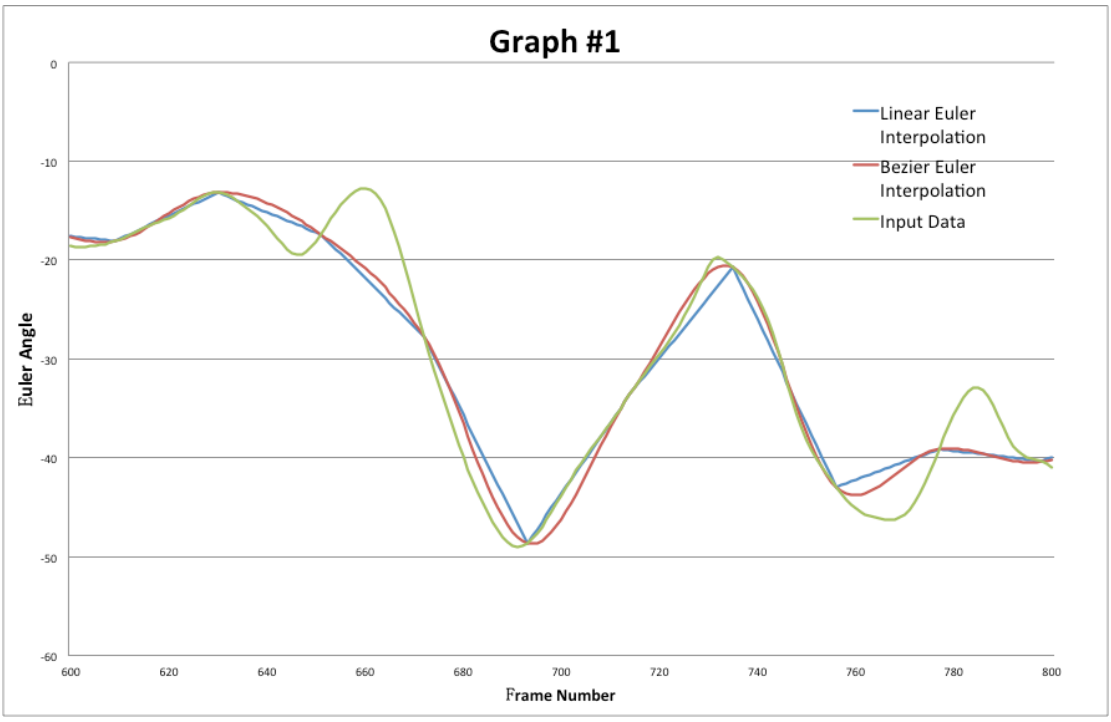
CSCI520 Assignment 2 Report

Each value in Euler angles represents the rotation in degrees around one of the 3 axises in 3d space. So Euler angle is most intuitive way of people thinks how rotation works in 3d space. Most of the time you will want to create angles using Eulers because they are conceptually the easier to understand. Euler angles are of great importance to physics, especially classical mechanics. The flaw is that Euler angles have a problem known as the gimbal lock that prevents certain rotations when two axises align. Therefor we have optional solution, which is quaternion.

Quaternion represents angles in the coefficients of 4 terms, instead of 9 coefficients in rotation matrix. Another advantage of using Quaternion is that it's easy to interpolate between two keyframes by using sphere linear Interpolation between unit quaternions. It also performance natural rotation on body joins (see Graph #3).

The easiest way to interpolate between two points is the use of linear interpolation. A straight line is however not useful to animations since a rotating joint is supposed to move along a smooth curve (see Graph #1 and Graph #2).

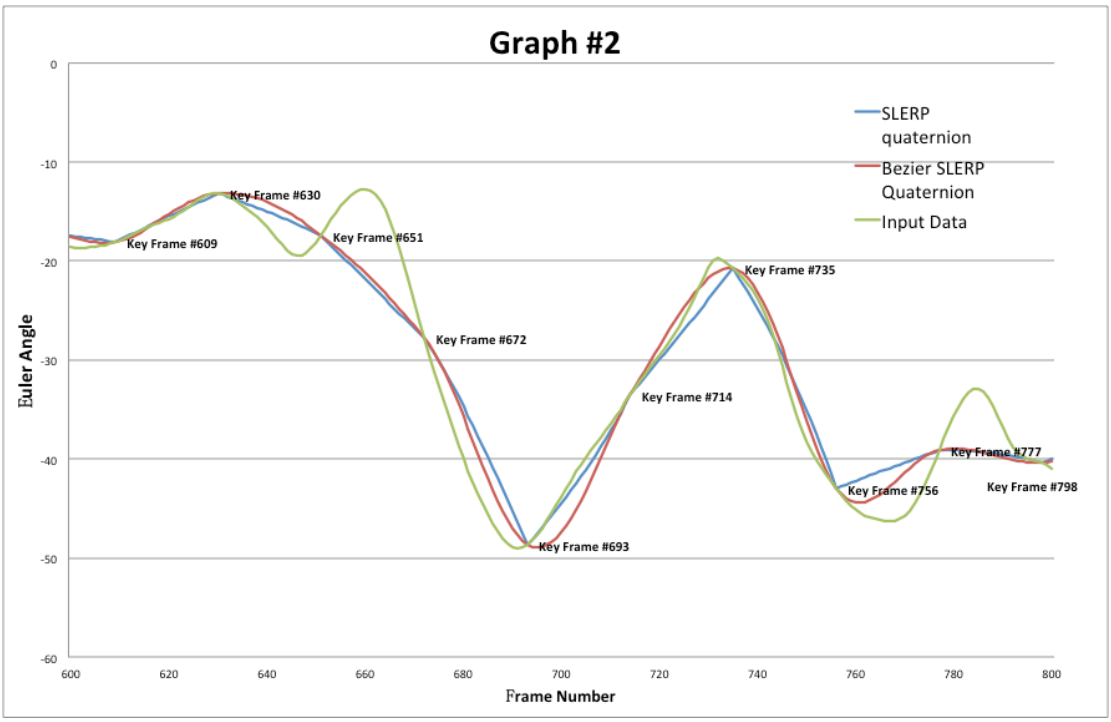
If we output and visualized interpolated angle, we can plot graph for 131_04-dance.amc input file with N = 20.



Graph 1: Linear Euler and Bezier Euler Interpolation comparison

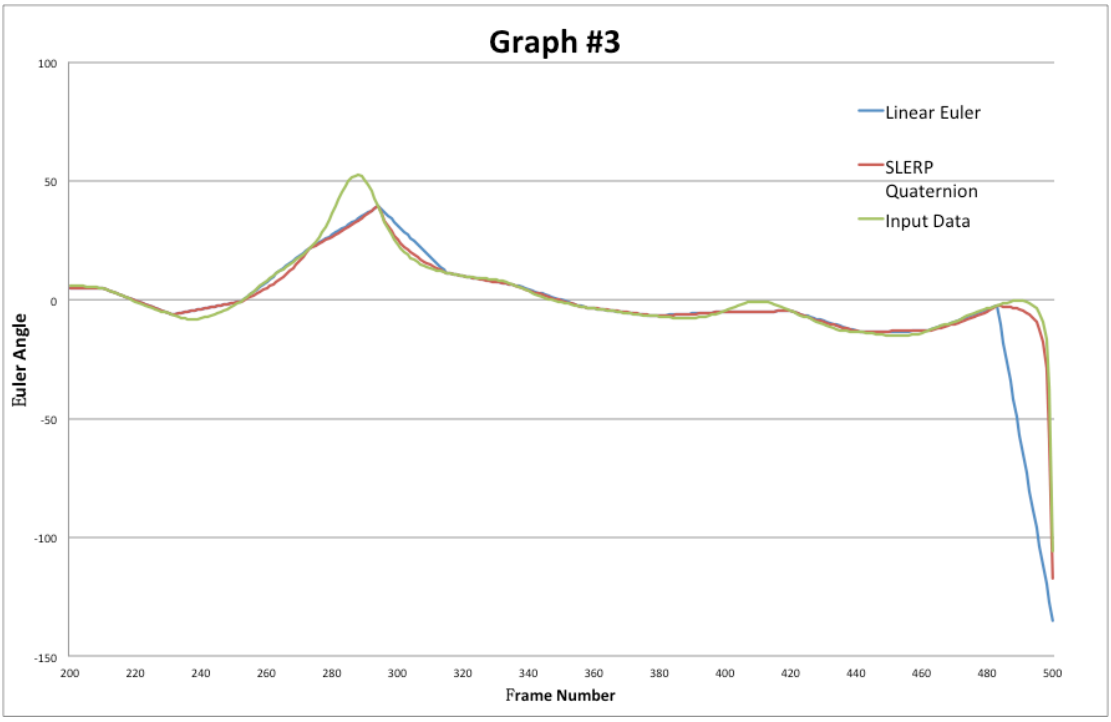
Graph #1 is comparison between Linear Euler interpolation and Bezier Euler

interpolation result. We can see both interpolation results in approximate curve from frame 600 to frame 800. Linear Euler interpolation shows straight line between each keyframe (plotted graph might be flawed due to Microsoft Excel plotting method, please check exact data in attached excel files). Bezier interpolation points form smooth curve showing natural gesture changes (joint point angle changes). Both linear and Bezier interpolations meet input data at same keyframes.



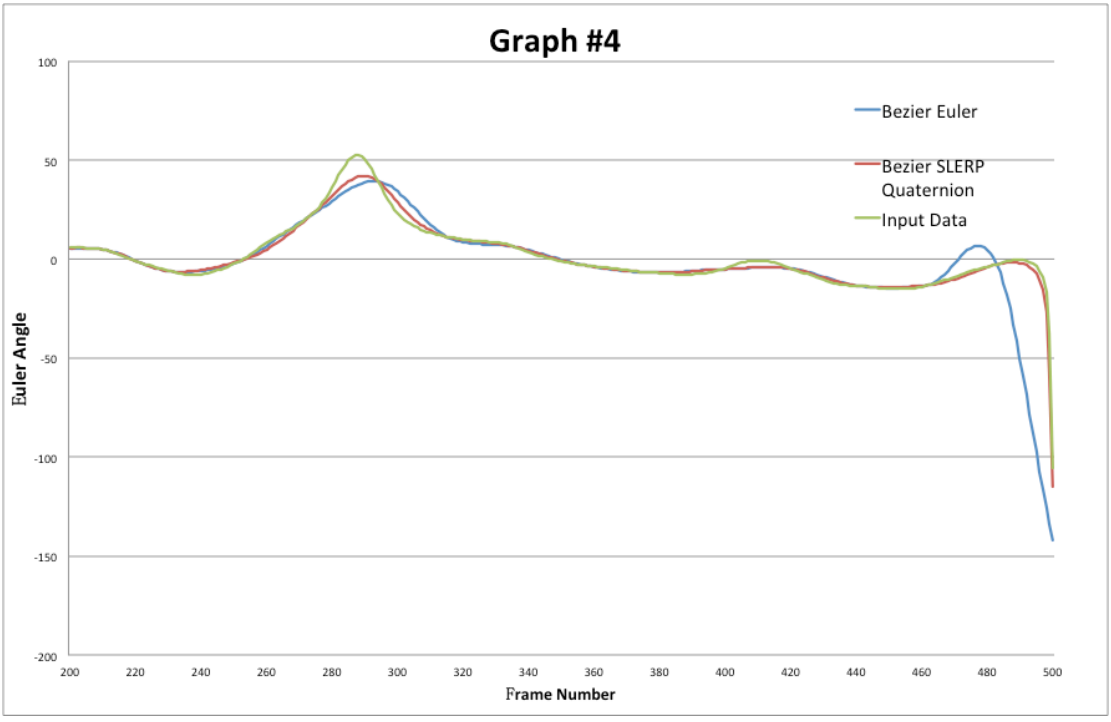
Graph 2: Linear Quaternion and Bezier Quaternion comparison

Graph #2 is comparison between Linear Quaternion interpolation and Bezier Quaternion interpolation result. It still reflects how Bezier works better on simulating angle shifting. For both graph #1 and graph #2, interpolation lost information of fluctuate data while it appears between two keyframes. However Bezier Interpolation by Euler and Quaternion can all slightly reflect such situation (clips around frame 670 to 740 and frame 760 to 780).



Graph 3: Linear Euler and Linear Quaternion comparison

While we look into Graph #3, which both interpolations are implemented in linear methods, interpolations are quite different, because of different functions decide how angles shifting from one to another.



Graph 4: Bezier Euler and Bezier Quaternion comparison

Graph #4 is comparison between Bezier Euler and Bezier Quaternion interpolation

result. Even Bezier function smooth the curve, rotation still shows strange in Euler angle around frame 480. However, Bezier Quaternion interpolation almost perfectly matches original data, except that big rotation without keyframe around frame 280.

Additional comparison:

In my program, I try to output running time of whole interpolation (without reading data and output data part) on 131_04-dance.amc file using individual methods, Linear Euler (0.034346s), Linear Quaternion (0.177121s), Bezier Euler (0.164703s) and Bezier Quaternion (0.518773s). Running time indicates that Linear Euler Interpolation cost the least and Bezier Quaternion cost the most. Bezier Euler and Linear Quaternion are competitive, maybe finding control point and doing quaternion transformation cost similar system resource (?). All running times are not absolute and vary, but relations among them are deterministic.

Appendix:

How to build fltk on MAC OS X, Xcode 4.2 and import fltk in starter code:

1. open fltk-1.3.0/ide/Xcode4/ FLTK.xcodeproj
2. Set the "Active Target" to fltk, fltk_gl. Then build release for both framework.
3. Locate the output frameworks, and put them into library/frameworks/.
4. Create new project and put corresponding .cpp, .h files and “add Files to” your project.
5. For compiling “interpolate”, you only need import “fltk.framework”
6. For compiling “mocapPlayer”, you need import “fltk.framework”, “fltk_gl.gramework” and original “OpenGL.framework” under path /System/Library/Frameworks/

To Yili: Graph is quite small, so I also put excel files including data in package for review. There are three videos named “Beizer Euler 40” “Beizer Qua 40” and “Linear Qua 40”, which are required video “Input motion and Bezier Euler”, “Input motion and Bezier SLERP quaternion”, “Input motion and SLERP quaternion” respectively. Sorry for that it is too large to upload images files, i don’t have enough network bandwidth.