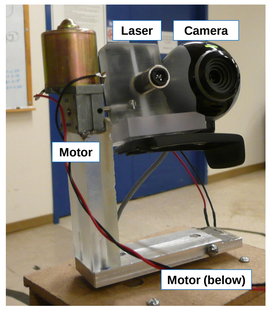
Project Cyclops

Mechatronic Design, Spring 2010

Arafat Amin, Ben Chu, Mark Martinez, Mikhail Podust, Stephen Tu

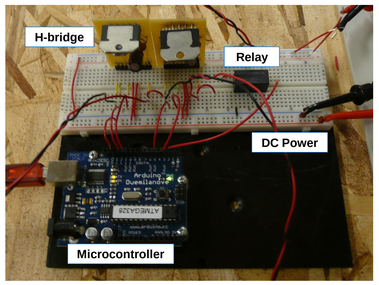
EECS

MECHANICAL HARDWARE



- Built from scrap aluminum and acrylic for strength and weight.
- DC motors with gearbox.
- Laser pointer to shoot target.
- Constrained to 2 rotation degrees of freedom: pitch and yaw (no translation motion).

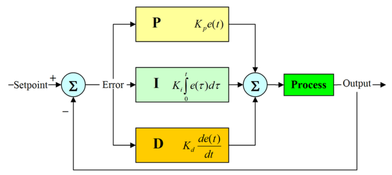
ELECTRICAL HARDWARE



- H-bridge circuit for control of motor direction and velocity.
- 2 separate circuits for pitch and yaw axes.
- Relay switch to activate laser.
- Powered by DC power supply.

FEEDBACK CONTROL

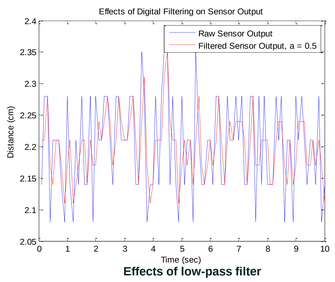
- PID control is used to send a PWM and direction signal to the motor.
- Gains tuned for response time and to minimize steady state error.



Block Diagram of a PID control loop

SIGNAL PROCESSING

$$y[i] = \alpha x[i] + (1-\alpha)y[i-1] \text{ where } 0 \leq \alpha \leq 1$$

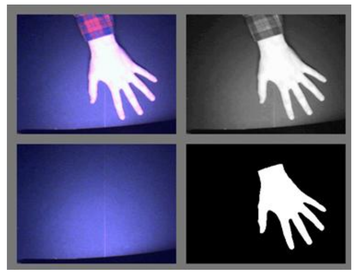


Effects of low-pass filter

- Discrete-time (digital) low pass filter for displacement data, implemented on the Arduino microcontroller.

SYSTEM OVERVIEW

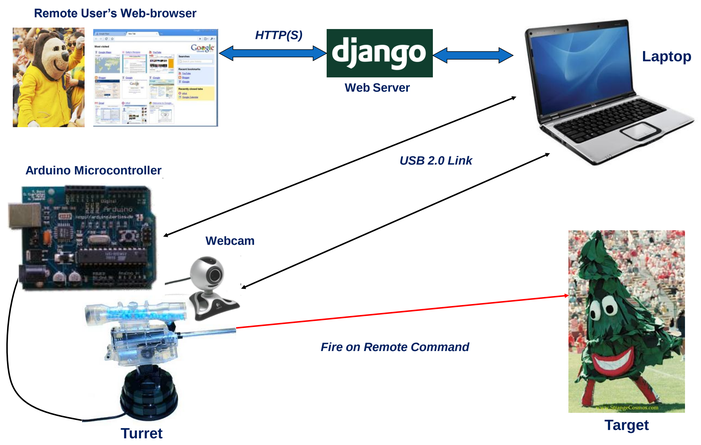
- Built a turret to track arbitrary colors and faces.
- Manual mode to move turret, such as in FPS video games.
- Change target to track in real-time.
- "Attack" target with laserpointer.



Color masking on an image

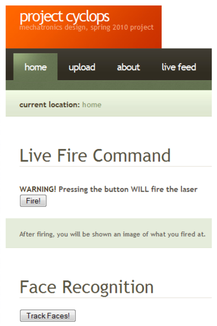
SYSTEM ARCHITECTURE

- Mechanical hardware: structural support and motors.
- Electrical hardware: relay switch, H-bridge circuit, DC power source, camera.
- Embedded software: feedback control, signal processing.
- Laptop software: image processing, facial recognition, communication.
- Server-side software: Web application for remote use.
- User connects to web server to activate system.



USER INTERFACE

- Web application, enabling remote control of turret.
- User has full control from a remote location.



FACIAL RECOGNITION ALGORITHM

- Haar Classifier
  - Machine learning based approach to classify human faces.
- Camshift Algorithm
  - Used to interpolate when the Haar Classifier is unable to find a match.

Haar Classifier performed on a test image

IMAGE PROCESSING ALGORITHM

- Implemented in C using OpenCV libraries.
- Main color tracking algorithm described below:

Algorithm 1 ColorTrack($x = (h, s, v)$, $y = (h_{tol}, s_{tol})$, k, k_{tol})

Require: $x \leftarrow$ desired hue, saturation, and value gains to track
Require: $y \leftarrow$ predetermined tolerance values
Require: $k \leftarrow$ number of binary convolutions to run
Require: $0 \leq k_{tol} \leq 8 \leftarrow$ number of neighbors in binary convolution
loop
$I[m,n,k] \leftarrow$ RGB image ($m \times n$ matrix, $k = 3$ channels) from camera
$I'[m,n,k] \leftarrow I[m,n,k]$ converted into HSV color space
$J[m,n] \leftarrow m \times n$ binary matrix such that

$$\begin{cases} J[i,j] = 1 & \text{if } |I[i,j,0] - h| < h_{tol} \text{ and } |I[i,j,1] - s| < s_{tol} \\ J[i,j] = 0 & \text{otherwise} \end{cases}$$

$K_0[m,n] \leftarrow J[m,n]$
for $s \leftarrow 1, 2, ..., k$ **do**
$K_s[m,n] \leftarrow m \times n$ binary matrix such that

$$\begin{cases} K_s[i,j] = 1 & \text{if } \sum_{(i',j') \in (i \pm 1, j \pm 1)} K_{s-1}(i',j') \geq k_{tol} \\ K_s[i,j] = 0 & \text{otherwise} \end{cases}$$

end for
$(X_I, Y_I) \leftarrow \left(\frac{m_{1,0}}{m_{0,0}}, \frac{m_{0,1}}{m_{0,0}}\right)$ such that

$$m_{a,b} = \sum_i \sum_j i^a j^b K_k[i,j]$$

Send (X_i, Y_I) to Arduino
end loop

SOFTWARE OPTIMIZATION

- Flush serial data receive buffer each iteration.
- Compress image data to 320x240 pixels.
- Compress serial transfer data to 8 bytes per packet, while still preserving byte alignment.