

Computer Vision / OpenCV

<http://goo.gl/Jw7ov9>

Outline

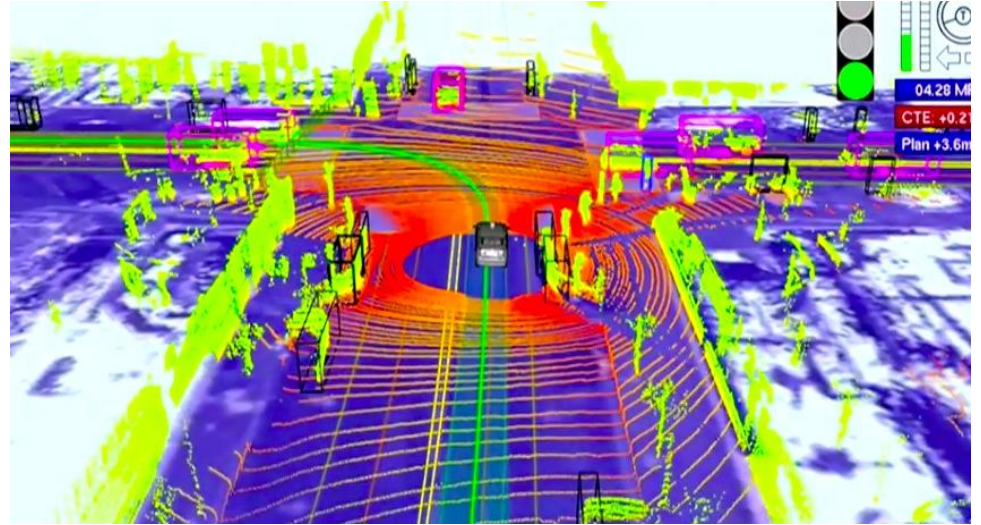
- What is computer vision? Why is it difficult/interesting?
- What is OpenCV?
- Code walkthroughs
- Examples + Demos!

What is Computer Vision?

- Acquiring,
 - Processing,
 - Analyzing,
 - Understanding images
- ... to make decisions.

Some examples

- Mars Rover
- Autonomous cars



What is OpenCV?

- Open Source Computer Vision
- Library for doing CV

Installing OpenCV

- Unfortunately can be quite confusing
- Mac
 - Follow [this](#) tutorial!
- Windows
 - Follow [this](#) tutorial instead!
 - **“Installing OpenCV from prebuilt binaries”**
section only - no Visual Studios or matplotlib needed

Code Snippet

```
import cv2
window_name = "Webcam!"
cam_index = 0 # Default camera is at index 0.
cv2.namedWindow(window_name, cv2.CV_WINDOW_AUTOSIZE)
cap = cv2.VideoCapture(cam_index) # Video capture object
cap.open(cam_index) # Enable the camera
while True:
    ret, frame = cap.read()
    if frame is not None:
        cv2.imshow(window_name, frame)
    k = cv2.waitKey(10) & 0xFF
    if k == 27: # Escape key
        cv2.destroyAllWindows()
        cap.release()
    break
```


Thresholding

- For each pixel, check some quantity
 - Usually the value or location of the pixel
 - e.g. $R+G+B > 100$
- If so, change pixel value to X
- Else, change pixel value to Y
 - e.g. if yes, make pixel black, else make pixel white

Thresholding



Thresholding

- Manually
- Using OpenCV

Blurring

- Replace each pixel with the average of its neighbors

Blurring

- Replace each pixel with the average of its neighbors
- Manually
 - loop through pixels
 - loop through neighbors, adding up their values
 - divide by the number of neighbors

Blurring

- Replace each pixel with the average of its neighbors
- Manually
 - loop through pixels
 - loop through neighbors, adding up their values
 - divide by the number of neighbors
- Using OpenCV
 - `cv2.blur(img, ksize)`
 - `ksize` - amount of blur

Convolution

- More general form of manual technique for blurring

Blurring

| | | |
|-------|-------|-------|
| $1/9$ | $1/9$ | $1/9$ |
| $1/9$ | $1/9$ | $1/9$ |
| $1/9$ | $1/9$ | $1/9$ |

Convolution

| | | |
|---|---|---|
| a | b | c |
| d | e | f |
| g | h | i |

Edge Detection

- How can we detect edges using convolution?

Edge Detection

- How can we detect edges using convolution?

Sobel filter!

| | | |
|----|----|----|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

| | | |
|---|---|----|
| 1 | 0 | -1 |
| 2 | 0 | -2 |
| 1 | 0 | -1 |

Circle Detection



Hough Circles

- A method to detect circles in an image
- Manually = complicated
- OpenCV = easy!

```
cv2.HoughCircles(image, cv2.cv.CV_HOUGH_GRADIENT,  
dp, minDist)
```

- `cv2.cv.CV_HOUGH_GRADIENT` - OpenCV constant
- `dp` - resolution
- `minDist` - minimum distance between the centers of two detected circles
- `minRadius/maxRadius` - optional arguments

Examples + Demos!

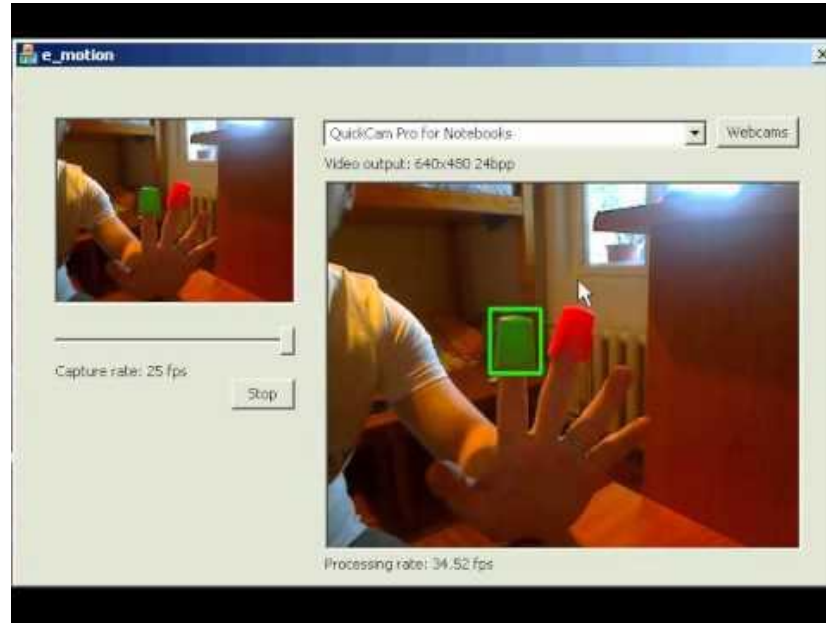


Detecting Color Objects
with OpenCV C++

<https://www.youtube.com/watch?v=JcDZxohDq2w&spfreload=10>

0:40 to 1:15

Fast color based tracking



<https://www.youtube.com/watch?v=UjXu1kqfK-A>

0:07 to 0:25

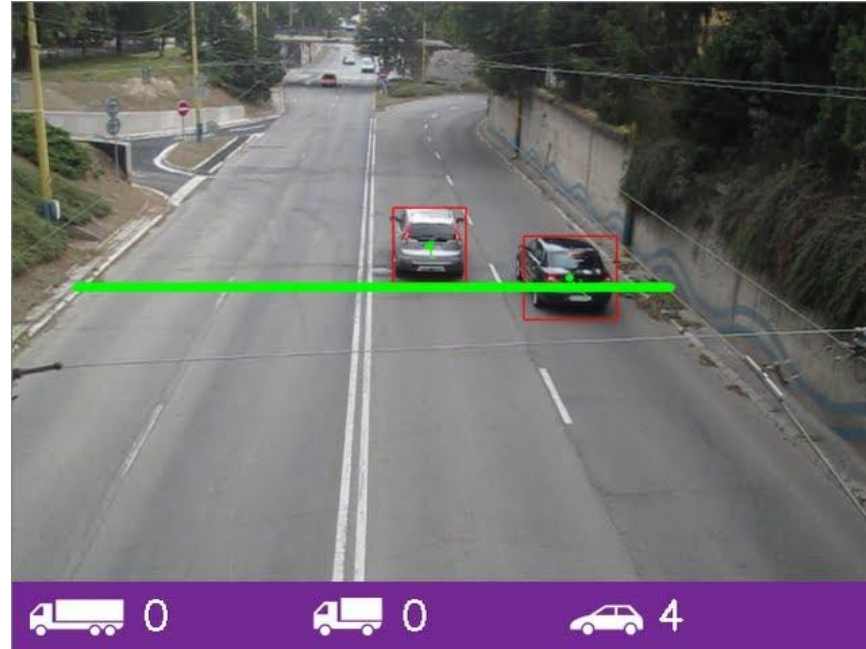
Realtime face detection



https://www.youtube.com/watch?v=HTk_UwAYzVk

0:00 to 0:20

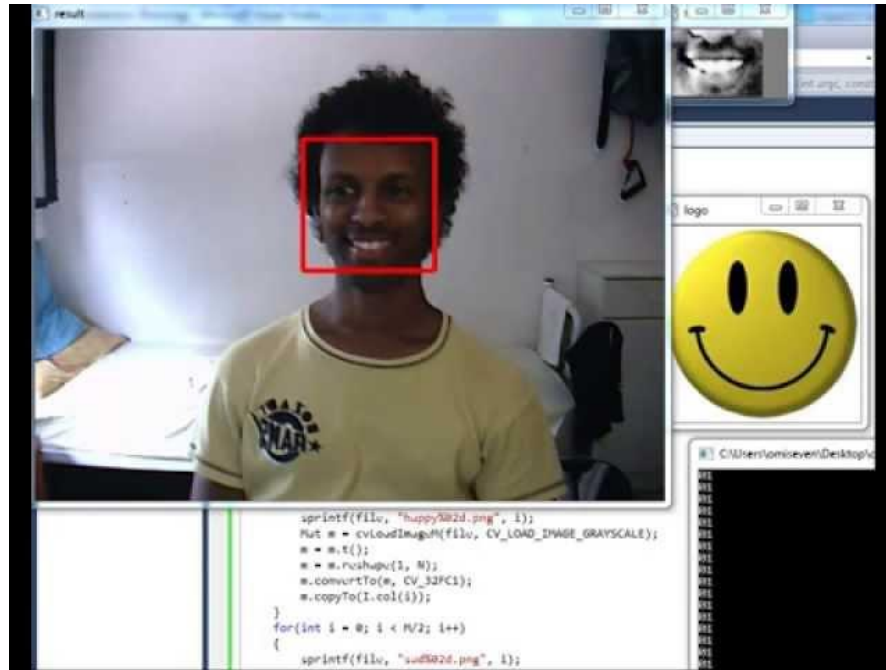
Classifying vehicles



https://www.youtube.com/watch?v=z1Cvn3_4yGo

0:00 to 0:25

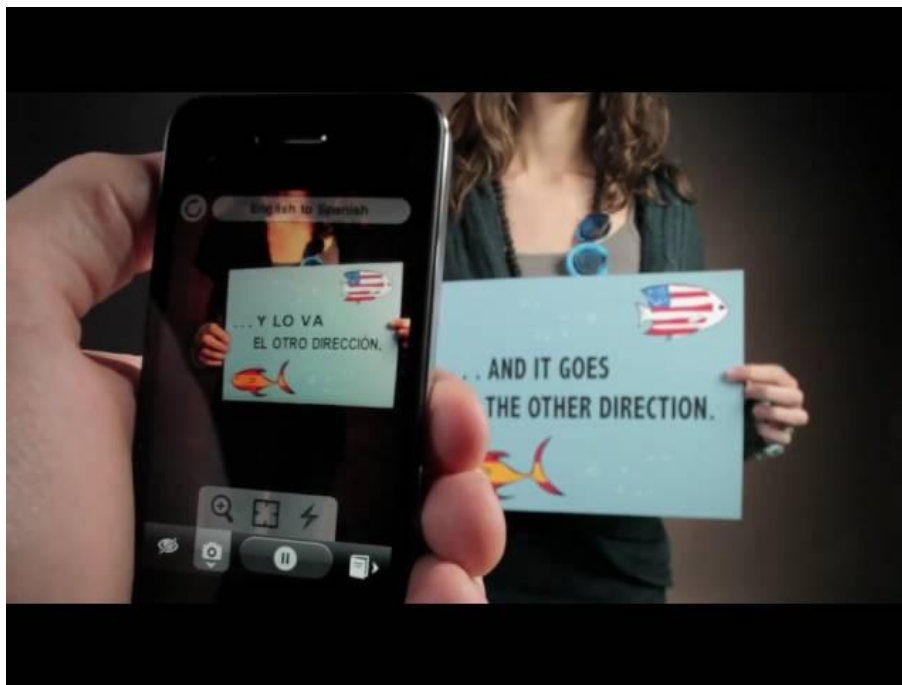
Smile detection



<https://www.youtube.com/watch?v=AA78saEk29E>

0:00 to 0:25

WordLens



<https://www.youtube.com/watch?v=h2OfQdYrHRs>

0:00 to 0:50

Term Projects!

- Photo Calculator
 - <https://www.youtube.com/watch?v=iX6qbuyKka4&feature=youtu.be>
- Rubik's Cube solver
 - <https://www.youtube.com/watch?v=QrLaajSDnTw>
- Baseball game
 - https://www.youtube.com/watch?v=AWREaR5q8MA&feature=em-upload_owner
- AudioShop
 - <https://www.youtube.com/watch?v=XO5OXKEO8rE&feature=youtu.be>
- Magneto's Maze
 - <https://www.youtube.com/watch?v=fev12WxDuJl>
- Hand/palm detection and drawing
 - <https://www.youtube.com/watch?v=oH0ZkfFoeYU>
- Playschool
 - <https://www.youtube.com/watch?v=LTNfe19qjlg>

Additional resources

- http://docs.opencv.org/trunk/doc/py_tutorials/py_tutorials.html
- <http://cbarker.net/opencv/>