The Zynga Tunnel

Keith Rasmussen
Isotropic Laboratories

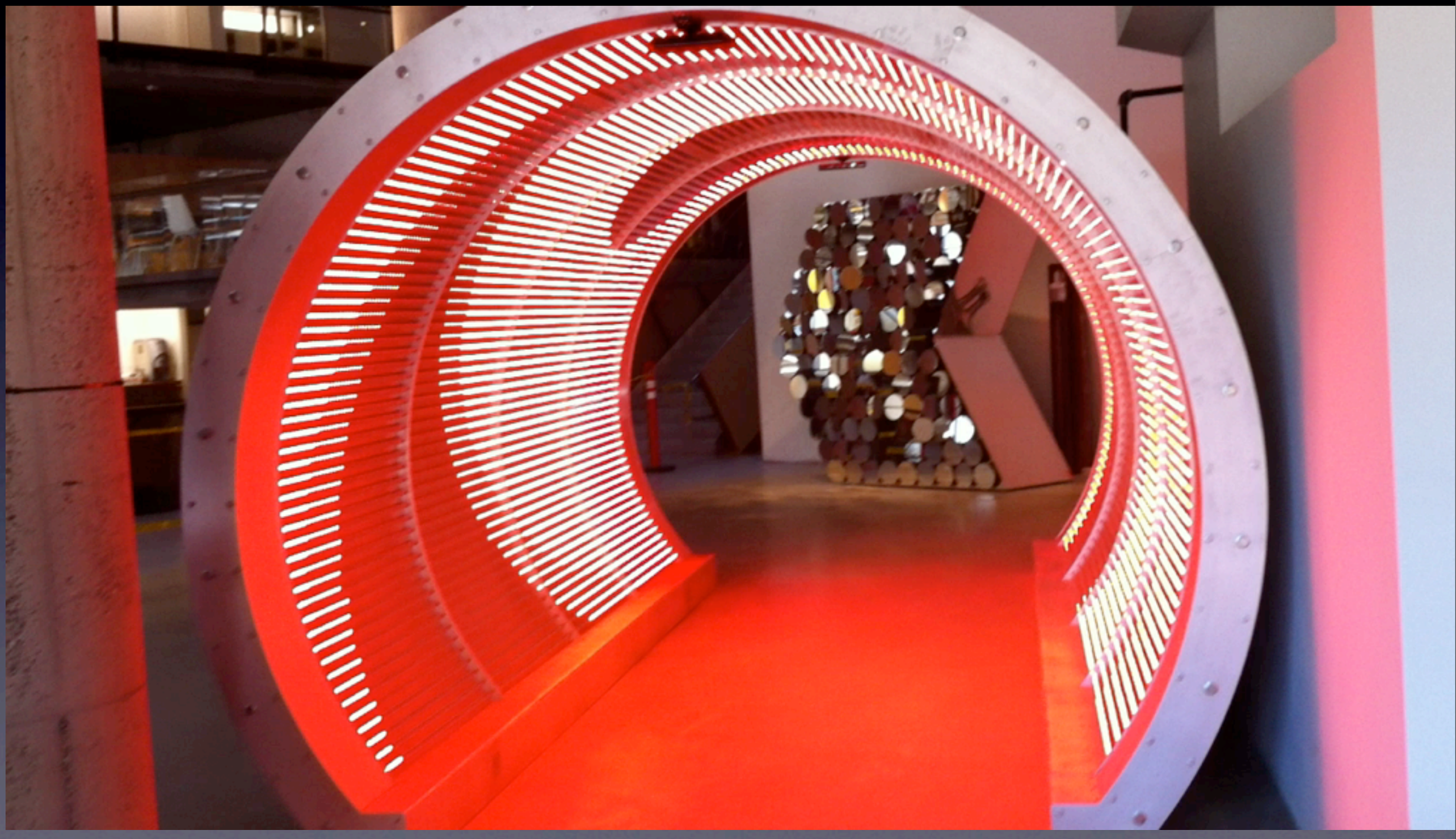
What Is It?

- 157 x 128 Fully Addressable RGB Display
  Refresh Rate >= 30 fps, 24 bit color
- 4 Speaker Stereo Sound
- 2 Kinect Sensor Arrays
- Dedicated Mac Mini
- Intranet Connectivity???

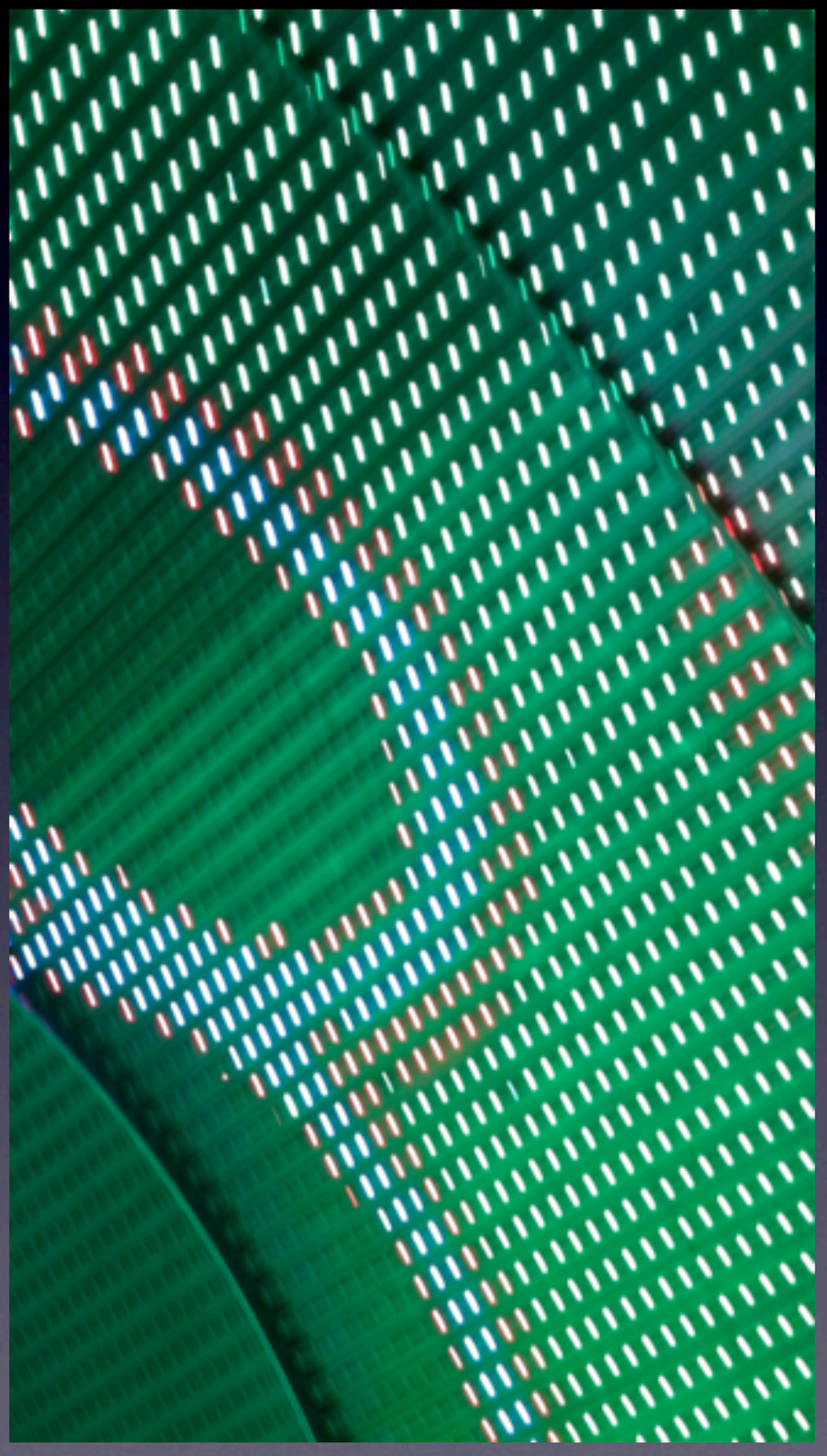
What does it mean?

- Transition? Portal? Gauntlet?
- It's what the Transient wills it...
- And...it's an opportunity for the programmer to *seed* the experience...

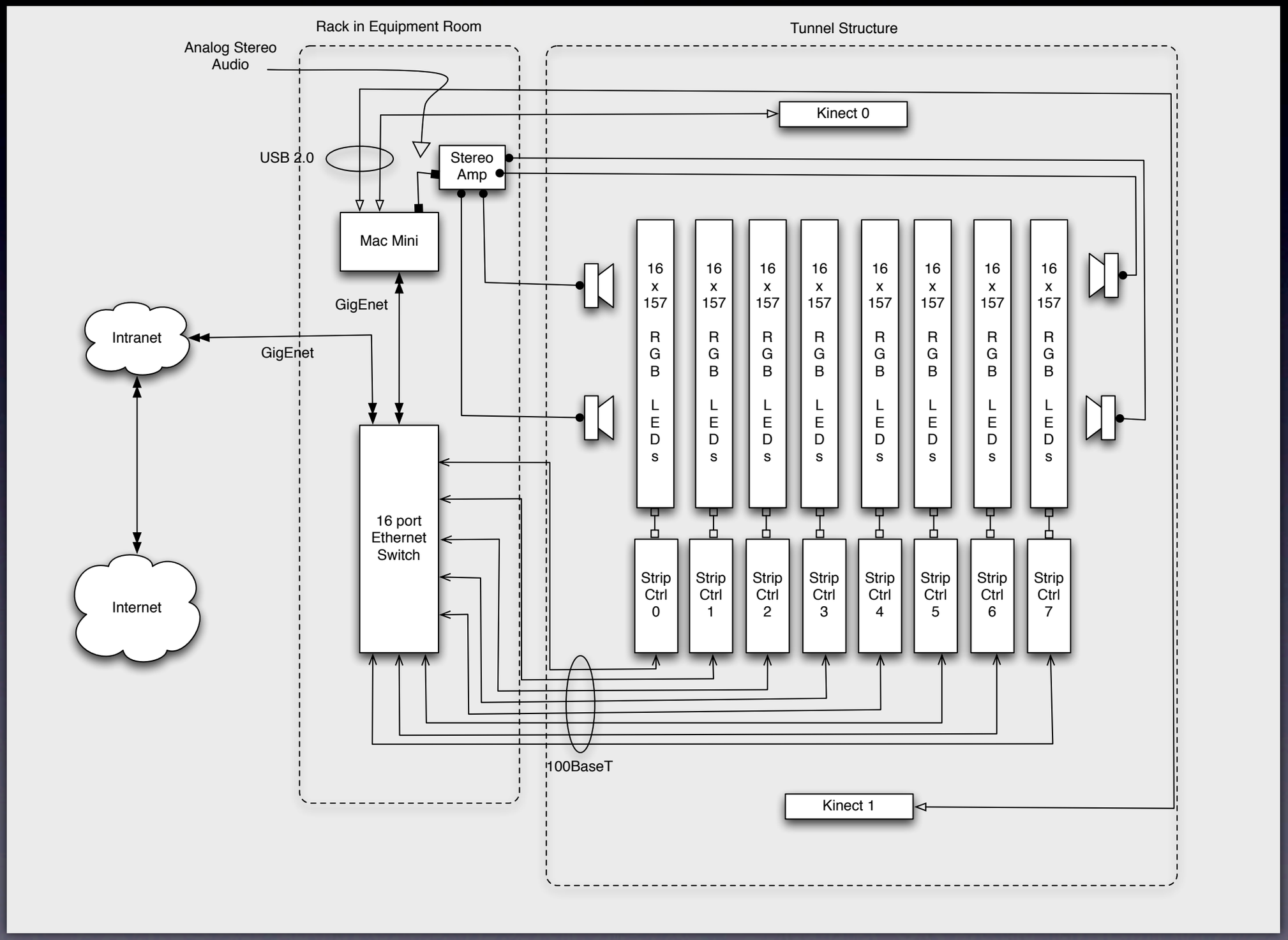
Algorithmic



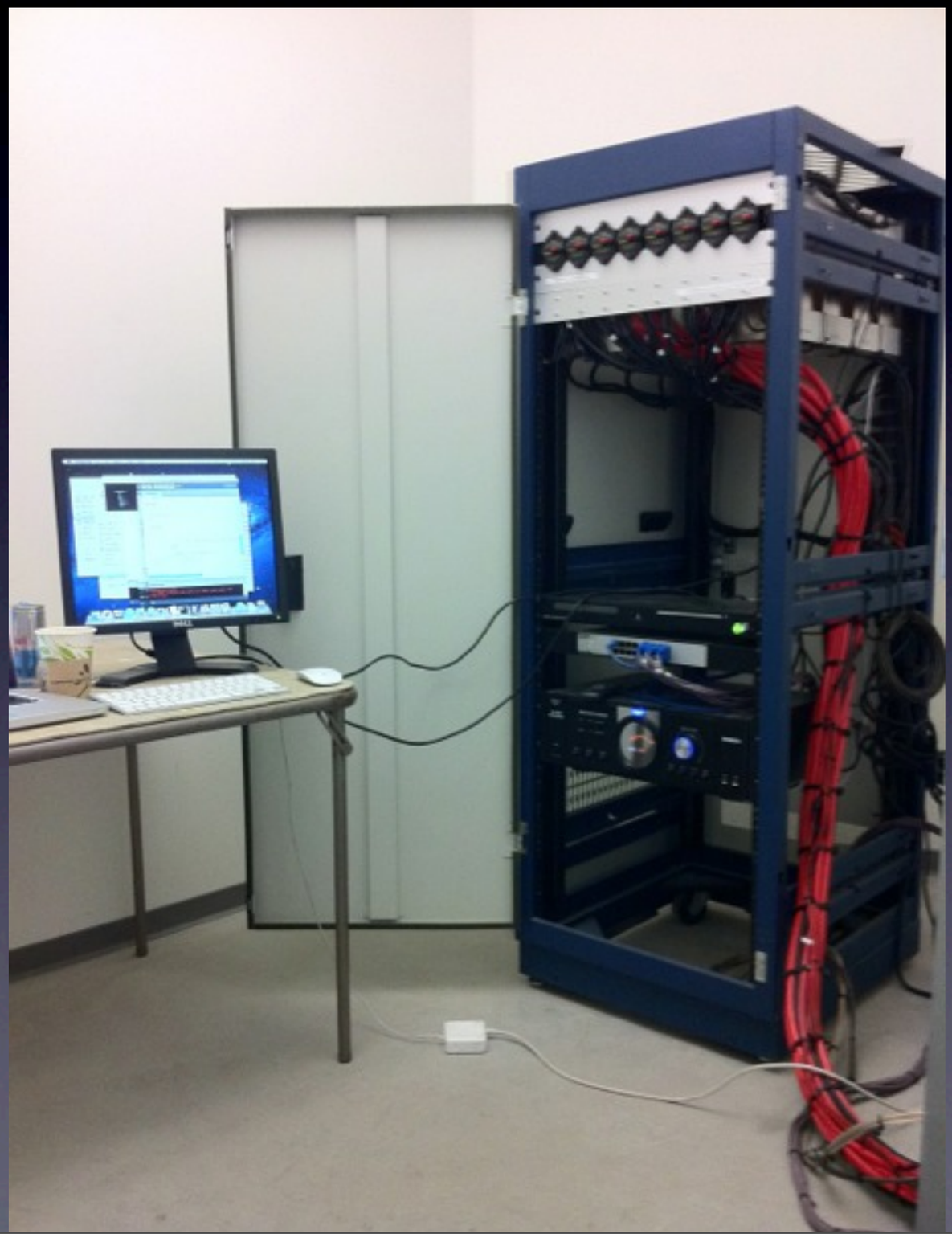
Quicktime



Block Diagram



The Rack...



Mechanics - The Rack

- Equipment Rack in Room Adjacent to Tunnel
- Eight 5 Volt, 100 Amp Power Supplies, each with Thermal Breaker
- Each 16 Strip Section Completely Independent - Power & Control
- 16 port GigEthernet Switch (must support Jumbo UDP Packets)

Rack Power Supplies

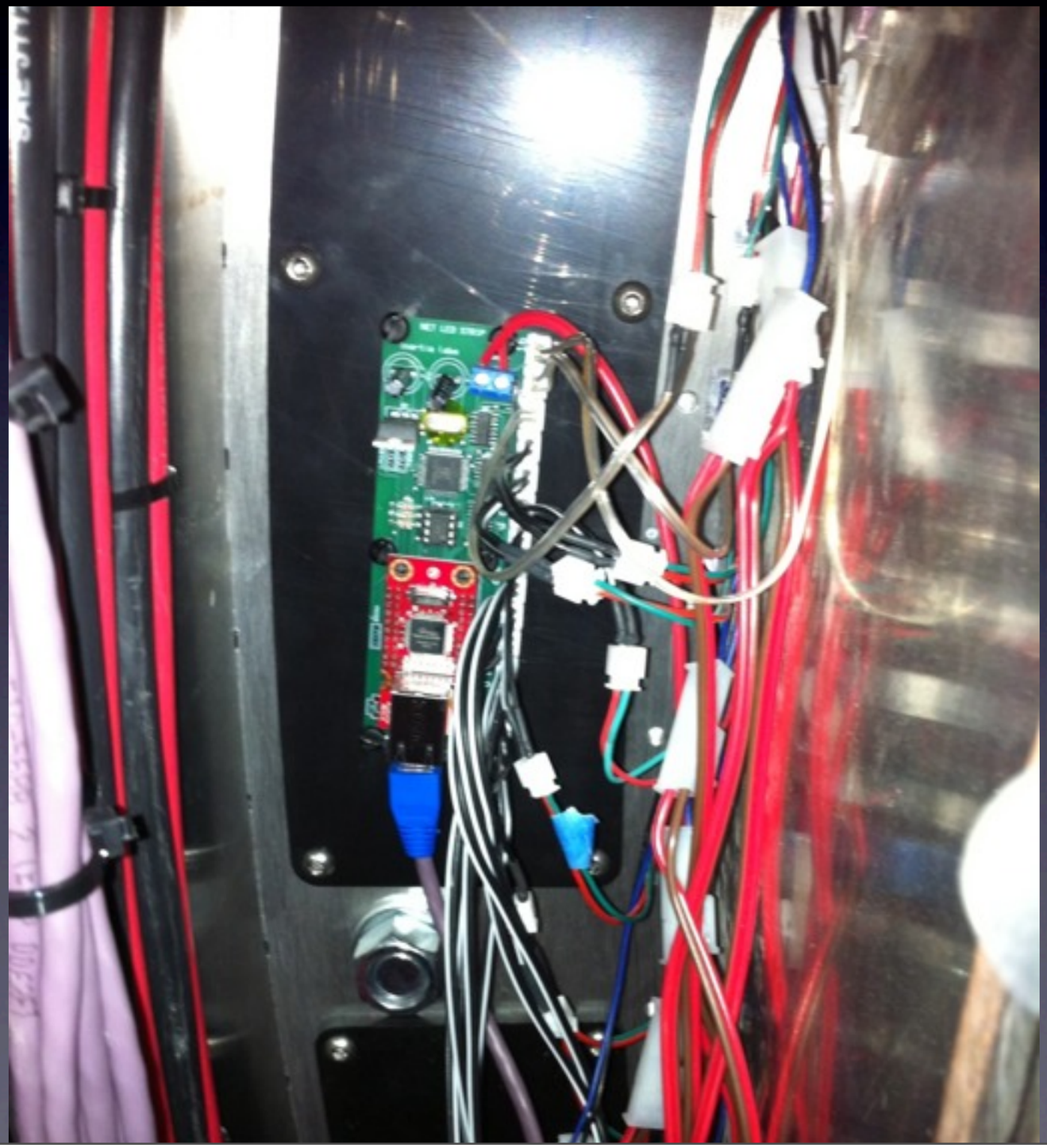
The Rack (continued)

- Rack Mounted Mac Mini (2.5GHz dual-core i5 with 4GB DDR3) running OSX.7
- Right-hand USB port on Rack Face available for USB Drives, etc.
- 1000W Stereo Amp [overkill] with Mac audio out connected to Tape input (drives 4 4” speakers mounted in the tunnel kick)

LED Strips

- Each Horizontal LED Strip Composed of 157 RGB LEDs
- Each RGB LED controlled by individual WS2801 IC, does 8 bit PWM for each color (R, G, B)
- Programmed serially from Strip Controller

Strip Controllers



Strip Controllers

- 8 Strip Controllers each handle 16 LED Strips (128 strips total)
- 8 Core 80 MHz Propeller + Wiznet Ethernet Chip + Drivers
- Receives Jumbo UDP packet with 24 bit RGB data for 16 strips (157 x 16 x 3 bytes)
- Serializes and shifts data to the 16 strips
- Fixed IP Addresses

UDP Packet Format

Protocol ID (value = 0)	4 bytes
Seq # (increments)	4 bytes
LED data 'B,G,R' Strip 0, col 0 Strip 0, col 1 ... Strip 15, col 156	3 x 157 x 16 bytes

IP Addresses and Port

- Strip controllers addressed consecutively, starting at 192.168.1.177 for strip controller 0 (16 strips on bottom South)
- Controllers listen to Port 6000

Who cares? Gimme code!

```
class TheTube {
        // Globals
        int ledsPerStrip = 157;
        int bytesPerLed = 3;
        int bytesPerStrip = ledsPerStrip * bytesPerLed;
        int numStripsPerPacket = 16;
        int numStripsPerSystem = 128;
        int bufSize = bytesPerStrip * numStripsPerPacket + 4 + 4; // Strip data
                                                                  // (471*16) + 8
                                                                  // byte UDP
                                                                  // header)+ 4
                                                                  // bytes type +
                                                                  // 4 bytes seq.
        byte[] buf = new byte[bufSize];
        // String ip = "192.168.1.177"; // the remote IP address base
        String[] ipaddr = { "192.168.1.177", // stripctrl0 remote IP address
                            "192.168.1.178", // stripctrl1
                            "192.168.1.179", // stripctrl2
                            "192.168.1.180", // stripctrl3
                            "192.168.1.181", // stripctrl4
                            "192.168.1.182", // stripctrl5
                            "192.168.1.183", // stripctrl6
                            "192.168.1.184" }; // stripctrl7
int destPort = 6000; // the destination port
int srcPort = 6000; // the source port
long seq; // tx packet sequence number
int secs = 0; // elapsed seconds
long timeStamp = 0; // Timestamp
int effectIdx = 0;
byte r = 0;
byte g = (byte) 0xFF;
byte b = (byte) 0xFF;

UDP udp; // define the UDP object

TheTube() {
        udp = new UDP(this, srcPort); // create a new datagram connection on
                                      // port 6000
        // udp. log( true ); // <-- print out the connection activity
        print("UDP Buffer Size: ");
        println(UDP.BUFFER_SIZE);
        udp.listen(true); // and wait for incoming message
        seq = 1; // seq # starts at 1
        for (int i = 0; i < bufSize; i++) {
                buf[i] = (byte) 0xFF;
        } // set pattern in buf
}

void copyToTube() {
        // this will load pixels from the drawn image into the "pixels" variable
```

The guts...

```
void copyToTube() {
    // this will load pixels from the drawn image into the "pixels" variable
    loadPixels();

    int ipidx = 0; // index into array of IP addresses for strip controllers

    for (int lineidx = 0; lineidx < numStripsPerSystem; lineidx += numStripsPerPacket) {
        int pixelIdx = ledsPerStrip * lineidx;

        for (int i = 8; i < bytesPerStrip * numStripsPerPacket + 8 - 1; i += 3) {
            color curPixel = pixels[pixelIdx];
            buf[i] = (byte) blue(curPixel); // Blue
            buf[i + 1] = (byte) green(curPixel); // Green
            buf[i + 2] = (byte) red(curPixel); // Red

            pixelIdx++;
        } // set pattern in buf
        // Put a type 0 in the first long to signify this is a Strip
        // Data packet
        buf[0] = 0;
        buf[1] = 0;
        buf[2] = 0;
        buf[3] = 0;

        // put a sequence # in the next 4 bytes of buf (little endien)
        buf[4] = (byte) (seq & 0xFF);
        buf[5] = (byte) ((seq & 0xFF00) >> 8);
        buf[6] = (byte) ((seq & 0xFF0000) >> 16);
        buf[7] = (byte) ((seq & 0xFF000000) >> 24);
        udp.send(buf, ipaddr[ipidx++], destPort); // the message to send
    }
    seq++;
}
```

Really, why all the low level stuff?

- Drive it from another Platform
- Drive it with your Language / Environment of Choice
- Drive it from the Cloud

What about Movies?

- Quicktime is easy in Processing (see TestMovieGS sketch for sample code)

  BUT...

- Un-preprocessed Home Movies will look like crap!
- Experiment!

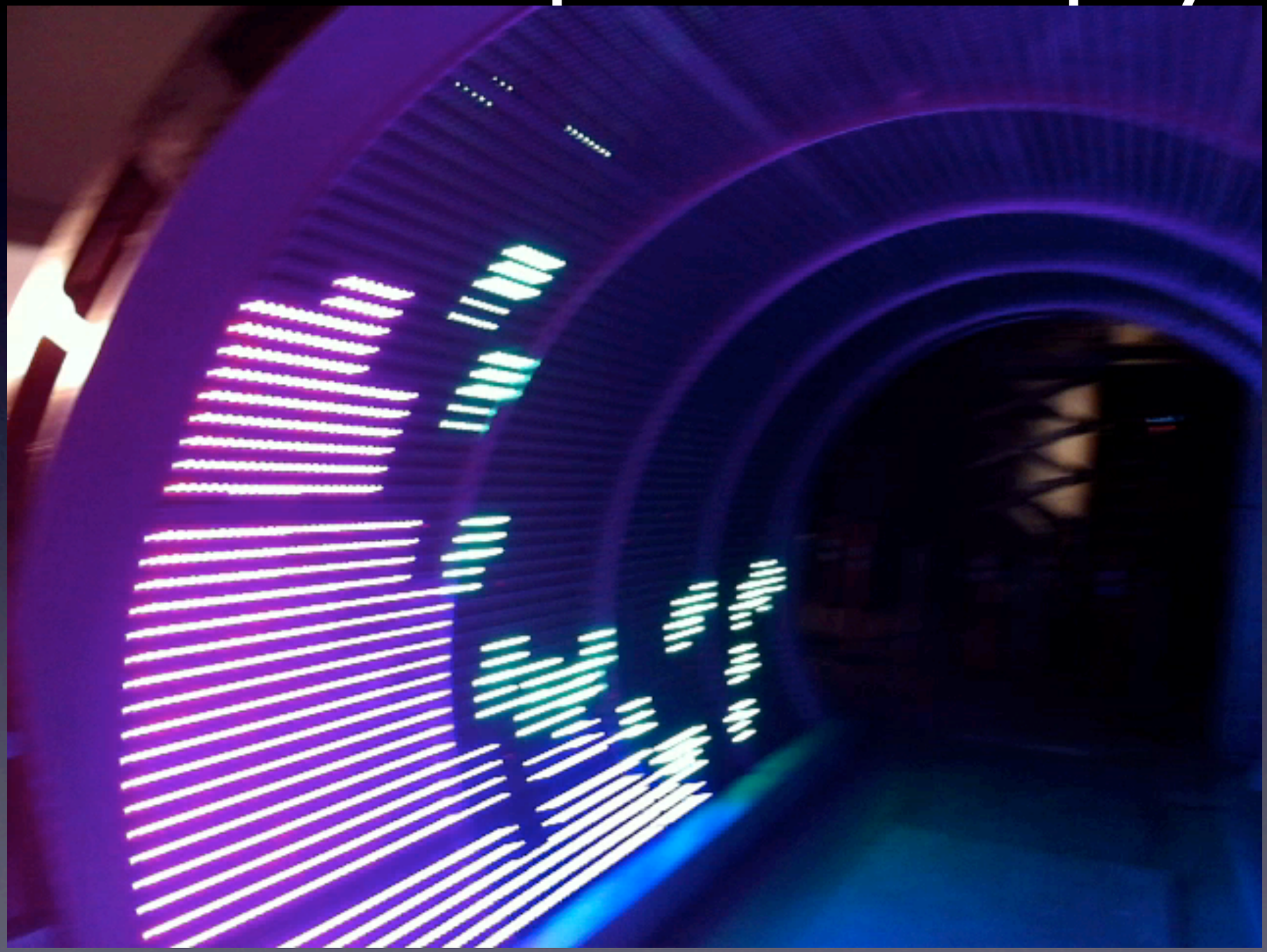
Other Movie Sources

- Cinema 4D
- After Effects
- Maya
- Code mixers, overlays, chroma keys...
- Mix or layer with algorithmic images...

What about Audio?

- Headphone Jack Out from Mac Mini
- Sample Processing code (tunnel_sim_udp_all) uses MiniM, FFT, BeatDetect to play music and warp visuals to music
- Volume Control on Amp is NOT under program control :(

FFT based Spectrum Display



Kinects?

- 2 Kinects, one at each edge of the tube, looking inward
- Both connected to Mac Mini USB ports

- Processing Libraries:
  org.openkinect.*
  org.openkinect.processing.*
- Other Languages Supported

Kinect Possibilities...

- Track masses down the tunnel...
- Interactive Visuals...
- Games?!
- Musical Instruments...

Intra and Internet Interaction

- IT in the house?
- Interaction with public data: Stock Price, Weather, etc.
- Interaction with protected data: Badge Reader, Customer Traffic Stats, Canine Accidents, roofCam, whatever...

Software Architectures

- Jukebox?
- Rules-based?
- Game Engine?
- Distributed Control???
- Experiment!

...and Now...

...it's YOURS.

Have fun, Play!

Code and Slides at:
http://public.me.com/keithrasmussen

keith@isotropiclabs.com