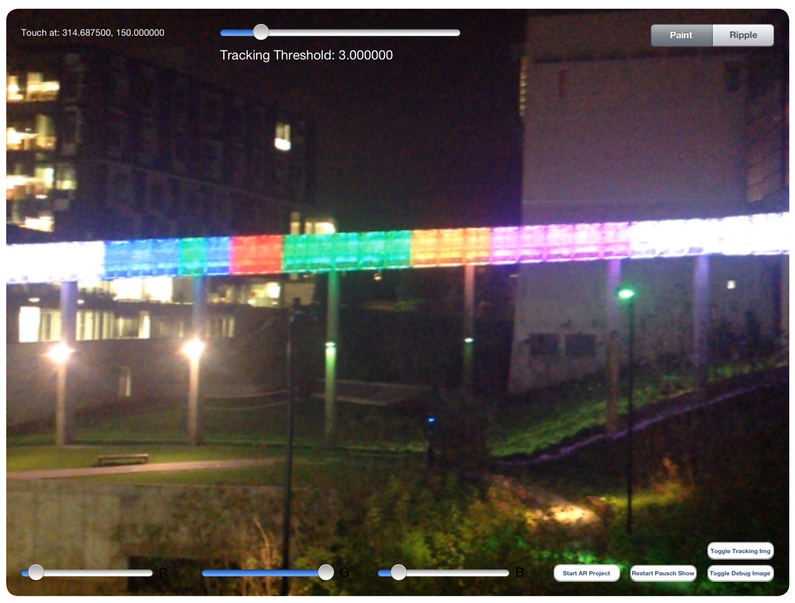
Augmented Reality Interaction with the Pausch Bridge

Evan Shimizu, BCSA '14
Kayvon Fatahalian, Advisor

SURG Grant Provided by IBM

Summary

This project presents an augmented-reality touch interface for controlling the Pausch Bridge lighting. It allows a user to virtually finger paint on the bridge.



System Statistics

- Developed on an iPad (A6X chipset)
- Display runs at 60 FPS, tracking runs parallel at 5-10 FPS
- Two interaction modes:
  - Paint: Drag over bridge with selected color to paint the panel that color
  - Ripple: Touch a panel to start a ripple at that location with the selected color
- Camera resolution: 640 x 480
- Reference Image resolution: 640 x 478

Results

- Fun way to interact with the Bridge
- Display running at 60 FPS created satisfying interaction
- Responsiveness critical to user experience

Future Research Ideas

- Multiple simultaneous users, games
- Integration into a live performance setting
- Creation of new system for programming bridge light shows

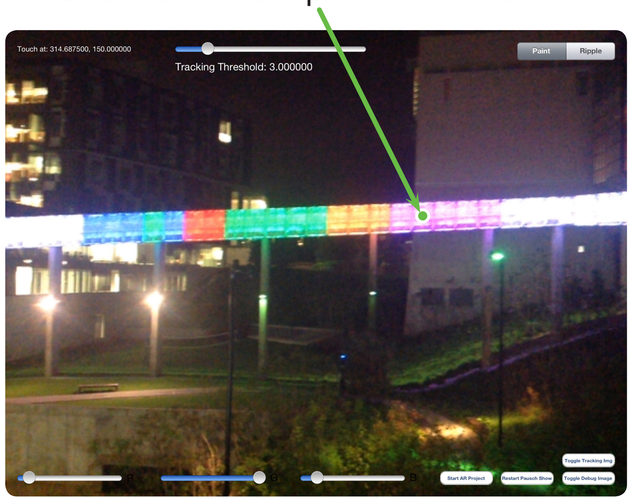
Key Optimizations

- Use of 2D tracking vs. 3D object recognition
- Reference image cropped for stability
- Tracking remains accurate at low resolutions
- Not necessary to link tracking and display rate
- Low tracking rate still provides good experience

System Overview

iPad

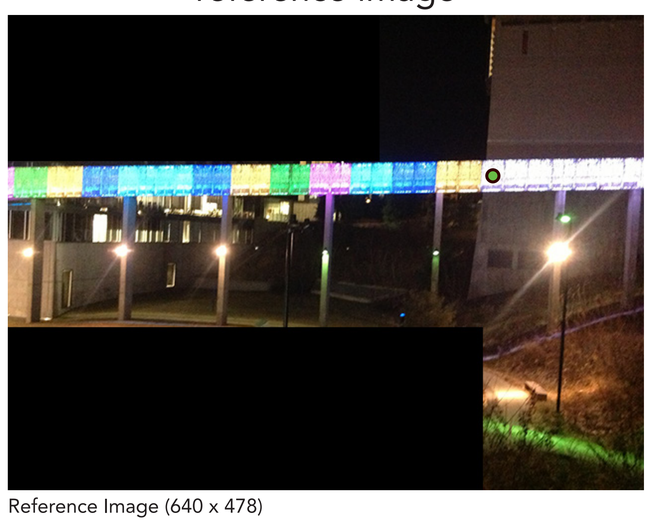
User touches a panel on screen



iPad screen: live video from iPad camera

(touch_x, touch_y)

Touch point is mapped to a point in the reference image



Reference Image (640 x 478)

(ref_x, ref_y)

Reference point is used to look up what panel was hit

Panel ID Map (640 x 478)

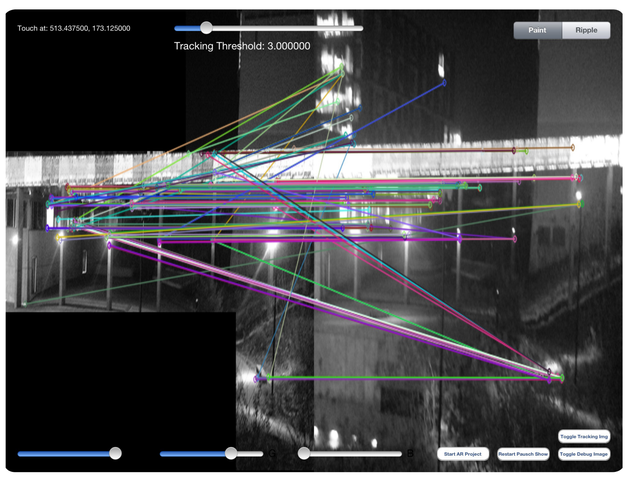
Color indicates ID of corresponding Pausch bridge panel

panel_id

If a panel is hit, the iPad sends a HTTP GET request to a proxy server

```
GET /trigger/51?var=10,255,35,0
```

Command Type
Panel ID
Red Channel
Green Channel
Blue Channel
Fade Time



Alignment of video frame with reference image via feature matching (visualization provided by OpenCV)

The image tracking uses OpenCV's ORB to detect and describe features

Runs in the background between 5-10 FPS, no significant difference in accuracy with slower speed

ORB performed better than OpenCV's FAST, SURF, BRISK and FREAK detectors

GET /trigger/51?var=10,255,35,0

Bridge Lighting Server

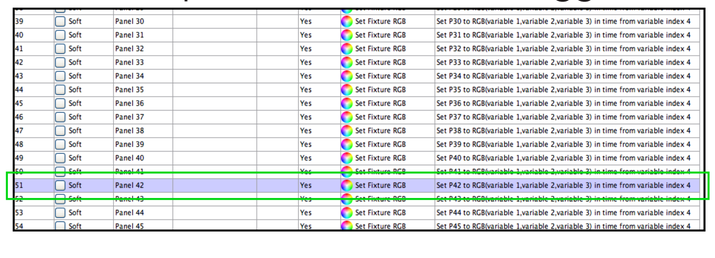
The proxy server receives the request and forwards it to the Pharos Lighting Controller

```
GET /trigger/51?var=10,255,35,0
```

The proxy server is connected to the entire campus network, while the Pharos controller is connected only to the proxy server and the bridge lights

GET /trigger...

The Pharos Lighting Controller receives the request and executes a trigger



[10,255,35...]

The change in color is visible on the bridge and on the screen of the iPad almost instantly

In Ripple Mode, the proxy receives a different request: `GET /ripple/51?var=10,255,35,0` and will send out a series of commands at timed intervales