wiiMote Headtracking and Laser Detection in Python

–John Harrison
–Insight Industries LLC
–
–Projects at:
–www.insightvr.com
–
–Utah Code Camp: April 26, 2008

Many Thanks To:

My Background

- Programming since 3rd grade
- BS in CS from Stanford
- 8 years with IBM
- Co-founder Insight Industries
  - We do data visualization and business intelligence software
- Like to write games as a hobby

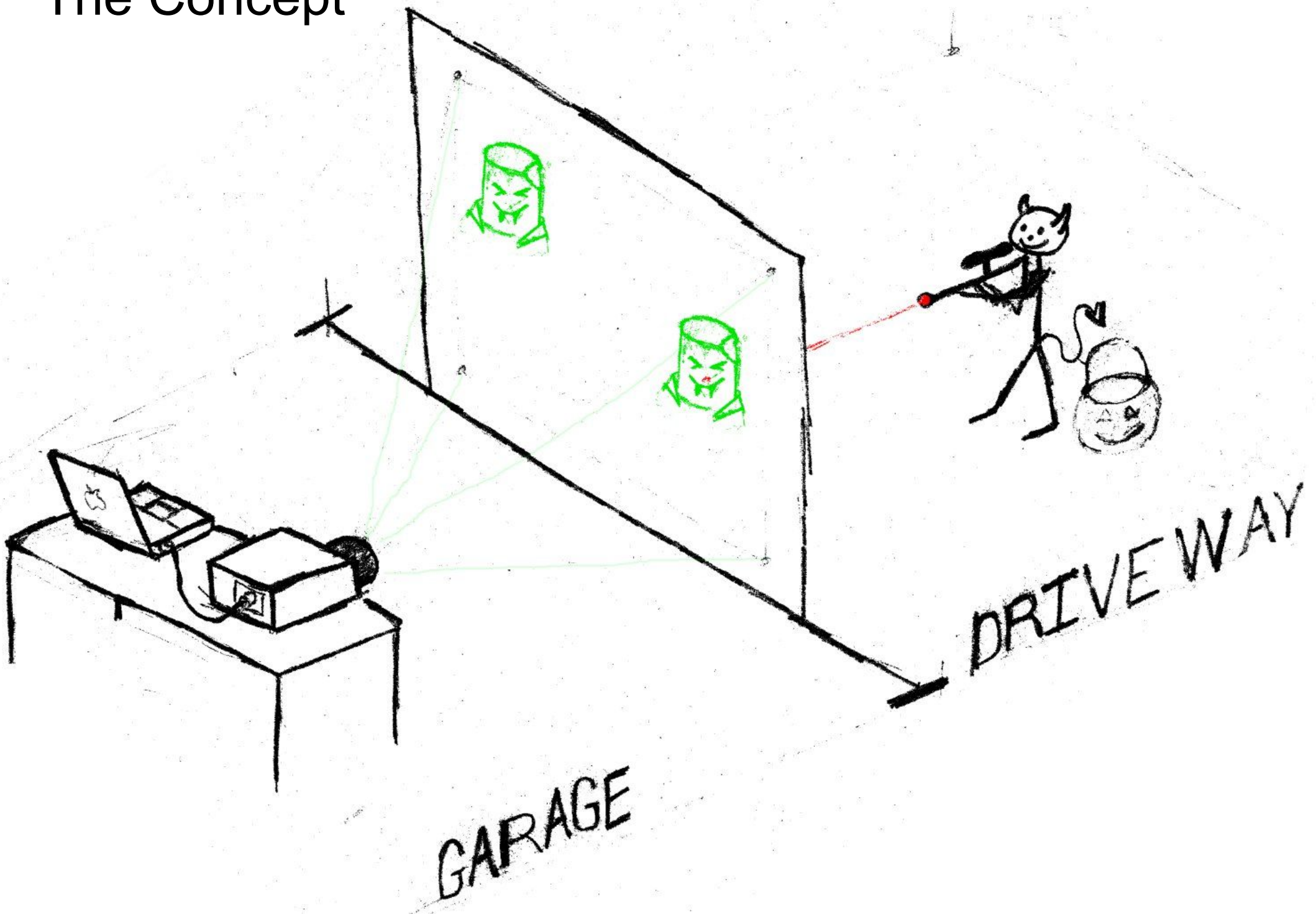
Agenda

- Have Fun
- Inspirations, Concepts, and Warnings
- Simple Laser Detection
- Better Laser Detection
- wiiMote Basics
- wiiMote Headtracking
- Putting it all together

The Inspirations

- Graffiti Research Lab's Laser Tag Project
  - http://youtube.com/watch?v=DKbtTPYZEig
  - Local copy
- H/Malloween
  - http://homestarrunner.com/malloween.swf
  - Local copy

The Concept



You'll Shoot Your Eye Out Kid!!

A Christmas Story

Laser Safety

- Safety Disclaimer: Lasers can be dangerous. I am not an expert. I bought low power (<1mW) laser pointers for Halloween as I was handing this to kids. I had no need for more power.
- Many laser pointers are <5mW. This is powerful enough to cause eye damage if you stare at it but your blink reflex should prevent accidents.
  - http://www.osha.gov/SLTC/laserhazards/
  - http://en.wikipedia.org/wiki/Laser_safety

The Problem

- What I had:
  - MacBook Pro with iSight
  - Laser Pointer
  - A garage
- What I lacked:
  - Projector
  - Screen
  - Software to access iSight
  - Game to play
  - Any clue about Python

PySight

- Easy Access to iSight
- Wraps CocoaSequenceGrabber

PySightTest

- PySight comes with PySightTest and is a simple and functional iSight app
- Has an event loop that gets bitmaps
- SequenceGrabberTest

Looking for Red in the PySightTest Event Loop

- Given RGB values, what is red? Two tests:
  - 1st test is simple and is performed on all pixels: red must meet a threshold value which is 2/3rds of the highest red seen so far
  - 2nd test only happens if 1st is passed:
    - redness = red*2 – green – blue
- Pixel with highest score for 2nd test is what we're interested in
- If nothing meets 1st test then there is no result

PyGame

- PyGame comes with an aliens.py demo that did roughly what I wanted – basically simple space invaders using arrow keys and space bar
- Modified to change how the aliens move

Error!

- I should have made a mouse based interface to speed testing without using laser and PySight

Linking the Two Programs

- Shared global variables used to communicate:

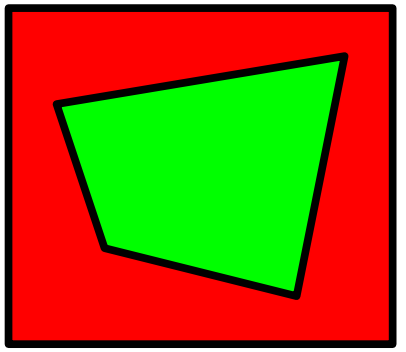
```
X = 0
Y = 0
SHOT = False
TOP = 0
BOTTOM = 480
LEFT = 0
RIGHT = 640
```

Performance Concerns

- Scanning is slow
  - Scanning every pixel of every frame makes the game too slow to play
- Only scan every 5^{th} frame
- Use calibration to limit scanning area
- Only scan 1 in 4 pixels

Calibration Thoughts

- Need to transform iSight coordinates to screen coordinates
- Derived a 4 point calibration that would perform an arbitrary transform

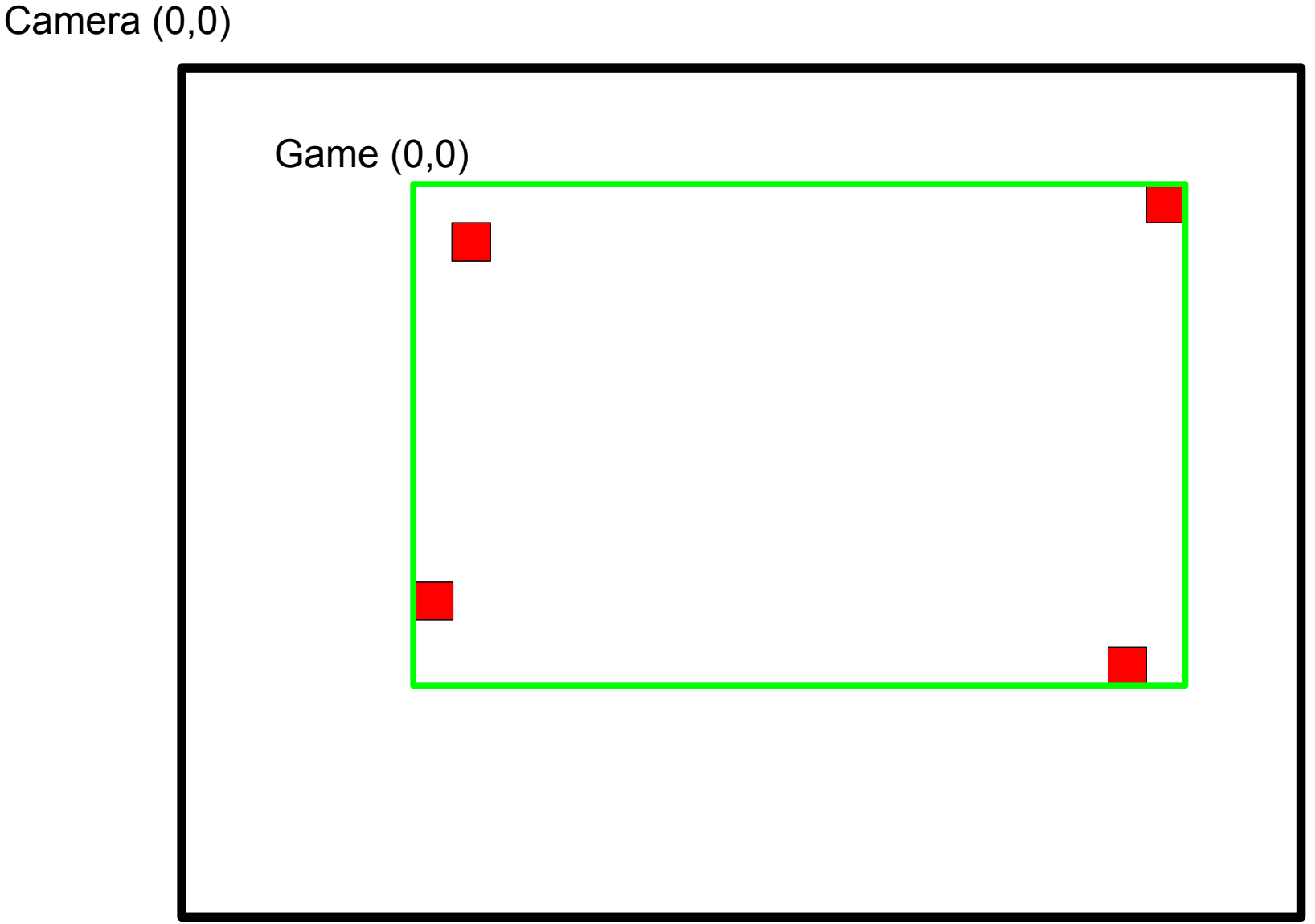


- Decided to try implementing a stupid simple method first
- Worked well enough since camera is roughly aligned with projector so I stuck with it

Simple Calibration

- Game puts a red dot in a corner, waits for iSight to “see” it and then put a dot in the next corner
- Top & bottom y camera coordinate saved
- Left & rightmost x camera coordinate saved
- Assume that image of screen is rectangular
- gameX = gameWidth * (cameraX – left) / (right - left)
- gameY = gameHeight * (cameraY – top) / (bottom - top)
- Note that camera code only scans between top & bottom, left & right, improving performance

Callibration Illustration



Constructing Laser Guns

- PVC Gun
  - Laser pointer with the button taped down in some PVC from Home Depot
  - Radio Shack switch worked poorly
- Star Wars Gun
  - Laser module put into normal gun in place of LED
  - Hardest part was getting the gun open without breaking it

Where I Bought Lasers

- Low powered pointers:
  - http://www.laserpointer.net/
- Giveaway lasers:
  - http://www.surpluscomputers.com/
- Modules:
  - http://mfgcn.com/

Demo

- Marshie Attacks

Results

- It works and was a big hit at Halloween
- Needs to be dark to work with low powered lasers and a bedsheet
- Inherently multiplayer
- Works best if you aim for a dark area
- Response is slower than I'd like
- Firewire DV cam: plug it in and it just works

Defects

- Tracking is lacking
  - Doesn't scan every frame (1 in 5)
  - Doesn't scan every pixel (1 in 4)
  - Only detects one laser at a time
- All of this is due to the fact that my method is slow
- Game is somewhat lame

Faster Laser Detection

- NumPy gives the ability to do matrix operations in Python with the speed of c
- Still need to use it carefully
- Requires a different way of thinking than iteration

Code

```
def process_buffer(self, image_buffer, rgba=4, flip_y=False, bgra=False):
    global MUL_ARRAY
    # convert from buffer type to numpy array
    flat_array = numpy.frombuffer(image_buffer,numpy.uint8)

    # convert from a (640*480*4) x 1 matrix to a (640*480) x 4 matrix
    image_array =numpy.reshape(flat_array,(NUM_PIXELS,rgba))

    # filter out one colum of the matrix in order to just get red values
    if bgra:
        reds = image_array[:,2]
    else:
        reds = image_array[:,0]

    #image_array =numpy.multiply(image_array,MUL_ARRAY)

    # Filter out any red value that is less than 180 (should be adaptive)
    mask = numpy.greater(reds,180)

    # get indices of pixels with a red value over 180
    original_indices = numpy.array(mask.nonzero())[0]
```

```
# pixel_list is an n x 4 matrix of candidate pixels
pixel_list = numpy.array(image_array[original_indices])
if bgra:
    MUL_ARRAY = BGRA_MUL_ARRAY

# next two lines multiply and then sum in order to get
# redness = 2*R - G - B + 0*A
pixel_list = numpy.multiply(pixel_list,MUL_ARRAY)
pixel_list = pixel_list.sum(axis=1)

# filter for redness over 300
pixel_indices = numpy.greater(pixel_list,250).nonzero()
red_pixels = numpy.array(pixel_list[pixel_indices])

# Work back to get the index of pixels that pass both filters
original_indices = original_indices[pixel_indices]
# create an n x 2 matrix with redness values and indicies
combined_array = numpy.column_stack((red_pixels,original_indices))

# sort by redness
combined_array = numpy.sort(combined_array,axis=0,kind='quicksort')
```

```
        clean_list = []

        # take reddest pixel first, put it in the list
        # then take next reddest pixel and put it
        # in the list if it isn't too close (sqrt(200) pixels)
        # to a pixel already in the list
        for pixel in combined_array[::-1]:  # this reverses the order from the
sort
            #print "pixel:", pixel
            if clean_list.__len__() > 5: #limits total number of dots tracked
                break

            x = pixel[1]%640
            y = pixel[1]/640

            add_elem = True
            for clean_elem in clean_list:
                if ((x-clean_elem[0])**2 + (y-clean_elem[1])**2) < 200:
                    add_elem = False
                    break
            if add_elem:
                clean_list.append((x,y))

        for elem in clean_list:
            self.push((elem[0],elem[1]), flip_y)
```

Demo

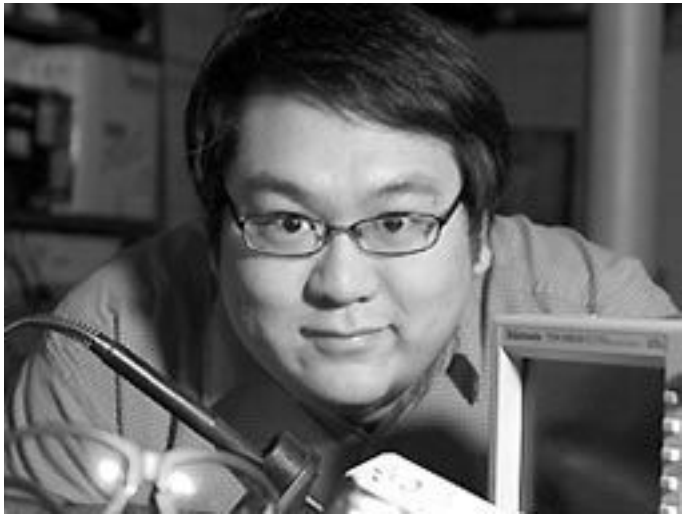
- Laser Missile Command

Limitations

- Technically doesn't “track” lasers, just reports points where they are detected
- Still has problems detecting a laser when surrounding area isn't black
- No obvious interface for movement, only pointing

An Additional Inspiration

- Johnny Lee wiiMote hacks
- http://youtube.com/watch?v=Jd3-eiid-Uw
- Note the power of putting your research on YouTube...

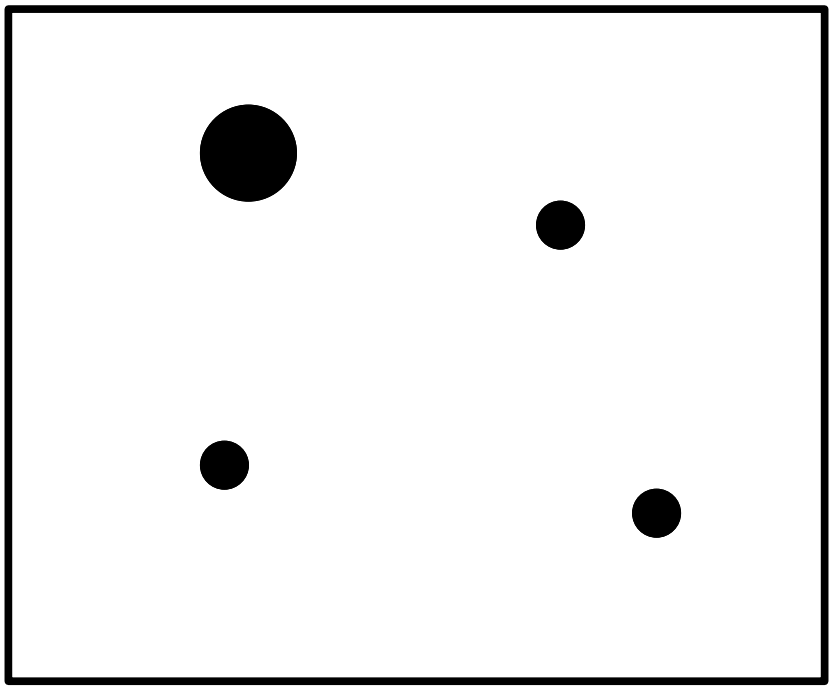


wiiMote

- Traditional controls:
  - thumbpad
  - buttons
  - trigger
- Bluetooth
- Accelerometers
- IR Webcam w/ hardware tracking of up to 4 infrared points

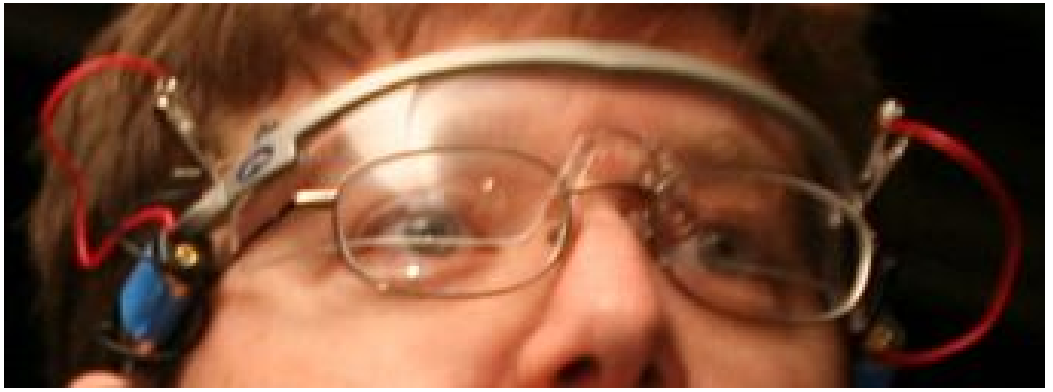
Infrared Web Cam

- Doesn't send images
- Gives coordinates and intensity of up to four points
- 100 fps – limited by Bluetooth
- 1024 x 768 resolution
- IR only is due to filter which can be removed

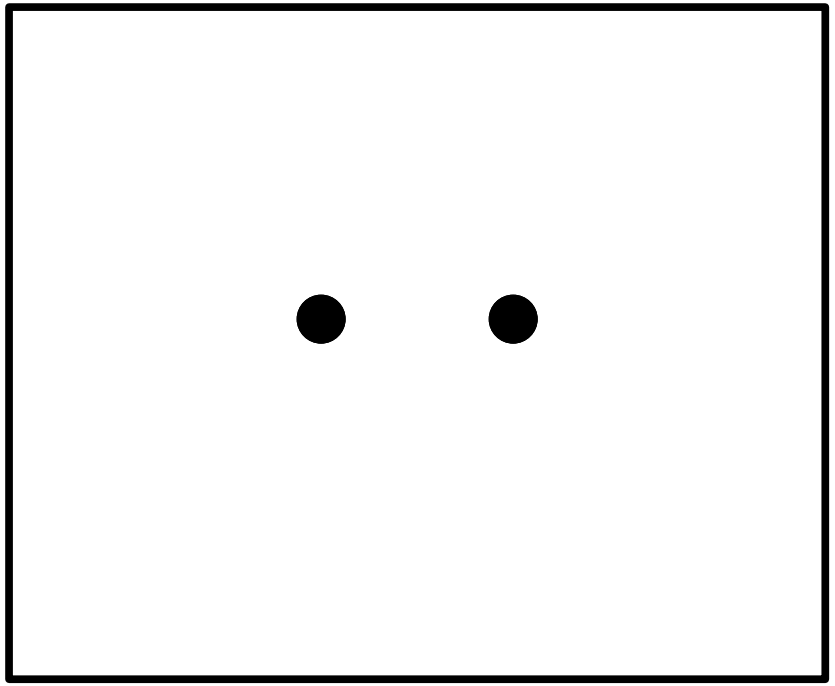


“Sensor” bar

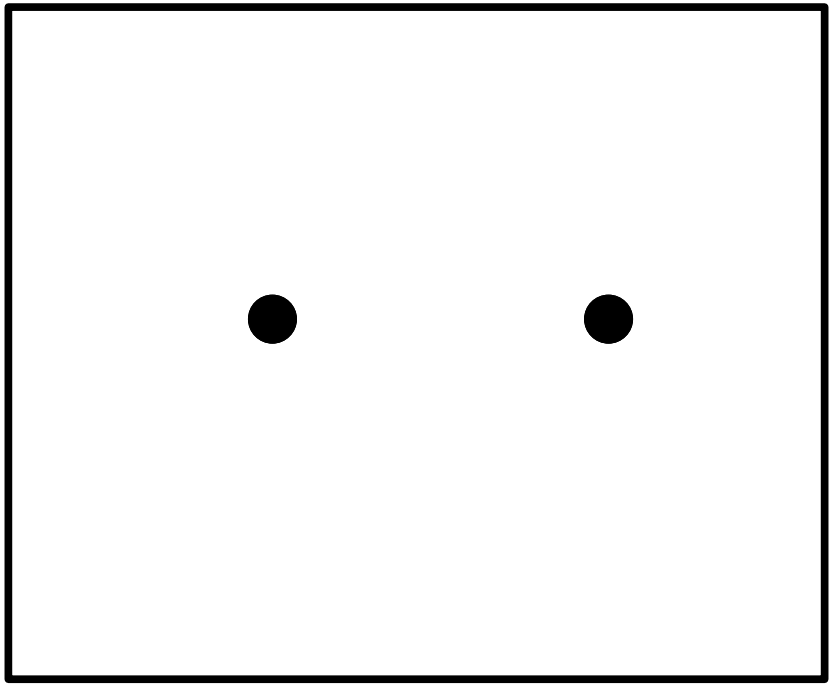
- Reversal of roles between wiiMote and sensor bar
- Need to put sensor bar on the player using glasses or a hat
- Currently requires wiiMote to be about at eye level



Headtracking Made Simple

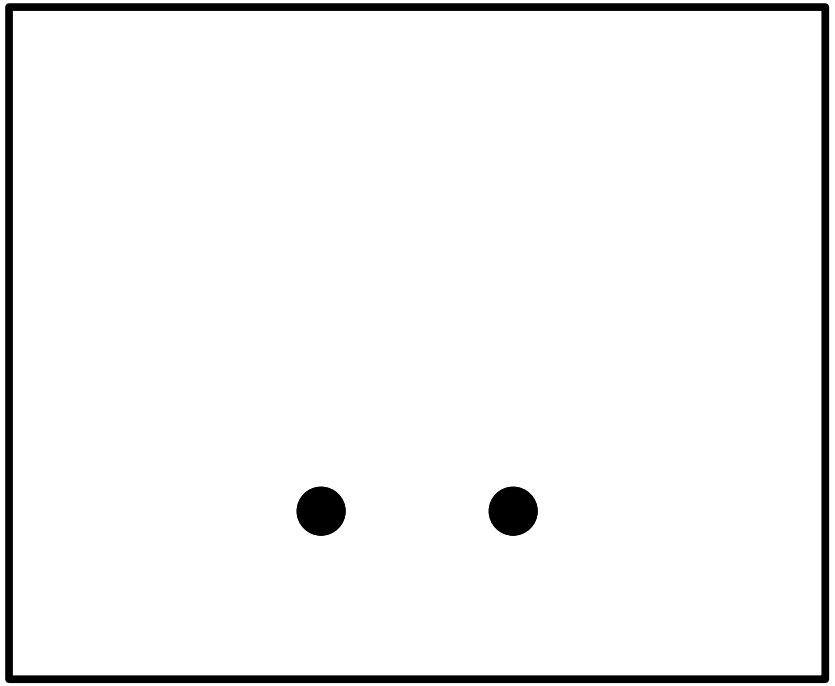


neutral

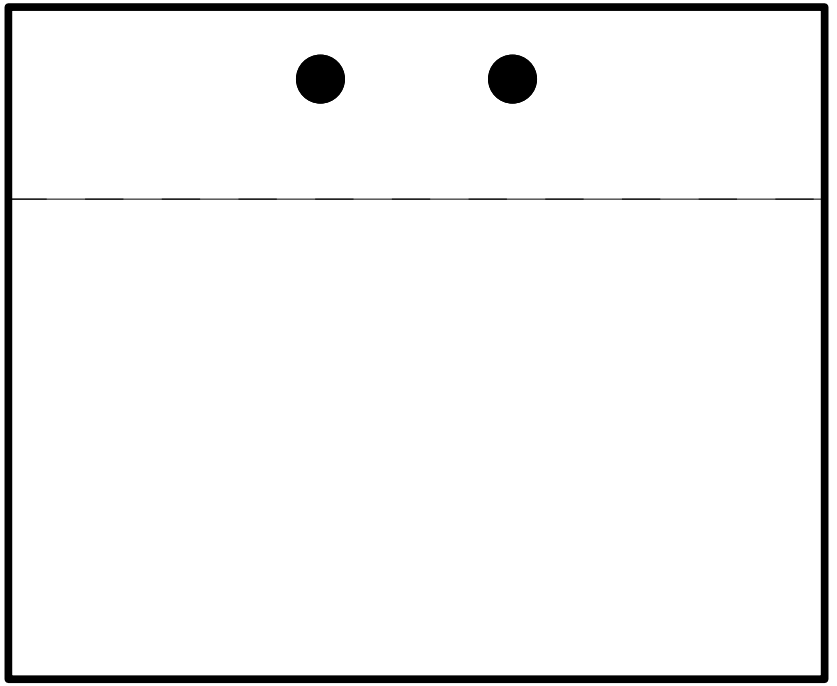


closer

Headtracking Made Simple

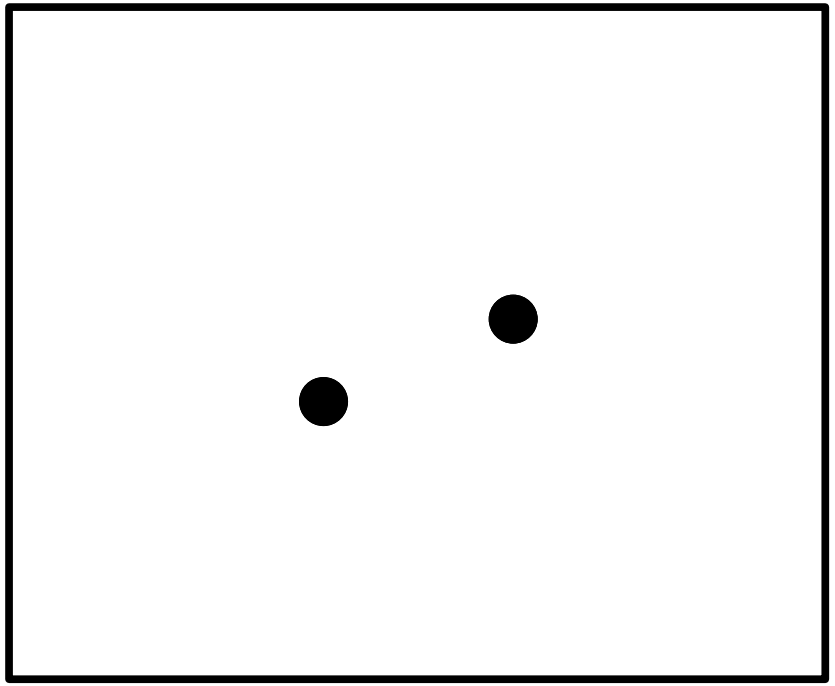


crouch

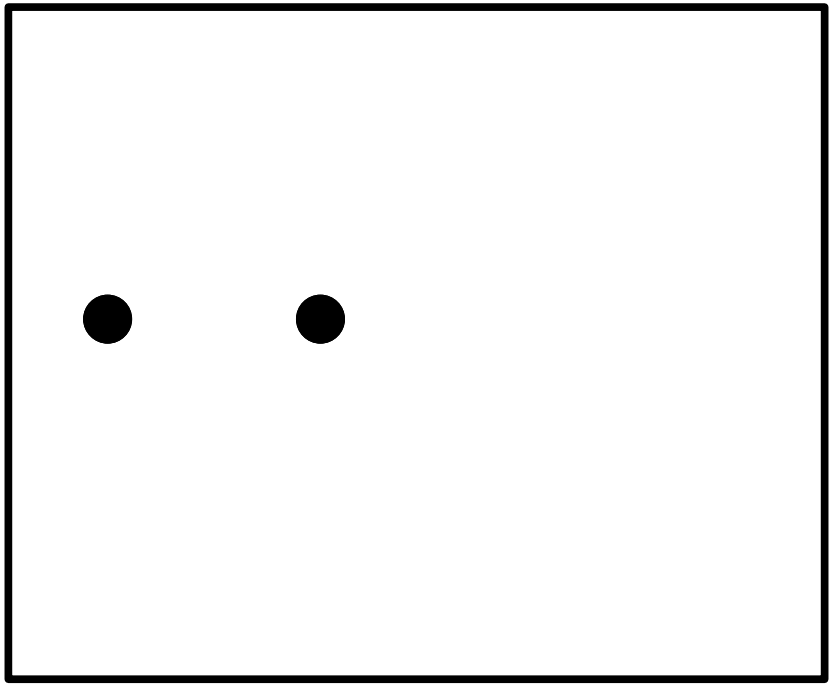


jump

Headtracking Made Simple

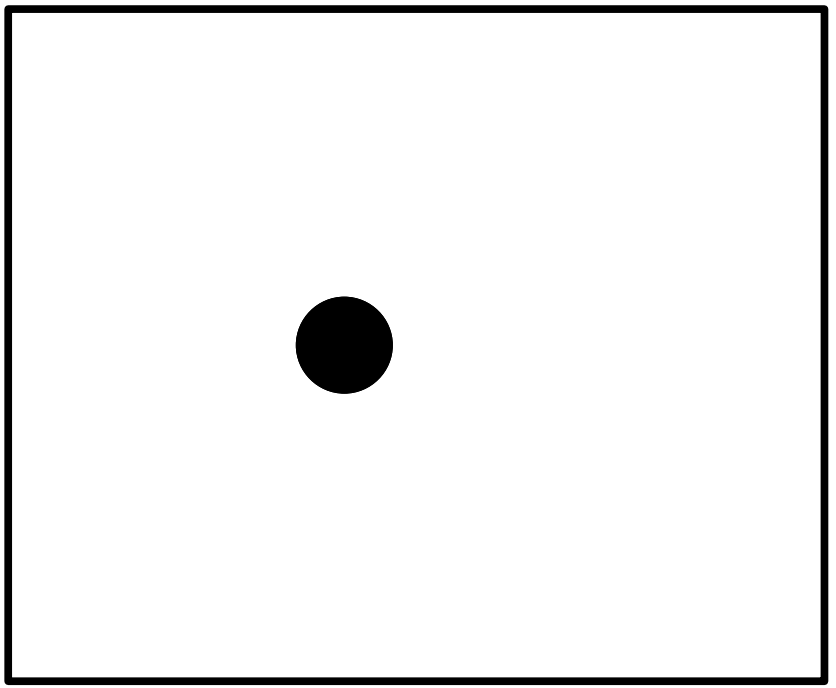


tilt



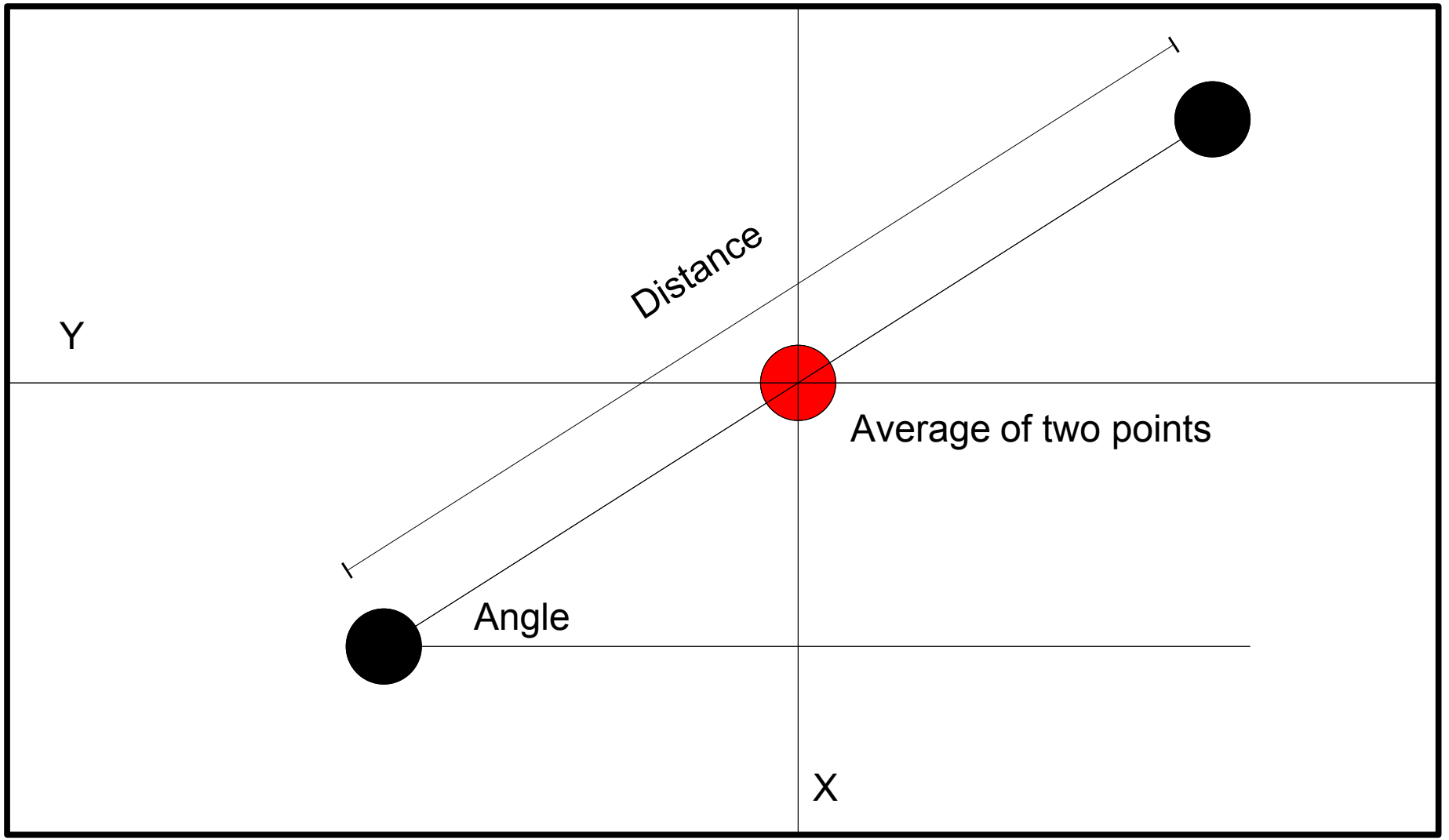
side step

Oh Noes!



u turnd ur head

Things to Measure



Headtracking Code

```
def set_points(x1, y1, x2, y2):
    global DIST
    global SLOPE
    global X
    global Y
    if (x1 < 1023 and y1 < 1023 and x2 < 1023 and y2 < 1023):
        DIST = math.sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2))
        if (x2-x1==0):
            SLOPE = 100
        else:
            SLOPE = (y2-y1)/float(x2-x1)
        X = (x1+x2)/2
        Y = (y1+y2)/2

def get_vals():
    return (DIST,SLOPE,X,Y)
```

3D Navigation w/ Headtracking

- Subtle differences from the Johnny Lee demo
- Player stands in neutral position to stand still
- Take a step forward to walk forward
- Take a step to the left to turn to the right
- Jumping, crouching
- Combos: player can run, turn, crouch at once
- Return to neutral position to stand still

Lasers in 3D

- Not as simple as it might seem
- OpenGL picking works but detects a hit on a polygon, not an exact point in 3D space
- How I cheat:
  - Only detect hits on a cube that surrounds world
  - Recursively subdivide wall that is hit to get point
  - Fire missiles towards that point
  - Do collision detection with missiles
- This works well enough

Demo

- 3D Game

Miscellaneous

- Explosions and missiles are each about 80 lines of code
- Collision detection hashes on locations
- Could drop the laser and use two wiiMotes
- Playable with keyboard and mouse
- Currently very much a demo
  - Need enemies to do something
  - Sound effects would be nice
- Could make a sensor bar lightsaber

Questions?

- Contact:
  - JohnHarrison@gmail.com
  - http://blog.insightvr.com
- Downloads:
  - http://insightvr.com