6.111 Final Project Status Report

Adam Gleitman, Andrew Shum, Tim Balbekov

OVERALL

Safety Goals

- Capture static left and right images of an object and store a single, anaglyph-filtered frame into the front buffer ZBT
- Display a static anaglyph image onto the screen from this buffer
- Track the location of a dot on two camera frames and triangulate its location in three-dimensional space
- Translate a gestural motion to a change in virtual camera position
- Render a 2D image of a wire-frame polyhedron as viewed from a hard-coded virtual camera position

Ambitious Goals

- Continually capture left and right images of an object and store a single, live-updated, anaglyph-filtered frame into the front buffer ZBT
- Provide a live anaglyph feed onto the screen from this buffer
- Obtain more advanced virtual camera movements, such as rotation and panning, by tracking more dots
- Render a 2D image of a wire-frame polyhedron as viewed from a virtual camera position specified using gestural motion

TIM

Checkoff List

- NTSC decoder controller automatically switches between two video feeds after a video field is recorded.
- Frame grabber converts NTSC signal to appropriate RGB values, and stores them to ZBT via the ZBT controller.
- ZBT Controller generates appropriate SRAM control signals in ModelSim.
- ZBT Controller has a functional hardware implementation, and timing constraints for the safety goal are met.
- ZBT controller copies from back to front buffer, prioritizing front buffer read access to the VGA controller, and back buffer write access by the frame grabber.

ANDREW

Checkoff List

- Demonstrate test jig with left pixel / right pixel as input into Anaglyph Filter module and resultant pixel as output. Validate the accuracy of the anaglyph rendering using a Python script and the Python Imaging Library
- Demonstrate test jig with hard-coded virtual camera position as input and 3D rendering frame data as output. Validate the accuracy of the wire-frame rendering using a Python script and the Python Imaging Library
- Demonstrate VGA controller rendering hard-coded anaglyph images and wire-frame

polyhedron models onto the screen. Verilog code for hard-coding these images into memory will be generated with a Python script

ADAM

Checkoff List

- Track the location of a single dot on a single camera frame using a center of mass calculator
- Calculate the position of a dot in three-dimensional given two centers of mass as seen from two different camera angles
- Translate a dot position into a change in a virtual camera angle
- Demonstrate functionality of modules using test jigs and live testing