# Exchangeable Humanoid Animation Using H-Anim and Motion Capture Data

Myeong Won Lee[1], Chul Hee Jung[1], Mingeun Lee[1], and Don Brutzman[2],

[1] IT College, The University of Suwon, Hwaseong,
Gyeonggi-Do, 445-743 South Korea
mwlee@suwon.ac.kr, whiskerfe@hanmail.net, adora98@empal.com
[2] Naval Postgraduate School, MOVES Institute
Monterey, CA 93943-5000 USA
brutzman@nps.edu

**Abstract.** We describe a method for generating H-Anim character animation using general graphics tools and motion capture data. An H-Anim character model can be designed with a general graphics tool using a procedure that defines the skeletal hierarchy of a human character. The character model is then provided with motion capture data. H Anim character animation is generated and displayed using a general X3D browser. The H-Anim character animation is obtained by converting captured motion data into X3D interpolator nodes. We have defined a conversion algorithm that provides H-Anim characters with motion by applying any motion capture data to the H-Anim character model. As well as an H Anim viewer, we have developed an H-Anim editor that is used to display and edit the hierarchy of an H-Anim character model and also generate its animation interactively using motion capture data.

**Keywords:** Humanoid Animation (H-Anim), H-Anim character model, motion data definition, H-Anim character animation, X3D interpolators, H-Anim editor.

## 1 Introduction

ISO/IEC 19774 H-Anim (Humanoid Animation) [1] is an international standard that defines a data structure for representing and exchanging human models and animation through networks. Corresponding H-Anim bindings for Extensible 3D (X3D) Graphics are found in [2] which can be expressed in multiple language encodings and compressed binary syntax. The H-Anim specification also includes definition of keyframe and kinematic animation using X3D interpolators. Other animation methods using motion capture, and dynamic and other algorithmic animations, are also defined by converting all animation parameters into X3D interpolators.

Several research projects related to H-Anim have been done since its inception. Tools, such as H-Animator and Web3D Toolbox [3][4], have been developed to generate H-Anim character animation. Other research describes ways to generate H-Anim animation using keyframe, inverse kinematics, or motion capture data[5][6][7]. All this research was focused on the generation of animation using H-Anim models.

Our work is intended to include motion definition in the H-Anim specification for the purpose of exchanging humanoid animation between different applications and/or between different platforms when generating animation using H-Anim character models and motion capture data. We presume the motion capture data is obtained independently of the H-Anim figure model. This means, therefore, that the motion captured model is completely separate from the H-Anim figure model. We consider the reusability of motion capture data so that any motion capture data can be applied to any H-Anim figure model. This paper describes the procedure and exchange format for generating H-Anim animation using motion capture data for an H-Anim character designed using a general graphics tool.

The H-Anim character model and motion capture animation are stored in an H-Anim X3D file, and then a general X3D browser displays the motion capture animation of the H-Anim model. Two converters are required for this procedure, one to convert the 3D character model (in this case, a VRML .wrl model) designed using a general graphics tool to an H-Anim X3D model, the other to convert from motion capture data to H-Anim X3D interpolators for motion definition. We have developed an H-Anim viewer and an H-Anim editor that includes the converters.

## 2 H-Anim Character Animation Process

This section describes an overall procedure for generating H Anim character animation. It defines motion for a character model designed using any graphics tool or generated from a scanning device. Motion data generated using an animation algorithm or obtained from capture devices can be used to define motion and to generate human animation sequences that are exchangeable through networks of heterogeneous computer systems. An H-Anim character model can be designed with any general graphics tool according to the hierarchical structure defined in the ISO/IEC 19774 H-Anim or the ISO/IEC 19775-1 X3D specifications. The character model is designed independently of motion data obtained or generated using specific algorithms. Motion parameters necessary for animating the H Anim model can be defined. Several kinds of motion data can be applied to generate animation sequences for the character model; conversely, one kind of motion data can be applied to multiple character models.

Fig. 1 shows the procedure for generating and storing H-Anim character animation sequences using motion capture data, keyframe animation, and algorithmic animation methods including kinematics and dynamics. They can be transferred and exchanged through different computing and application environments. The motion of an H-Anim character model designed using one graphics tool can be applied as the motion for another character designed using a different graphics tool.

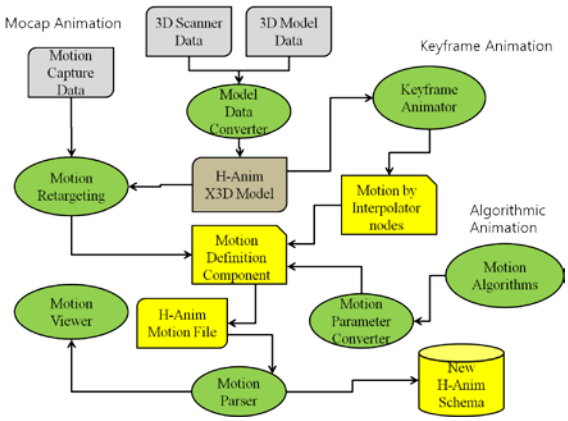


**Fig. 1.** H-Anim character animation

## 3 H-Anim Character Modeling

In order to define motion in the H-Anim specification, we need a process to prepare the modeling data for an H-Anim human figure. Generally, character animation data have special characteristics that combine segments and joints according to their motion generating roles. Each segment has motion dependency on its related joints and should be uniquely identified. Therefore, it is necessary to follow certain guidelines when preparing modeling data so that each segment and joint can be identified and its related motion parameters can be retrieved consistently.

To uniquely identify each segment, we use the H-Anim specifications precisely defined naming scheme. The other modeling requirement is that a character model must be created by combining separately designed segments. Using any graphics tool, each segment must be designed separately, each with a predetermined name. Then, the separate segments are integrated into a combined character model. Lastly in the modeling process, the combined model is stored in a VRML or an X3D file in the same way as when generating 3D models for Web3D applications. Many commercial graphics tools provide this functionality.

At this last step of the procedure, it is important to define the center of each joint, also stored in the VRML or X3D file. VRML or X3D data can be converted to X3D H-Anim format with the center of each joint consistently maintained.

## 4 Motion Definition of an H-Anim Character Model

There are various motion generating algorithms for 3D character models such as keyframe, interpolation, kinematics, and dynamics for human figures. These are optionally selected based on the application. Details about the algorithms themselves need not be provided in the specification, but their interfaces should be included.

Necessary parameters should be provided so that the H Anim character can be animated using the specified algorithms. This section describes which algorithms can be selected and the kinds of parameters that must be defined for each.

When defining the motion of a character using motion generating algorithms or motion capture data, motion parameters and motion parameter values go hand in hand. In addition to the motion parameters themselves, motion parameter values must be able to be transferred between various applications. Therefore, motion parameters and motion parameter values should be defined in the motion definition part of the H-Anim specification. This section describes how to define a unique interface for motion parameter values from various motion algorithms and motion capture data. We can use X3D interpolators when defining keyframe animation. Animation sequences generated using other algorithms and motion capture data can also be defined using X3D interpolators in H-Anim.

# 5 Conversion of Motion Capture Data into X3D Interpolators

This section describes an algorithm to take motion capture arrays and create the X3D interpolators, PositionInterpolator and OrientationInterpolator[8][9]. Given its broad and unconstrained availability, we presume to use BVH motion data for motion capture[10]. The following sections describe the procedure for converting BVH data into X3D interpolators, step by step. Further details can be found in [11].

## 5.1 Interpolator Key Array

In X3D, an interpolator key is typically a fraction from 0 to 1.0. Take the total time interval of the BVH file, e.g. 8.01667 seconds, divided by the number of intervals, e.g. 481. The resulting key array would consist of fractions incrementing by (1.0 / 481) = 0.016667. The result is:

```
key = "0 1/481    2/481      3/481     ...etc... 1"
```

Note that the X3D Interpolator keyValue array will be the same length as the rotation array. We must convert each triple of Euler angles to a corresponding SFRotation. The result will be, e.g:

```
keyValue="0 1 0 0.0, 0 1 0 0.111, 0 1 0 0.123 ...
etc ..." etc.
```

### 5.2 TimeSensor key Array

TimeSensor fraction_changed output is ROUTED to OrientationInterpolator set_fraction. Result:

```
<TimeSensor DEF="AnimationClock" cycleTime="8.01667" />
<OrientationInterpolator DEF="X3DMotionArray"
  key = "0 0.002079  0.004158   0.006237   ...etc... 1"
  keyValue="0 1 0 0.0, 0 1 0 0.111, 0 1 0 0.123 ... etc
..." />
 <ROUTE fromNode="AnimationClock"
fromField="fraction_changed" toNode="X3dMotionArray"
fromField="set_fraction"    />
```

### 5.3 Computing keyValue Arrays

This section examines how BVH values are excerpted and converted into various keyValue arrays. In other words, we take two frames of a BVH file to produce PositionInterpolator and OrientationInterpolator nodes. Below is an example of a unmodified, unmarked BVH frames:

```
BVH Motion
  Frame 0:
  1.662 31.427 60.304   -1.249 -4.859 -3.582   4.463 1.354 0.075     -13.732 3.052 3.999     95.677
1.705 -1.512     5.541 -3.491 0.339     1.259 -3.022 1.790     6.765 2.405 -4.446     -91.027 -7.187
4.910     -3.633 0.867 0.043     -2.879 0.120 -5.688     -1.132 -1.858 0.809     -2.969 -8.472 1.461     -
1.304 3.919 -2.045     1.054 9.006 -0.191     2.695 -1.341 -0.615     0.361 4.452 4.756     0.484 8.095
0.193     -6.340 -0.815 1.224
```

Next, we use excerpts from the above BVH frame values to show example conversions into X3D PositionInterpolator and OrientationInterpolator. The keyValue array for the PositionInterpolator is obtained from the first three values at all frames. The keyValue array for each OrientationInterpolator per joint are obtained by Euler-to-SFRotation conversion from the corresponding three values at all frames for each corresponding joint.

Color codes: translation values (red), joint 1 rotations (orange), joint 2 rotations (green), and joint 3 rotations (blue)

```
BVH Motion
  Frame 0:
  1.662 31.427 60.304   -1.249 -4.859 -3.582   4.463 1.354 0.075     -13.732 3.052 3.999 ...
  <PositionInterpolator DEF='HumanoidRootTransInterp' key='0 1/481 2/481 3/481 ...' keyValue='1.662
31.427 60.304,   1.659 31.427 60.307, ...' />
  <OrientationInterpolator DEF='HumanoidRootRotInter' key='0 1/481 2/481 3/481 ...' keyValue='f(-
1.249 -4.859 -3.582), f(-1.268 -4.835 -3.588), ...'/>
  <OrientationInterpolator DEF='sacroiliacRotInterpo' key='0 1/481 2/481 3/481 ...' keyValue='f(4.463
```

```
1.354 0.075), f(4.487 1.352 0.080), ... '/>
  <OrientationInterpolator DEF='l_hipRotInterpolator' key='0 1/481 2/481 3/481 ...' keyValue='f(-13.732
3.052 3.999), f(-13.802 3.059 3.999), ...'/>
```

where f(phi,theta,psi) means euler-to-SFRotation conversion. Using other expressions, here is a pattern for the conversion mapping from BVH to X3D keyValue arrays. The meaning of these BVH value labels are as follows: "0-tz" means "frame time 0, translation along z axis". "1-rotz2" means "frame time 1, single Euler-rotation angle about z axis, for joint 2"

### 5.4 Euler to SFRotation Conversion Algorithm

X3D viewers only use axis-angle SFRotation values, not Euler angles. For example, in an X3D OrientationInterpolator, if (1 0 0 0.5) means x-axis rotation with 0.5 radian, then (1 1 1 0.5) means an arbitrary axis from (0 0 0) to (1 1 1) with 0.5 radian rotation about that axis, using the right-hand rule for direction. Define the function f(phi,theta,psi) for converting a single BVH Euler-angle triplet into a single X3D SFRotation axis-angle value.

The transformation from BVH Euler angles to SFRotation has considerations as follows: Simply put, we must transform a triple value (x-rot y-rot z-rot angles in degrees) into a quadruple value (x y z w angle in radians). Therefore, our transformation algorithm is as follows:

1) Choose the largest rotation angle amongst the three Euler angles.

2) Define the axis value as 1 for the rotation axis with the largest rotation angle for the quadruple.

3) Convert the largest angle in degrees into an angle in radians. This will be w for the quadruple.

4) The remaining axis values are calculated from the proportion of the rotation angle of the largest rotation angle.

For example, consider Euler angles (in degrees) x=90, y=45, z=-180. Based on the algorithm, SFRotation is x=0.5, y=0.25, z=-1, w=3.14. For another example, consider Euler angles x=30, y=30, z=30. Based on the algorithm, SFRotation is x=1, y=1, z=1, w=3.14/6. The following is the implementation code of the algorithm:

```
void HAnim::ConvertEulerToSFRotation(float& x, float& y, float& z, float& w){
        #define piover180 0.01745329252f
        float fAbsoluteX = abs(x);
        float fAbsoluteY = abs(y);
        float fAbsoluteZ = abs(z);
    float fMaxAngle = 0;
        if(fAbsoluteX > fMaxAngle) fMaxAngle = fAbsoluteX;
        if(fAbsoluteY > fMaxAngle) fMaxAngle = fAbsoluteY;
        if(fAbsoluteZ > fMaxAngle) fMaxAngle = fAbsoluteZ;
        float fRatio = 0.0f;
```

```
        if(fMaxAngle > 0.0001f)    fRatio = 1.0f / fMaxAngle;
        x = x * fRatio;
        y = y * fRatio;
        z = z * fRatio;
        w = fMaxAngle * piover180;
        w = abs(w);
}
```

## 6 Implementation of an H-Anim Viewer and Editor

We have implemented both an H-Anim viewer and an H-Anim editor. The viewer reads an X3D character converted from a WRL character designed using a general graphics tool. Using the viewer, the X3D character is converted into an H-Anim X3D character with a hierarchy representing relationships between joints and segments. Then, a motion capture file, such as BVH, is imported. Finally, the viewer retargets the motion at each joint of the motion capture character to the motion of the H-Anim X3D character using the motion capture conversion described above. Fig. 2 shows the H-Anim viewer displaying a sample motion capture animation of an H-Anim character model.

The editor provides functionality for interactive changeable skeletal structure for an H-Anim character model. This means that the hierarchical structure of a humanoid model can be modified interactively. Motion capture data can also be updated in the editor interactively. An H-Anim X3D character model and its motion capture animation are displayed and controlled in the editor. The editor can also be used to generate an H-Anim X3D animation file from an H-Anim model and a motion capture file. Fig. 3 shows the H-Anim editor. On the left is an edit window for the hierarchical structure of the H-Anim model, and on the right is an edit window for motion capture data applied to the H Anim model. The H-Anim model is displayed and synchronized according to the information in the edit windows.

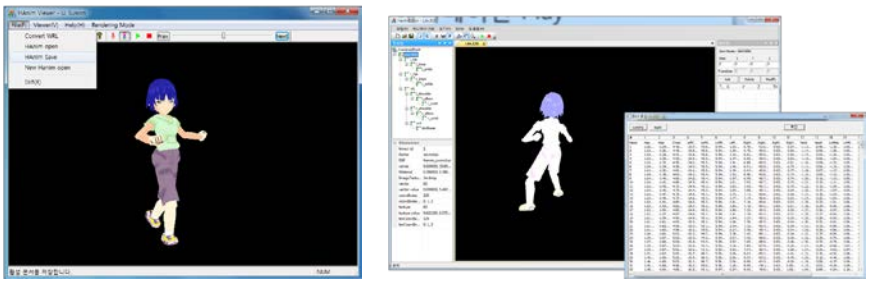

**Fig. 2.** H-Anim viewer

**Fig. 3.** H-Anim editor

## 7 H-Anim X3D Authoring Tools Strategy

Using the above methods, H-Anim X3D humanoid animation is obtained using motion capture data. X3D browsers display this motion capture animation for an H-Anim X3D model. This section describes the strategy for generating H-Anim X3D animation sequences using our H-Anim authoring tools. Fig. 4 shows the procedure for generating motion capture animation for an H-Anim X3D model. It illustrates that H-Anim animation definition can be exchanged between different animation tools. An H-Anim X3D model and motion capture data is imported into the H-Anim editor and converted into X3D interpolators. Then, motion capture animation of the H-Anim X3D model can be displayed using the X3D browser because the captured motion is applied to the H-Anim X3D model with the interpolators. Current work includes further verification of algorithm correctness and completeness by creating an independent implementation in the open-source X3D Edit authoring tool [12].

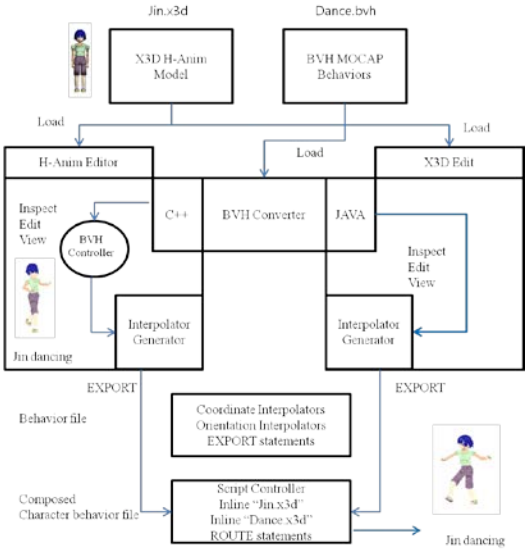

**Fig. 4.** H-Anim authoring strategy for composing X3D H-Anim models with mocap-derived animation behaviors.

## 8 Considerations about Possible Revisions to H-Anim Specification

As described in the previous section, motion capture data for an H-Anim character model can be converted into X3D interpolators which are used for generating keyframe animation. In order to generate the motion capture animation of an H-Anim model, a converter must be provided. When converting the motion capture data into X3D interpolators, large sets of keys and duplicate keywords are generated. These can cause inefficiency when using motion capture data for H-Anim character animation.

It is important to point out that motion capture animation is not keyframe animation. This is because motion capture data already has all parameter values at all frames. X3D interpolators are appropriate for keyframe animation that requires in-between parameter values, that is, between key values. Therefore, motion capture animation has no need for the concept of keys and key values. As a result, interpolators are unnecessary when an H Anim viewer is able to display all the motion capture frames consecutively after matching the joints of the motion capture data to the joints of the H-Anim model.

In this section, another representation of motion capture animation of H-Anim X3D models is considered. It is, simply, to add motion capture data to H-Anim X3D model data after matching joints between a motion capture and an H-Anim character. Although this matching is a prerequisite function for an H-Anim viewer, it is easily obtained since motion capture data generally has a similar joint hierarchical structure to H-Anim. The new representation includes the definition of a joint node and a motion node. The joint node is almost the same as before except two fields are added: number of channels, and channel parameter values (Fig. 5). The motion node is newly defined by the number of frames, frame time, and transformation parameter values (Fig. 6).

```
Interface Joint {
  // the same as the existing joint node
     float[3] bboxCenter 0 0 0
     float[3] bboxSize -1 -1 -1
     float[3] center 0 0 0
     sequence<Object> children []
     sequence<Object> displacers []
     sequence<float[3]> llimit []
     float[4] limitOrientation 0 0 1 0
     string name ""
     float[4] rotation 0 0 1 0
     float[3] scale 1 1 1
     float[4] scaleOrientation 0 0 1 0
     float[3] translation 0 0 0
     sequence<float[3]> ulimit []
  // define additional fields
     int[2] ChannelsNumber
     sequence<string> Channels
}
```

**Fig. 5.** Joint node with proposed field additions

```
Interface Motion {
        int Frames
        float FrameTime
        sequence<float> transformation
}
```

**Fig. 6.** Motion node, proposed

## 9 Conclusions

In this paper, motion capture animation using H-Anim X3D models and X3D interpolators was defined by providing a conversion algorithm to convert from captured motion to X3D interpolators. The captured motion is retargeted to the H-Anim model and is then displayed with a general X3D browser. With this conversion algorithm, motion capture data is easily defined in an H-Anim X3D file for the motion of an H-Anim model.

That said, because X3D interpolators are designed for keyframe animation, note that the converted motion capture H-Anim file includes keyframe animation concepts and keywords, such as key, key values, and interpolators, that are not necessary for motion capture animation. In order to simplify H-Anim motion data definition, we have proposed a method to directly combine an H Anim model and motion capture data by defining an updated joint node and a new motion node for the H-Anim specification.

All these works have been progressed in the H-Anim Working Group of the Web3D Consortium in cooperation with the ISO/IEC JTC 1/SC 24 standardization group. The H-Anim working group is developing ISO/IEC standards for H-Anim revisions. The work described in this paper was done as a part of the development of the standards.

**Acknowledgments.** We express our appreciation for the members of the H-Anim Working Group of the Web3D Consortium including Joe D. Williams, William O. Glascoe, and Richard F. Puk who have graciously provided us with a great deal of valuable comment on this work. In addition, we extend our appreciation to Sarah Ashlie for her review and edits. This research was supported by the National Standards Technology Promotion Program of the Korean Agency for Technology and Standards, MOTIE (Ministry of Trade, Industry, and Energy).

## References

1. Humanoid animation (H-Anim) International Specification V1, http://www.web3d.org/files/specifications/19774/V1.0/HAnim/HAnim.html
2. ISO/IEC IS 19775-1:2013, X3D Architecture and base components V3, Nov. 2013 (http://www.web3d.org/realtime-3d/specification).
3. Sanna, A., Montuschi, P., A new algorithm for the rendering of CSG scenes. The Computer Journal, 9:555-564, 1997
4. Maillot, P.-G., Using quaternions for coding 3D transformations, A. S. Glassner, editor, Graphic Gems, pages 498-515. Academic Press, Boston, MA, 1990
5. Buttussi, F., Chittaro, L., Nadalutti, D., H-Animator: a visual tool for modeling, reuse and sharing of X3D humanoid animations, Proceedings of the eleventh international conference on 3D web technology, ACM, 109-117, April 2006
6. Cobo, M. , Bieri, H., A Web3D Toolbox for Creating H-Anim Compatible Actors, Computer Animation 2002, 120-125, June 2002
7. Ortiz, A., Oyarzun, D., Aizpurua, I., Posada, J., Three-dimensional Whole Body of Virtual Character Animation for its Behavior in a Virtual Environment Using H-Anim and Inverse Kinematics, Proceedings of Computer Graphics International, 307-310, 2004
8. Brutzman, D. and Daly, L., X3D, Extensible 3D Graphics for Web Authors, Morgan Kaufmann Publishers, 2007
9. X3D Basic Examples Archive (including open-source H Anim and Medical scenes), http://www.web3d.org/x3d/content/examples/Basic
10. Biovision Hierarchy (BVH) file format, https://en.wikipedia.org/wiki/Biovision_Hierarchy
11. ISO/IEC 19774 H-Anim Working Group Public Wiki, Web3D Consortium, http://web3d.org/wiki/index.php/H-Anim
12. X3D-Edit authoring tool, https://savage.nps.edu/X3D-Edit