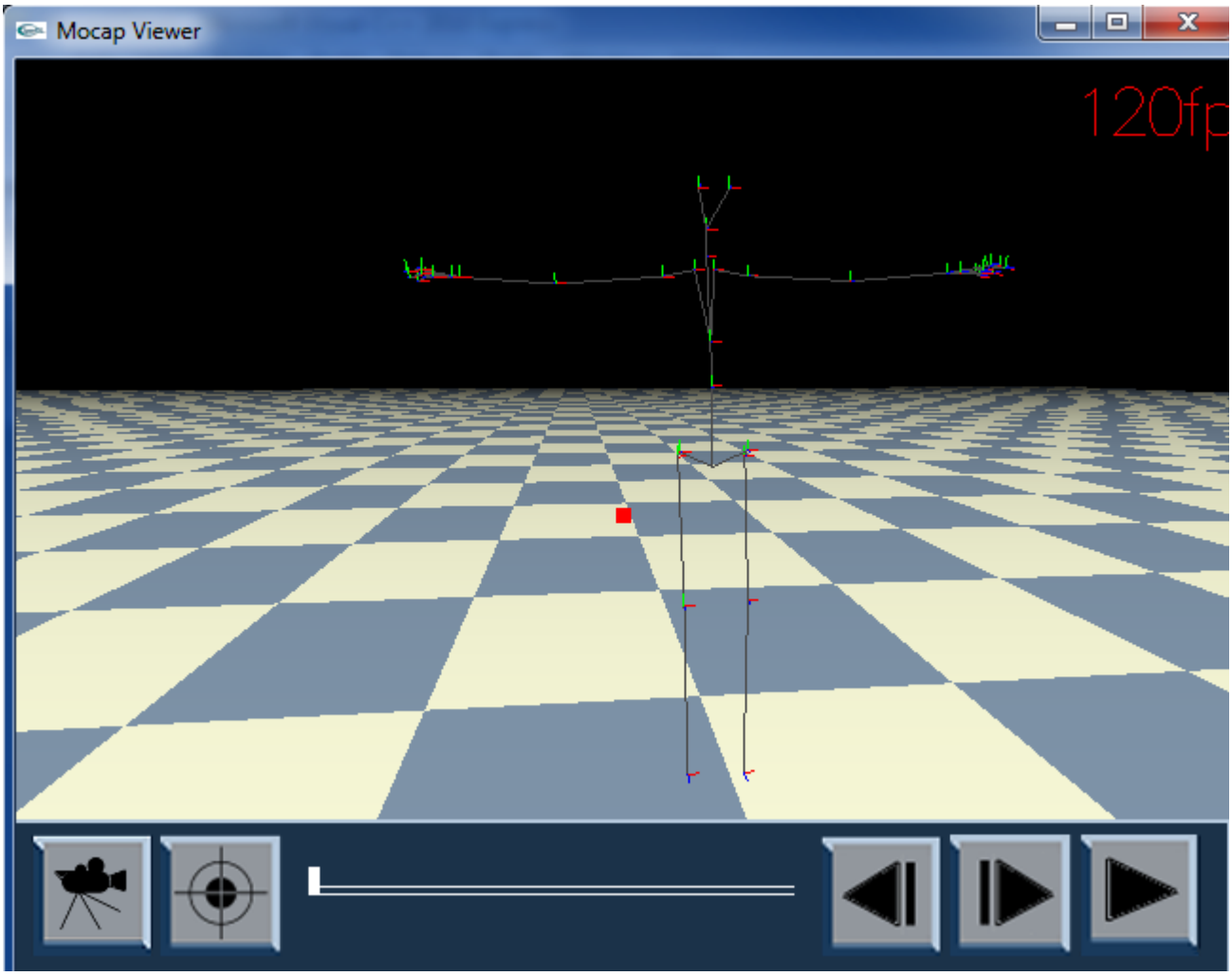
Language/Libraries: C++, openGL, working on Qt, custom vector, quaternion, camera, ui, and parser code.

Parser:

Our parser is tailored to parse BVH files. It accesses a file once, constructs a node hierarchy then couples the appropriate motions to each node depending on degrees of freedom. Each node contains an offset to the parent, links to their respective parent and children (if applicable), motions (which are accessed based on frame), and a history of previous positions (for drawing "streamers" for each node.

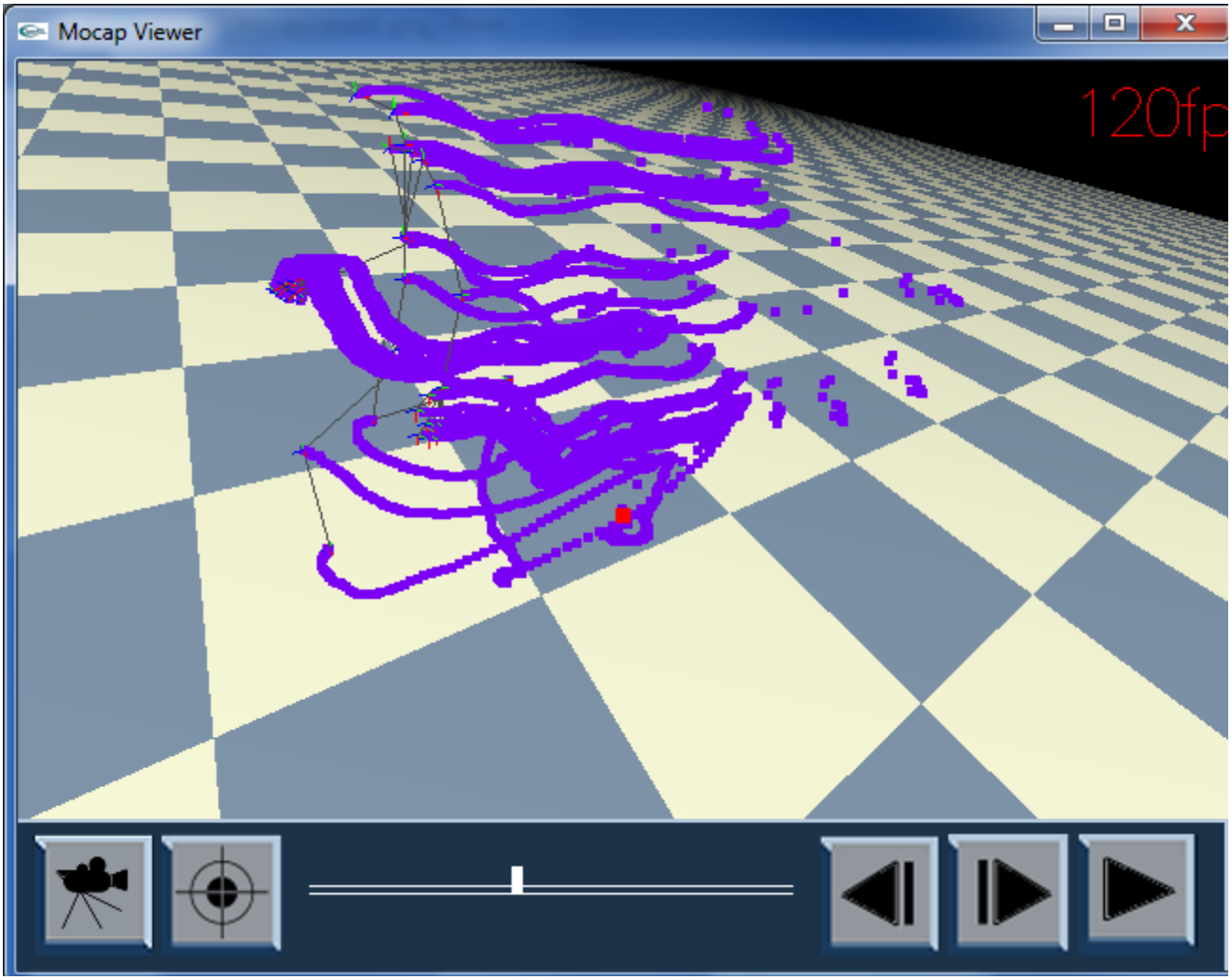


Motion:

Our framework currently draws a BVH motion capture figure as line segments though we do draw the local axis at each node as a means to test and visualize the rotations at the joints. Volumes for bones are in progress. We used BVH Viewer to verify the correctly of our display, ours matches the utility with the same or even greater visual quality.

Tracking:

Each node keeps a record of previous positions. As each node is being drawn for its respective frame its history is drawn as well (in the form of points). Once the drawing is complete, the node's history is updated with the position of the node in the current frame.

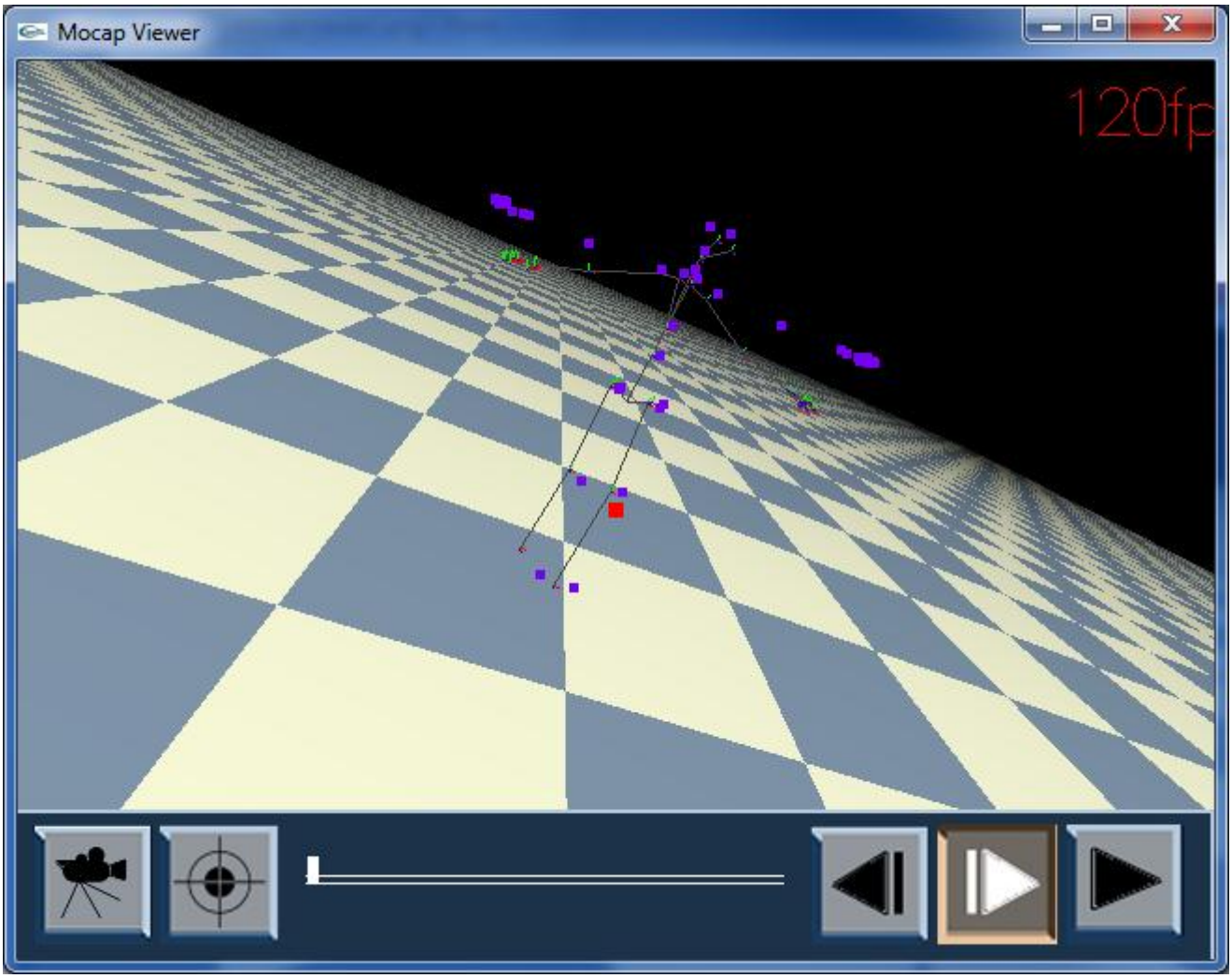


Framerate/scrubbing:

We calculate the refresh based on the motion capture data and display the text on screen. You can hit the "play" button, move one frame forward or backwards, and move a slider to control the motion. Adjusting the frame rate is trivial at this point as well, though no keys or buttons are mapped for this.

Camera:

We have a great camera. It uses quaternions for rotations and supports are basic camera movements: trucking, panning, dollying, rolling, and orbiting (around a target). We can lock the camera onto the target so it moves as the character moves. Mouse movement is also supported.



Multiple file:

Framework can easily support multiple files, but the user has no say as to which files. Unfortunately, must be specified at compile-time for now (need to change the file name string)

Splicing:

There is currently no interface or keyboard command that allows the user to arbitrarily select frames. However, the splicing is present within the program itself. Splices are specified and stored as vec2df, which in turn stored in a vector container. Each splice corresponds to exactly one motion capture set (see loading multiple motions). As is, the program runs each loaded motion based on the splice range (which are currently hard coded values).

Internal Rotations:

We are uses quaternions for our rotations; however it's very inefficient at the moment. I need to work on getting the quaternions concatenated properly with the current setup.

Write-out:

Did not do.