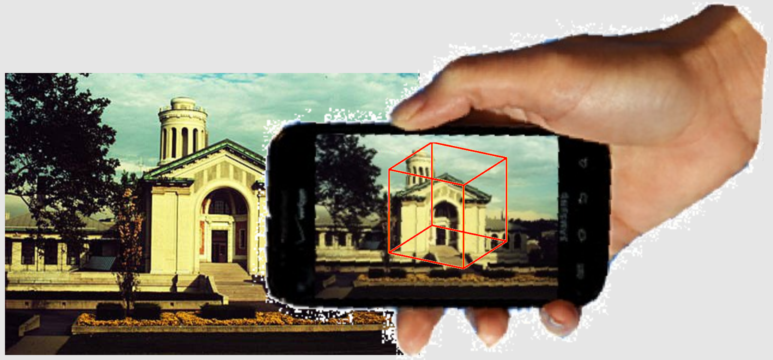
Team 5

Concept and Motivation

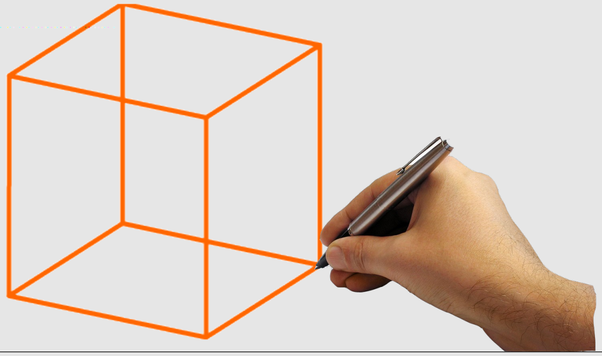
WHAT?

- This project will allow users to view and add virtual 3D objects embedded in reality via their mobile phones – idea can be expanded for many applications.
- The prototype integrates smart phones with graphics technology to create a 3D presentation tool.



WHY?

- People today have the chance to experience augmented reality through current smart phone apps but they are usually fixed in content and non-interactive.
- Using our prototype system, the user is in control of a rich and collaborative multimedia environment.



Competitive Analysis

StreetMuseum

- StreetMuseum is an iPhone app that allows you to browse historical photographs in various parts of London in real time.

- Our prototype differs in that we are able to add to the existing virtual objects, as well as view the objects in 3D.

Google Goggles

- Google Goggles is visual search app for Android phones that allows you to search for text/landmarks/books/contact info… based on what is within the frame of reference on your phone.

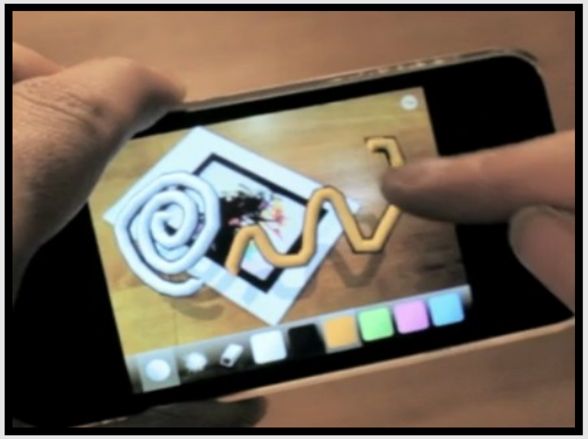
- Our prototype differs in that we can add objects in real time, and view from all angles.

11:03
Google goggles labs
Landmark
Golden Gate Bridge
San Francisco, United States
Map
Web
Options
Golden Gate Transportation District

Scrawl

- Scrawl is an iPhone app that allows for 3D spatial drawing in augmented reality. The user can move the object to draw at different angles.

- Our prototype differs in that we allow for collaboration amongst various users.



Technical Specifications

Hardware

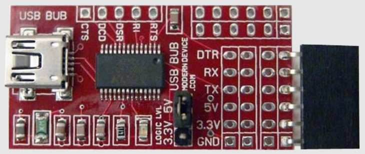
- Android: T-Mobile G1, Firmware version 1.6, updated to 2.1
- Drawing Tool/Remote Control: JeeNode Main MCU, JeeLink Server Interface, JeeLabs Power Supply Board, USB BUB Board, LEDs
- Kinect

Software

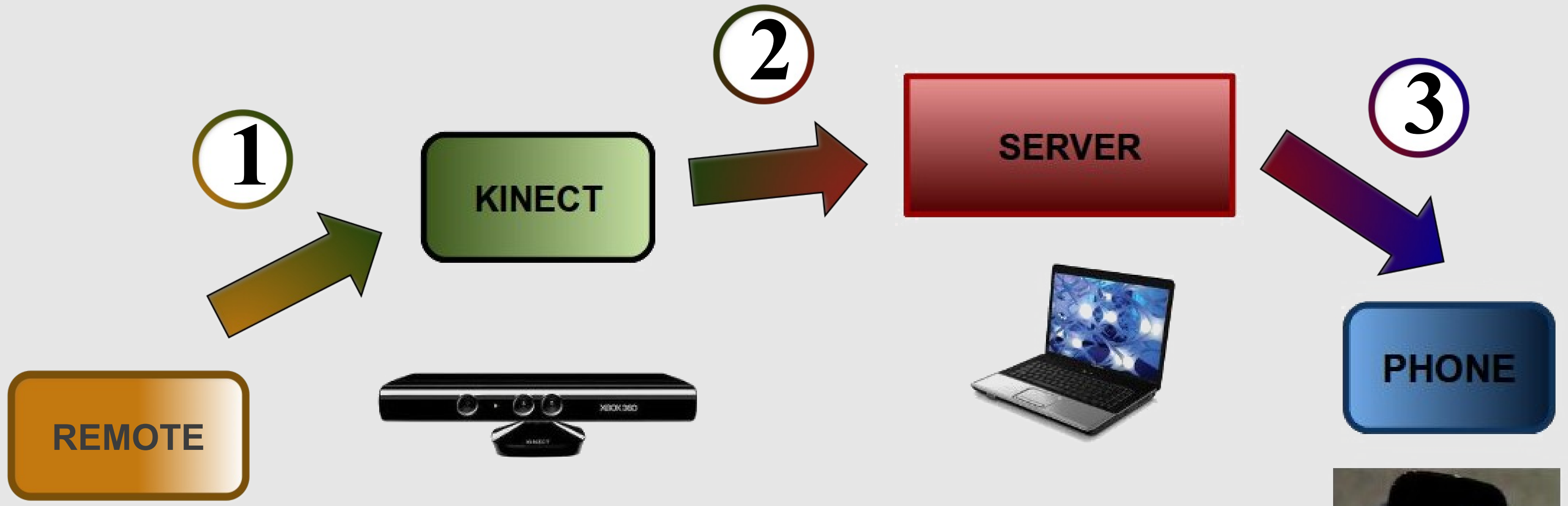
- Languages: C/C++, Java
- 3D rendering using OpenGL
- Android OS

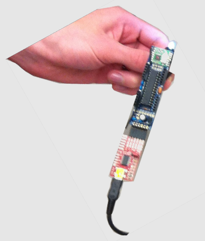
Communication Protocols

- Wi-fi network for phones to share information
- 3G network for phones without Wi-fi
- USB communication between Kinect and server



Architecture





The Kinect detects the bright LED light of our remote/drawing tool, reads the points and packs them into data packets.

Data packets are sent from the Kinect and read by the Server via USB Connection.

The server forwards the packets to the Android phone via Wi-fi connection. The phone reads and unpacks the data packets into points for rendering on its screen.

Test Results

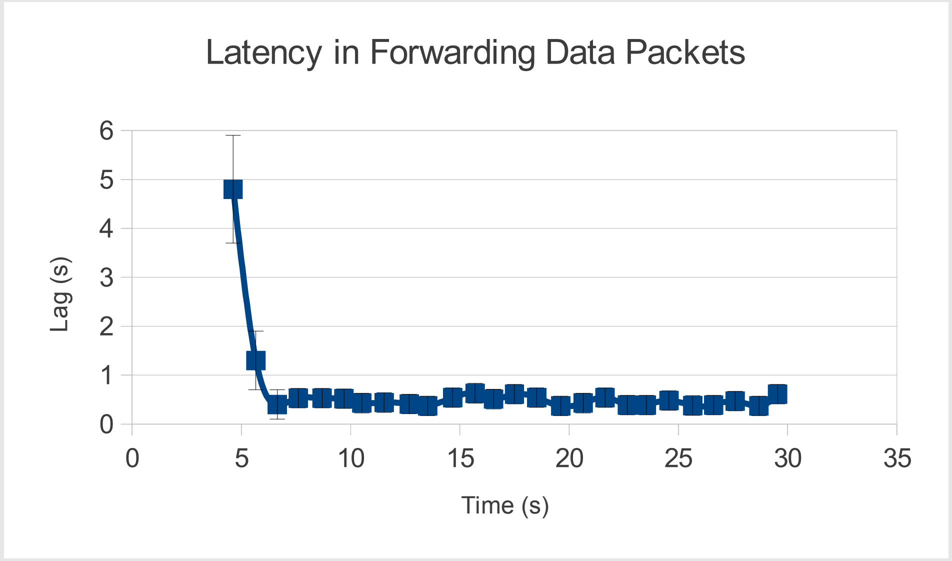
General Latency

- Tracking phone location: ~ 2.3 fps (dev: .26 fps)
- Graphics rendering: ~ 18.4 fps (dev: 2.3 fps)

Kinect: not a limiting factor since it is connected via USB

Latency in forwarding of points from the Kinect to the Android phone

- Initial spike: ~ 4.8s (dev: 1.1s) lag due to the initial set-up of the Android phone and commencement of the rendering of points
- Constant lag: plateaus at ~0.4s (dev: 0.15s) due to sending of data packets over the network



Lessons Learned

- Despite having a project that was easily decomposable into components that we could work on individually, we still got the most done when we worked together
- Network latency is painful
- Optimizing algorithms for embedded processors is painful
- Cell phone hardware progresses quickly