**Kinects in Unity**

**UCSB Computer Science Capstone 2013**

**The Hex Pistols**

**SPECIAL THANKS TO**
- Chandra Krintz UCSB
- Tim Sherwood UCSB
- Janet Kayfetz UCSB
- Victor Luo JPL
- Garrett Johnson JPL

---

**The Problem**

- Field scientists need to be able to safely and conveniently interact with the environments they study.
- Unlike earth geologists, it is not easy for planetary geologists to travel beyond our planet to study other celestial bodies.
- The Varrier Autostereoscopic Virtual Reality Display is an example of a virtual reality system that displays a first-person motion-tracked perspective via multiple displays.
- JPL currently has an equivalent infrastructure working, but employs extremely expensive equipment and bulky devices.
- Our goal was to develop an inexpensive and intuitive way to present a virtual window perspective of a 3D environment.

**Astrogeologists vs. Geologists**

- Field scientists need to be able to safely and conveniently interact with the environments they study.
- Unlike earth geologists, it is not easy for planetary geologists to travel beyond our planet to study other celestial bodies.

**Our Solution**

- Create a system that transforms displays into virtual windows that look onto a virtual environment.
- Take advantage of the Microsoft Kinect and its included Software Development Kit (SDK).
- Provide a self-calibrating system that allows variable placement of displays (virtual windows).

**The Data Flow Process**

1. Using skeletal, depth, and color (S/D/C) data from the Kinect SDK, recognize and establish the placement of the screens and user in a unified coordinate system (UCS).
2. Establish communication through network interfaces (between the Kinect processor machine and each Unity 3D display rendering machine) to promote scalability of the VR system.
3. Load placement and user position data based on the UCS into our custom Unity3D game rendering engine scripts.
4. Using linear algebra (matrices and vectors), calculate projection matrix data set for each display at each frame.
5. Render the virtual window display as each S/D/C frame from the Kinect is updated (at 30 frames/second).

**The Technology**

- Using skeletal, depth, and color (S/D/C) data from the Kinect SDK, recognize and establish the placement of the screens and user in a unified coordinate system (UCS).
- Establish communication through network interfaces (between the Kinect processor machine and each Unity 3D display rendering machine) to promote scalability of the VR system.
- Load placement and user position data based on the UCS into our custom Unity3D game rendering engine scripts.
- Using linear algebra (matrices and vectors), calculate projection matrix data set for each display at each frame.
- Render the virtual window display as each S/D/C frame from the Kinect is updated (at 30 frames/second).

---

**The Team**

- Anthony Narsi
- Sea Pong
- Kevin Sheridan
- Boyang (Jerry) Peng
- Alex Scarlett

---

**Data Flow Diagram**

- KINECT SENSOR(s)
- LARGE FORMAT DISPLAY
- KINECT SDK
- UNITY 3D
- SCENE CONTROL STREAM(s)
- MULTI KINECT PROCESSING & COMMUNICATION LIBRARY