Adapting Motion Capture Data to Follow an Arbitrary Path

Mark Whitfield
Supervised by Shaun Bangay and Adele Lobb

Abstract

This paper presents a method to adapt motion capture data to follow an arbitrary path. The central idea is to create a path which represents an abstraction of the motion. The motion is then represented relative to the path. Thus altering the path alters the motion. Results using this method (which does not incorporate constraints) exhibit undesirable side-effects like footskate. However, if the path corresponds closely to the actual motion we can minimize these effects.

1 Introduction

The subtle details of human motion are present in motion capture data. These subtle details are key to the realism of the motion. Furthermore, a variety of locomotion styles may be captured. Thus motion capture data provides a good basis for creating realistic avatar movement.

Motion capture is an expensive and time-consuming process. Therefore, it is desirable to be able to alter previously recorded data. For example we may have motion capture data of a man walking in a straight line, but we need him to walk around an obstacle. Instead of spending time and money on recording more data, a motion capture data manipulation technique can be employed.

Gleicher (GLEICHER 2001) introduces a method for editing the path of a motion. The path is an abstraction of the motion (Section 3). The motion is represented relative to the path (Section 3.1), and thus by altering the path we can alter the motion accordingly. The initial path is created by using a least squares fit of motion capture data to a polynomial (Section 3.2). The root node translational data contained in the motion capture file is used for the fitting.

The path is parameterized by time, and thus altering the path may alter the speed. Arc-length parameterization (Section 4) is used to make sure that the speed along the edited path is the same as the original. The avatar's position on the path is mapped by arc-length rather than time.

2 Related Work

There are three main schools of thought for getting an avatar to follow an arbitrary path using motion capture data:

- The desired path is followed by editing a base motion (Section 2.1)
- A new motion is synthesized that fits the desired path (Section 2.2)
- A cyclic motion is applied over and over again as the avatar is displaced along the path (Section 2.3)

2.1 Altering a Base Motion

The motion of an avatar can be specified by the root node translation and orientation at each frame, as well as each joint angle at each frame. Each of these parameters (e.g. knee joint) can be represented as a function of time. A motion curve is one of these parameters as a function of time. Witkin and Popovic (WITKIN and POPOVIC 1995) use a technique based on warping motion curves. Keyframes are used as constraints in the motion warp. Although Witkin and Popovic do not directly address the problem of path transformation, we could warp the motion curves so that the avatar follows the desired path by setting certain keyframes which the avatar must pass through.

With methods which involve altering a base motion, blending may be used to create the base motion by joining two or more motions together, thus creating a longer path. Constraints may be required to prevent undesirables like footskate.

2.2 Synthesis

With synthesis methods, a sequence of motion capture clips is put together so as to follow the desired path. This sequence is extracted from a library of clips, usually stored in a graph structure.

Kovar et al. (KOVAR et al. 2002) construct a directed graph from a library of motion capture data. This graph consists of nodes which represent transition points within the database. The user specifies the transition threshold which will determine the connectivity of the graph. A cost value is associated with each possible transition, and motion is generated by building walks on the graph. A branch and bound search is used to find sequences which satisfy user demands such as sketched paths.

Lee et al. (LEE et al. 2002) use a similar technique to Kovar et al. They also construct a directed graph from motion clips containing possible transitions. A cost formula incorporating joint angles, velocities and positions is used to identify the cost of possible transitions, and this data is stored in a matrix. The sketched path is considered as a sequence of goal positions to be traversed. The tracking algorithm used by Lee et al. is based on a best-first search through the directed graph.

Synthesis methods also require constraints to avoid undesirables like footskate. Blending is an integral part of synthesis methods as the two frames associated with a transition may not be similar enough for a smooth transition to occur.

2.3 Cyclic Motions

A common technique used by animators is to define locomotion cycles and loop these cycles while the character is moving along. This technique has been used since the earliest productions (LUTZ 1920).

This idea can be extended to motion capture data. Motion capture sequences which transition onto themselves are used. This single cycle is applied over and over as the avatar is displaced along the path (ROSE et al. 1998). This method does not provide a way to alter an existing complex path.

3 Paths

Motion capture data does not explicitly contain a path, but rather a series of translations and orientations. In order to give meaning to the motion for the user, a path is constructed based on the translations in the motion capture data. The path can be thought of as an abstraction of the motion, where smaller details are disregarded.

3.1 Representing the Motion Relative to the Path

We can describe the avatar's position relative to the path by taking the difference between the avatar's actual position and the path's position at a given time. This is known as the translation residual. The direction of the path at a given time is defined as its tangent vector. Thus we can also describe the avatar's orientation relative to the orientation of the path by calculating the transformation required to get from the path's orientation to the avatar's orientation at a given time. This is known as the orientation residual.

The motion is represented relative to the path. Thus altering the path alters the motion. When the motion is rendered, effectively we transform onto the path and then apply the translation and orientation residuals.

3.2 Creating the Initial Path

The root node translation data is extracted from the motion capture data. The initial path is created by applying a least squares fit to the root node translation data. Code for a least squares fit in two dimensions was obtained (KAVANAGH 2003). However, we need a least-squares fit in three dimensions. Three separate functions are used for the fitting:

- f(t) represents the least-squares fit of the x vs. time data points
- g(t) represents the least-squares fit of the y vs. time data points

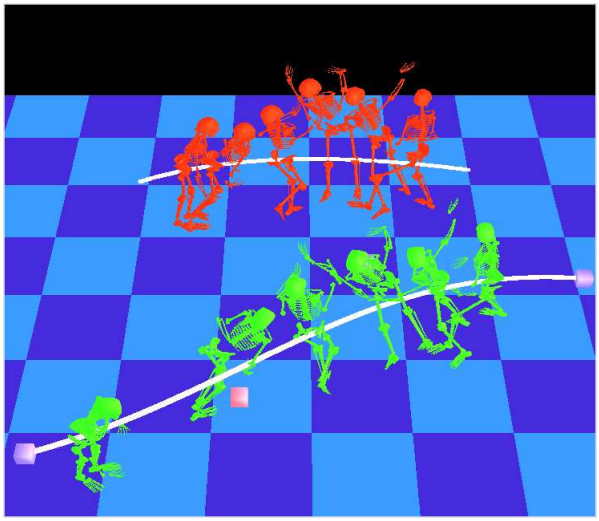

Figure 1: In the background is the original motion and the original path. In the foreground is the adapted motion and the corresponding b-spline and control points. The frames displayed are frames 1, 69, 81, 89, 98 and 116. These frames were selected from the 155 frames that make up the motion because they are fairly evenly distributed in space.

- h(t) represents the least-squares fit of the z vs. time data points

The position of the initial path at time t is the point $[f(t), g(t), h(t)]$. The orientation of the initial path at time t is the tangent vector $[f'(t), g'(t), h'(t)]$.

3.3 Editing the Path

The path is represented as a B-spline with a number of control points. The knots are designed so that the spline passes through the first and last control points. The user has direct control over the control points.

4 Arc-Length Reparameterization

Since the path is parameterized by time, altering the path may also alter the speed of the avatar. For example if we lengthen the path, the speed of the avatar will increase (Figure 1). This is generally undesirable. By parameterizing the path by arc-length, the original velocities are maintained (Figure 2).

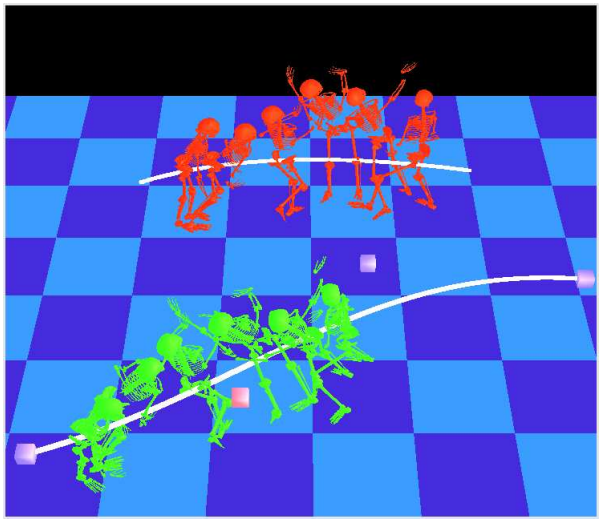

Figure 2: In the background is the original motion and the original path. In the foreground is the adapted motion and the corresponding b-spline and control points. The frames displayed are the same frames displayed in Figure 1. Arc-length reparameterization is now being used.

4.1 Calculating Arc-Length

The arc-length of frame f is the distance along the path from the start of the path to the position on the path at frame f. We calculate the arc-length along the original path for each frame. We use Equation 1 (OKIKIOLU 2003) to calculate the arc-length between time a (the start of the path) and time b (the time of the frame).

$$L(t) = \int_a^b \sqrt{(dx/dt)^2 + (dy/dt)^2 + (dz/dt)^2}dt \quad (1)$$

Euler-step integration (FITZPATRICK 2003) is used to calculate the arc-length at each frame. The idea behind Euler's method is to approximate a curve with a series of straight lines.

4.2 Finding the Co-Ordinates on a New Path

If we wish to maintain the speed of the avatar, frame f in the new path must have the same arclength as frame f in the original path. Given a new path, we can find the point on the path that corresponds to a given arc-length. To find this point we sample the arc-length along the path at small time intervals. When the arc-length along the path becomes greater than the desired arc-length, we take the co-ordinates of the path at that time. Although the final sampled arc-length is unlikely to be exactly the same as the desired arc-length, a sufficiently small timestep provides enough accuracy for our purposes.

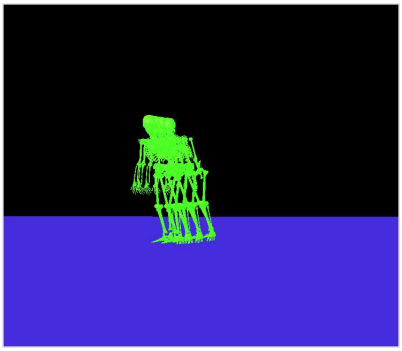

Figure 3: Footskate resulting from altering the path

4.3 Transformation

The user alters the original path. We already have the arc-length at each frame. We use the arc-length to find the point on the new path to transform onto. We calculate the tangent vector at this point to to be used for the orientation transformation. Finally, we transform onto the path before transforming using the translation residual and the orientation residual.

5 Results

For each frame we first transform onto the path, and then transform using the residual. Even if the motion is stationary for a period, the transformation onto the path will move along the curve. For example if we have a sequence of 4 stationary frames in row, the points that they correspond to on the path will be different, as will their residuals. However, with the original path the avatar will be stationary because by applying the transformation onto the path, and then the residual transformation, we end up with the same overall transformation for all 4 frames. When we alter the path, the residuals are still the same, but the transformation onto the path at each of these 4 frames has changed. Thus the avatar will slide. This is known as footskate.

This technique is not suitable for altering motions where there are lengthy stationary periods. Typically this leads to significant footskate (Figure 3) once the path has been altered.

A very complex motion with an initial path of relatively low degree will result in the adapted path being vastly different from the actual path followed by the avatar. Since the adapted path has very little correlation to the actual path followed by the avatar, it is difficult to guage how and where to change it to obtain the desired change in the actual path.

6 Conclusions

The most suitable motions for this technique are:

- motions which have a least-squares fit that corresponds fairly closely to the actual path followed by the avatar. A tighter fit may be obtained by adjusting the degree of the polynomial used for the least-squares fit.
- motions which do not have stationary periods.

In some cases there is a stationary period at the beginning and/or at the end of a sequence of motion capture data. It is fairly easy to directly alter the motion capture file to cut out the first few and/or last few frames, thus removing the stationary parts.

The results achieved would have been better if constraints had been implemented, especially as far as footskate is concerned.

References

FITZPATRICK, RICHARD (2003). Euler's method. `http://farside.ph.utexas.edu/teaching/329/lectures/node60.html`, 2003.

GLEICHER, MICHAEL (2001). Motion path editing. In *Proceedings of the 2001 symposium on Interactive 3D graphics* (2001), ACM Press, pp. 195–202.

KAVANAGH, RICHARD (2003). Least-squares curve fit of arbitrary order. `http://www.david-taylor.pwp.blueyonder.co.uk/software/components.html`, 2003.

KOVAR, LUCAS, M. GLEICHER and F. PIGHIN (2002). Motion graphs. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), ACM Press, pp. 473–482.

LEE, JEHEE, J. CHAI, P. S. A. REITSMA, J. K. HODGINS and N. S. POLLARD (2002). Interactive control of avatars animated with human motion data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), ACM Press, pp. 491–500.

LUTZ, E.G. (1920). *Animated Cartoons: How They are Made, Their Origin and Development*. Charles Scribner's Sons, 1920.

OKIKIOLU, KATE (2003). Math 21c calculus and analytical geometry. `http://math.ucsd.edu/~okikiolu/21c/l12.pdf`, 2003.

ROSE, C., M. COHEN and B. BODENHEIMER (1998). Verbs and adverbs: Multidimensional motion interpolation. In *IEEE Computer Graphics & Applications Volume 18, No. 5* (1998).

WITKIN, ANDREW and Z. POPOVIC (1995). Motion warping. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), ACM Press, pp. 105–108.