



NYU

POLYTECHNIC SCHOOL
OF ENGINEERING

Robotic Fish

Dhaval Palsana and Paul Phamduy
Advanced Mechatronics Raspberry Pi Project
May 8, 2014





Science, tech and toys of the future at DC expo



Robotic fish

- Robotic fish, Commodore, has been exhibited at Washington DC, New Jersey, and New York.
- To enhance informal science learning and increase interest in biology and engineering.
- Currently, controlled by iPad applications.

Exploration, Rubik's Cube and Robot Fish



NYU-Polytechnic Second Annual Research Expo in New York



Underwater Camera

- Everyone like to take pictures when they go on vacations. Interesting pictures are taken at hard to reach places.
- GoPros are a viable option for an underwater camera that can be viewed by an app. However, these are very programming constrained and have to be handheld.
- A controllable robotic fish may present a solution.





Look through the eyes of a robotic fish!

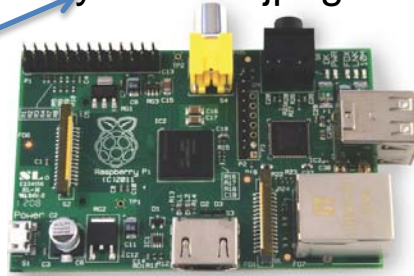
- Looking through the eyes of the robotic fish can give children a more interesting. Super fun!
- If we can do it in Xcode, why not try for Android too? Good practice in android app development.
- Apply concepts in the classroom to practical knowledge and the development of an underwater camera system for a robotic fish. (Raspberry Pi, Android, Arduino and OpenCV)



Logitech Webcam



Python + mjpeg



Raspberry pi



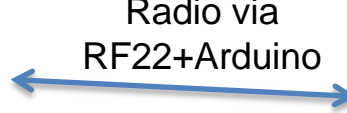
UDP comm



UDP comm



Radio via RF22+Arduino



Google Nexus 7

Netgear router





Hardware

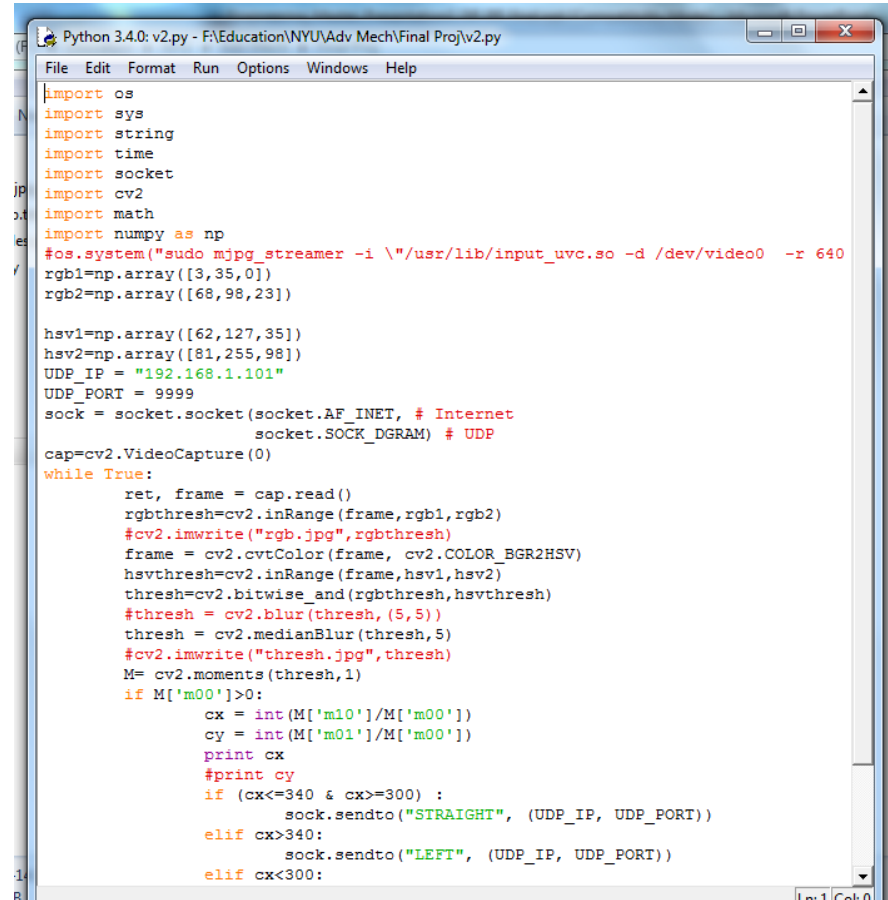
- One robotic fish named the Commodore. Multi-linked motorized tail with a waterproofed box housing an Arduino Pro mini as the processor.
- Android, Arduino, and RF22 to handle communications over the NETGEAR router using UDP protocol.
- A web camera and Raspberry Pi encased in a portable water proofed container.





Raspberry Pi

- Uses a Rasbian build.
- Uses python script, UDP communication protocol and opencv.
- Uses MJPEG Streamer to stream the video feed to Android tablet and computer.
- Not powerful enough to stream and process the images at the same time.



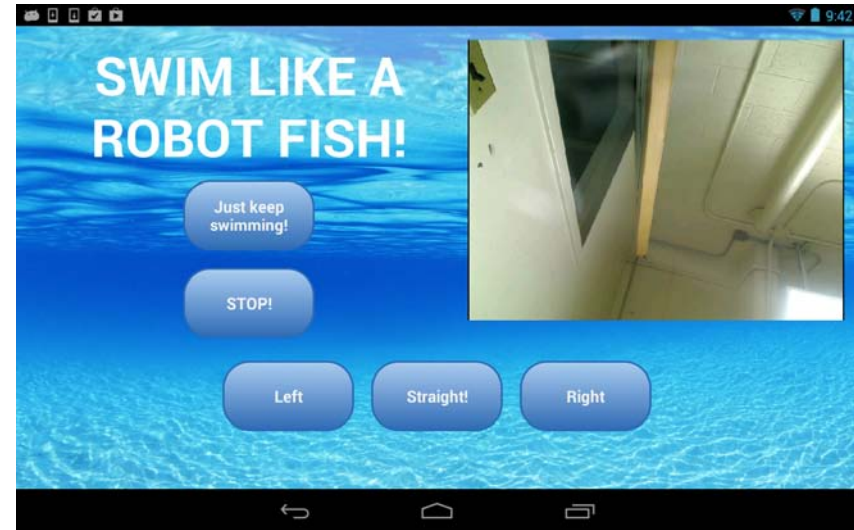
```
Python 3.4.0: v2.py - F:\Education\NYU\Adv Mech\Final Proj\v2.py
File Edit Format Run Options Windows Help
import os
import sys
import string
import time
import socket
import cv2
import math
import numpy as np
#os.system("sudo mjpeg_streamer -i \"/usr/lib/input_uvc.so -d /dev/video0 -r 640
rgb1=np.array([3,35,0])
rgb2=np.array([68,98,23])

hsv1=np.array([62,127,35])
hsv2=np.array([81,255,98])
UDP_IP = "192.168.1.101"
UDP_PORT = 9999
sock = socket.socket(socket.AF_INET, # Internet
                     socket.SOCK_DGRAM) # UDP
cap=cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    rgbthresh=cv2.inRange(frame,rgb1,rgb2)
    #cv2.imwrite("rgb.jpg",rgbthresh)
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    hsvthresh=cv2.inRange(frame,hsv1,hsv2)
    thresh=cv2.bitwise_and(rgbthresh,hsvthresh)
    #thresh = cv2.blur(thresh,(5,5))
    thresh = cv2.medianBlur(thresh,5)
    #cv2.imwrite("thresh.jpg",thresh)
    M= cv2.moments(thresh,1)
    if M['m00']>0:
        cx = int(M['m10']/M['m00'])
        cy = int(M['m01']/M['m00'])
        print cx
        #print cy
        if (cx<=340 & cx>=300) :
            sock.sendto("STRAIGHT", (UDP_IP, UDP_PORT))
        elif cx>340:
            sock.sendto("LEFT", (UDP_IP, UDP_PORT))
        elif cx<300:
```



Android

- Creates a one screen interface that lets users choose the direction to control the robotic fish and a stop button. Two buttons for on/off. Three buttons for steering.
- Located on IP address 192.168.1.100
- Collects the video stream from the mjpeg streamer from the router and displays it on the app.





Arduino (Commodore/Base station)

- A base station converts UDP commands from router to radio signals using the RF22 transmitter chip.
- Arduino is programmed to listen to the radio signals in the format “Mode:Manual”, “s0f1”, or “s0o10”
- Robotic fish responds by beating its tail following a carangiform motion based on the received message.

```
by Michael Margolis

This code is in the public domain.
*/

#include <SPI.h>           // needed for Arduino versions later than 0018
#include <Ethernet.h>
#include <EthernetUdp.h>  // UDP library from: bjoern@cs.stanford.edu 12/30/2008

|
// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 2);

unsigned int localPort = 9999;    // local port to listen on

// buffers for receiving and sending data
char packetBuffer[UDP_TX_PACKET_MAX_SIZE]; //buffer to hold incoming packet,
char ReplyBuffer[] = "acknowledged";      // a string to send back

// An EthernetUDP instance to let us send and receive packets over UDP
EthernetUDP Udp;

void setup() {
  Serial.begin(9600);
  Serial.println("try begin");
  // start the Ethernet and UDP:
  Ethernet.begin(mac,ip);
  Serial.println("try udp");
  Udp.begin(localPort);
  Udp.flush();
  Serial.println("setup done");
}

void loop() {
```



**Thank you for your attention!
We would be happy to answer any
questions.**

5/29/2014