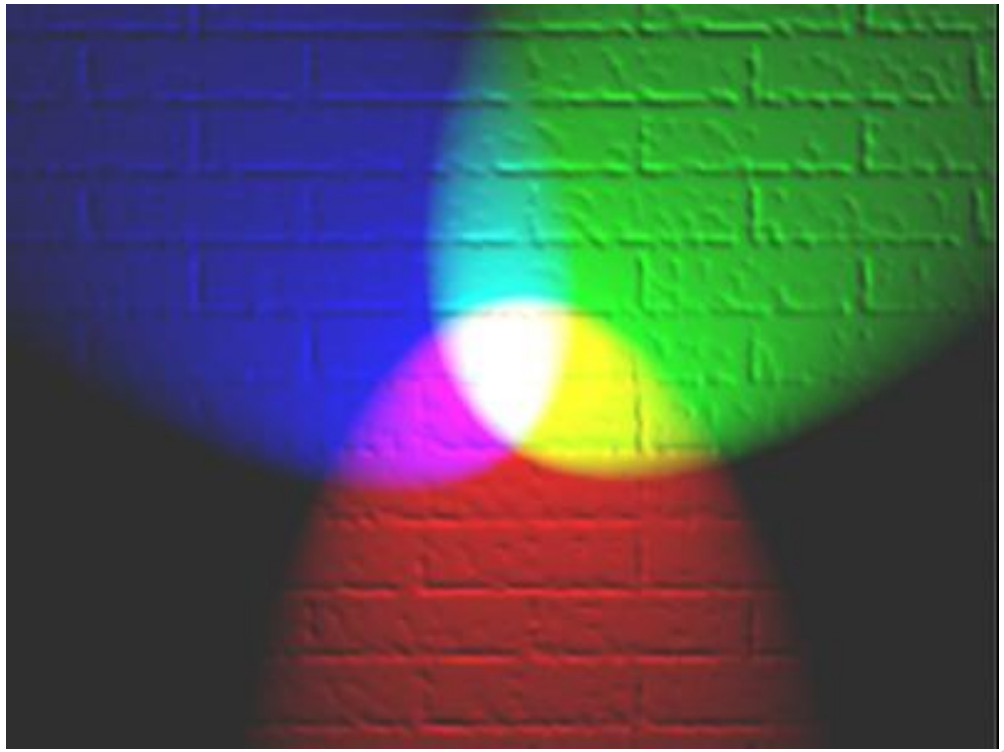
Lecture 12

Recap

- HTML Canvas
  - Comparison to SVG browser graphics
  - Drawing shapes
  - Basic Trigonometry
  - Basic collision detection
  - Animation and User Interaction
    - Examples
  - Advanced collision handling
  - Linear Algebra and Transformations
- SVG
- D3
- 3D – WebGL/Three.js
- Computer Vision – Edge and corner detection

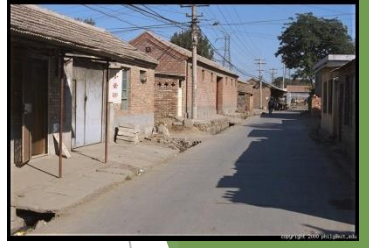
Color spaces

- How can we represent colour?

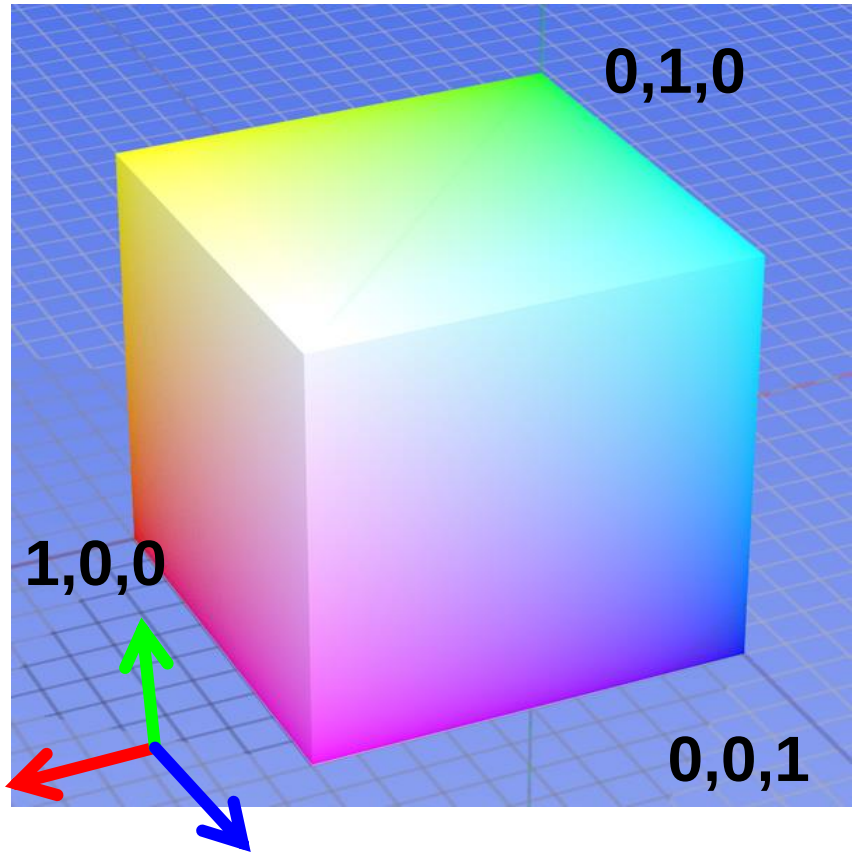


http://en.wikipedia.org/wiki/File:RGB_illumination.jpg

Color spaces: RGB



Default color space



R
(G=0,B=0)

G
(R=0,B=0)

B
(R=0,G=0)

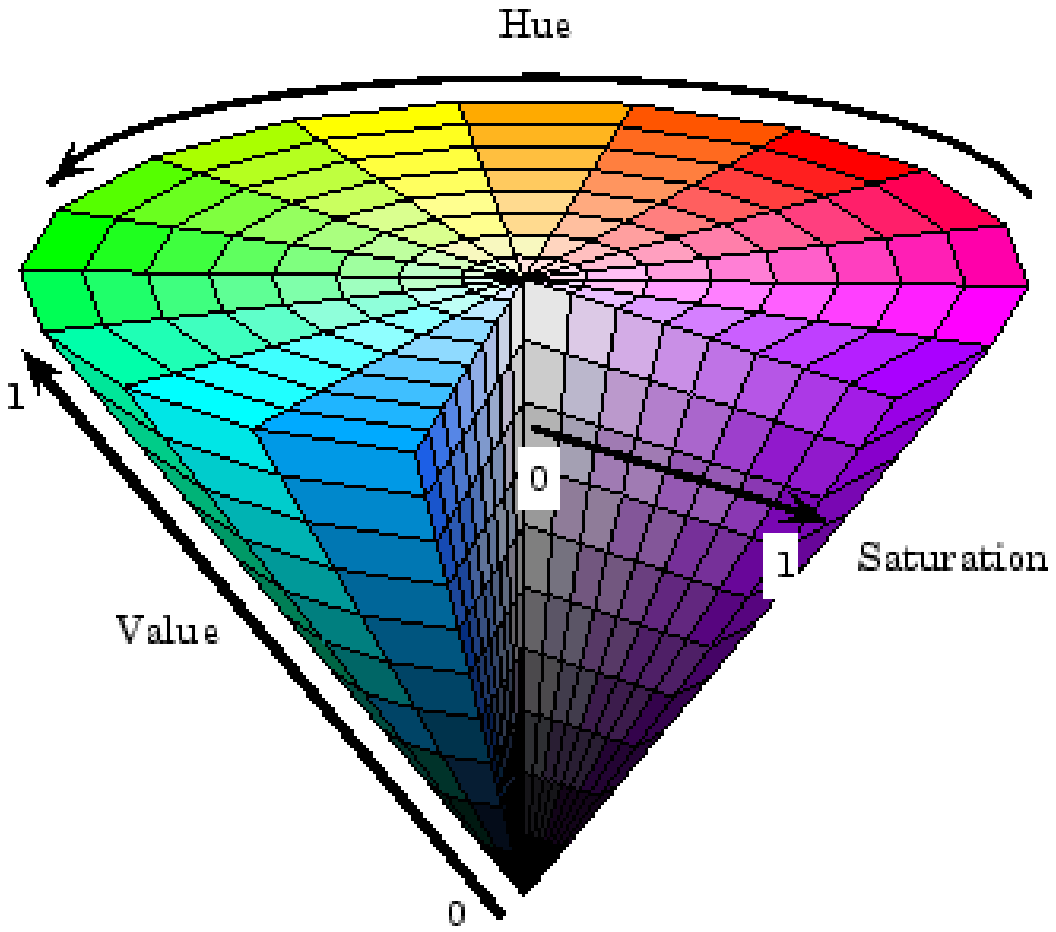
Some drawbacks

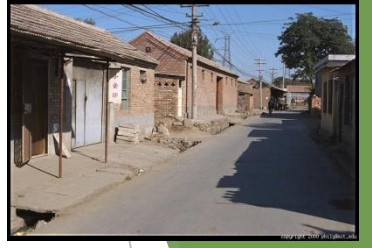
- Strongly correlated channels
- Non-perceptual

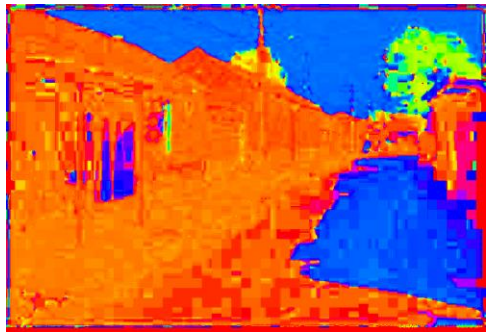
Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Color spaces: HSV

Intuitive color space







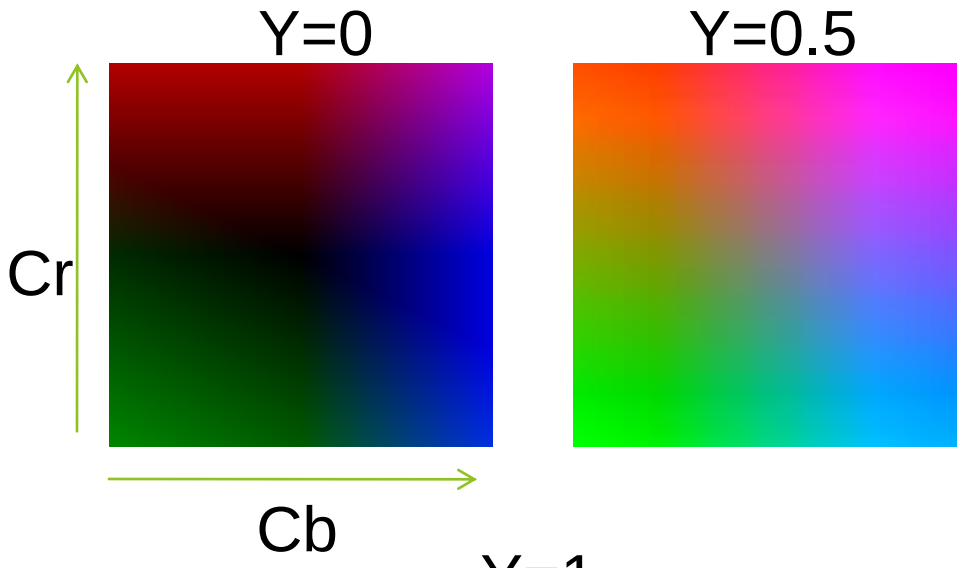
H
(S=1,V=1)

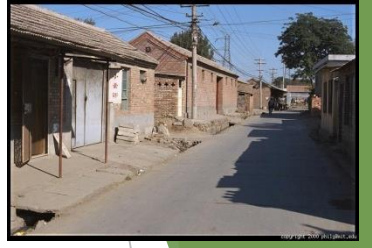
S
(H=1,V=1)

V
(H=1,S=0)

Color spaces: YCbCr

Fast to compute, good for compression, used by TV





Y (luma)
(Cb=0.5,Cr=0.5)

Cb (Chrominance - blue)
(Y=0.5,Cr=0.5)

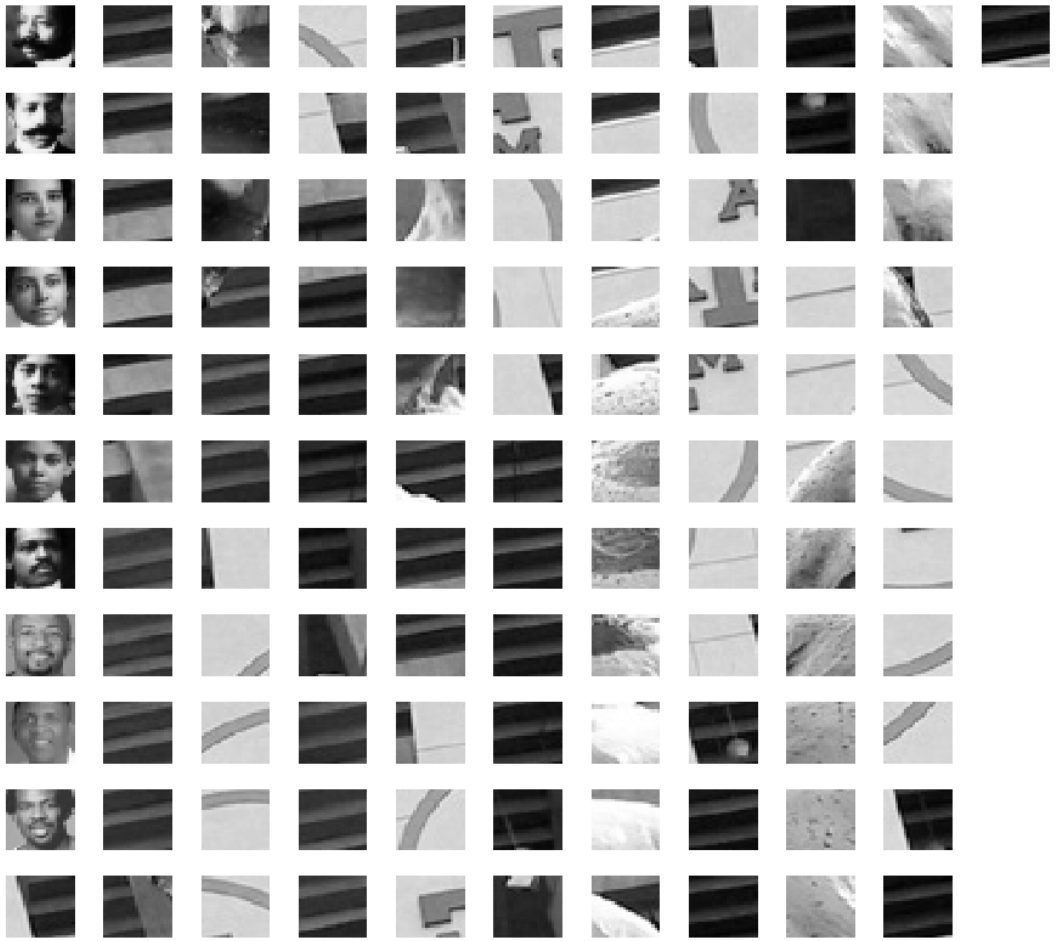
Cr (Chrominance - red)
(Y=0.5,Cb=05)

Learning to Detect Faces (Viola Jones)

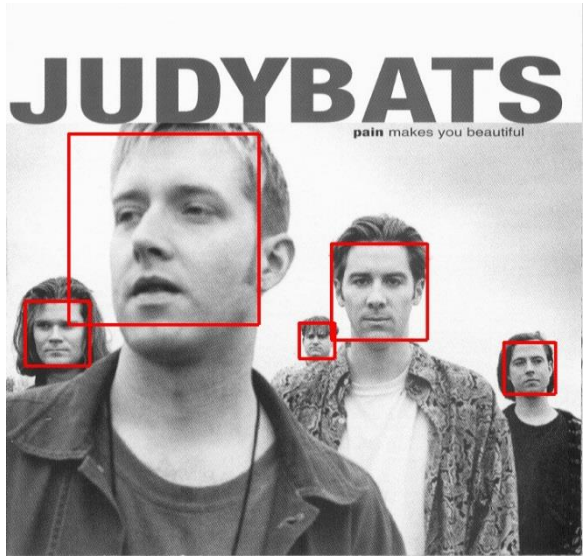
- Training - Examples of 24x24 images with faces

Learning to Detect Faces (Viola Jones)

- Training



Notice detection at multiple scales



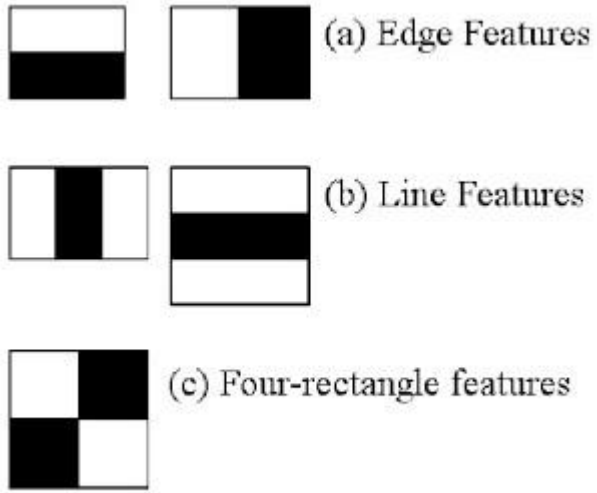
Viola Jones face detection

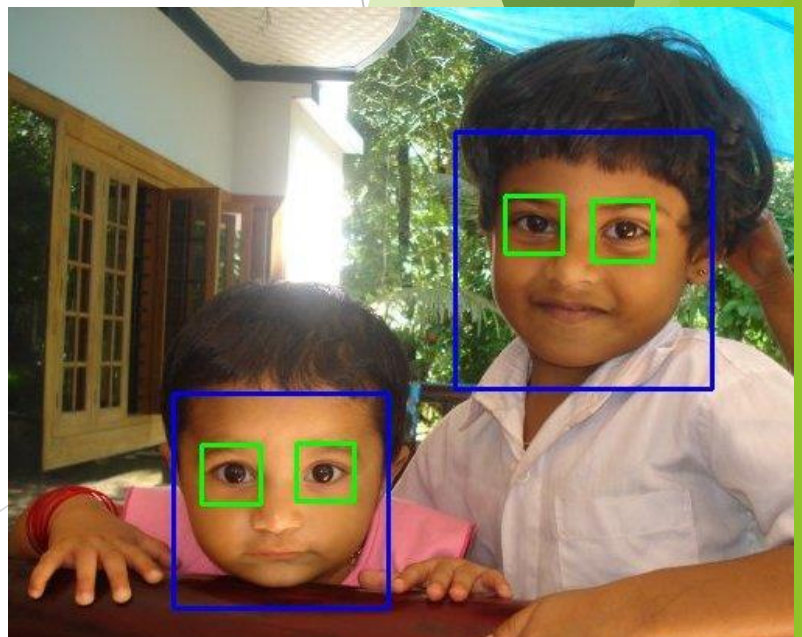
```
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

img = cv2.imread('sachin.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

- Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it.
- For this, haar features shown in below image are used.
- Each feature is a single value obtained by subtracting the sum of pixels under the white rectangle from the sum of pixels under the black rectangle.

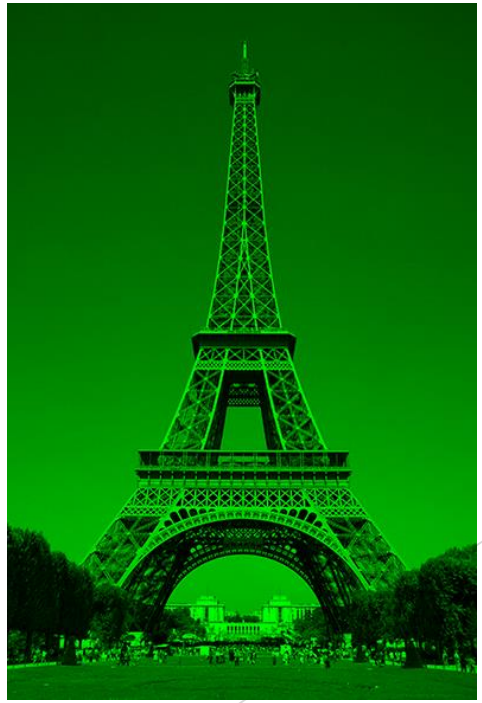


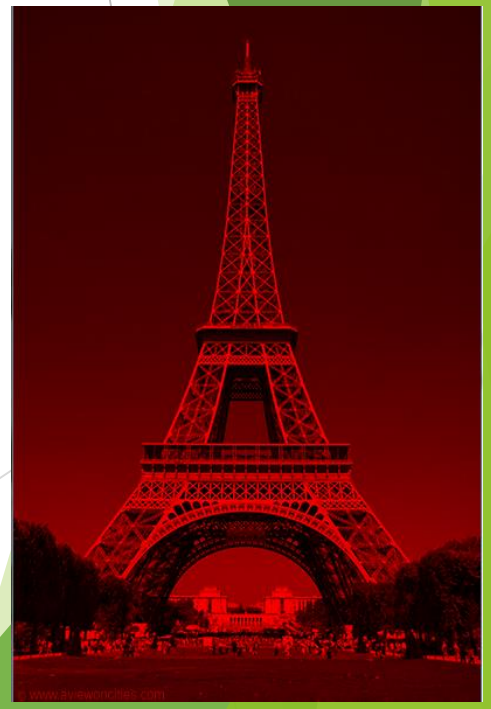


Labs

- Splitting image into colour channels:
  - Using nested for loop to achieve this

```
for i in range(len(img)):
    for j in range(len(img[i])):
        for k in range(len(img[i][j])):
```





Labs

- Transform image into HSV and splitting in same way
- Face detection
  - http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html#face-detection

Lab Submission

- Lab 10,11 and 12
- Python files and screenshots from labs
- Due 11th December, 2016