Affordable Off-The-Shelf Augmented Reality in Python

Thomas Perl
http://thp.io/2013/europython/

2013-07-04 – EuroPython 2013, Florence, Italy
About Me

Thomas Perl
http://thp.io/
m@thp.io
@thp4
http://github.com/thp
Hardware

- **PS Move Motion Controller**
  - USB (pairing, charging) and Bluetooth
  - Accelerometer, Gyroscope, Magnetometer
  - Glowing RGB ball, 8 digital buttons, 1 analog trigger

- **PS Eye Camera**
  - USB 2.0, 4 microphones (not used here)
  - 640x480 @ 60 FPS, 320x240 @ 120 FPS
6DoF (six degrees of freedom)

- 3-axis position
  - tracked with OpenCV via camera, sphere

- 3-axis rotation
  - tracked with AHRS algorithm via inertial sensors

“Blinking Calibration”

1: LEDs on
2: LEDs off
3: Difference image
4: Thresholded diff

Use 4 as mask for 1, average color of biggest blob = color of sphere in camera
Tracking Algorithm (2/2)

Sphere Size Calculation

Find the two points A, B in the blob with the maximum distance

Line from A to B

Center of sphere: Center of Line

Diameter of sphere: Length of Line
In a Nutshell, PS Move API...

...has had 414 commits made by 11 contributors representing 27,550 lines of code

...is mostly written in C++
    with a well-commented source code

...has a young, but established codebase
    maintained by a large development team
    with increasing Y-O-Y commits

...took an estimated 7 years of effort (COCOMO model)
    starting with its first commit in March, 2011
    ending with its most recent commit 8 days ago
Hands-On
How to use the API in Python
Let's write an AR game in Python

- “Whack a cube”
  - Grid of 3x3 cubes floating in space
  - Cubes light up randomly, hit to score
Whack a cube: Design (1/2)

- Highlight happens randomly
- Maximum number of highlighted cubes
- Timeout (before highlight disappears) also random
- Minimum time between two consecutive hits
Whack a cube: Design (2/2)

• Collision detection using distance
• Time split into “ticks” (20 ms)
• Rendering: Camera image + colored cube
Whack a cube: Implementation

- Focus on API usage and AR, not visuals
- **Button**: “Whackable cube”
  - Highlight state, position, hit handling
- **Highlighter**: Picks button for highlight
  - Also takes care of maximum highlights
3D Rendering

- Model-View Matrix: **Controller** (6DoF)
- Projection Matrix: **Camera** Projection

- Placing objects on the controller sphere
  - Object at origin (x=0, y=0, z=0)
  - Apply Model-View Matrix

Image source: songho.ca
OpenGL Shader Pipeline (1/2)

![Diagram showing the OpenGL Shader Pipeline]

Image source: developer.apple.com
OpenGL Shader Pipeline (2/2)

Image source: developer.apple.com
Hands-On
Whack a cube
Performance

- Vision Tracker Frame Rate (tracking 1 controller)
  - $\sim 50 - 68$ FPS

- End-to-End System Latency
  - Initially: $60$ ms (+/- $3$ ms)
  - While tracking is in progress: $\sim 15$ ms

- Maximum Sensor Update Rate:
  - $\sim 87$ updates / second (hardware limit)
More Info

http://thp.io/2013/europython/

Project
http://thp.io/2010/psmove/
http://code.google.com/p/moveonpc/

Thesis
http://thp.io/2012/thesis/

Google Summer of Code 2012