - "NO".

5.2 Durations and Trajectories

The timing attribute for each morph is a sequence of up to 7 tokens, each with codes that map to constants defined in the config.xml file, with the following purpose:

1) Whether the morph is anchored to the start of the interval during which it is played.
2) The attack time (the time spent ramping up from zero to the full amount).
3) The attack trajectory (the manner in which it approaches the full amount).
4) The sustain time (the time spent holding the morph at its full amount).
5) The release time (the time spent ramping down to zero).
6) The release trajectory (the manner in which it ramps down to zero).
7) Whether the morph is anchored to the end of the interval during which it is played.

The first and last token is either 'x' (anchored) or 'e' (elastic). These tokens can be omitted, and default to 'x' and 'e' respectively. Each time is either a real number of seconds, or one of the following tokens:

f	fast
m	medium speed
s	slow
-	zero

Each trajectory is one of the tokens "t" (targetted) or "l" (lax). The targetted trajectory makes a greater acceleration and deceleration towards its endpoint. Typically one would use "t" for everything except the release trajectory of the last morph.

6. Conclusion

The requirements to enable an avatar to perform deaf signing in the UEA JASigning software are essentially in addition to the standard specification for any avatar that can be animated. The only additions to the standard specification are the extra morph targets specific to deaf signing. The representation of the standard data can be converted to the format used in the avatardef.arp file already described. The other additional data required for signing is held in the asd.xml, config.xml, and nonmanuals.xml files. We believe these additions to the requirements for a standard virtual human character definition will be necessary in any system that synthesises authentic animated sign language.

7. Acknowledgements

We acknowledge with gratitude that the work described here has been partially funded under the European Union's 7th FrameworkProgramme, through the Dicta-Sign project (grant 231135).

References

Elliott, R., Glauert, J.R.W., Jennings, V., and Kennaway, J.R., "An Overview of the SiGML Notation and SiGML-Signing Software System", In Fourth International Conference on Language Resources and Evaluation, LREC 2004, Edited by Streiter, O. and Vettori, C., Lisbon, Portugal, pp. 98-104, 2004.

Elliott, R., Glauert, J.R.W., Kennaway, J.R., Marshall, I., and Safar, E., "Linguistic modelling and language processing technologies for avatar-based sign language presentation", Universal Access in the Information Society, vol. 6, no. 4, pp. 375-391, 2007.

Prillwitz, S., Leven, R., Zienert, H., Hanke, T., Henning, J., et al. "Hamburg Notation System for Sign Languages—An Introductory Guide", International Studies on Sign Language and the Communication of the Deaf (5). Institute of German Sign Language and Communication of the Deaf, University of Hamburg, Hamburg, 1989.

Kennaway, J.R., Glauert, J.R.W., and Zwitserlood, I., "Providing Signed Content on the Internet by Synthesized Animation", ACM Transactions on Computer Human Interaction, vol. 14, 3, no. 15, pp. 1-29, 2007.

Hanke, T., "HamNoSys representing sign language data in language resources and language processing contexts", In Fourth International Conference on Language Resources and Evaluation, LREC 2004, Edited by Streiter, O. and Vettori, C., Lisbon, Portugal, pp. 1–6, 2004.

[ARP] http://vh.cmp.uea.ac.uk/index.php/ARP