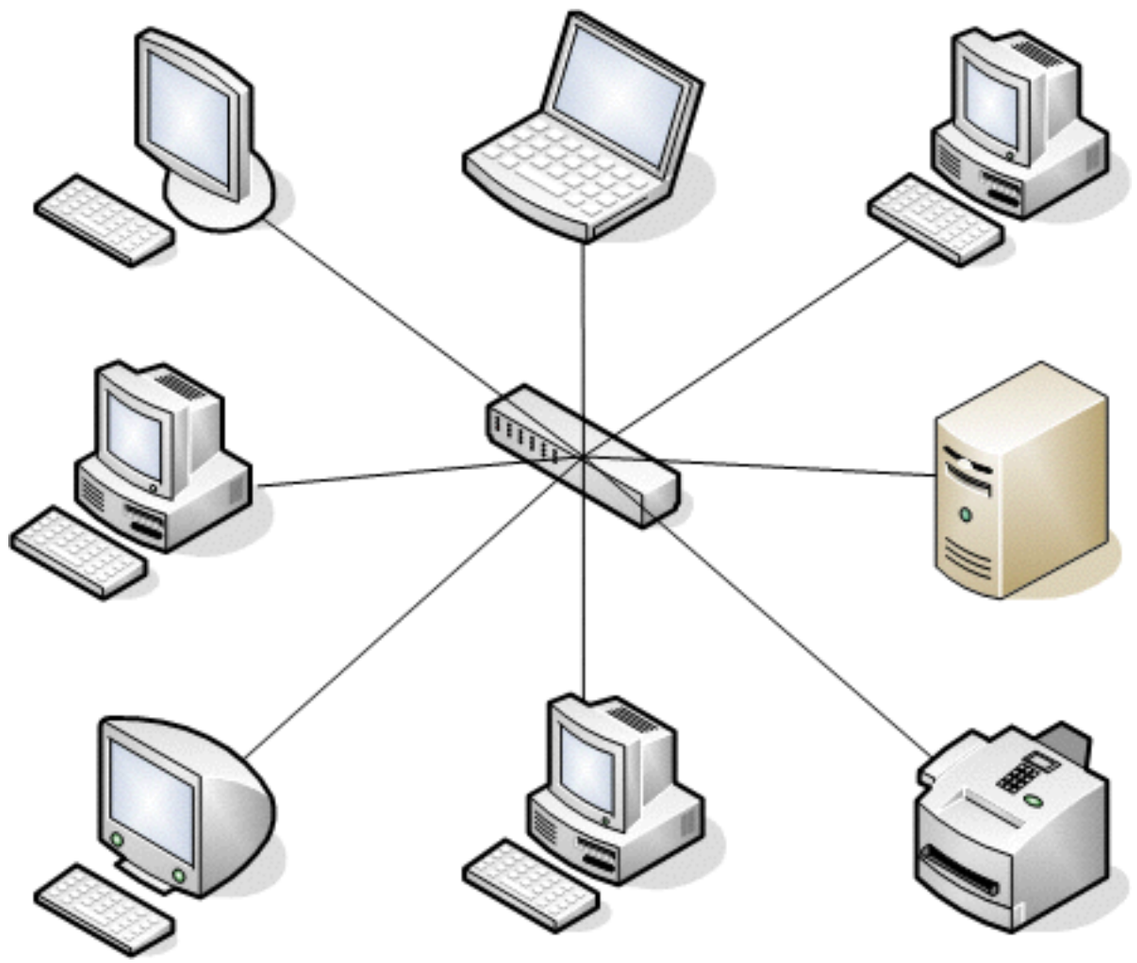

# Gluing Apps & Computers Together with Open Sound Control (OSC)

Dan Wilcox
Visiting Teaching Assistant Professor
Emergent Digital Practices

Winter 2016

class.danomatika.com



OSC is for when you need to send messages between apps and/or computers

(GIF: The City - on Creative Applications)

# Ex: A/V performance using multiple interfaces across multiple computers

OSC was originally developed at Berkley CNMAT for sharing musical performance data (1997)

spiritual successor to MIDI (1983)

“Sound” is part of name, but OSC has become a common method for inter-app comm in the creative community

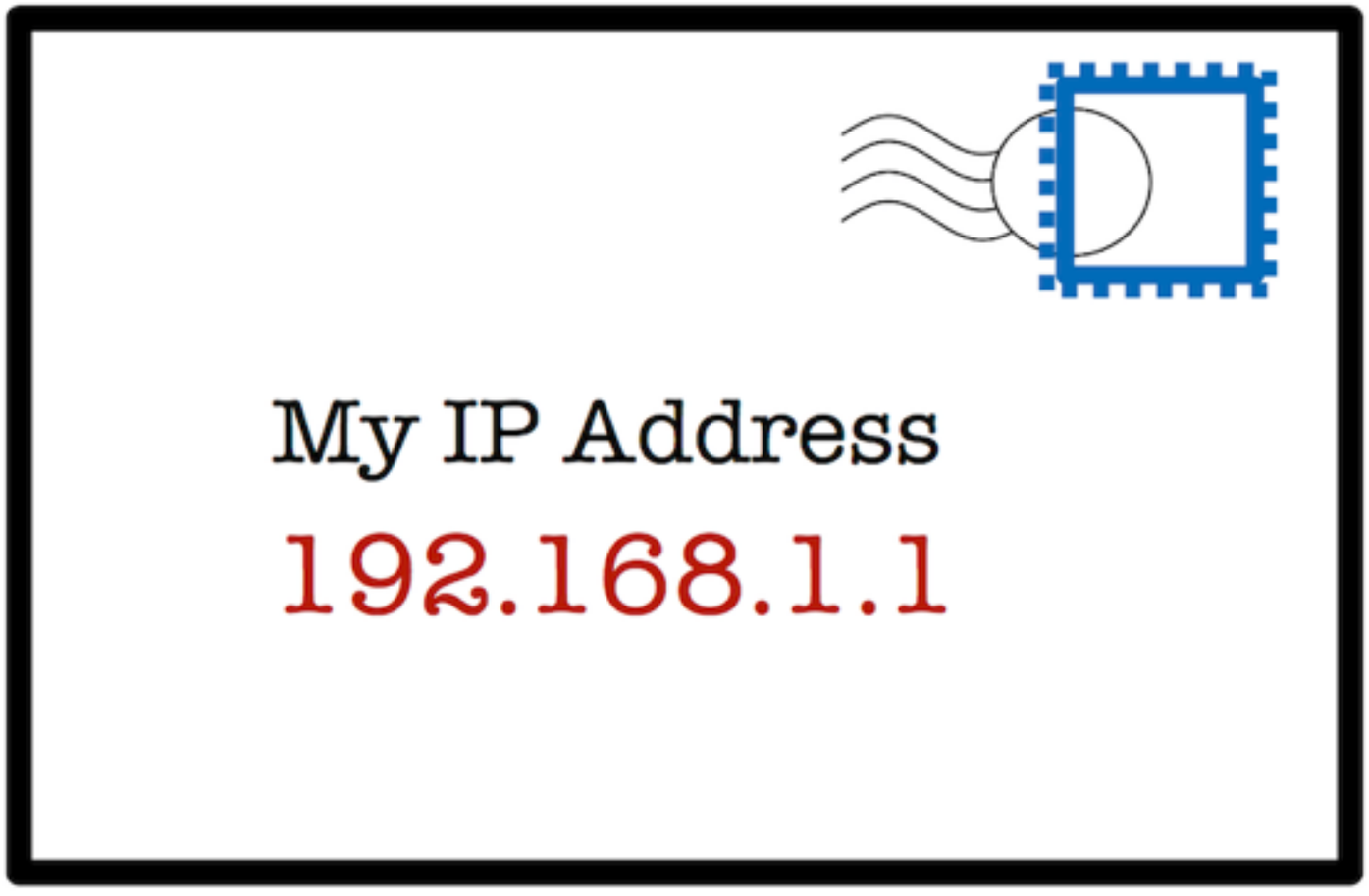
“just another networking data scheme”

uses IP/UDP over Ethernet

you don’t need any special cables or hardware

works over wireless networks too

# Networking 101




every computer has an address, you send messages back and forth

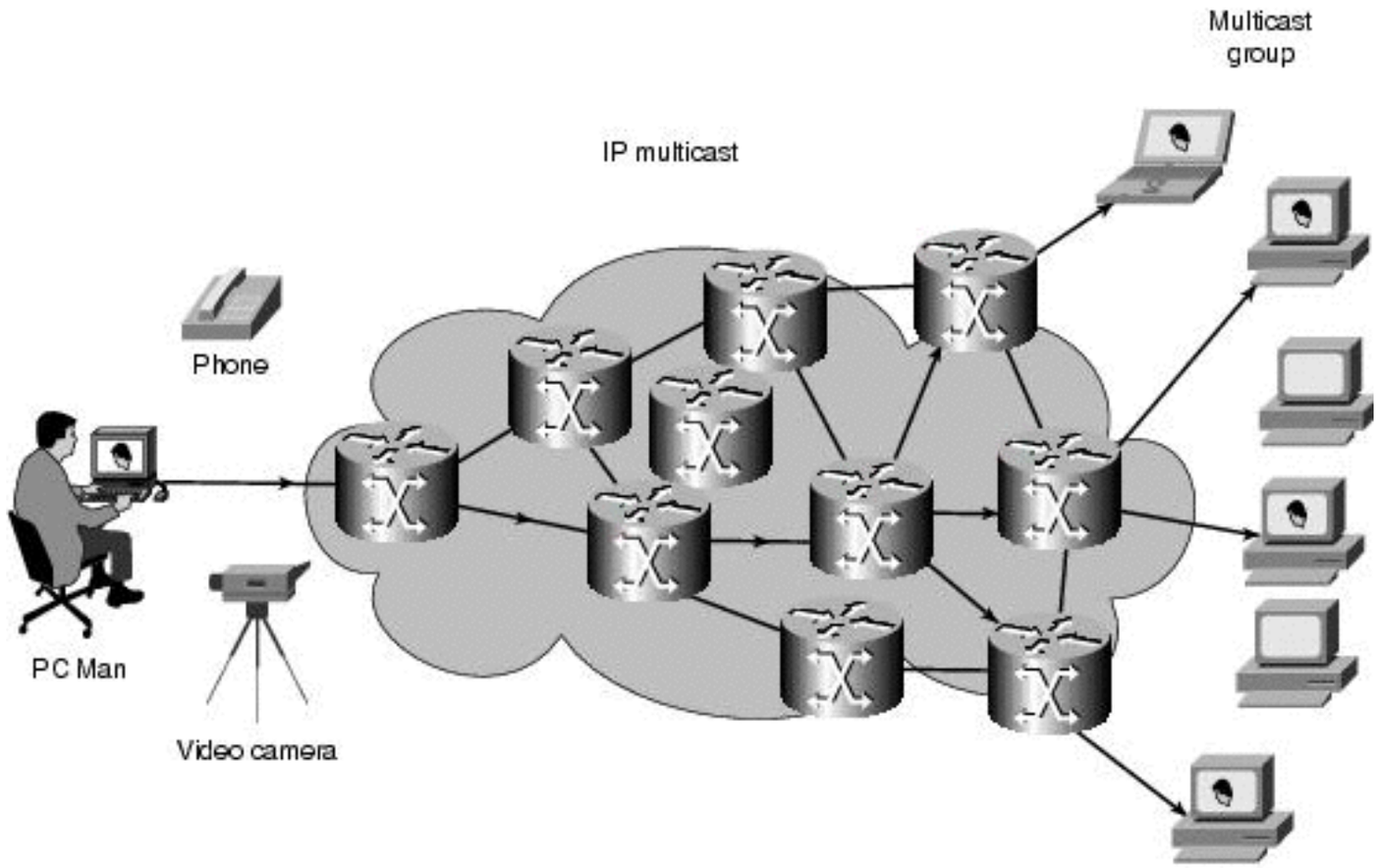
# Networking 101


we can send messages internally between applications via the ethernet loopback address: 127.0.0.1

# Networking 101


we can send messages to other computers if we know their addresses on the network (or over the internet)

# Networking 101




we can also broadcast messages to all computers on a local network, this is called *multicasting*

# OSC Basics

messages
&
bundles of messages

# OSC Basics

Message:

address

&

arguments

basic OSC communication object

# OSC Basics

Address:

address pattern is kind of like a URL

*/synth/knobs/volume*

you can organize any way you want

*/hello/world/foo/bar*

# OSC Basics

Address:

you can group things together

*/video/1/brightness*
*/video/1/speed*
*/video/1/volume*


# OSC Basics

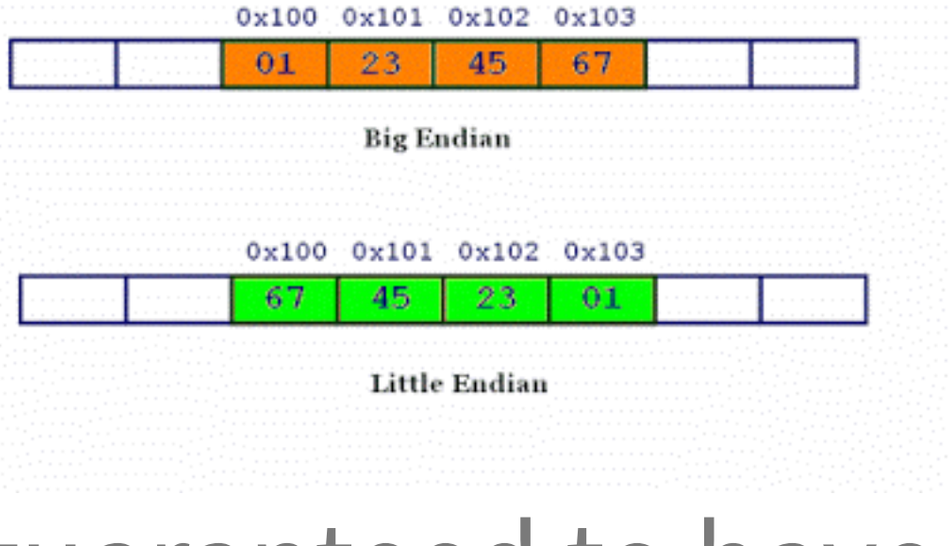
## Arguments:

just a list of data types after the address

*/hello 1 1.23 “hello”*

int, float, string, midi bytes,
timestamp, binary data “blob”, etc


# OSC Basics




types are guaranteed to have the same values on the other side

even if receiver has a different bit order (endianness) or int size (32 vs 64 bit)

# OSC Basics

Bundle:

timestamp
&
message

a bundle of messages, allows you to send a group of messages at once

# OSC Basics

Timestamp:

when the message in the bundle was sent

used to keep correct order if messages arrive at different times

# OSC Libraries

OSC is an open standard

numerous libraries implement this spec:

liblo (C)
oscpack (C++)
CocoaOSC (Obj-C)
etc…

# OSC Interfaces

creative apps/toolkits use OSC libraries
to provide OSC interfaces:

Processing (via oscP5)
OpenFrameworks (via ofxOsc)
Max/MSP (CNMAT OSC object)
PureData
etc…

# OSC Interfaces

OSC has also been picked up by commercial VJ applications for remote control:

VDMX
Modul8
Isadora
etc…

# OSC Devices

OSC is also used for sending data from hardware controllers and A/V devices:

Monome (grid controller)
AudioCubes (tangible controller)
Kyma (DSP workstation)
Milkymist One (live video effects)
etc…


# OSC Apps

you don’t have to write your own software to work with OSC

numerous apps map UIs & input devices to OSC

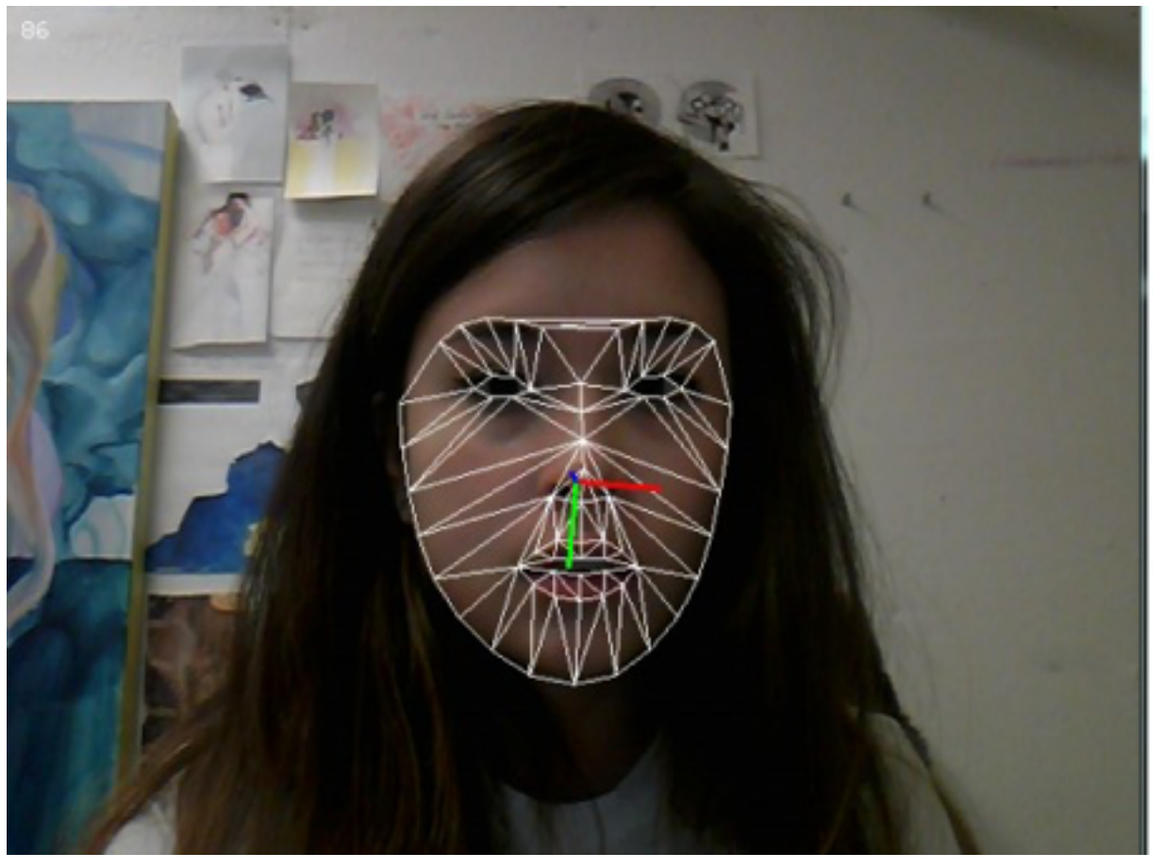
# OSC Apps


TouchOSC: mobile UI/control

# OSC Apps


Osculator: input device events

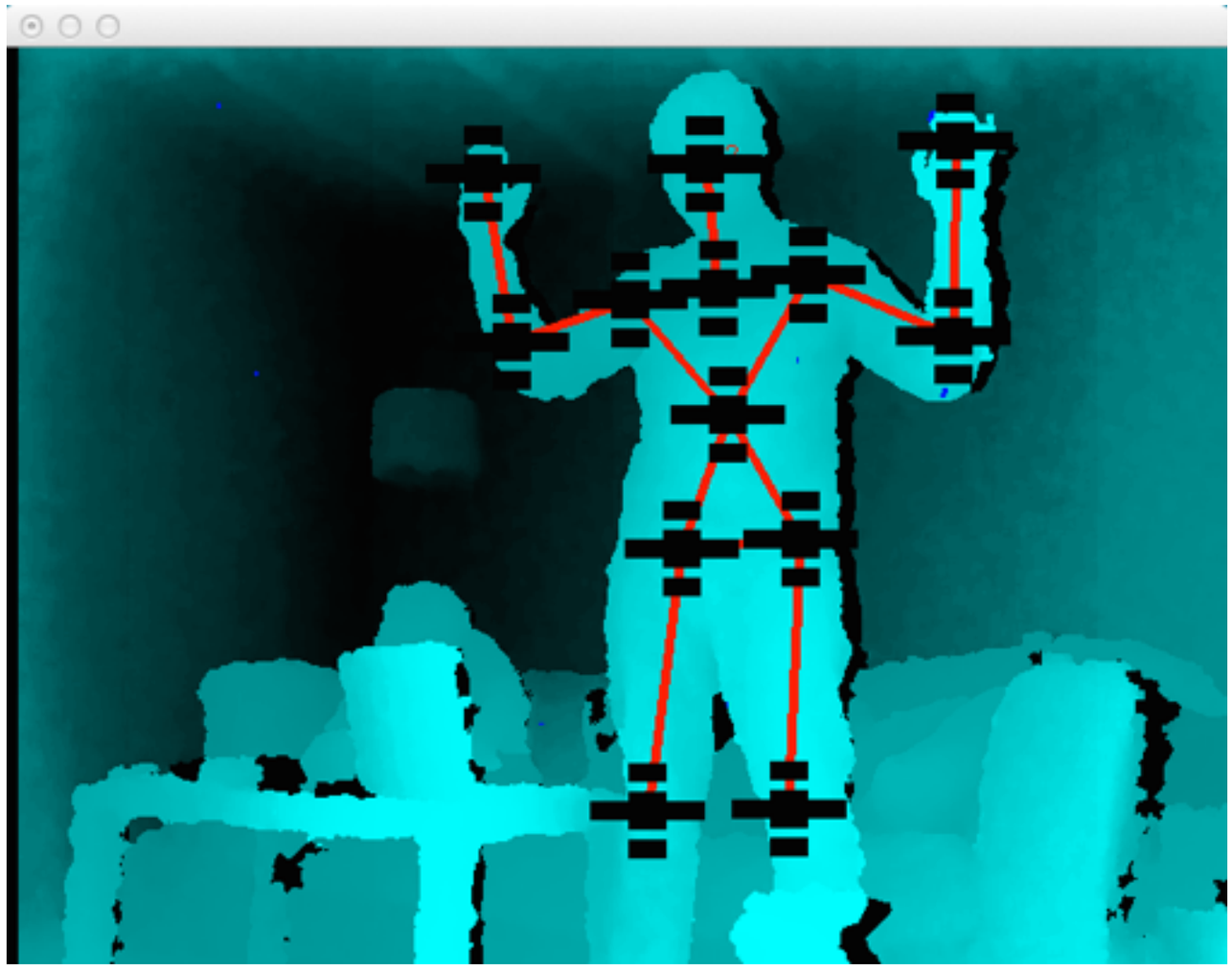
# OSC Apps




(Face OSC Monster Mash - Alex Wolfe)

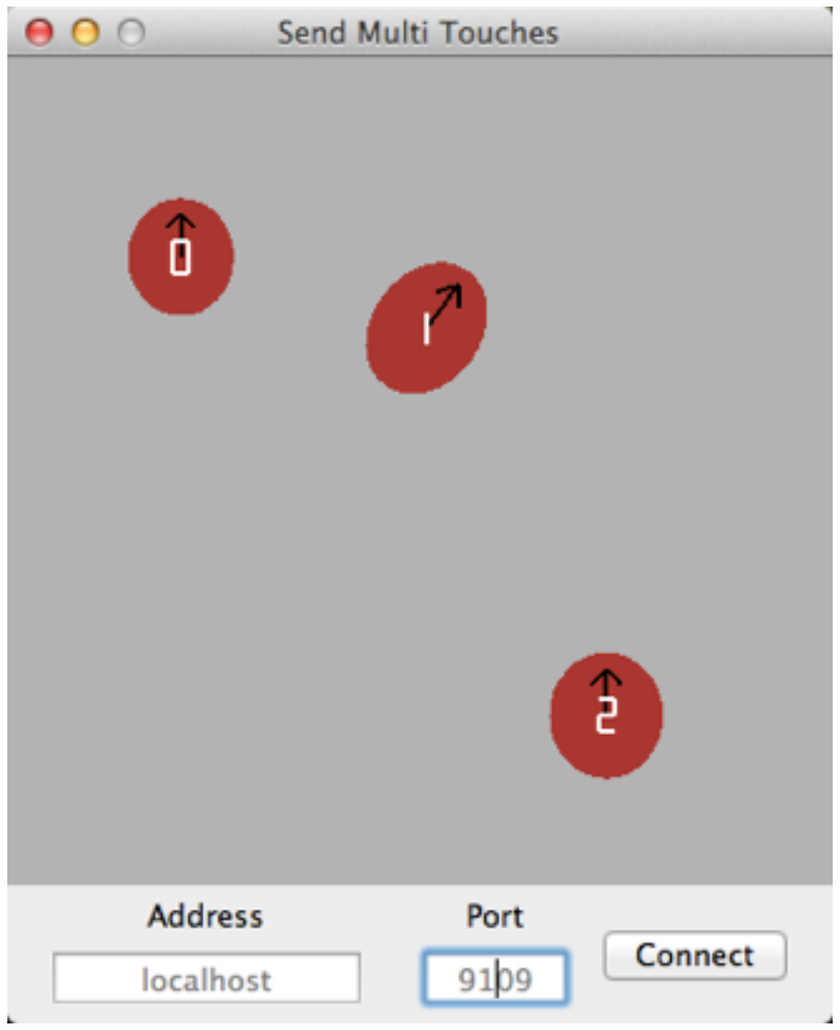
## FaceOSC: face tracker

# OSC Apps



Synapse: Kinect skeleton tracker

# OSC Apps

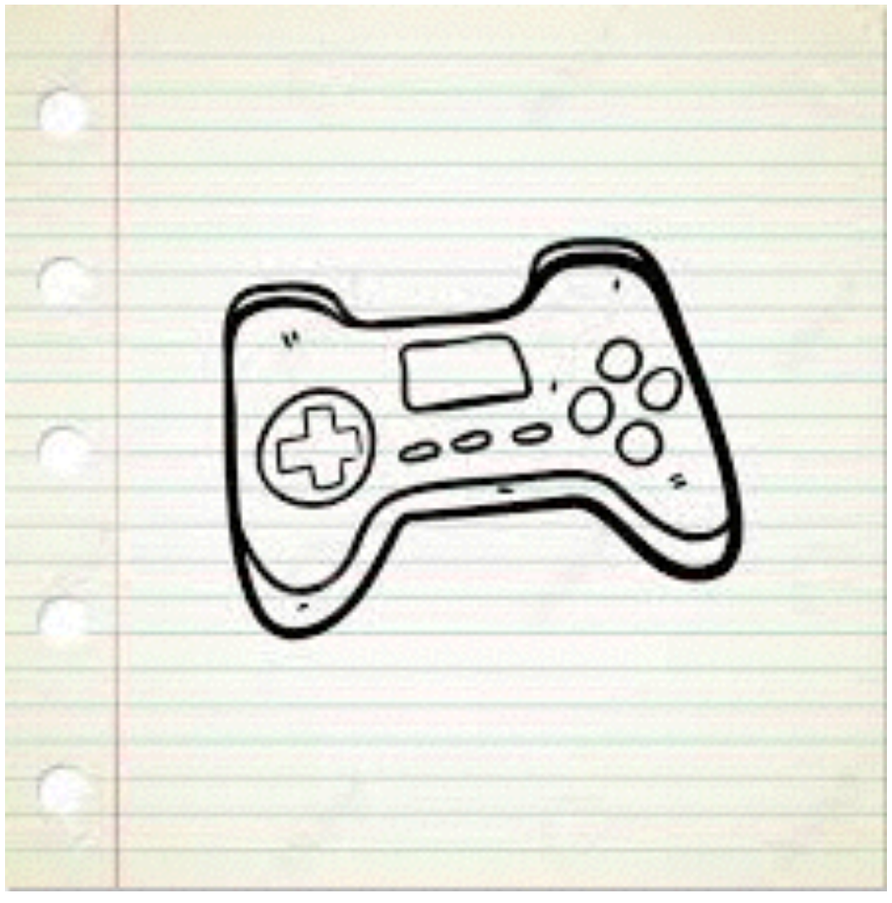



Send Multi Touches: Mac trackpad

# OSC Apps


QDTracker: simple Kinect head tracker

# OSC Apps



joyosc: gamepad events

# DEMO TIME