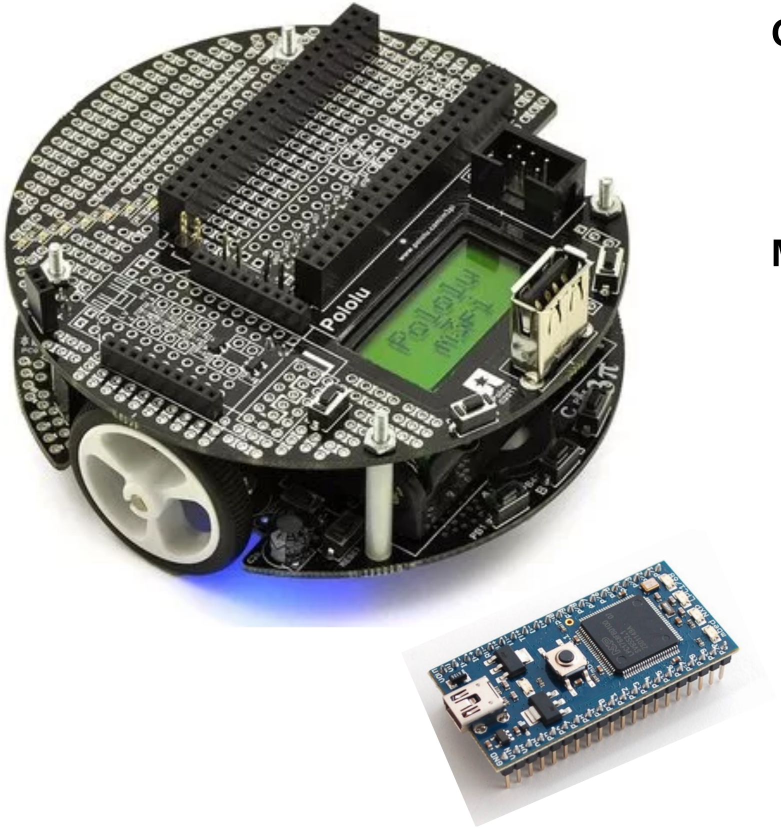
PixelBot

By John Wilkey, Alec Guertin, Chester Chu

EECS 149 - Fall 2014

Overview



Objective for the project

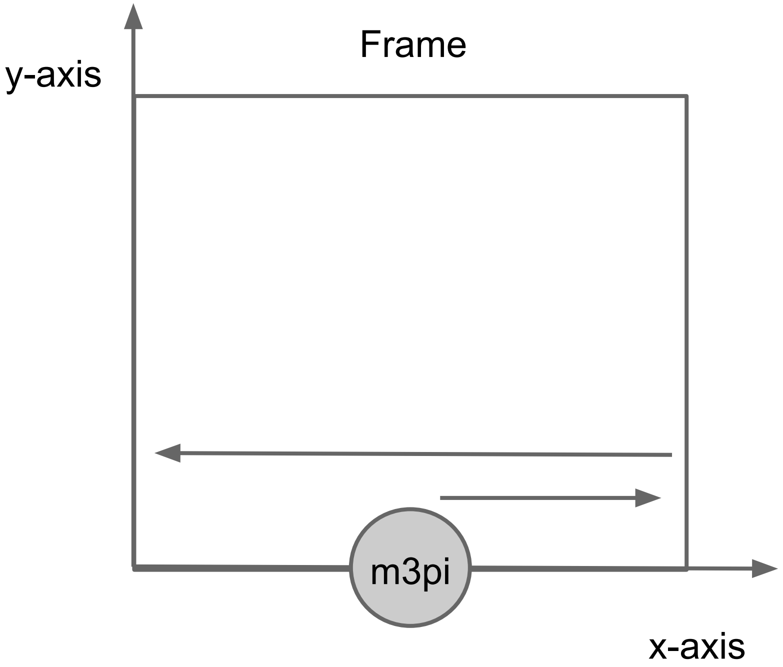
- Design/create a device capable of drawing geometric shapes supplied by user input from a drawing script.

Materials Used

- **m3pi** = 3pi (Atmel Atmega AVR microcontroller-powered robot) + MBED LPC1768 high level controller.
- Digital camera with long exposure lens
- Electrical Tape
- C/Python APIs
  - http://developer.mbed.org/cookbook/m3pi#library-and-api
  - http://developer.mbed.org/handbook/mbed-library-internals#mbed-api
  - http://svn.python.org/projects/python/trunk/Demo/tkinter/guido/paint.py

Part 1: Line Tracking

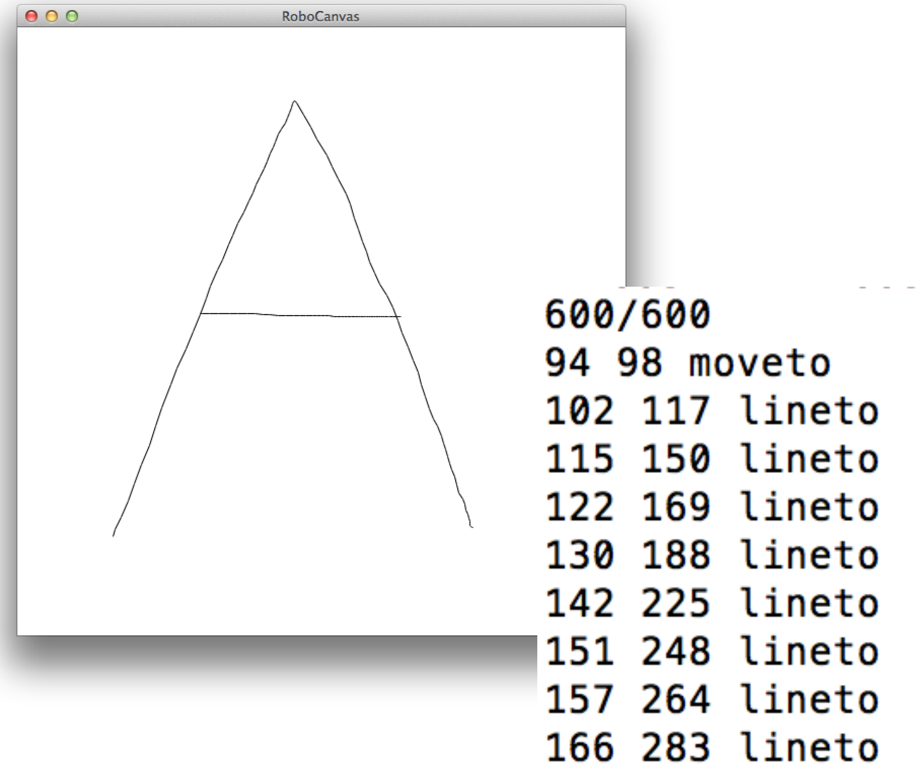
- Reflective sensors mounted under the m3pi detect electrical tape frame
- Use sensor data to follow frame and align m3pi with x-axis
- Search for corners to detect length (assuming a square frame)



Part 2: Drawing

- m3pi receives a series of simplified instructions (see figure)
- Instruction file provides data on coordinates and vectors on a frame with defined size
- Parser determines next state (TURN, MOVE, DRAW, FINISH) from next line of
- m3pi records overall angle from x-axis, calculates remaining angle needed to turn to face new coordinates
- Calculates distance between points and converts to physical distance based on measured frame length

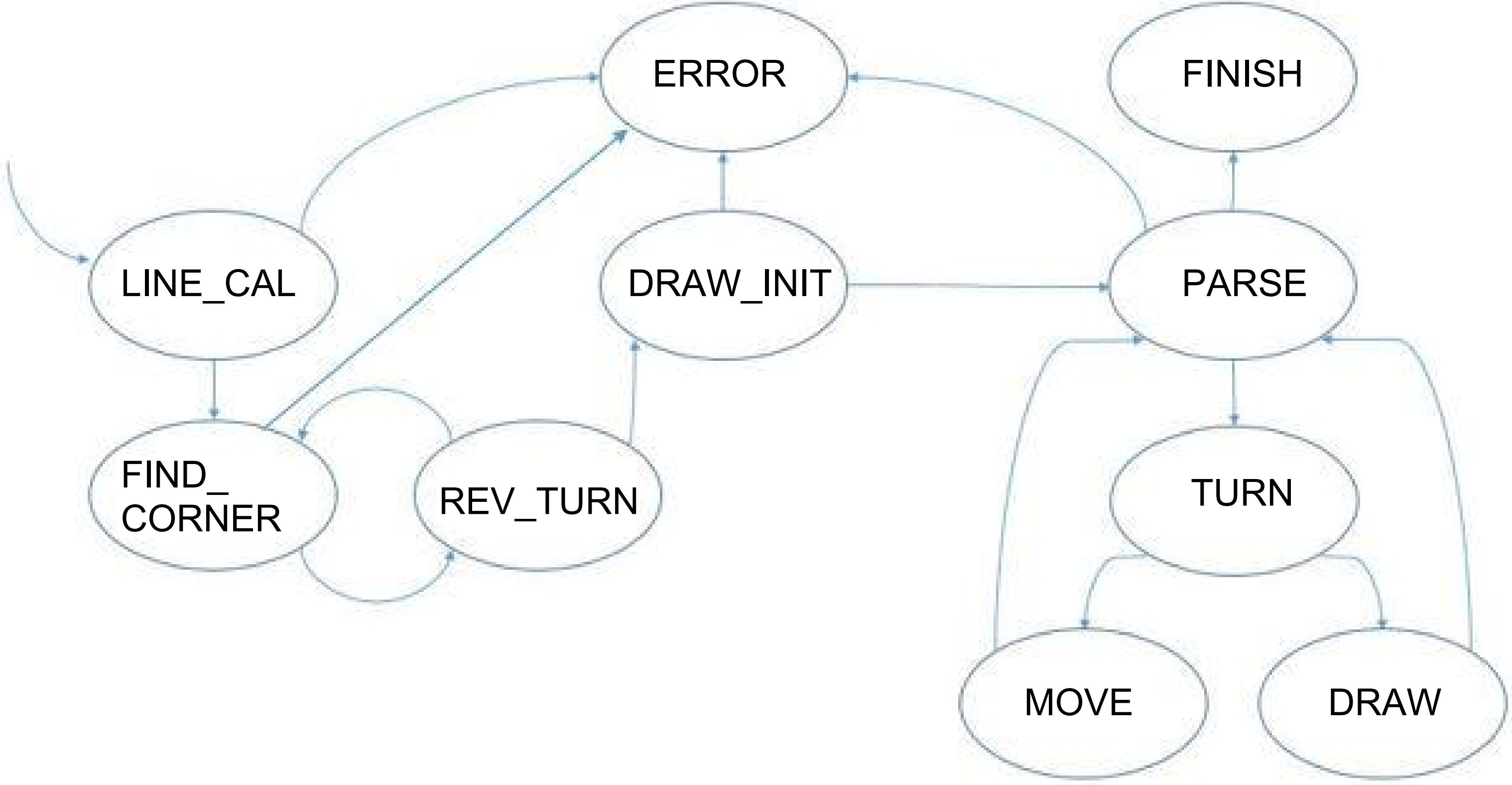
Part 3: User Drawing



```
600/600
94 98 moveto
102 117 lineto
115 150 lineto
122 169 lineto
130 188 lineto
142 225 lineto
151 248 lineto
157 264 lineto
166 283 lineto
```

- User draws with mouse on Tkinter window
- Python program generates a grid of 600 by 600 pixels representing the drawing
- Output is read by robot and treated as (x,y) coordinate vectors to draw

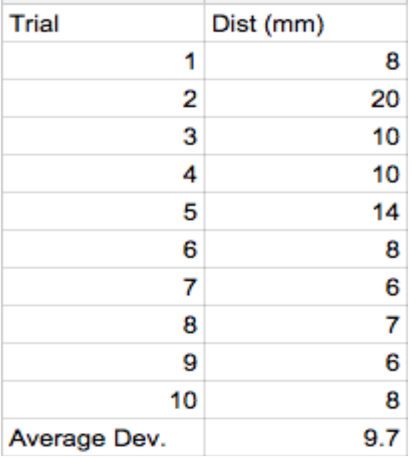
Modeling


ERROR
FINISH
LINE_CAL
DRAW_INIT
PARSE
FIND_
CORNER
REV_TURN
TURN
MOVE
DRAW

Error Measurements

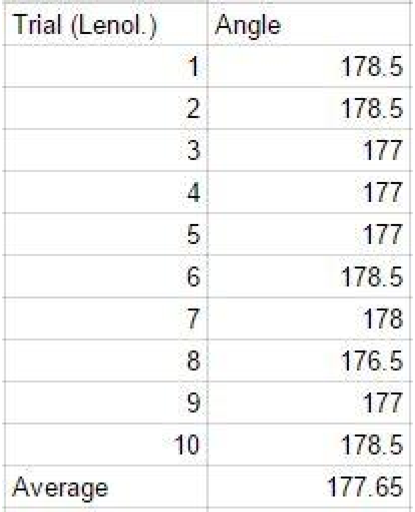
- **Symptoms**
  - Inaccurate turn angles
  - Arc when driving straight
  - Build up error (accumulated across multiple instructions
- **Sources**
  - Caster wheel placement
  - Weight distribution
  - Sensor/Actuator precision
  - Pixelation
  - Scaling
  - Surface Type

Driving Trials:



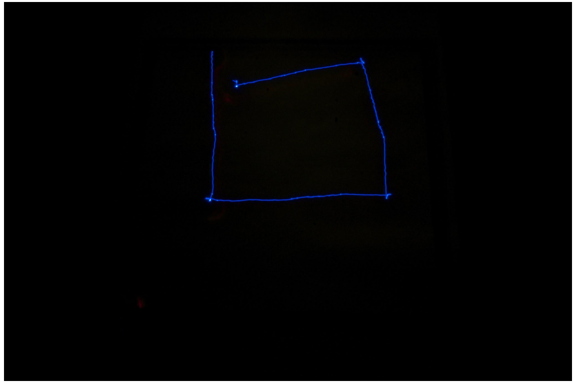
Trial	Dist (mm)
1	8
2	20
3	10
4	10
5	14
6	8
7	6
8	7
9	6
10	8
Average Dev.	9.7

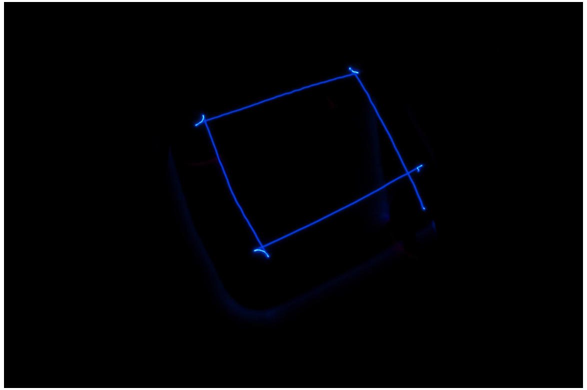
Turning Trials (180°):

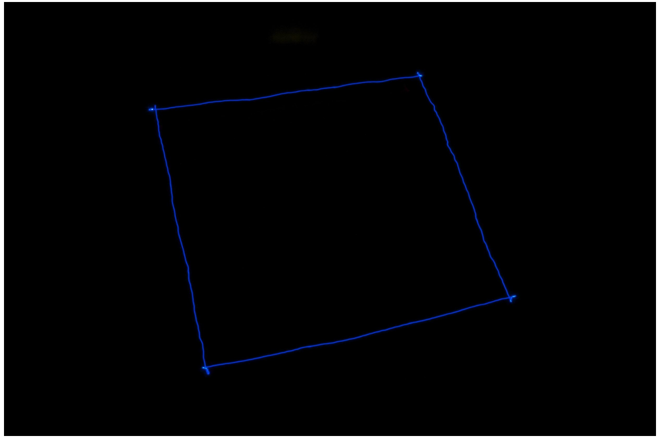


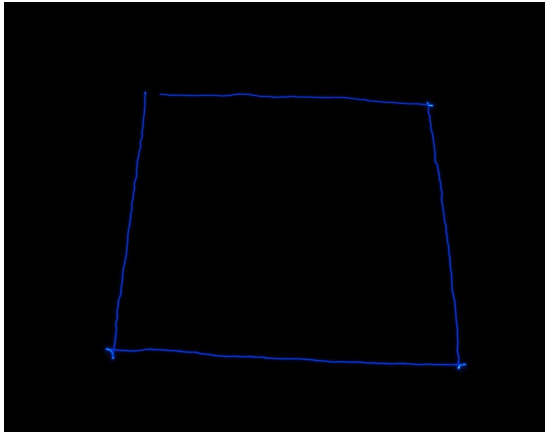
Trial (Lenol.)	Angle
1	178.5
2	178.5
3	177
4	177
5	177
6	178.5
7	178
8	176.5
9	177
10	178.5
Average	177.65

Trials/Calibration









Mitigations

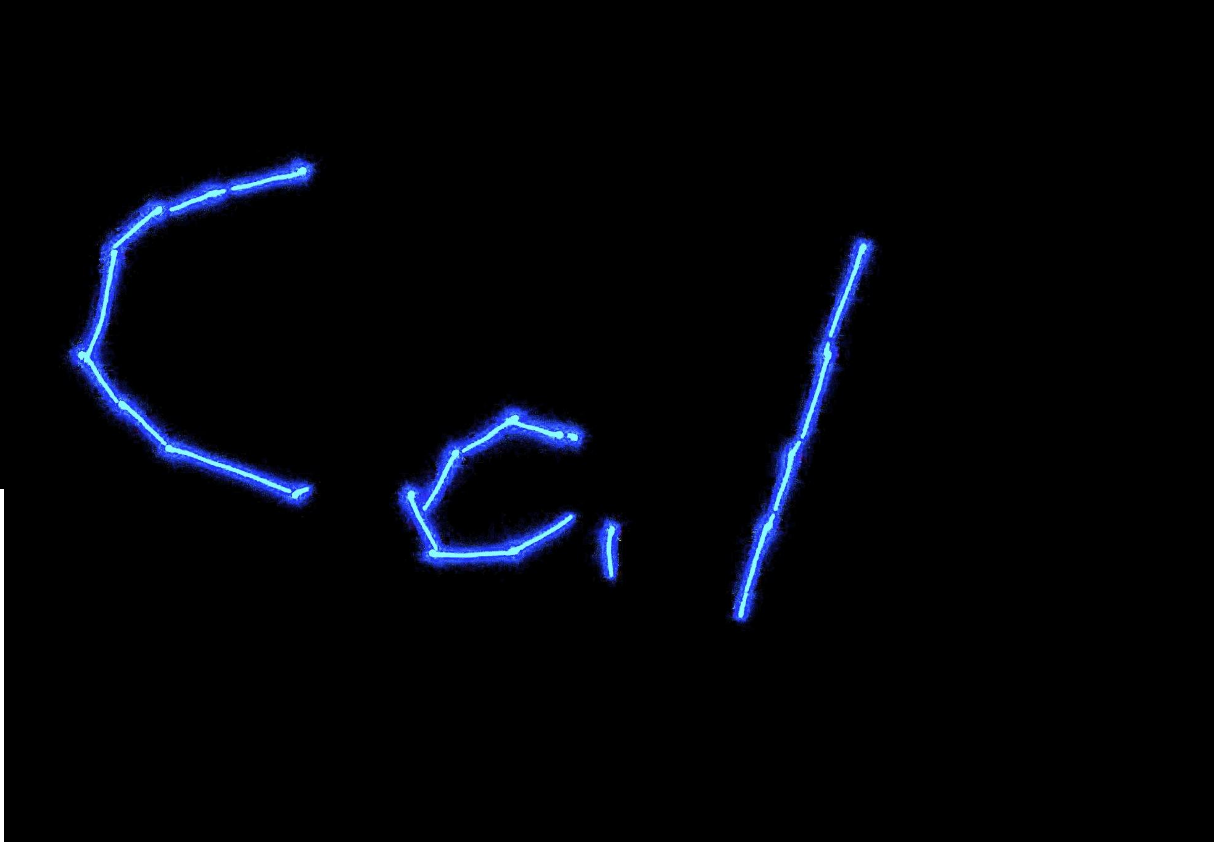
- Calibrate turning angle to compensate for biases
- Calibrate forward drive to compensate for motor drift
- Filtering / Smoothing line drawing commands.
  - reduce number of short lines due to pixelation
  - smooth curves and lines into larger
- Issues that remain: Inherent inconsistency of robot’s motions make static calibrations less useful

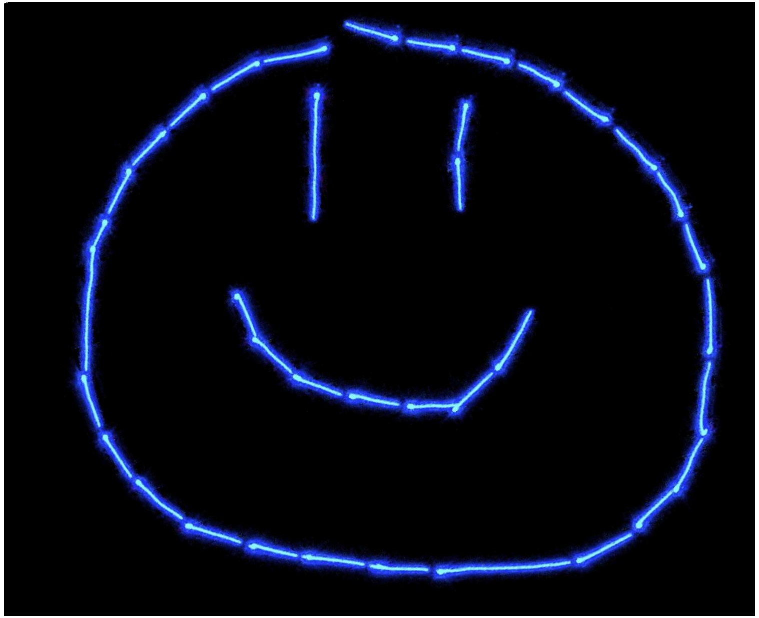
  Result: Many drawings are good but can be easily thrown off by a single errant turn

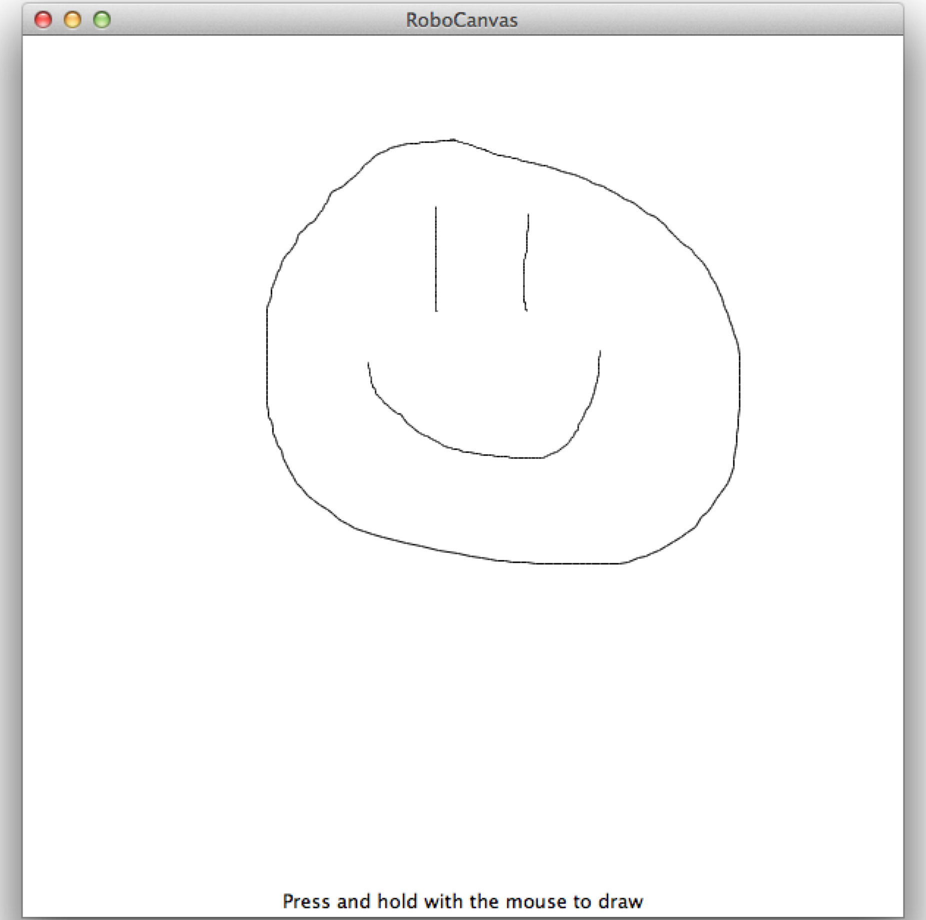
Final Design

- Changes from initial design:
  - What we deemed wouldn’t work and why.
    - The solenoid proved intractable. No reliable way to mount, and external power requirements results in too much weight on the robot. (9 volt battery ~ 50 oz results in deviation of ~ 3-4 cm/s).
  - What we changed to better make use of the robot’s abilities.
    - A mechanical apparatus was deemed inappropriate for our robot. We modified the project to make ‘light-drawings’ using long-exposure photography; this eliminates mechanical engineering obstacles while still preserving the spirit of the project.

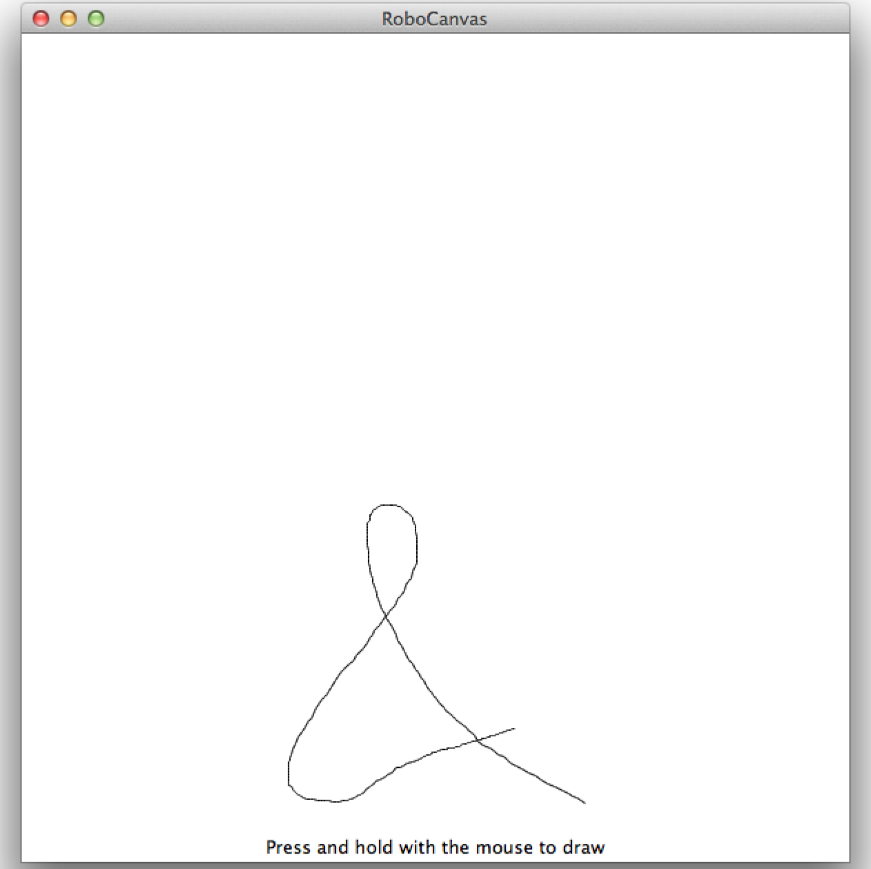
Examples...



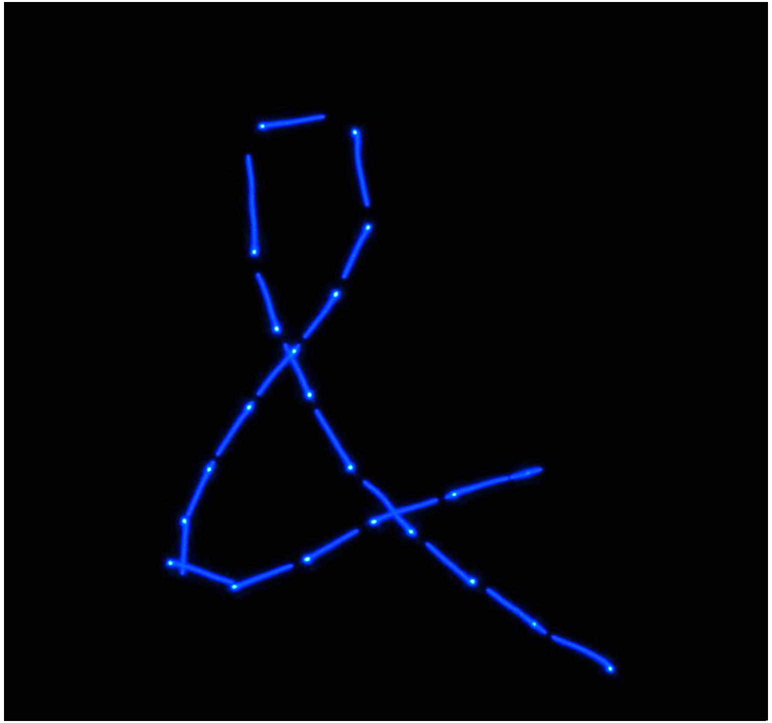




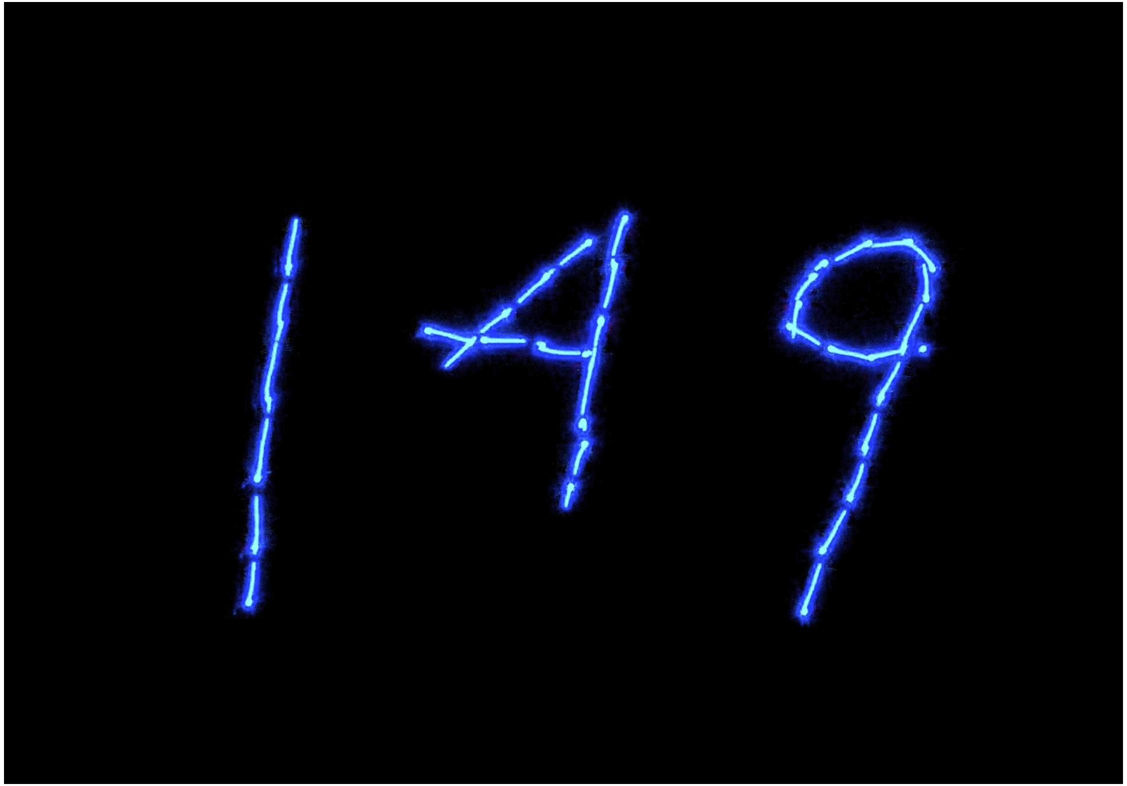
Press and hold with the mouse to draw



Press and hold with the mouse to draw



Thanks...



A