Lecture 11

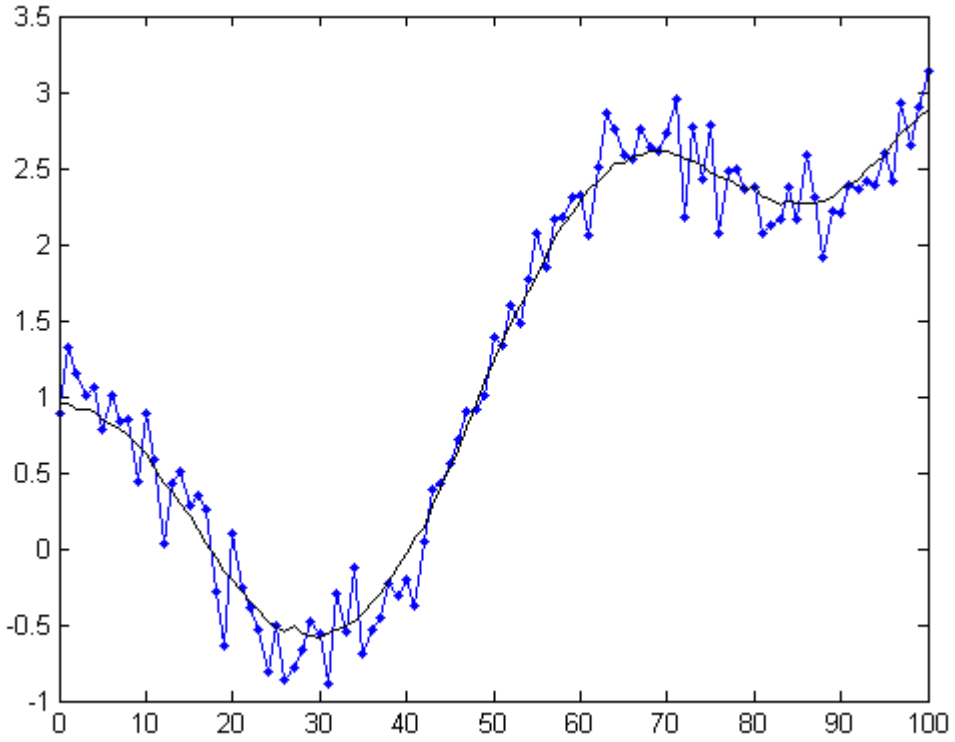
Recap

- 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

Edges – recap

- Similarities/Differences

Image filtering – moving average – 1d



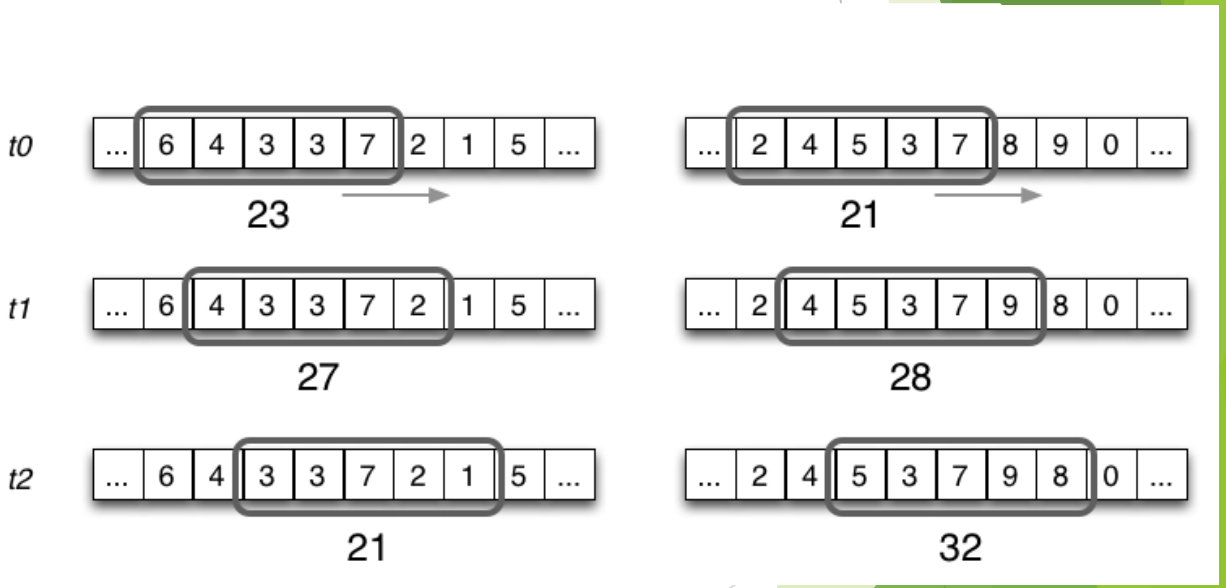
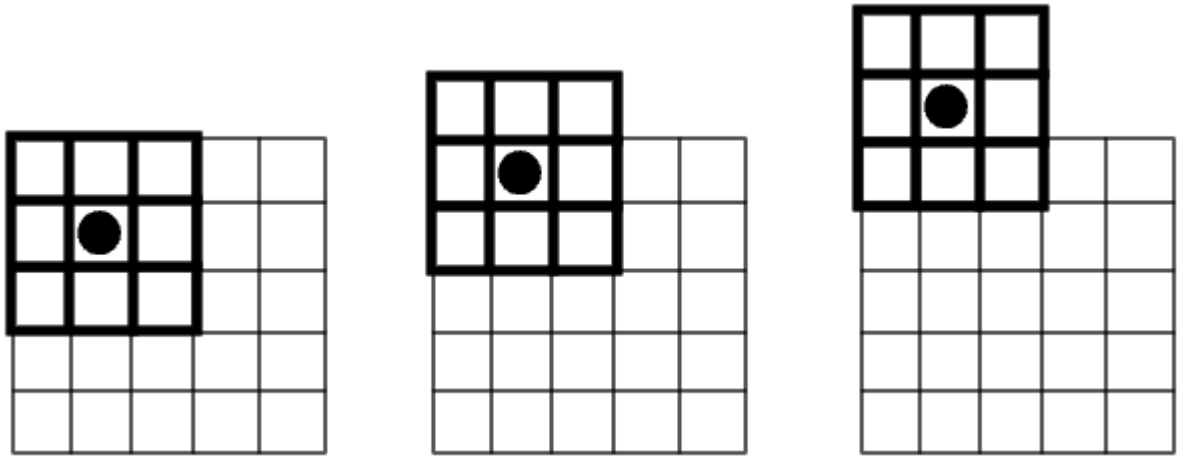
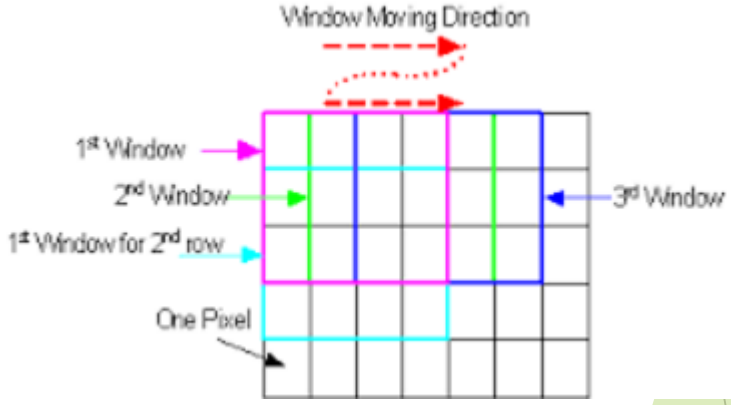


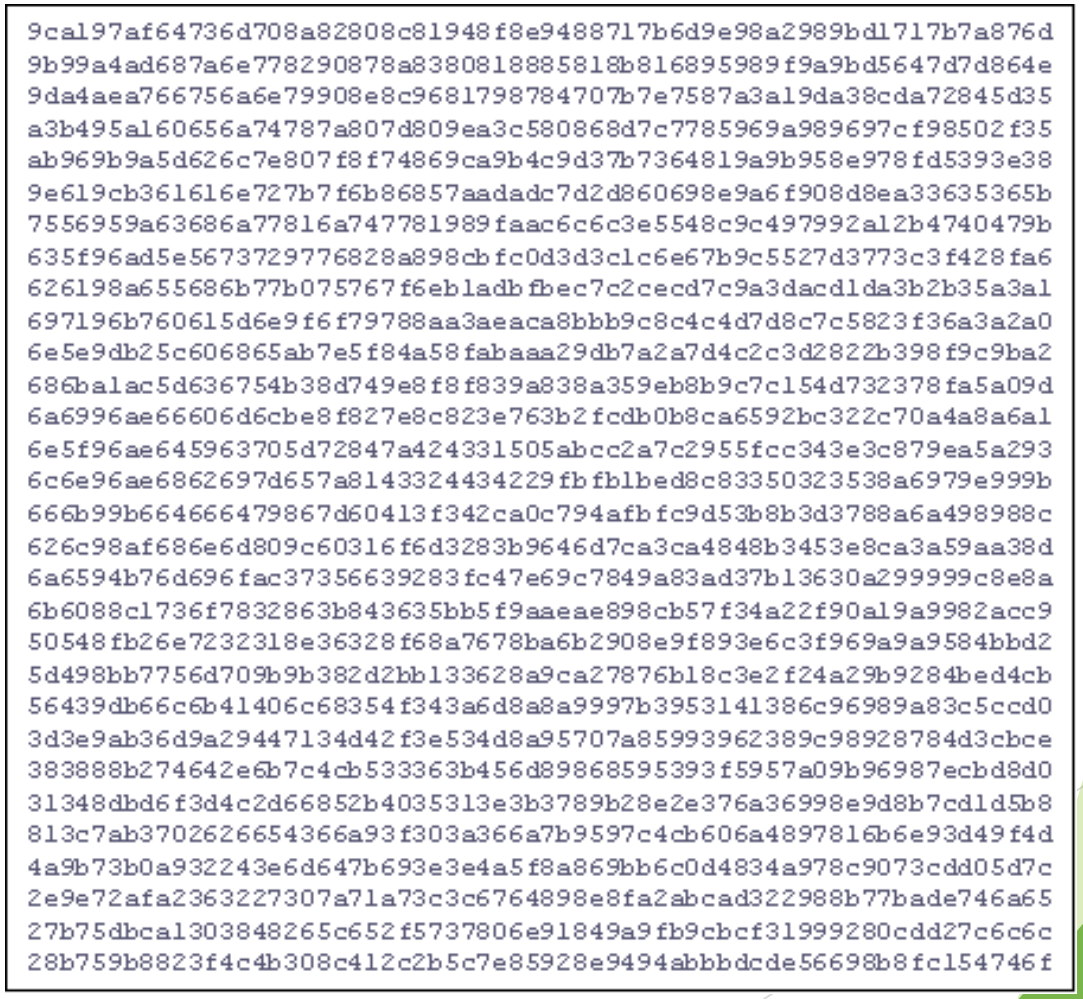
Image filtering – moving average – 2d



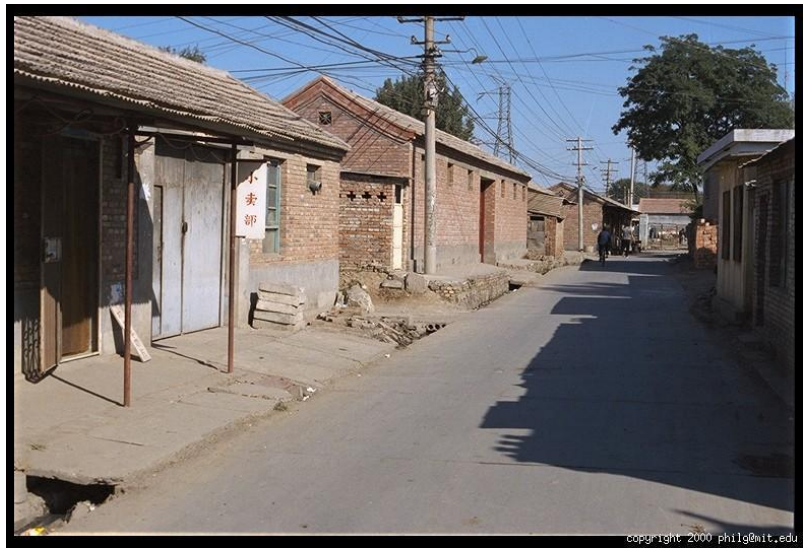


Images in OpenCV

- Grayscale – 2d array
- Colour – 3d array



Colour Images



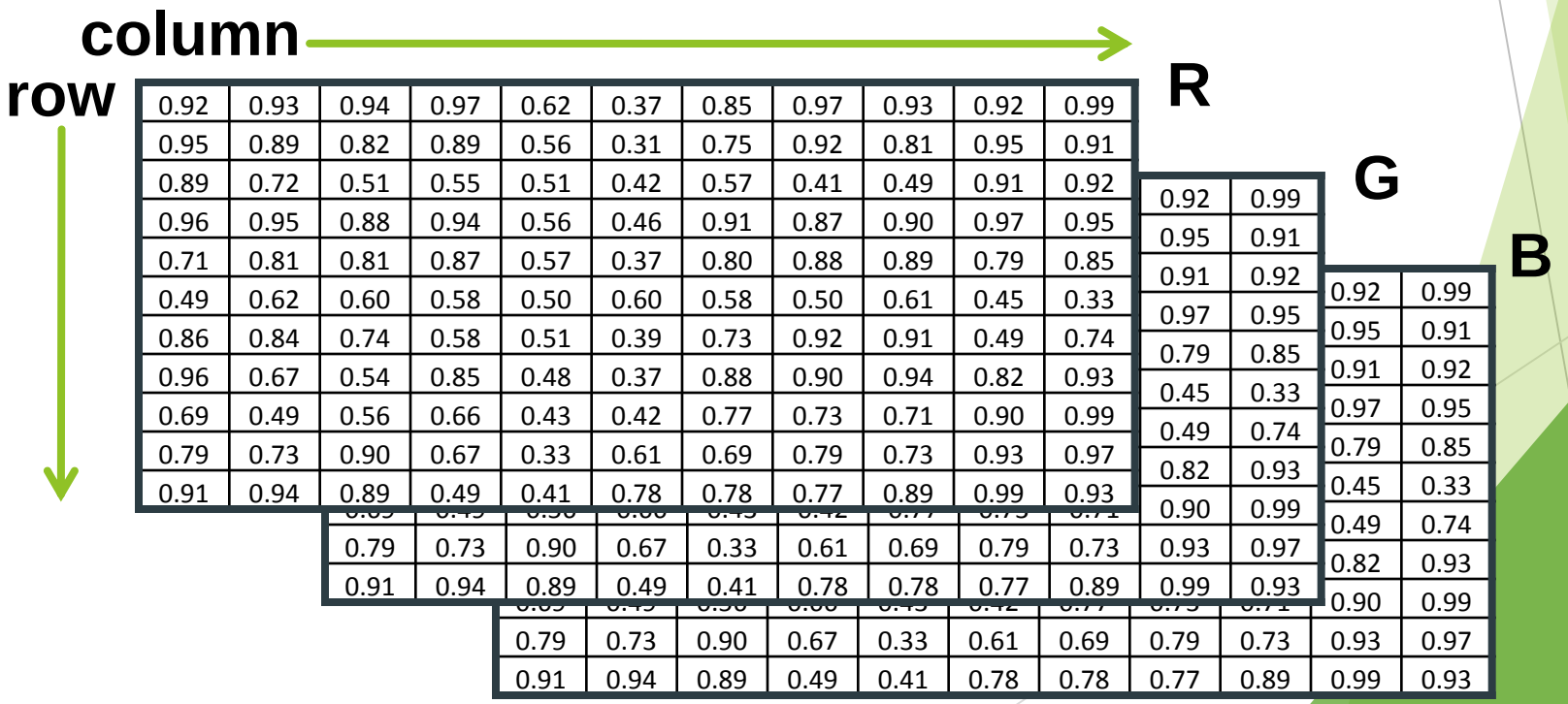
R

G

B

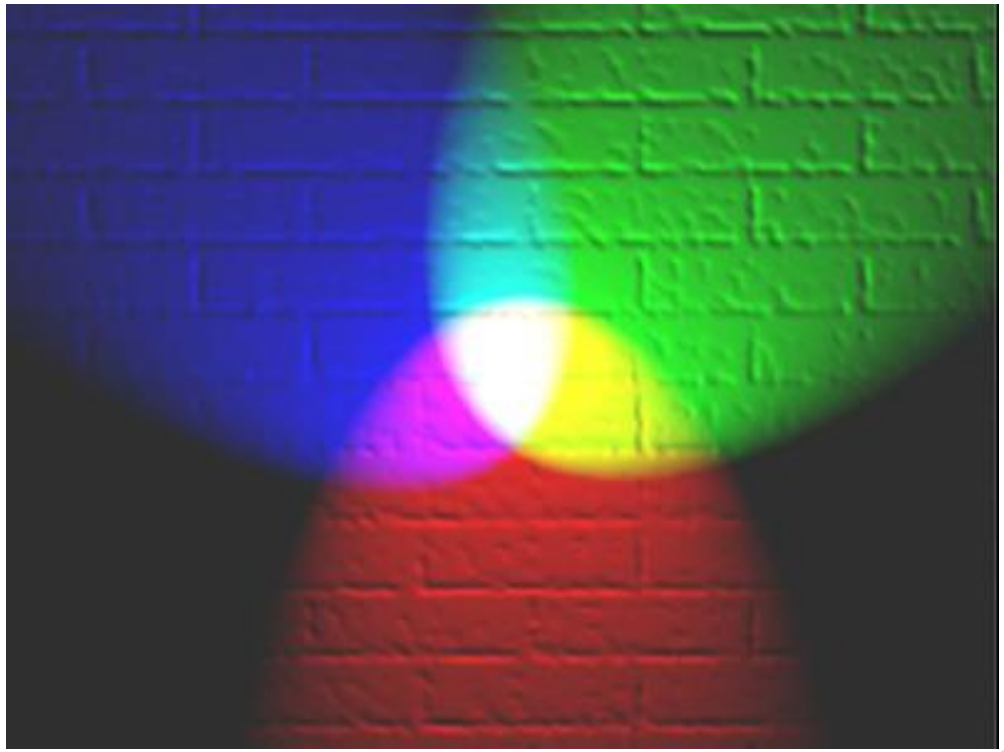
Images in Matlab/Octave/OpenCV

- Images represented as a matrix
- Suppose we have a NxM RGB image called “im”
  - im(1,1,1) = top-left pixel value in R-channel
  - im(y, x, z) = y pixels down, x pixels to right in the z^{th} channel
  - im(N, M, 3) = bottom-right pixel in B-channel
- imread(filename) returns a uint8 image (values 0 to 255)
  - Convert to double format (values 0 to 1) with im2double



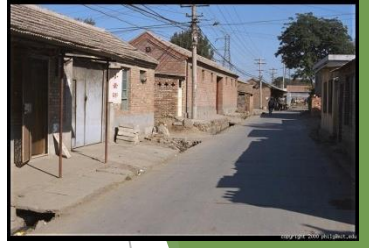
Color spaces

- How can we represent colour?

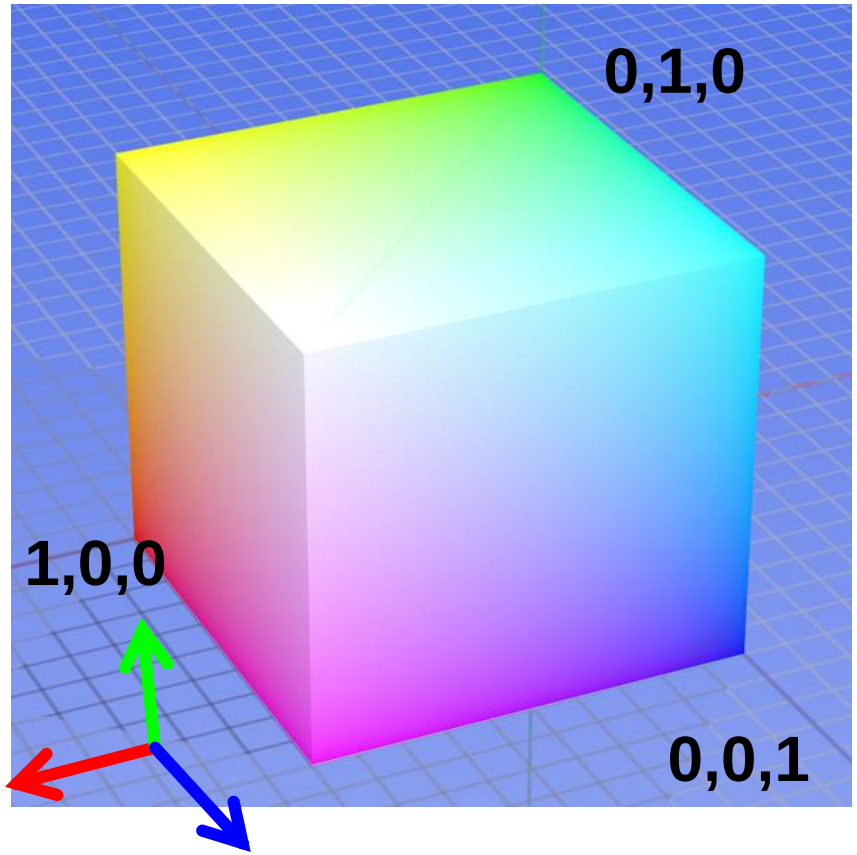


http://en.wikipedia.org/wiki/File:RGB_illumination.jpg

Color spaces: RGB



Default color space



R
(G=0,B=0)

G
(R=0,B=0)

B
(R=0,G=0)

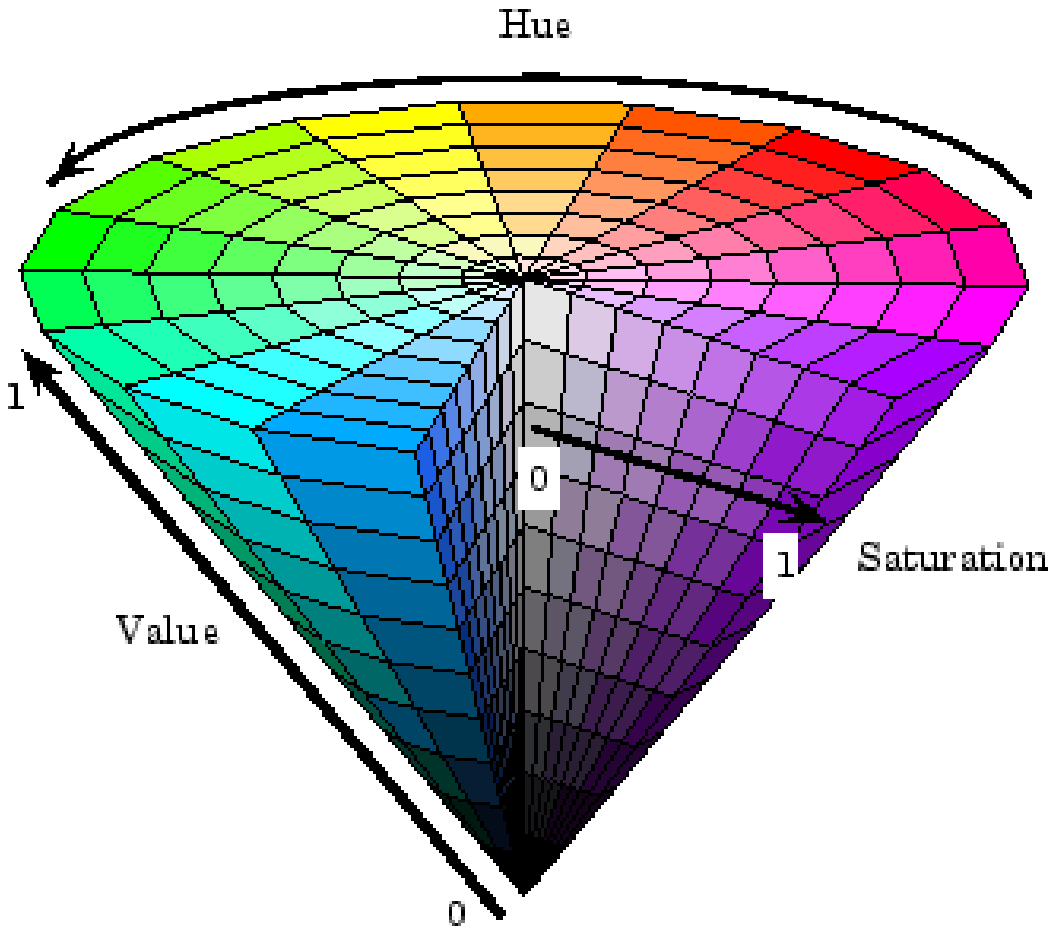
Some drawbacks

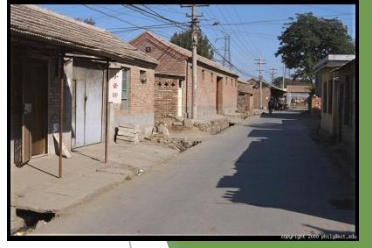
- Strongly correlated channels
- Non-perceptual

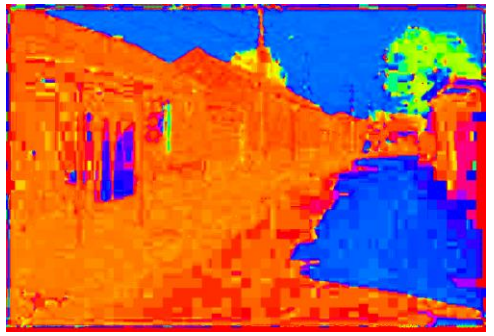
Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Color spaces: HSV

Intuitive color space







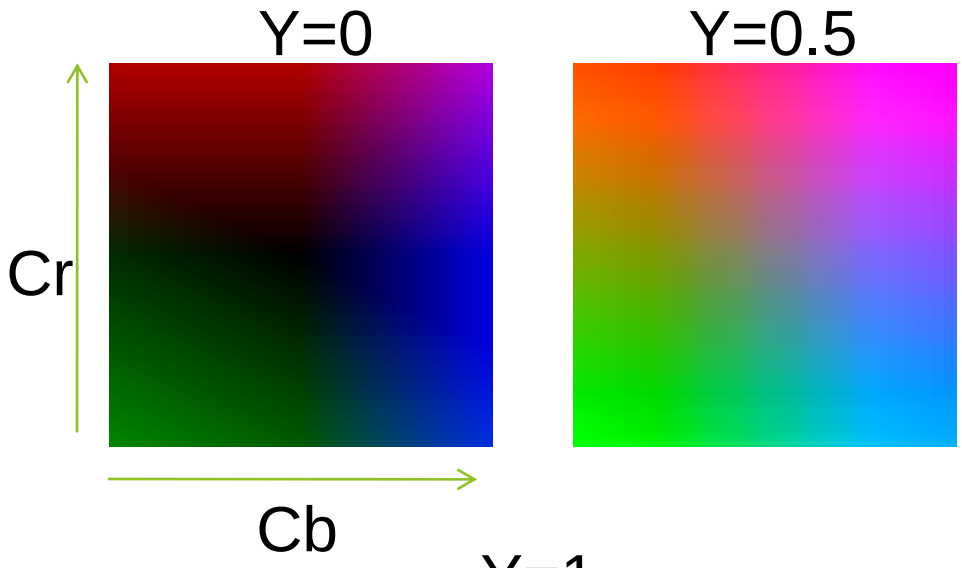
H
(S=1,V=1)

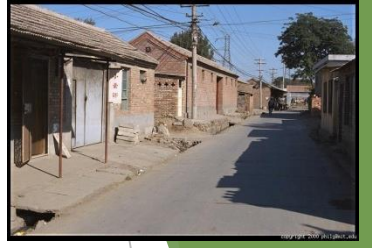
S
(H=1,V=1)

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)

Features in Computer Vision

- We've seen edges
- Corner detection also commonly used
- Like making a jigsaw puzzle
  - How does a computer know what it's seeing from a small part of an image
  - How does a human do it
    - Look for specific patterns or specific features which are unique, which can be easily tracked and easily compared

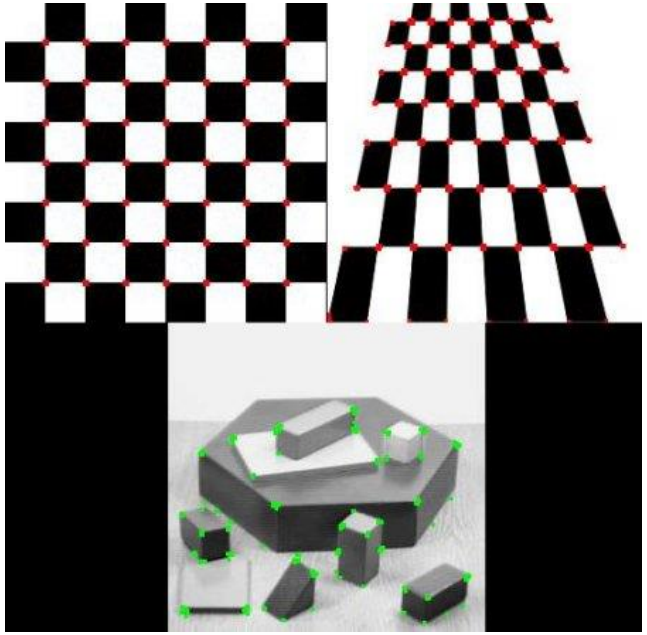
Features

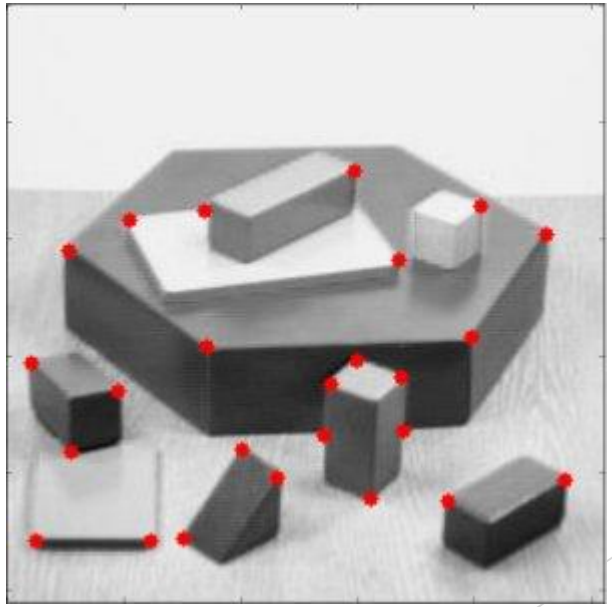
Features

- A and B are flat surfaces, and they are spread in a lot of area. It is difficult to find the exact location of these patches.
- C and D are much simpler. They are edges of the building. You can find an approximate location, but exact location is still difficult. It is because, along the edge, it is same everywhere. So an edge is a much better feature compared to flat area, but not good enough (It is good in jigsaw puzzle for comparing continuity of edges).
- Finally, E and F are some corners of the building. And they can be easily located. Because at corners, wherever you move this patch, it will look different. So they can be considered as a good feature.

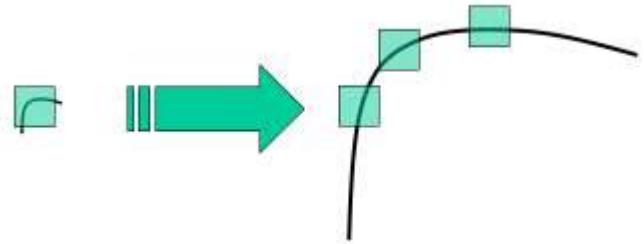
Several corner detection algorithms

- Corners are regions in the image with large variation in intensity in all the directions.
  - Harris – Early attempt (1988)
  - Shi Tomasi - Improvement (1994)
    - More appropriate for tracking





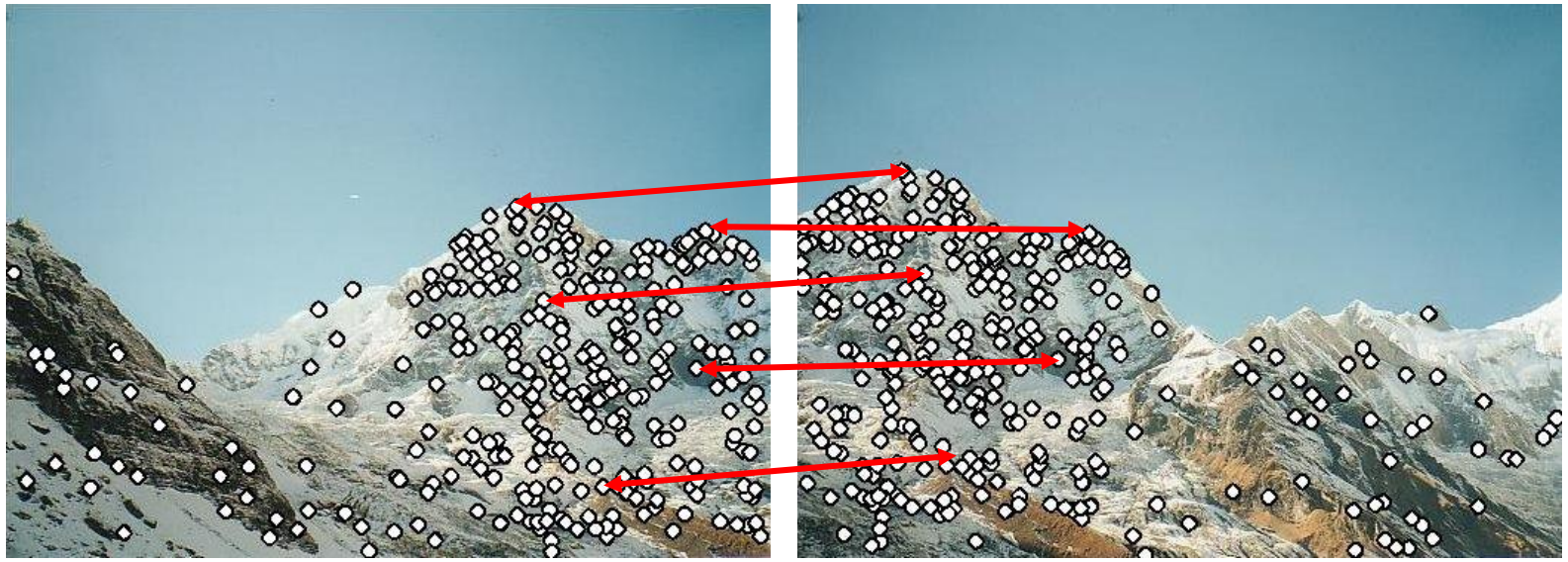
Several corner detection algorithms



- Harris and Shi Tomasi algorithms are rotation-invariant, which means, even if the image is rotated, we can find the same corners. It is obvious because corners remain corners in rotated image also.
- What about scaling? A corner may not be a corner if the image is scaled. For example, check a simple image above. A corner in a small image within a small window is flat when it is zoomed out. So Harris/Shi Tomasi is not scale invariant.
  - SIFT/SURF
    - Scale invariant

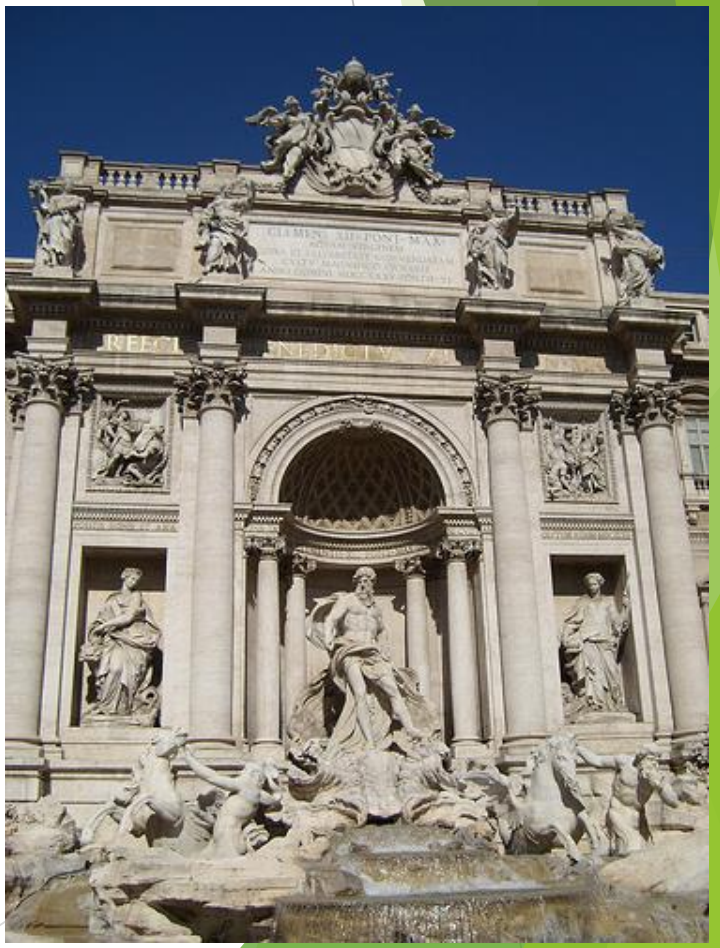
Feature Matching

- What things in this image matches with things in the other image?



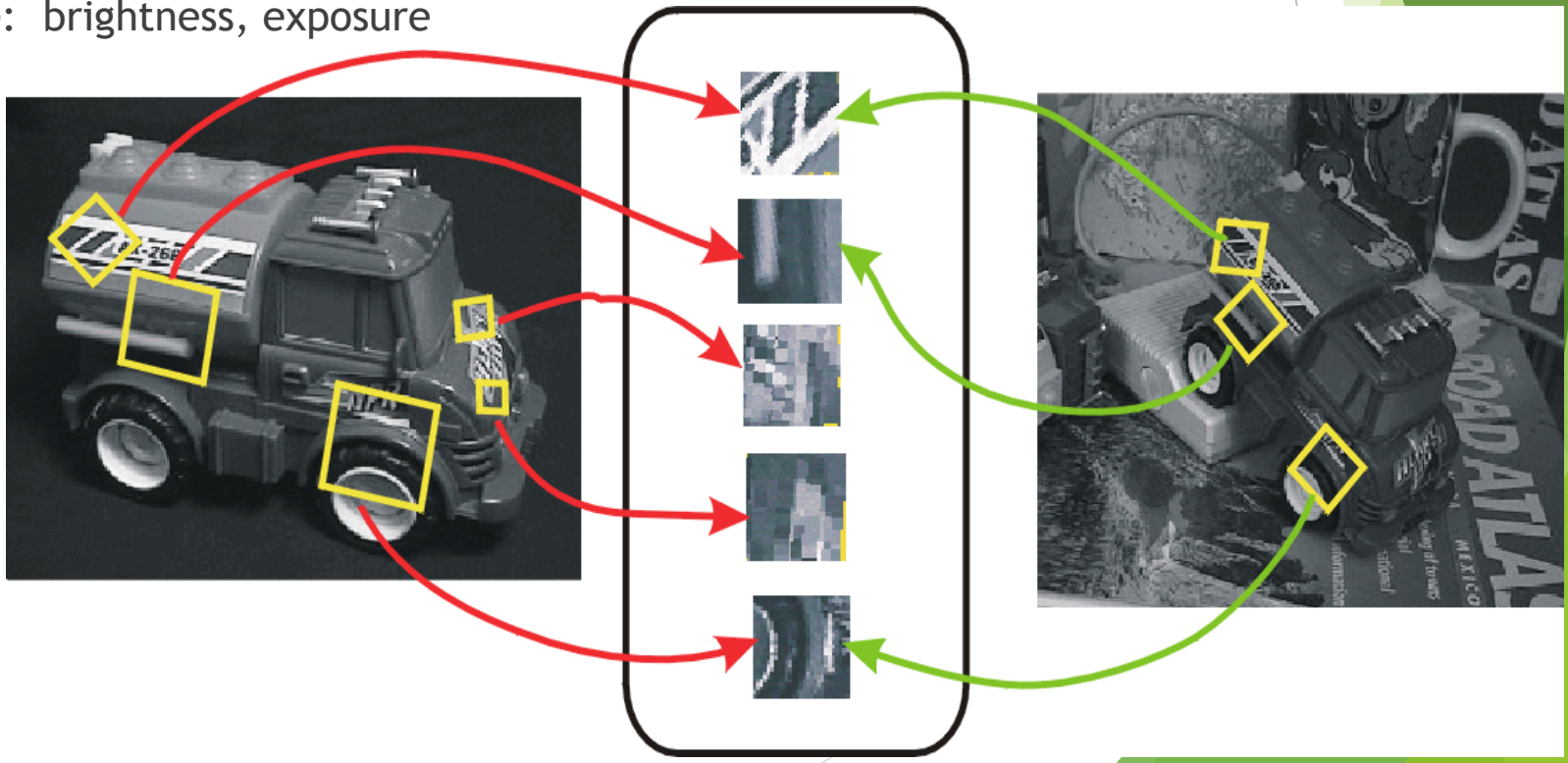
Feature Matching

- What things in this image matches with things in the other image?



Invariant local features

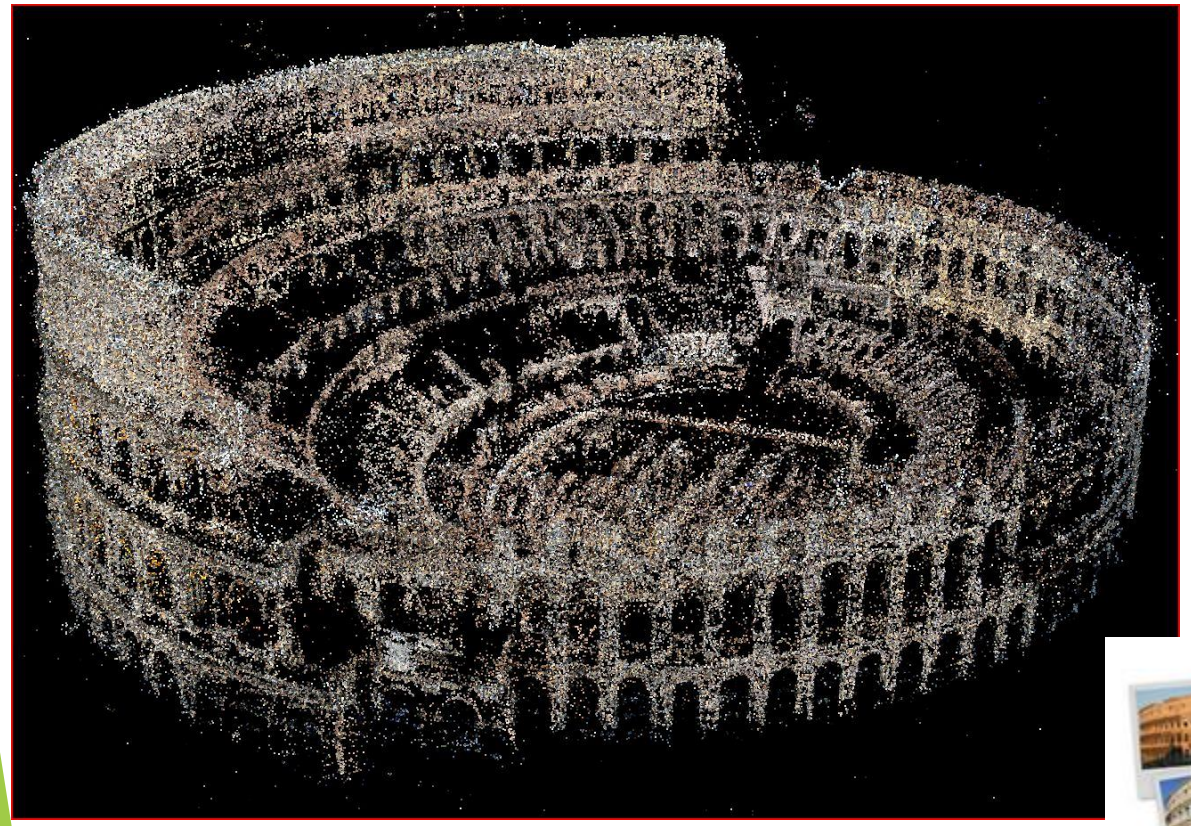
- Buzzword is “invariance”
  - Geometric invariance: translation, rotation, scale
  - Photometric invariance: brightness, exposure



Feature Detection/Matching Applications

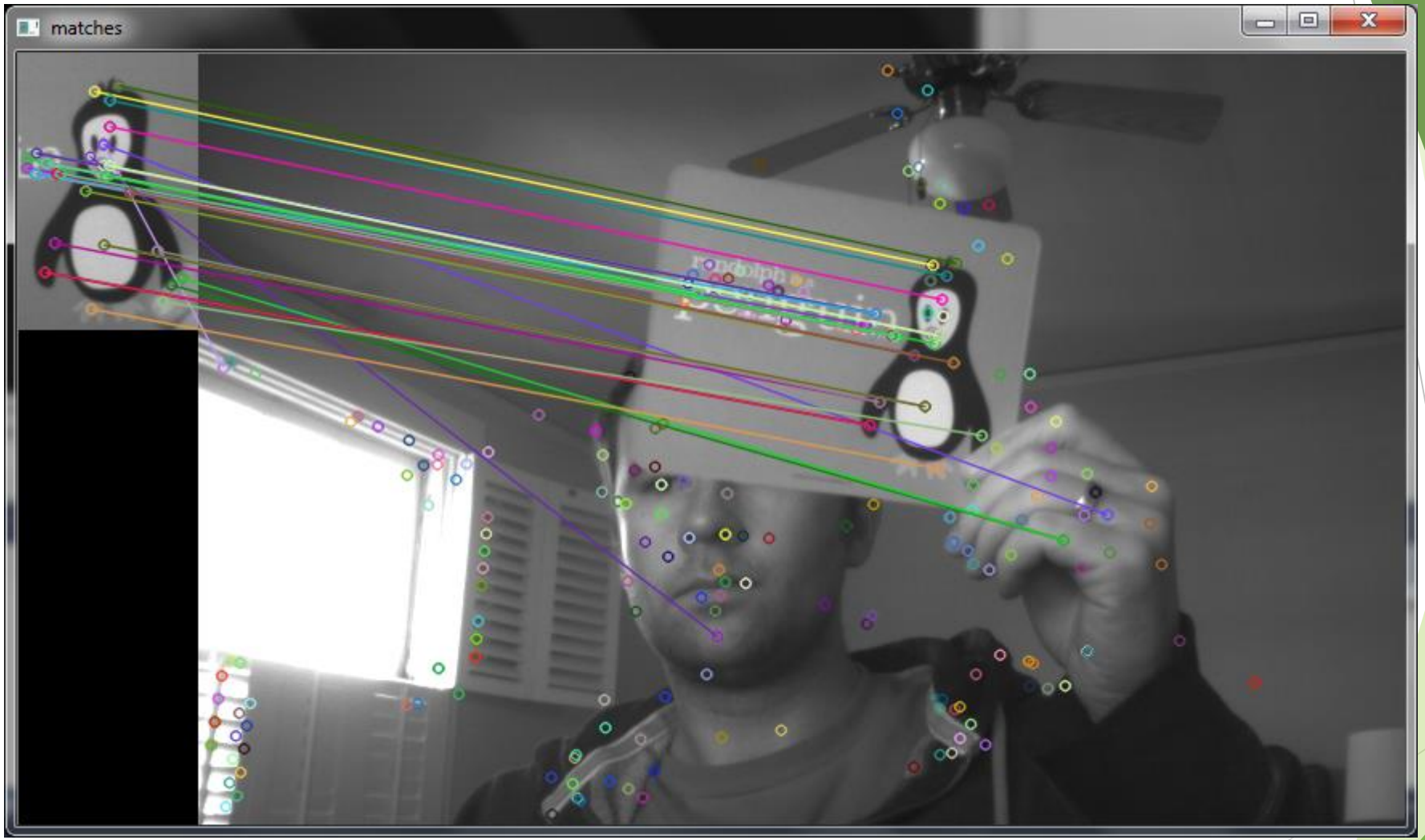
- Object detection and tracking
- Panorama stitching
- Stitching photos together in animation
- Training of object detection, object recognition, and image retrieval systems
- Camera calibration for single and stereo cameras
- Stereo vision, including rectification, disparity calculation, and 3-D reconstruction
- 3-D point cloud processing

Structure from Motion (SFM)

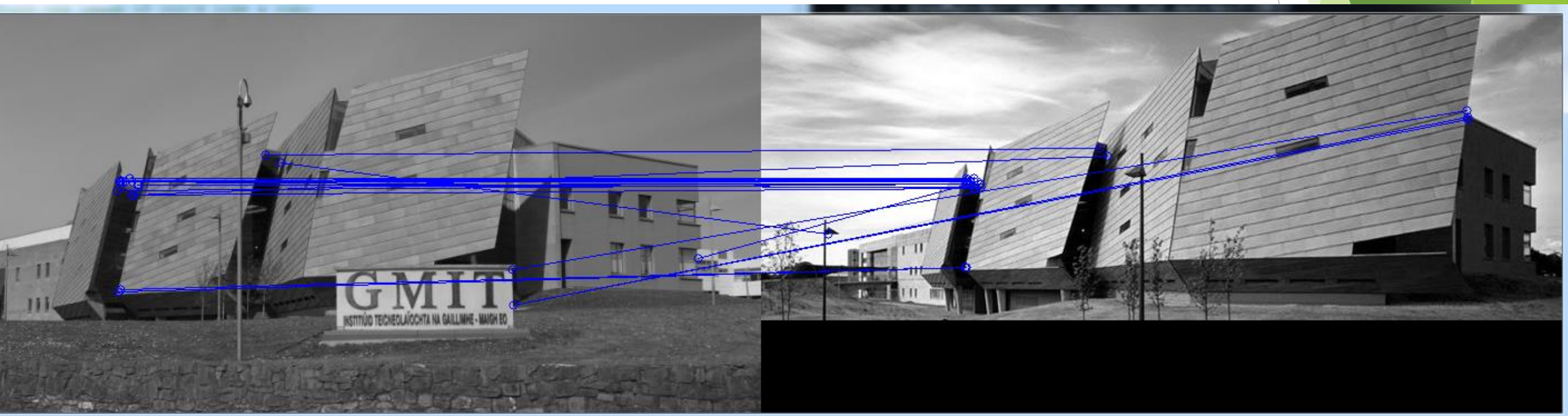


Feature Matching

- OpenCV provides two feature matching algorithms
  - Brute-force method
    - Takes one feature in first image and is matched with all other features in second image using some distance calculation. And the closest one is returned.
    - cv2.BFMatcher() takes 2 parameters
      - normType – set to cv2.NORM_L2
      - crossCheck – experiment with true and false settings (false by default)
    - cv2.drawMatches() displays the feature matches between the two images
  - FLANN (Fast Library for Approximate Nearest Neighbors)
    - Much faster than BFMatcher for large datasets.


matches

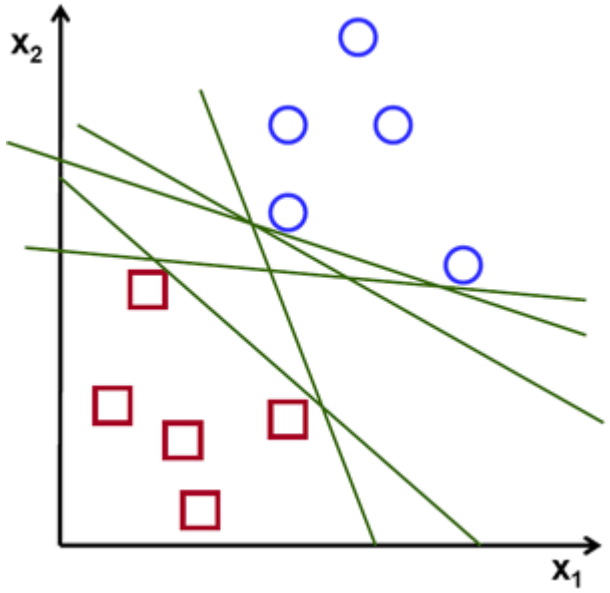

Matched Features

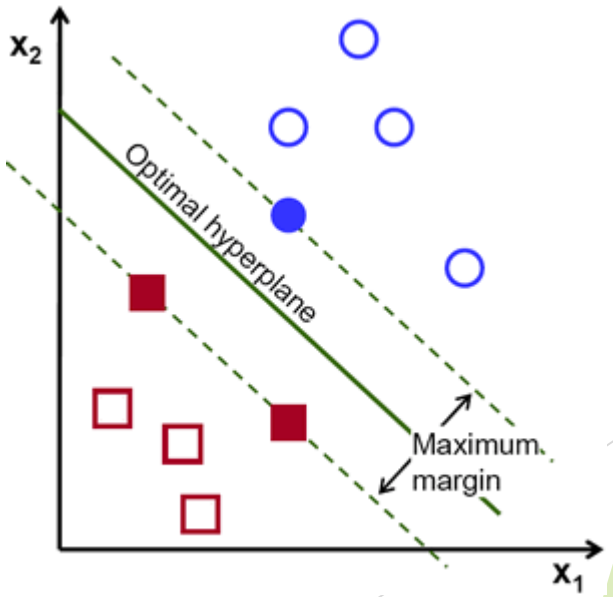

GMIT


0
500
000
500
0
500
1000
1500
2000
2500
3000

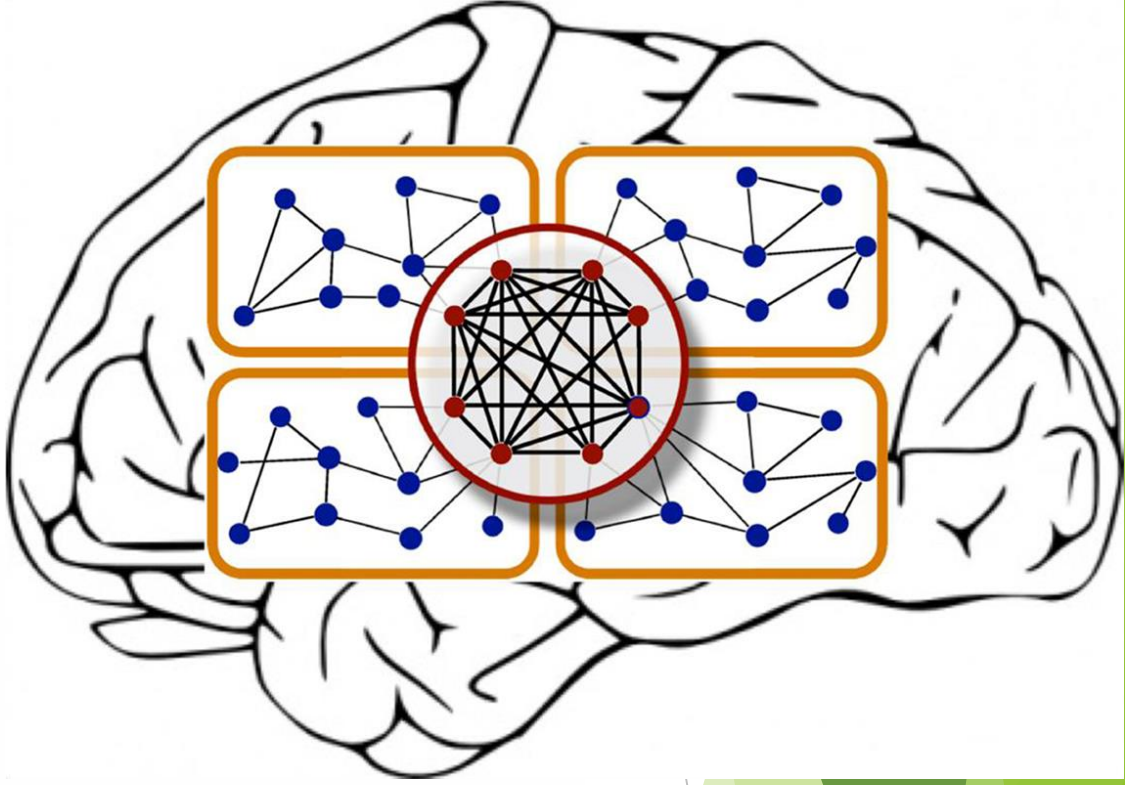
Image Classification

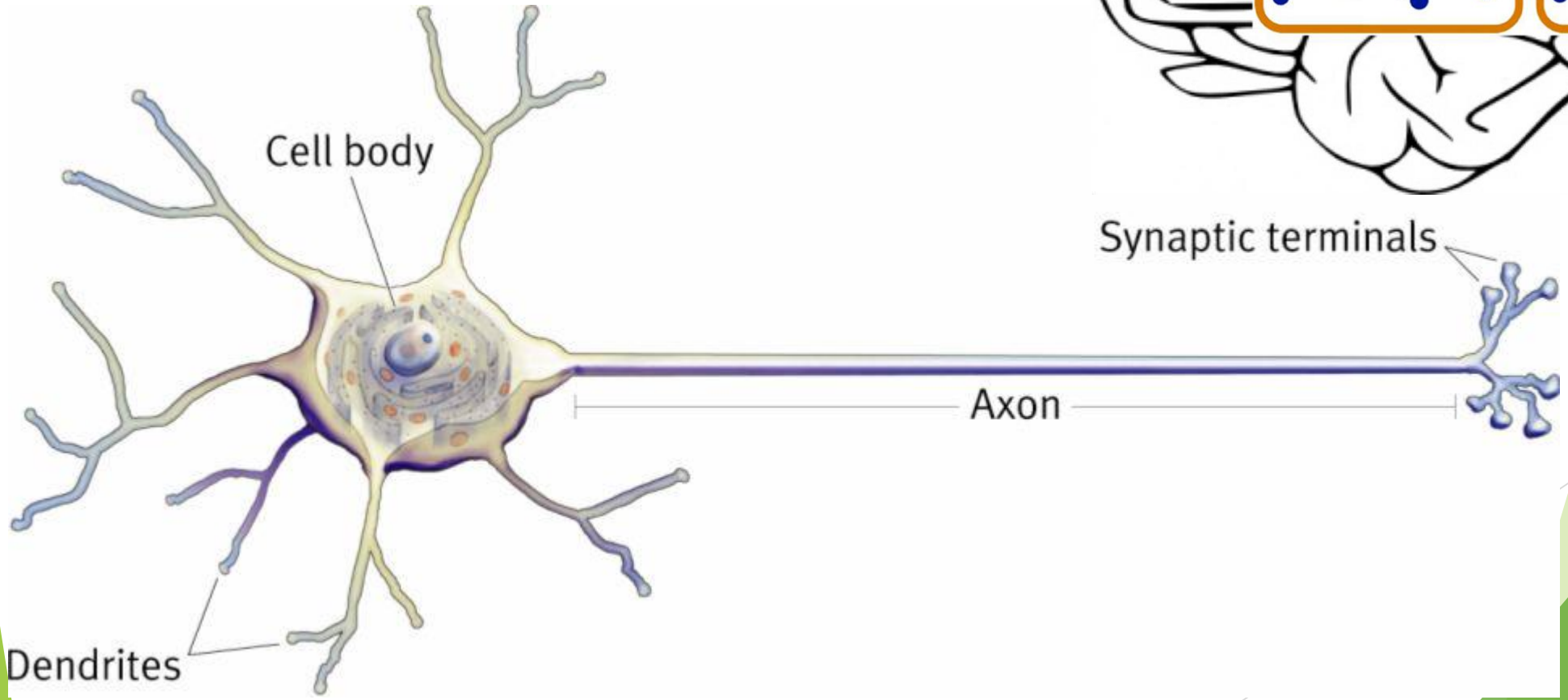
- Different classifiers
  - Neural networks
  - Support Vector Machines (SVM)
  - Boosted cascade classifiers (Adaboost)



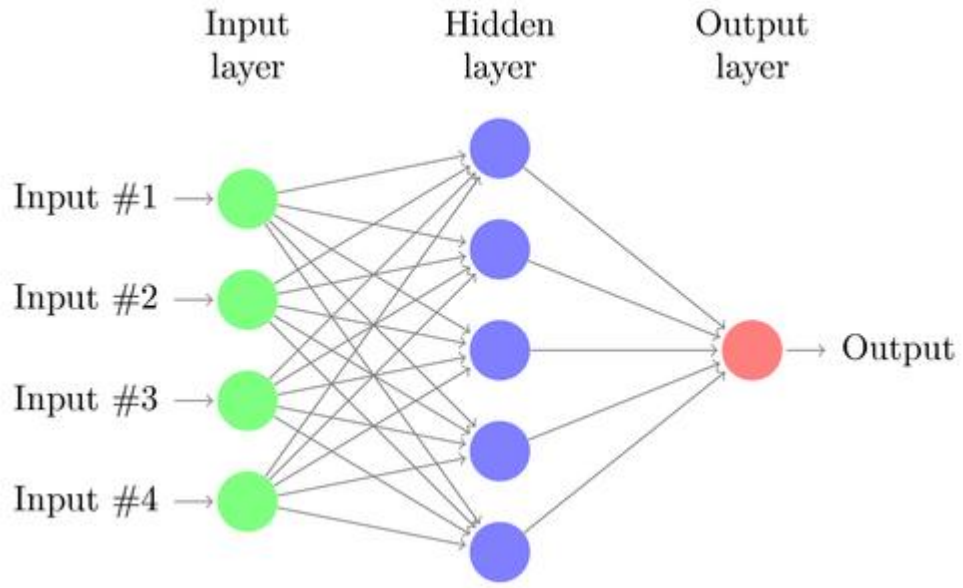


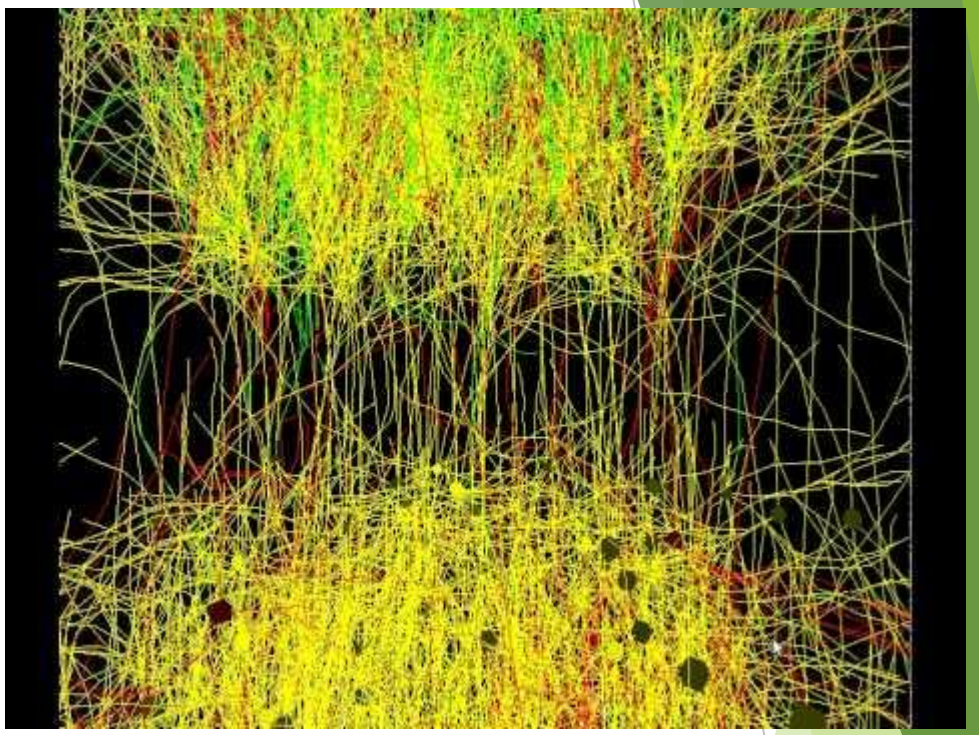
Neural Networks

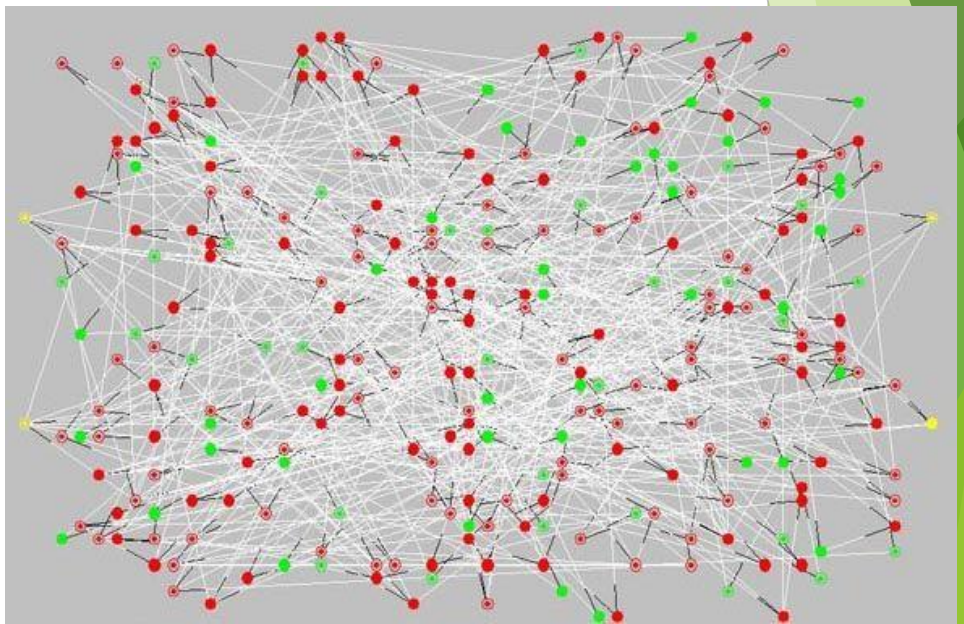




Neural Networks



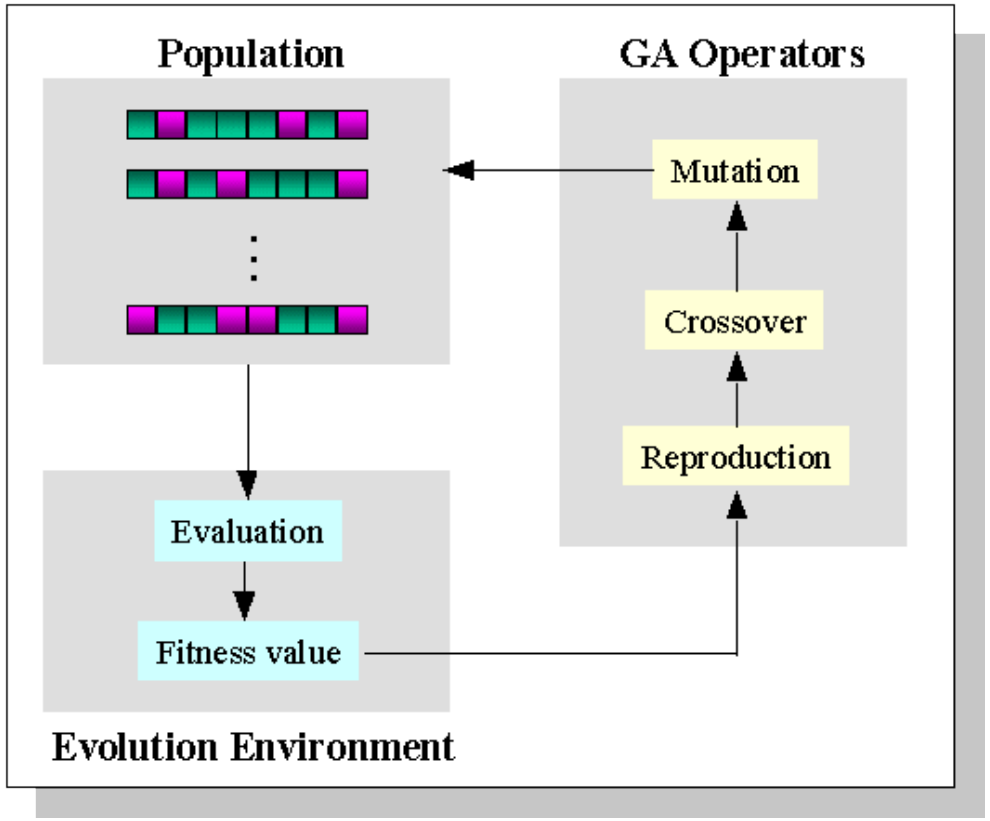


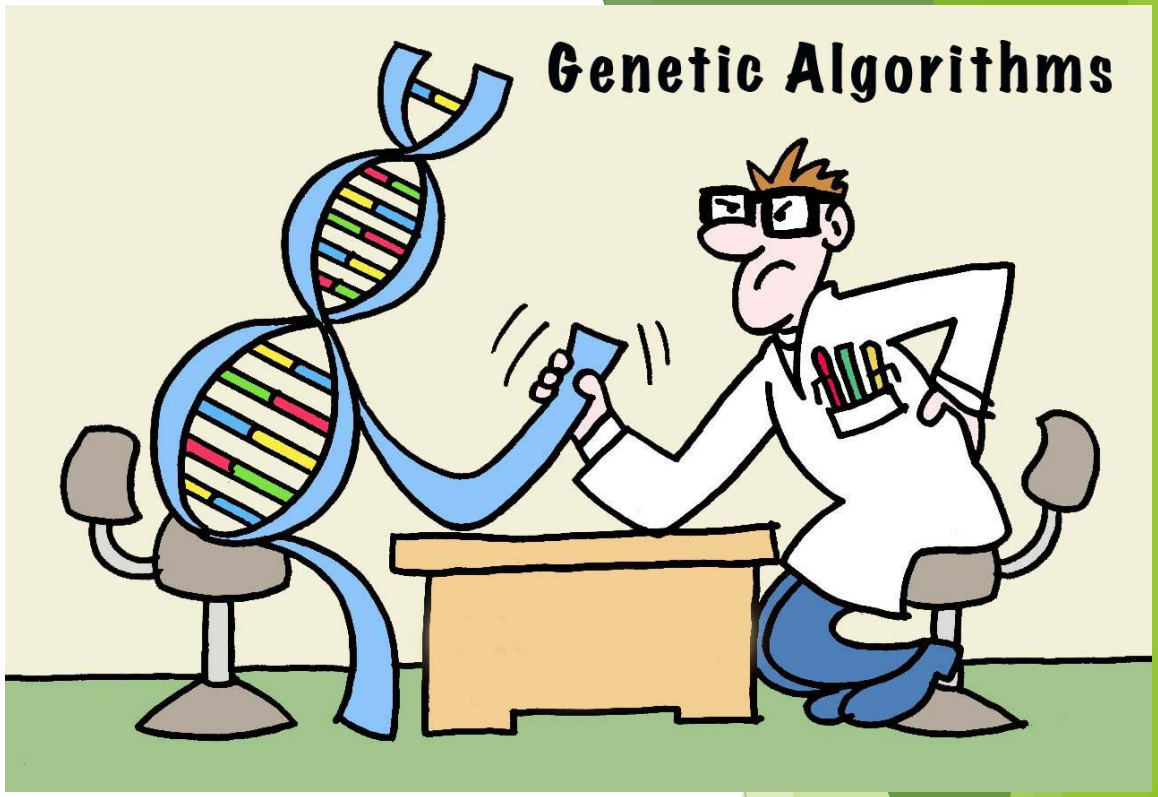


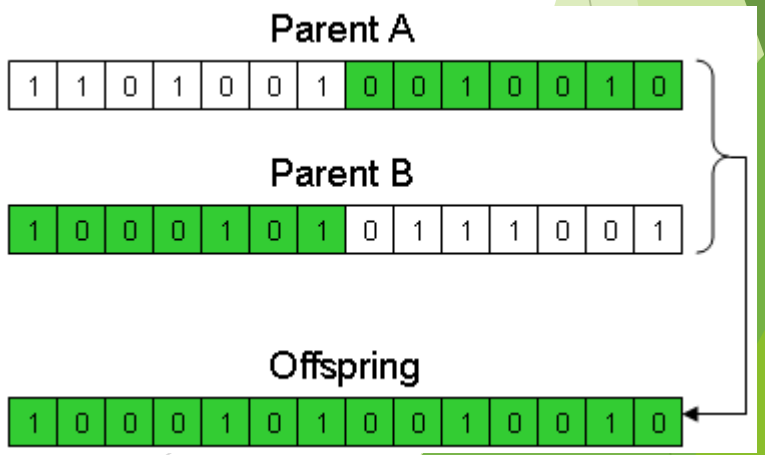
What is an Artificial Neural Network?

- A large set of nodes (neurons)
  - Each neuron has input and output
  - Each node performs a simple computation
- Weighted connections between nodes
  - Connectivity determines the intelligence of the network
  - Connections have direction
  - What can be computed by the SNN is determined by connections and strengths of connections.

Genetic Algorithms







Deep Learning

- Can see cats

Deep Learning

- Can caption photos
  - **Human:** “A group of men playing Frisbee in the park.”
    Computer model: “A group of young people playing a game of Frisbee.”

Deep Learning

- Can caption photos
  - **Human:** “A young hockey player playing in the ice rink.”
    Computer model: “Two hockey players are fighting over the puck.”

Deep Learning

- Can caption photos
  - **Human:** “A person riding a dirt bike is covered in mud.”
    Computer model: “A person riding a motorcycle on a dirt road.”

Deep Learning

- Can caption photos
  - **Human:** “A green monster kite soaring in a sunny sky.”
    Computer model: “A man flying through the air while riding a snowboard.”

Deep Learning

- Google is using Deep Learning in its image search
- Facebook is using deep learning to make inferences about our photos and about our intentions
- Qualcomm is using Deep Learning to help robots recognise objects the haven’t seen before or navigate themselves to a new location
- Still a new field

Humans vs Computers

- http://www.nytimes.com/2016/03/14/world/asia/south-korean-gets-priceless-victory-over-computer-in-go-match.html?mabReward=A1&moduleDetail=recommendations-0&action=click&contentCollection=Science&region=Footer&module=WhatsNext&version=WhatsNext&contentID=WhatsNext&src=recg&pgtype=article

Evolved Neural Network Art

Deep Learning - GO

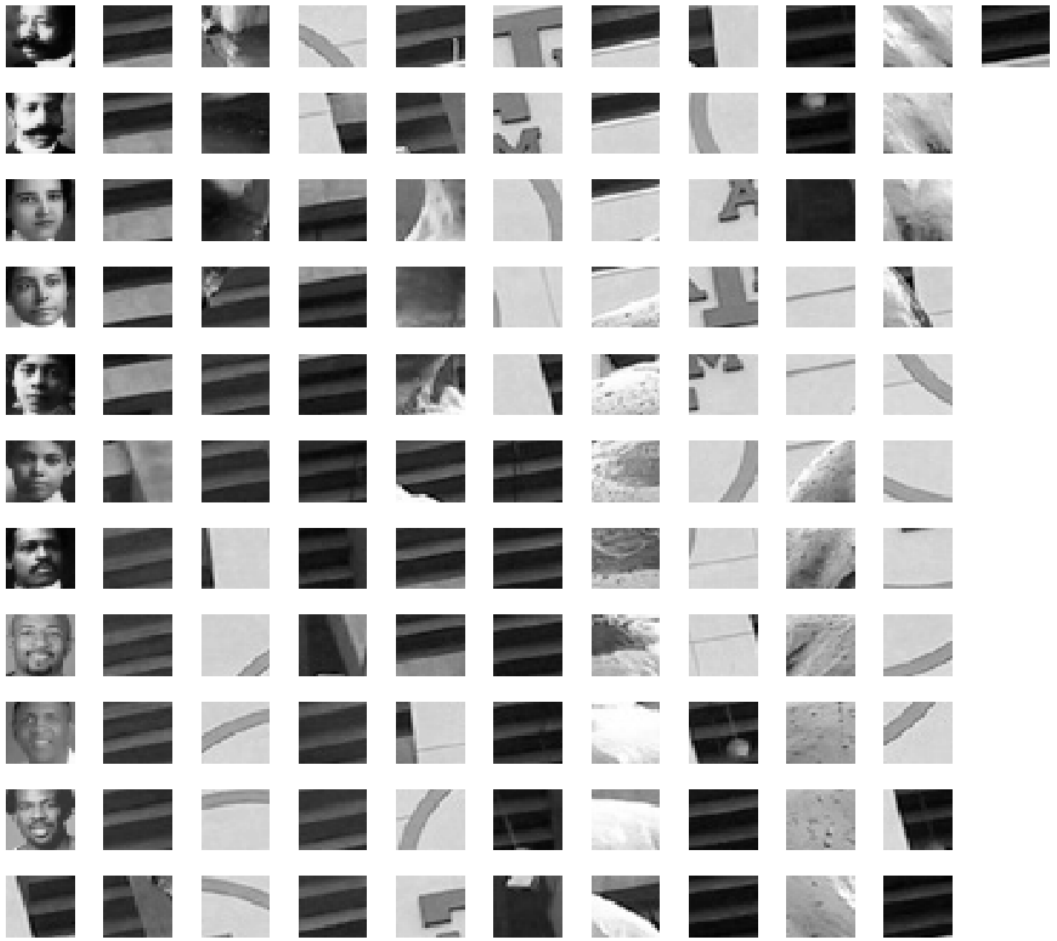
http://www.wired.com/2012/06/google-x-neural-network/

Learning to Detect Faces (Viola Jones)

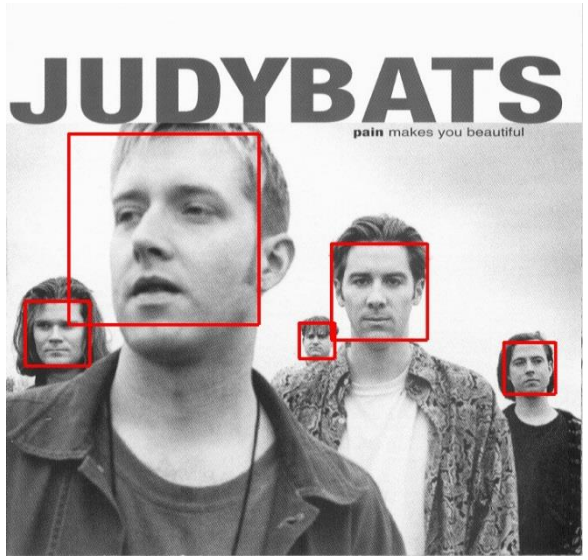
- Training - Examples of 24x24 images with faces

Learning to Detect Faces (Viola Jones)

- Training



Notice detection at multiple scales



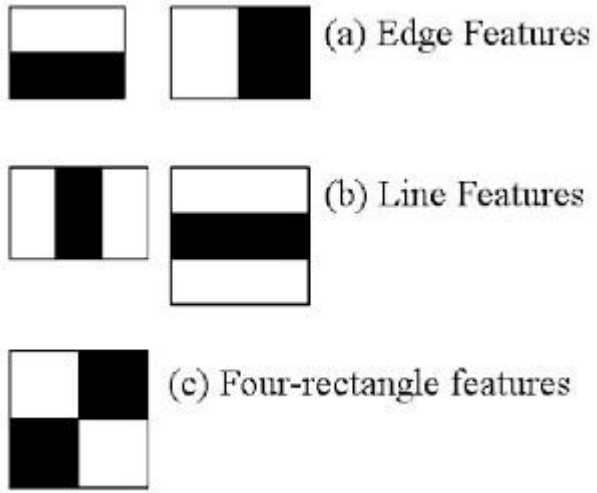
Viola Jones face detection

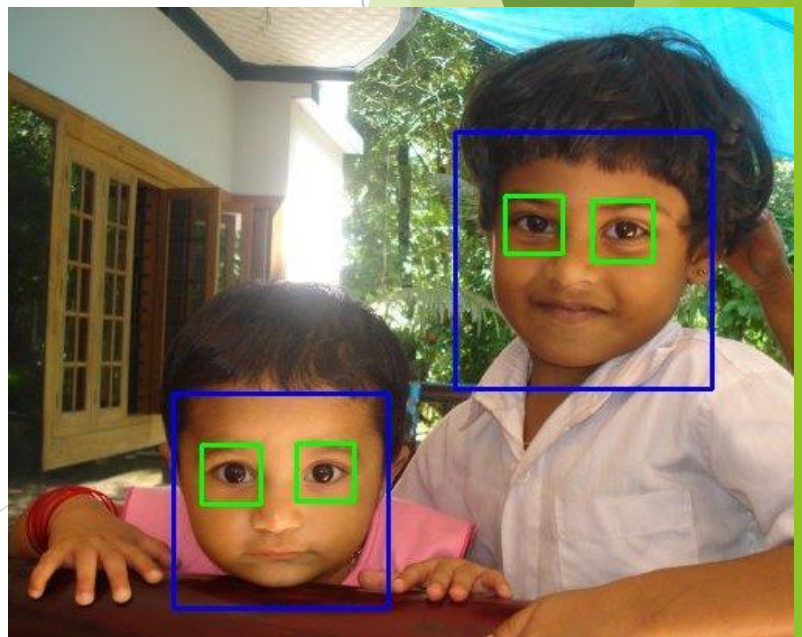
```
import numpy as np
import cv2

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.





Lab Submission

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