Assignment 5 Report

Liu Haixiang and Tomislav Pejsa

As part of this assignment we have implemented the following features:

1. **Parsing and playing back motions in BVH format.** Currently we support 3ds Max Biped-friendly BVH files. This is convenient because the entire CMU database has been converted to this format and made available for download: <https://sites.google.com/a/cgspeed.com/cgspeed/motion-capture/3dsmax-friendly-release-of-cmu-motion-database>. So far we have tested our system on 12 randomly chosen motions from the database, from 10 different capture sessions, and it was able to correctly load and display all of them.
2. **Displaying motions.** Our GUI application renders motions using OGRE. The skeleton is represented as a set of bone segments visualized as oriented, non-uniformly scaled boxes.
3. **Scrubbing and playing at different speeds.** These features are partially supported. Our program provides a slideable timeline which can be used for scrubbing, and the user can also modify playback speed. However, we do not currently support frame-locked playback, as we have not yet completed our implementation of a fixed frame rate render loop in wxWidgets (the framework we use for our GUI application). Additionally, for historic reasons, our animation system does not explicitly assume uniformly sampled motion – instead, it represents motion as a collection of joint tracks, each containing keyframes arbitrarily positioned along a continuous time axis. While this representation lends itself well to keyframe reduction and keyframe-based editing techniques, additional logic must be programmed to support easy playback control of uniformly sampled motion.
4. **Camera control.** Our program supports camera rotation, panning and zooming using the mouse. So far we have had no need to implement automatic camera repositioning and zooming based on motion extents, because all motions in the 3ds Max-friendly release of the CMU database have been edited to have the same scale, and to begin at the origin. However, we have implemented logic for computing motion extents, and we can easily add the camera repositioning feature if we end up needing it.
5. **Loading multiple motions.** Our program is currently set up to automatically load motions from a specific directory on startup. The loaded motions are then displayed in a list control, and can be selected for playback by double-clicking. Our current approach makes working with large motion datasets untractable; the solution is to load motions on selection (our resource management system already has support for this), and possibly have a background thread for loading resources.
6. **Joint marker and traces.** Our animation system can provide world positions for any joint. In our GUI program, we use this feature to render purple markers on joints. Similarly, we support joint "traces" – piecewise linear curves representing joint trajectories in the world space. The user can enable or disable markers and traces for any joint using appropriate checklists.
7. **Splicing motions into sequences.** Our program can play multiple motions as a sequence. When transitioning from one motion to the next, the target motion is rigidly transformed so that root position and orientation are continuous. Our animation system can also perform blend-based transitions, but we do not expose this functionality in the GUI program.
8. **Quaternions.** All rotations in our system are represented as quaternions.
9. **Creating new motion clips and writing them out.** Our program allows the user to select a segment of an existing motion clip and create a new motion clip out of it. This can make those long motions easier to handle. The program does not yet expose functionality for writing out motion files, but our animation system does have loaders and serializers for a custom XML file format, so adding this feature would be fairly trivial.

Screenshots illustrating the various features can be seen below:

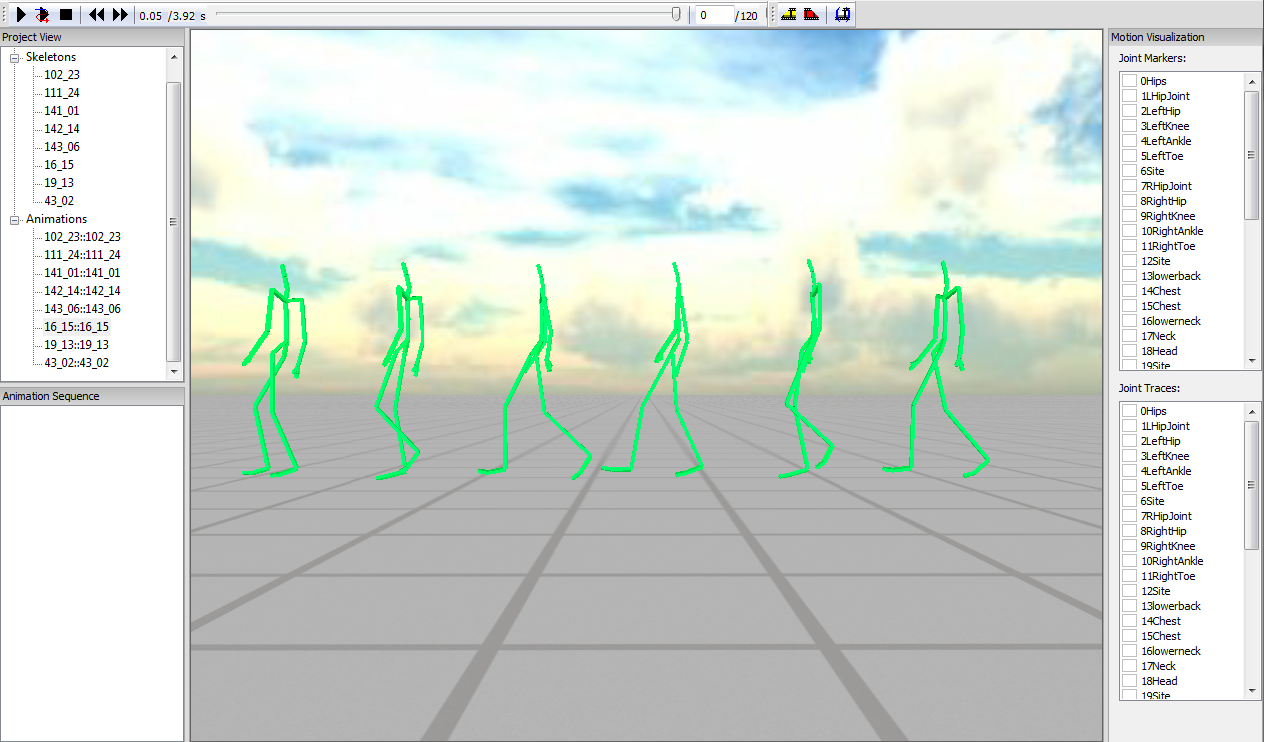


Figure 1: Viewing a walking motion (16\_15 from the CMU database).

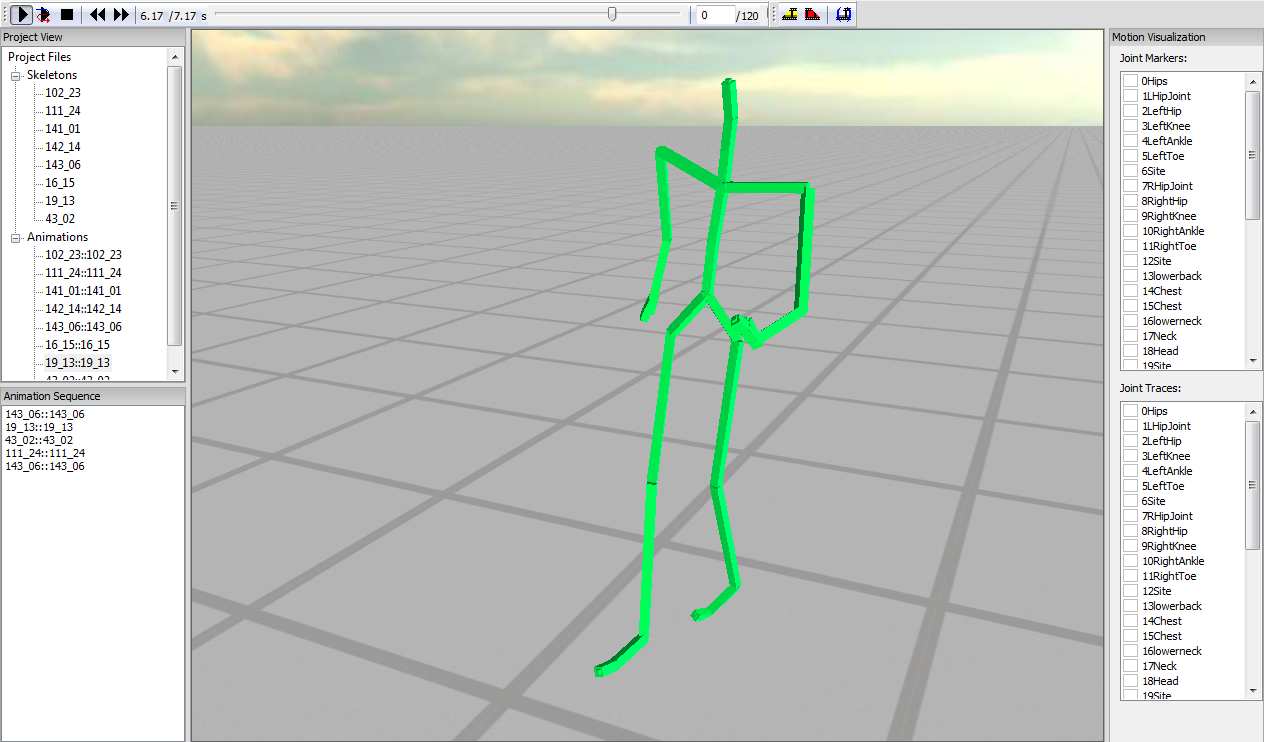


Figure 2: Program interface. (1) Playback controls – Play/Pause, Play Sequence, Stop, Step to Previous/Next Frame, Timeline, Frame Rate. (2) Editing controls – Select/Deselect Animation Segment, Create New Animation From Selection. (3) Project View, showing all currently loaded skeletons and motions. (4) Animation Sequence – motions can be queued here by dragging and dropping them from Project View. (5) Motion Visualization – for specifying which joints should have markers and traces on them.

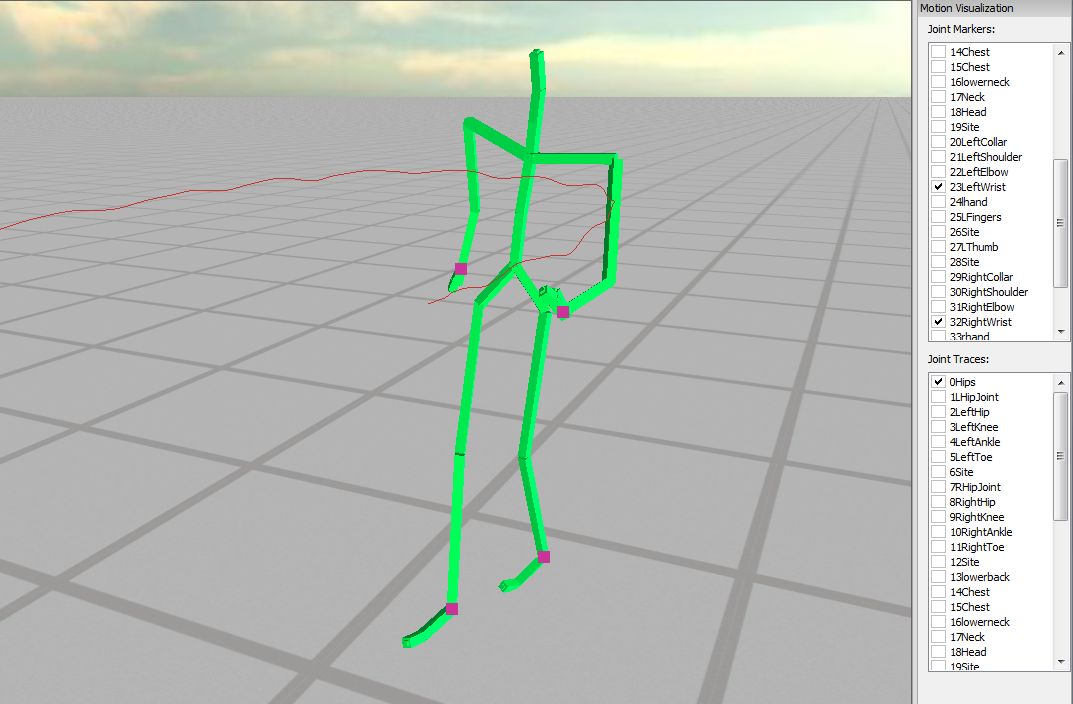


Figure 3: Helpful motion visualization features. Joint markers are indicated with purple points (placed on end-effector joints in this example), while the red curve represents the path of the root joint (currently rendered as overlay, hence incorrect occlusion).