Custom Vision Software in FTC

Sachin Shah

Who am I?

History of Vision

Velocity Vortex (16-17)

Vuforia VuMarks

Optional Use

Gears	LEGO®
	
Tools	**Wheels**
	

Relic Recovery (17-18)

Vuforia VuMarks

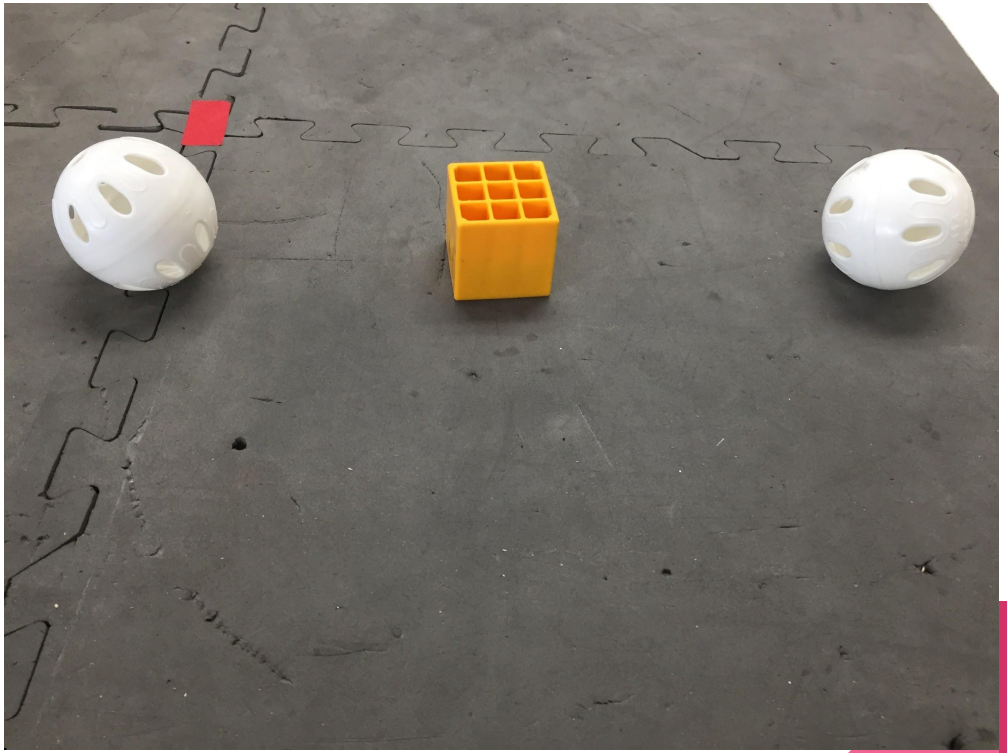
Pictographs

Rover Ruckus (18-19)

External Cameras

DogeCV

TensorFlow



The Case for Custom Vision

The Alternative to Commercial Solutions

Full range of control

Flexibility at competitions

Knowledge is power

Custom Vision

The Alternative to
Commercial Solutions

Set up for flexibility

Basic Algorithms

Optimization for speed

Vuforia for Custom Analysis

- Camera Class
- Init Vuforia
- Get a Vuforia Image
- Convert to Bitmap

- Get RGB values method

- Method to save images to the phone sdcard

Algorithms

Color Analysis

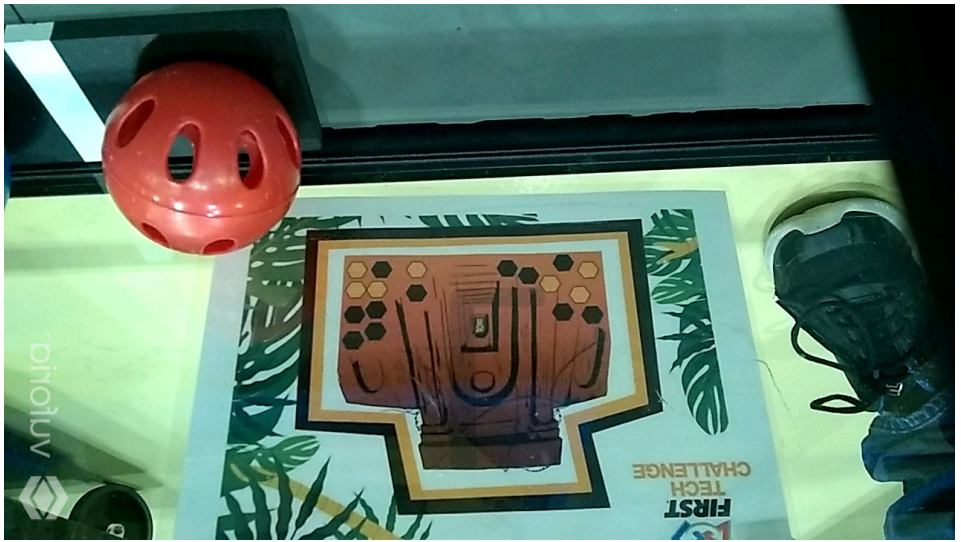
Count particular color pixels.

Simple to write

Functions like a color sensor

Singular color requires a threshold

Multiple colors can compare to each other

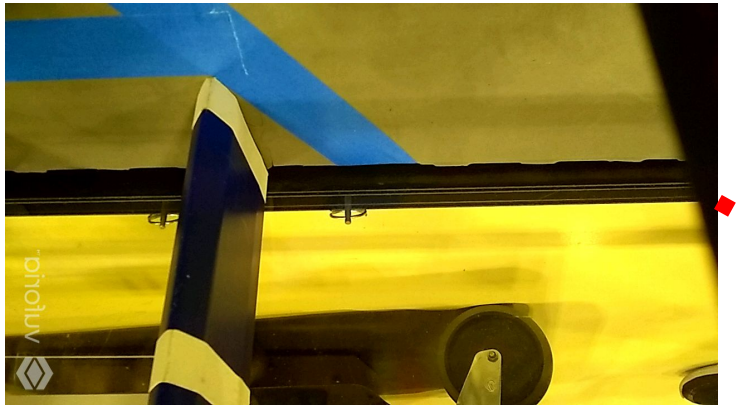


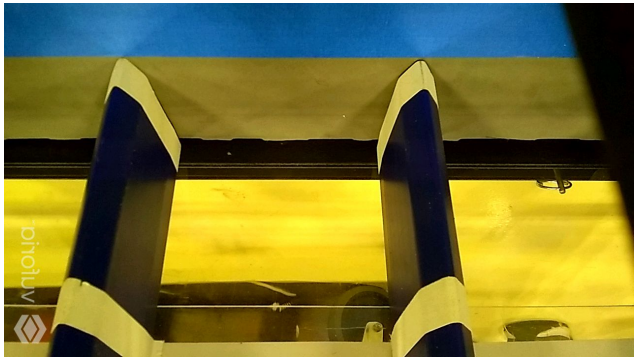
More red than blue

Sigma Difference Comparison

Add up the pixel level difference between the current image to saved ones.

Whichever comparison has the lowest sum, is the current image.

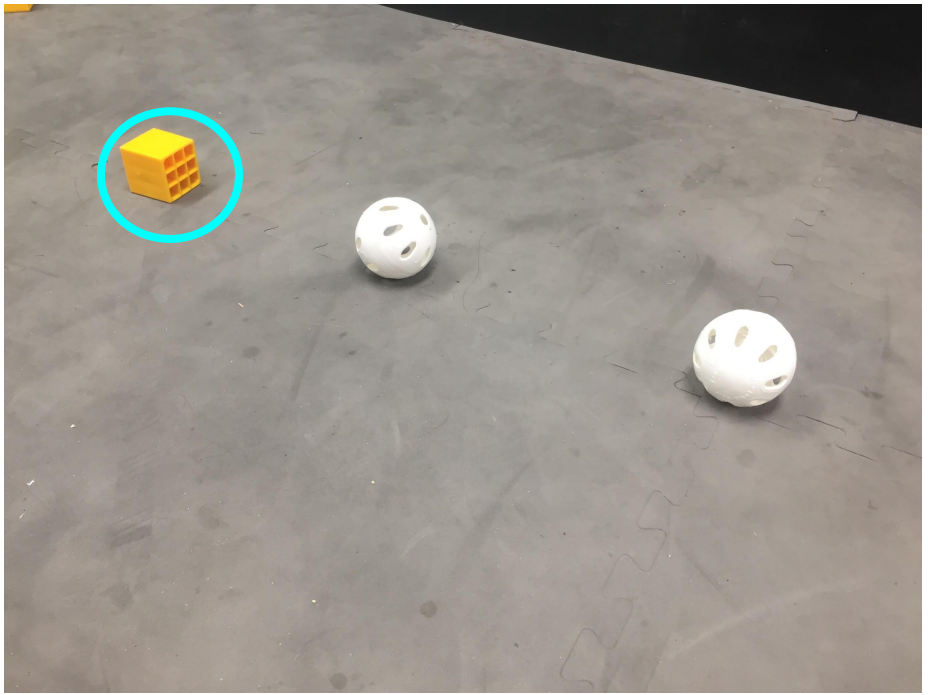




Average Color Position

Take the average of the x and y coordinates of every pixel of a particular color tag

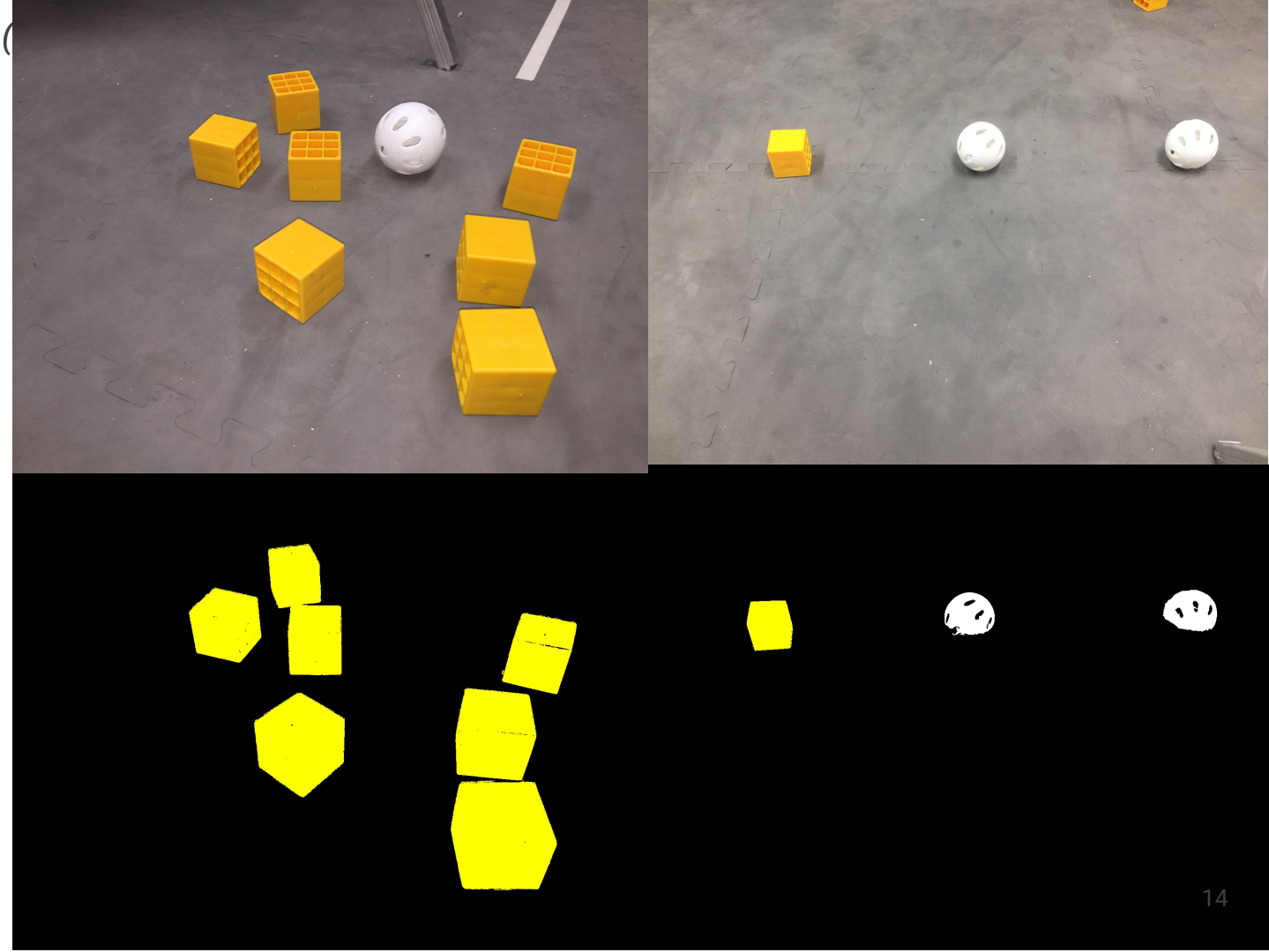
Use the color's coordinates for navigation



Flood Fill

Finds connected color blobs

Each object has a point, width, and height



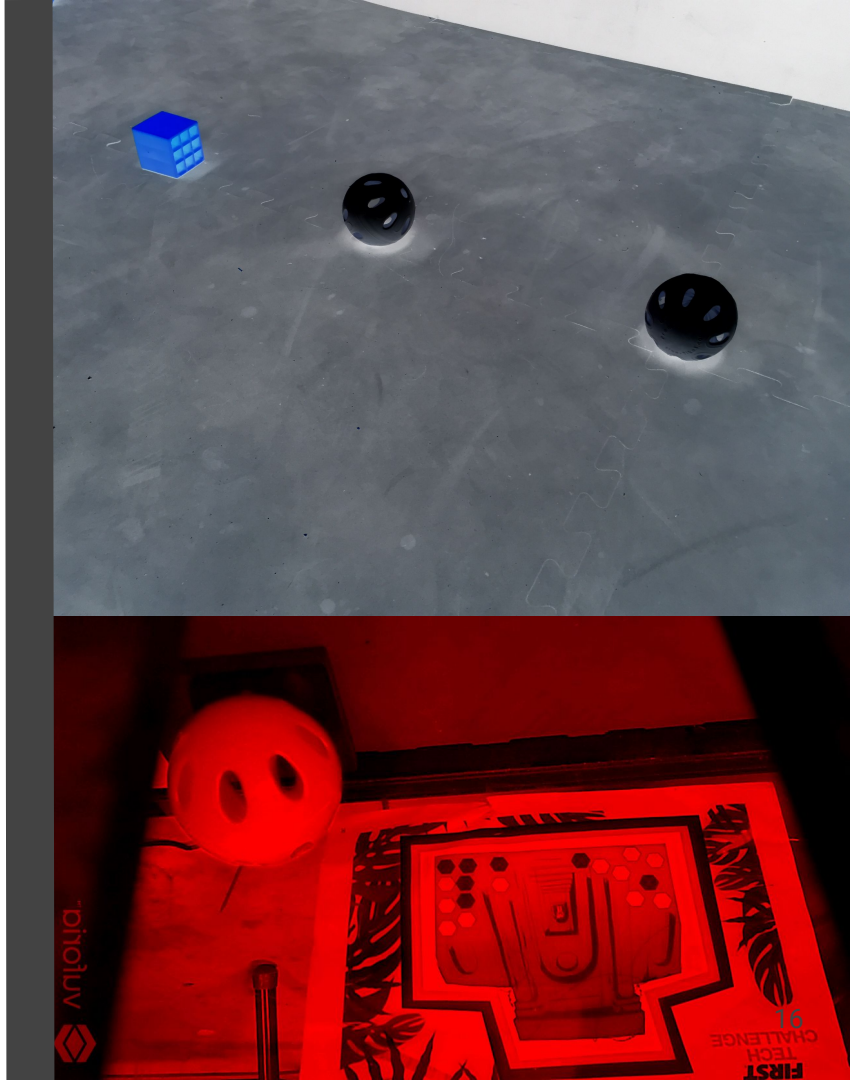
Filters

Math Functions

Transform pixel RGB values

Normalize image based on base values

Weight color channels based on importance



Kernel Convolutions

0	5	10	5	0
5	10	5	10	5
10	5	10	10	10
5	10	10	5	5
0	5	10	10	5

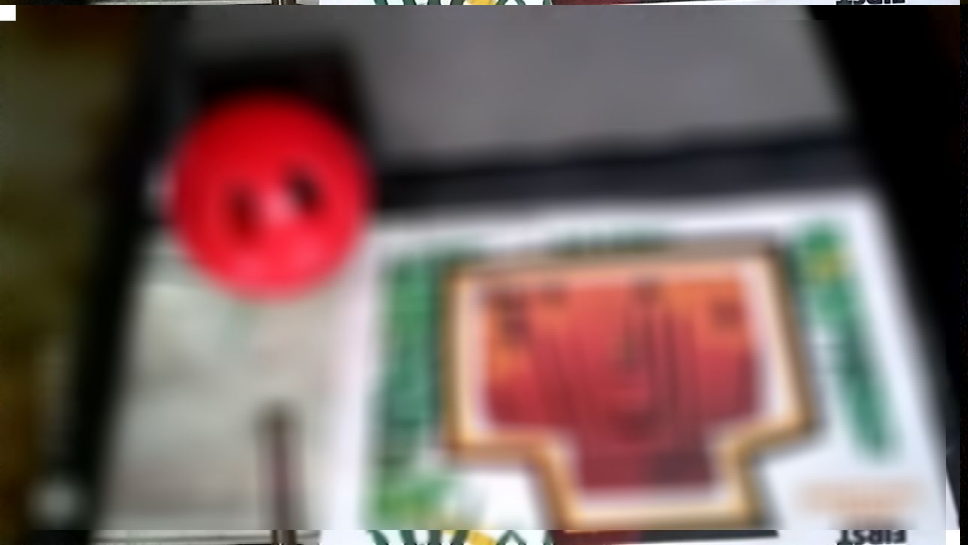
1	1	1
1	1	1
1	1	1

Blur Filter

	6.7	7.8	7.2	
	7.8	8.3	7.8	
	7.2	8.3	8.3	

(1)(0+5+10+5+10+5+10+5+10) = 60 → 60/9 = 6.7


FIRST
TECH
CHALLENGE



Optimization

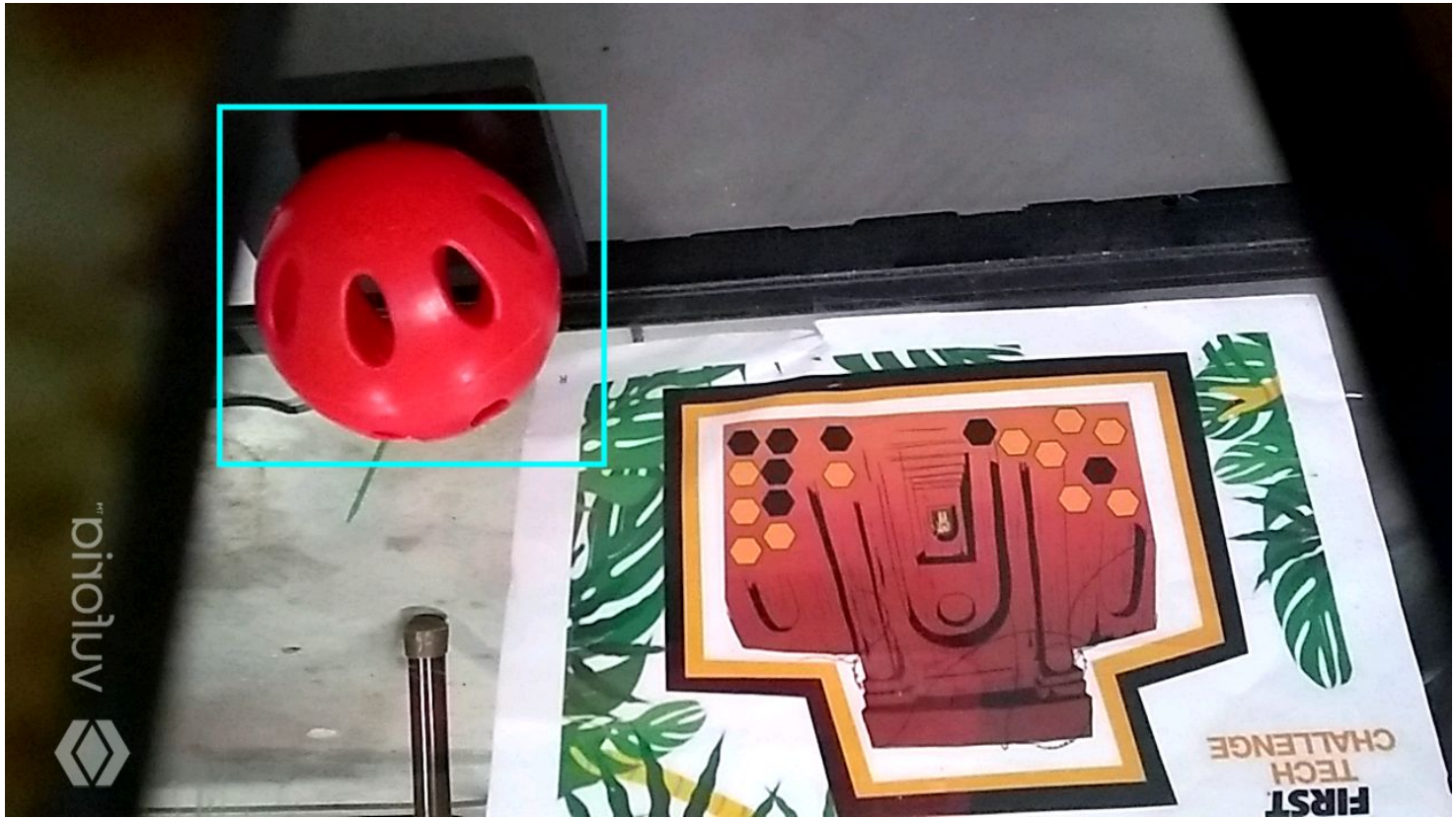
Overview and Notation

Complexity - O(n) where n is the number of pixels analysed (width * height)

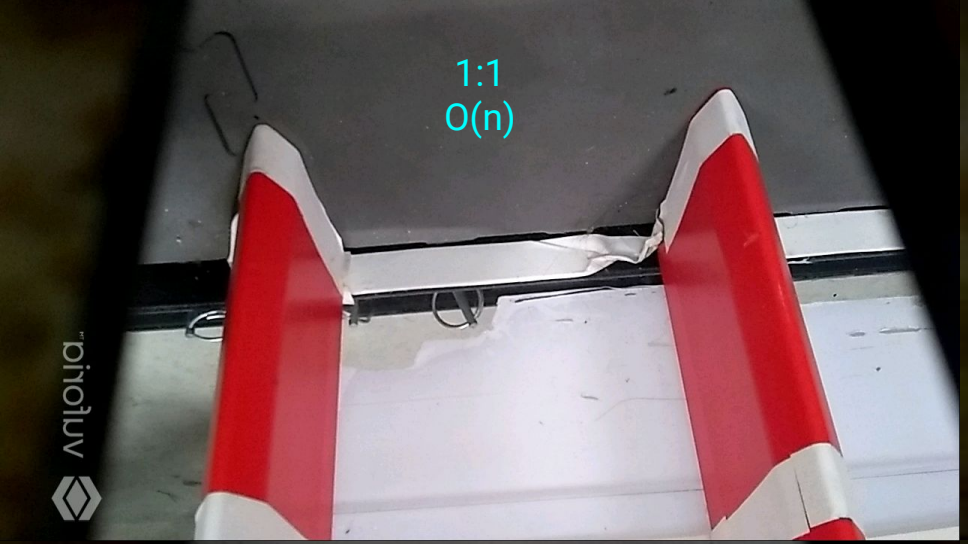
Accuracy - successes over trials

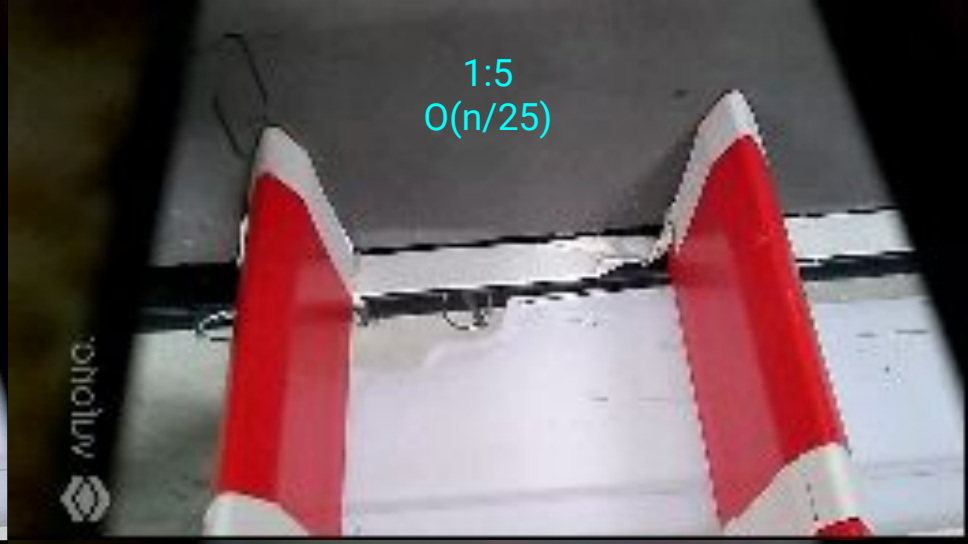
Processing Time - milliseconds algorithm took from start from completion

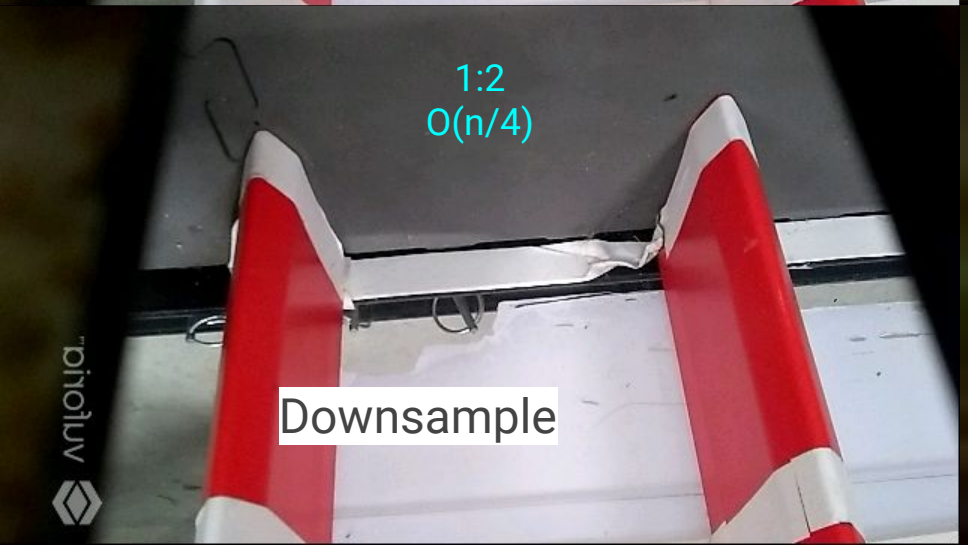
Pixel Polling - Average times a pixel RGB data is requested

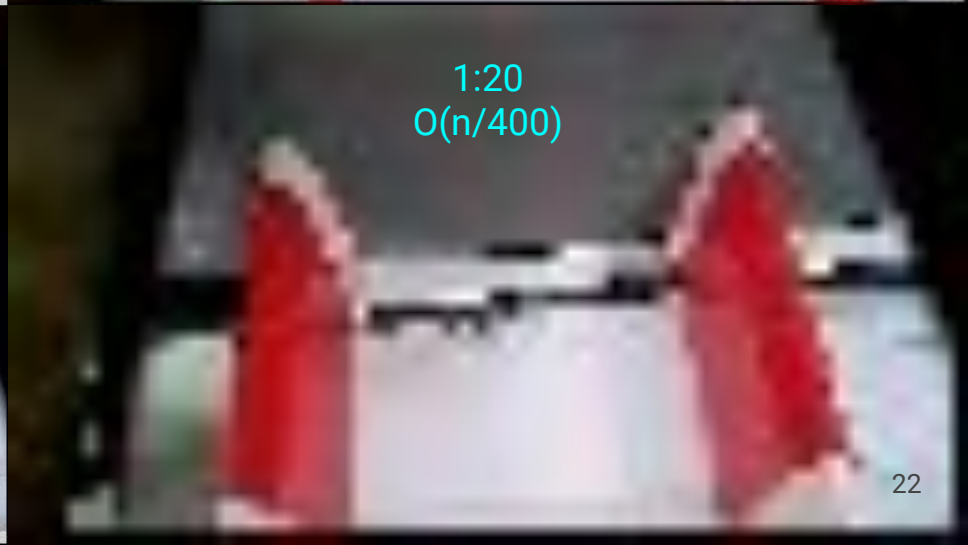


Target Area Bounding Region


1:1
O(n)


1:5
O(n/25)


1:2
O(n/4)
Downsample


1:20
O(n/400)

Use C/C++ and Integer Arrays

Machine Learning

The Solution to Variability

General Overview

Basic Algorithms

Good Practices

Perceptron

Linear model

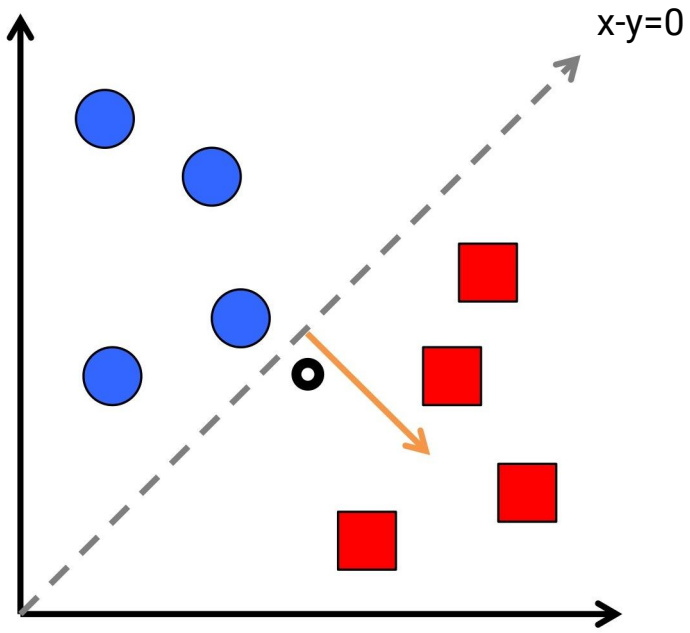
Weights features

Ax + By + C > 0

Requires user labeled data

Any number of features

3 features for colorspace

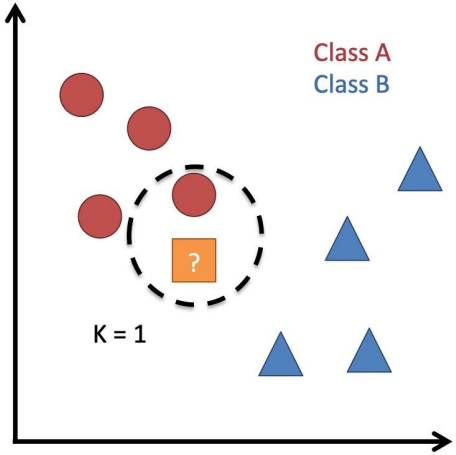


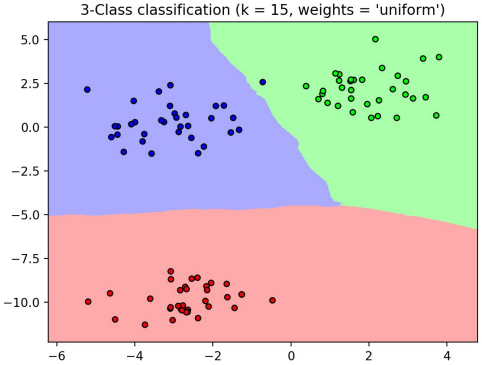
K-Nearest Neighbors

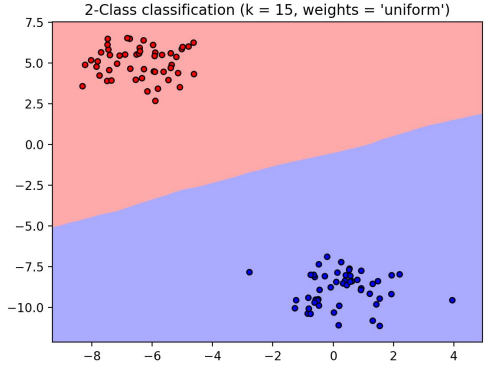
Simple classification

Majority voting system

The kth elements around the new value



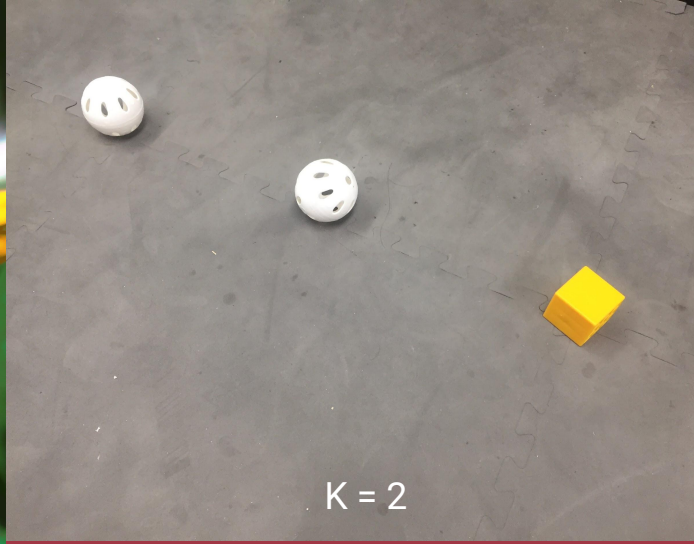




K-Means

Image Segmentation

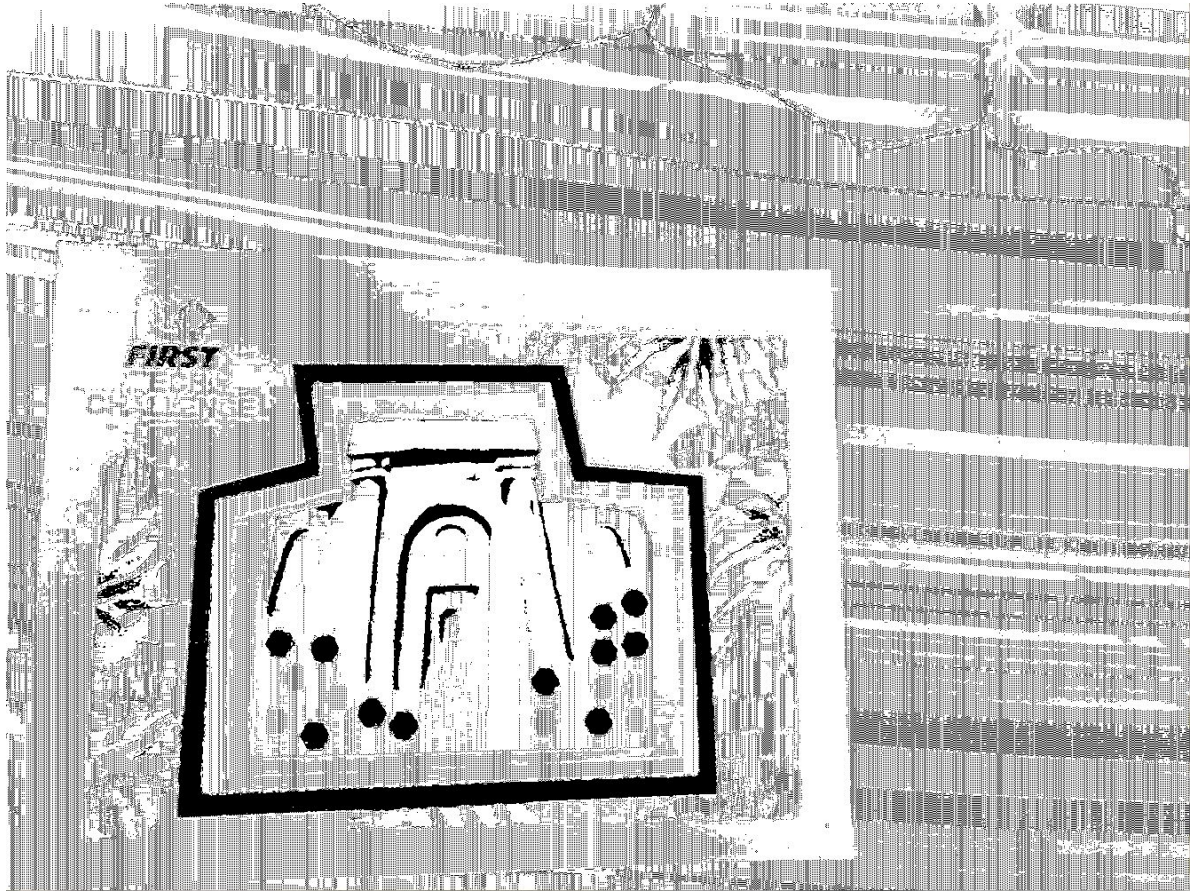
Unsupervised



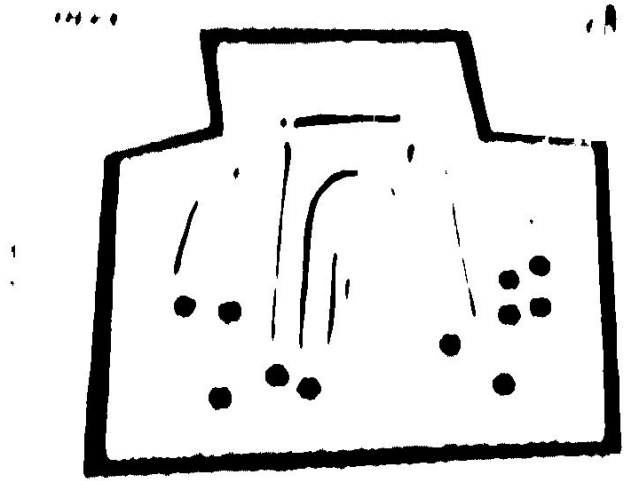
Pictograph Walkthrough

Step 1: Source - from phone camera

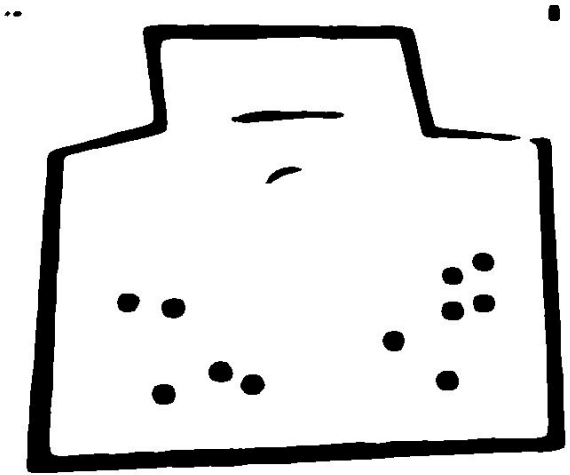
Step 2: Increase edge difference - break up image



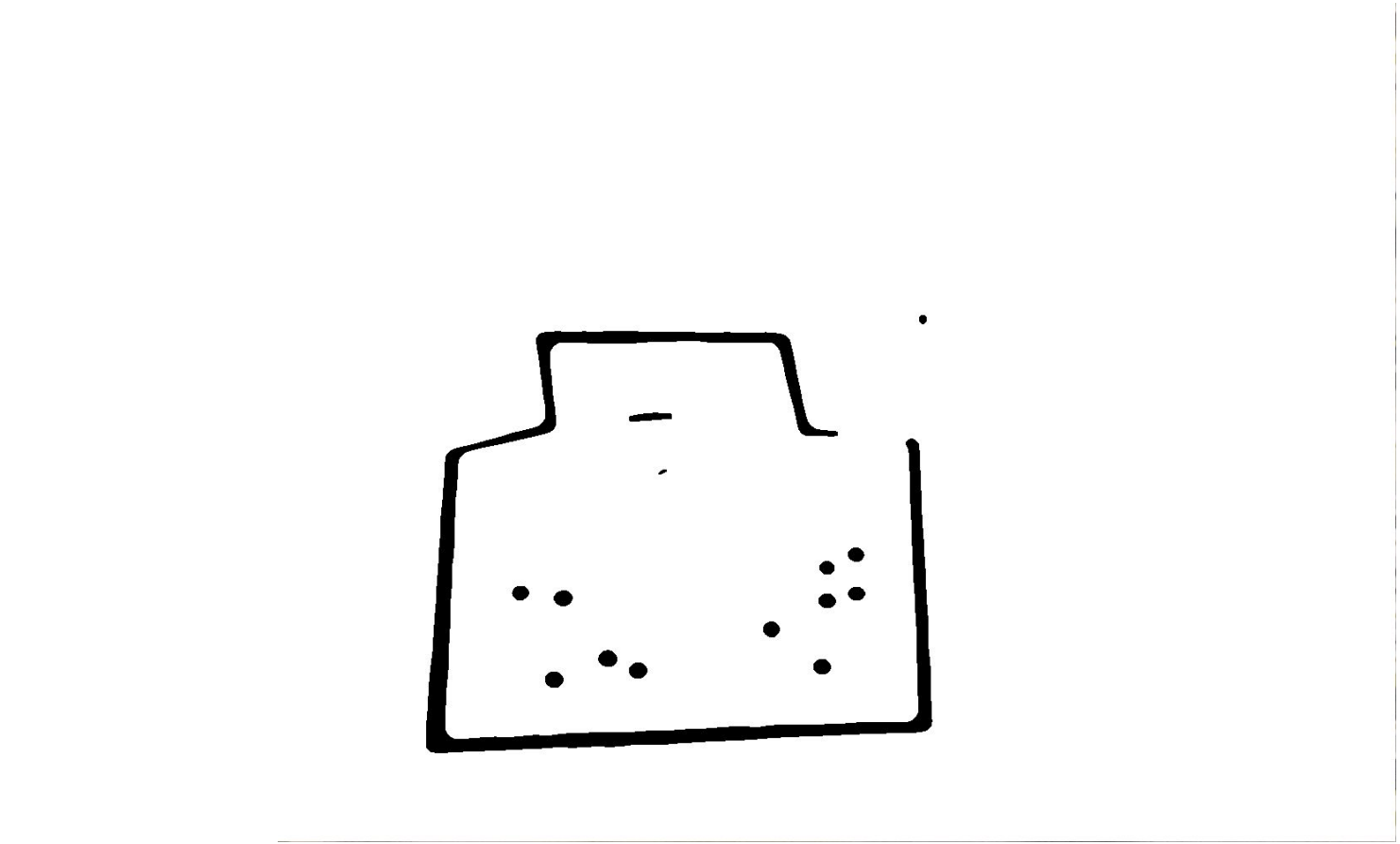
Step 3: Threshold black filter - Convert to binary colors



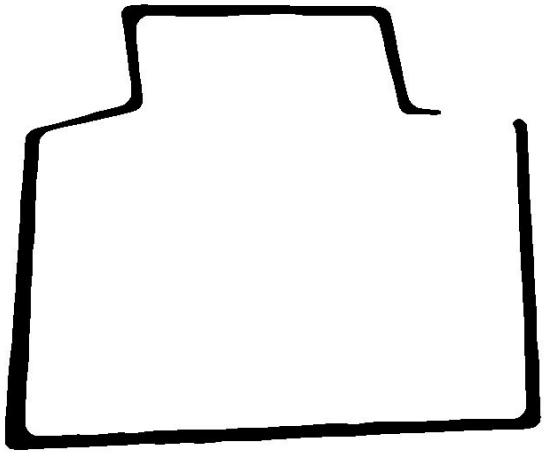
Step 4: Vertical density filter - first noise reduction pass



Step 4: Horizontal density filter - second noise reduction pass



Step 5: Square density filter - third noise reduction pass



Step 6: Find boarder - count dots inside

Contact

Sachin Shah

inventshah@gmail.com

Github: inventshah