Digital Image Processing (CSE 478)
Lecture16: Image morphing

Vineet Gandhi

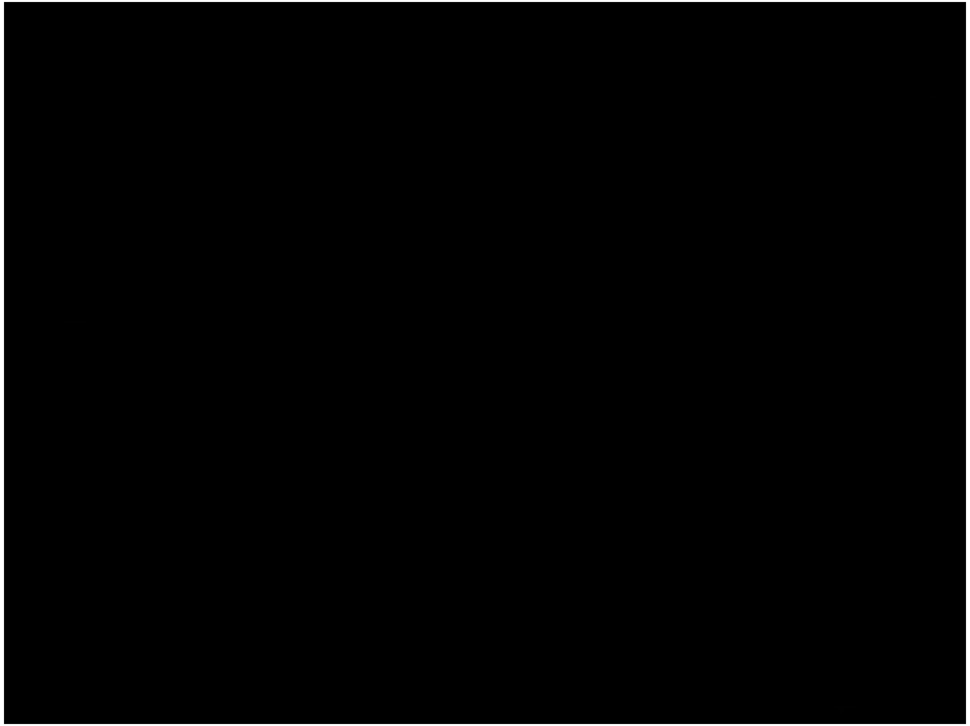
Center for Visual Information Technology (CVIT), IIIT Hyderabad

Most slides from Alexei Efros, Derek Hoeim

500 years of female portrait

https://www.youtube.com/watch?v=L0GKp-uvjO0

0 to 65 and back in a minute



https://www.youtube.com/watch?v=L0GKp-uvjO0

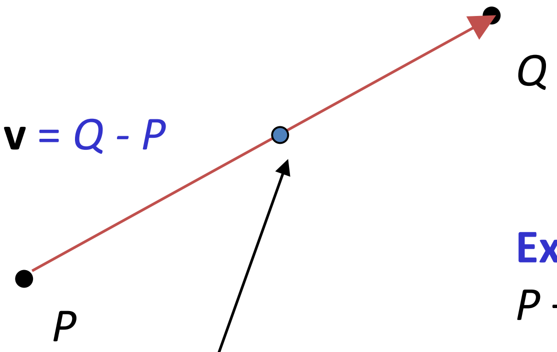
Averaging vs Morphing



- The aim is to find “an average” between two objects
  - Not an average of two images of objects…
  - …but an image of the average object!
  - How can we make a smooth transition in time?
    - Do a “weighted average” over time t

Averaging points

What's the average of P and Q?



Extrapolation: $t<0$ or $t>1$

$P + 1.5\mathrm{v}$

$= P + 1.5(Q - P)$

$= -0.5P + 1.5\,Q$ *(t=1.5)*

Linear Interpolation

New point: $(1-t)P + tQ$

$0<t<1$

$P + 0.5\mathrm{v}$

$= P + 0.5(Q - P)$

$= 0.5P + 0.5\,Q$

- P and Q can be anything:
  - points on a plane (2D) or in space (3D)
  - Colors in RGB (3D)
  - Whole images (m-by-n D)… etc.

Idea-1: cross dissolve

- Interpolate whole images:
- $\text{Image}_{\text{halfway}} = (1\text{-}t)*\text{Image}_1 + t*\text{image}_2$
- This is called **cross-dissolve** in film industry

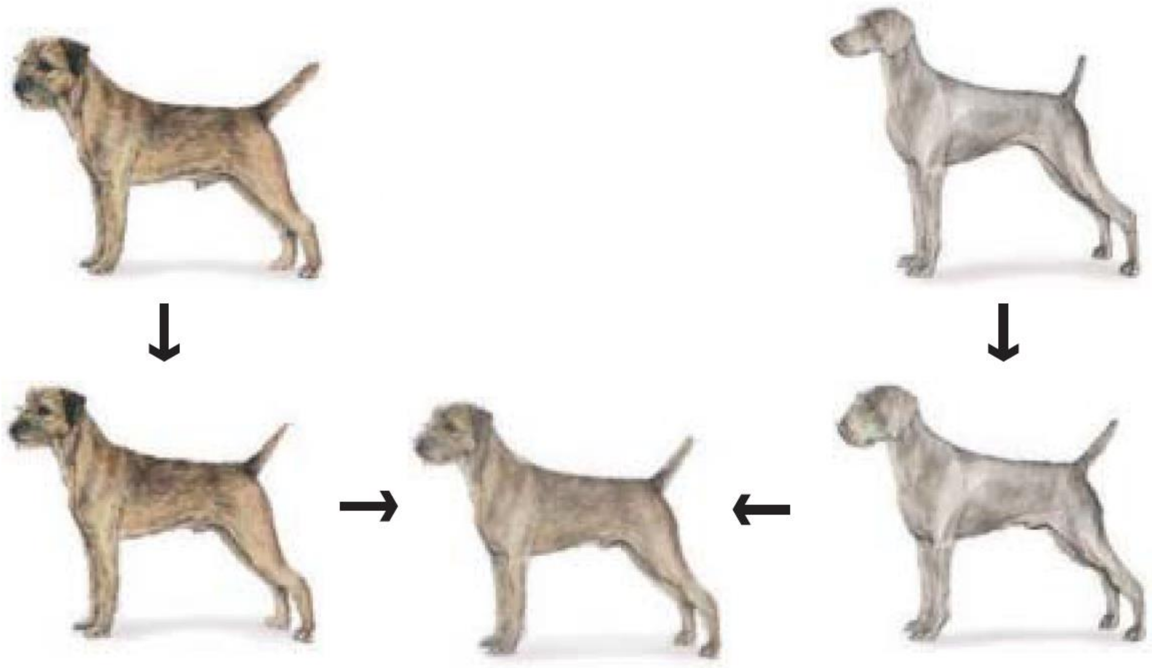
- But what if the images are not aligned?

Idea-2: align, then cross dissolve

Dog averaging

- What to do?
  - Cross-dissolve doesn't work
  - Global alignment doesn't work
    - Cannot be done with a global transformation (e.g. affine)
  - Any ideas?
- Feature matching!
  - Nose to nose, tail to tail, etc.
  - This is a local (non-parametric) warp

Idea-3: Local warp, then cross-dissolve



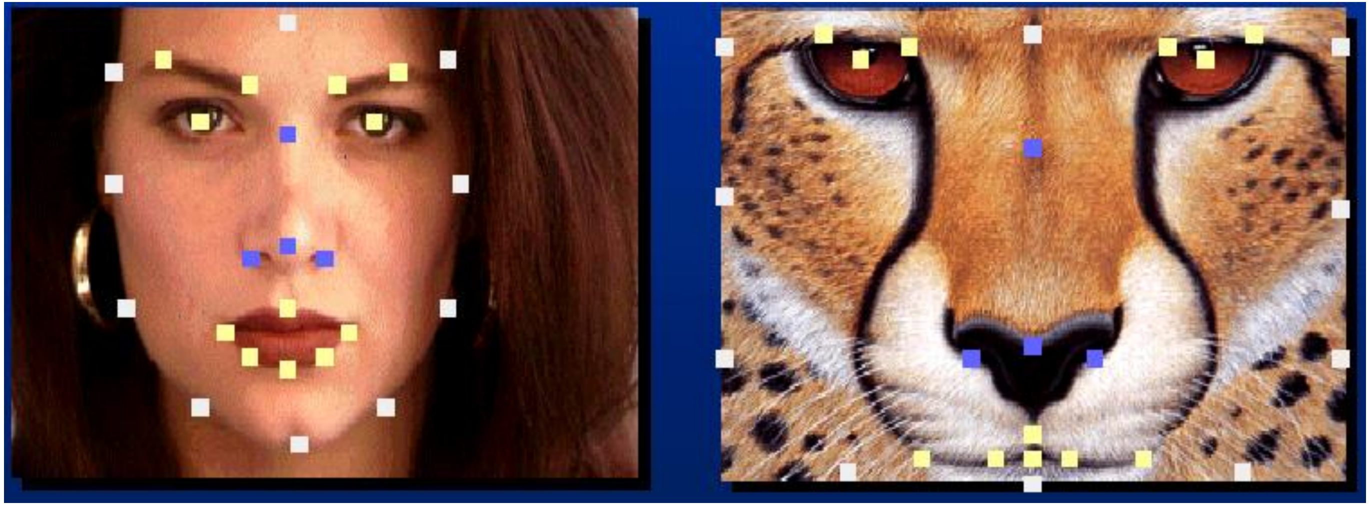
- *For every frame t,*
1. Find the average shape (the "mean dog"☺)
   – local warping
2. Find the average color
   – Cross-dissolve the warped images

Warp specification

- How can we specify the warp?

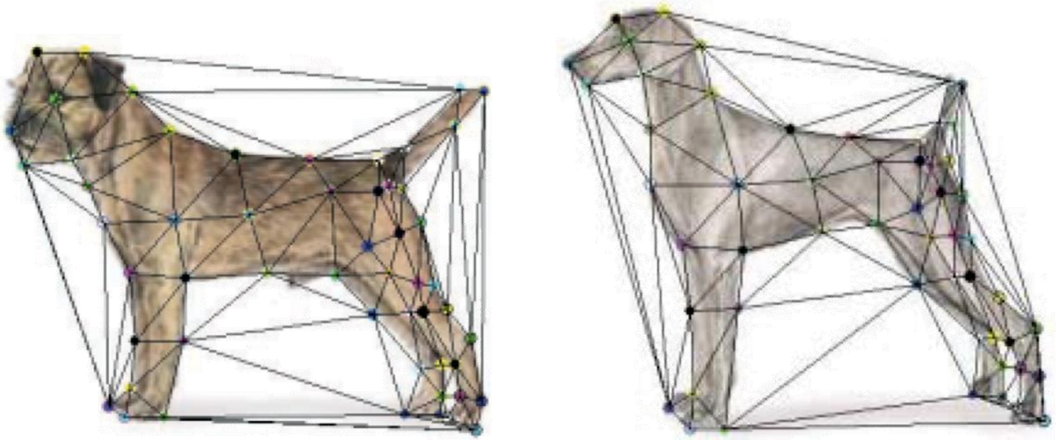
  Specify corresponding *points*

  - *interpolate* to a complete warping function
  - How do we do it?



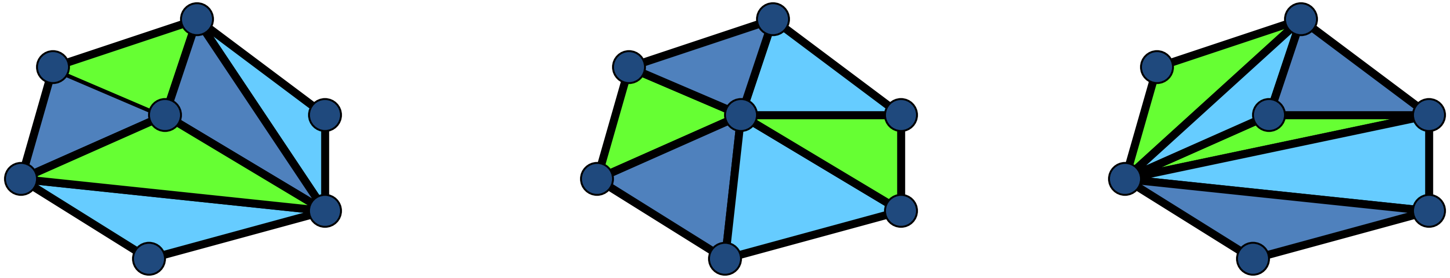
Triangular Mesh

1. Input correspondences at key feature points
2. Define a triangular mesh over the points
   - Same mesh (triangulation) in both images!
   - Now we have triangle-to-triangle correspondences
3. Warp each triangle separately from source to destination
   - Affine warp with three corresponding points



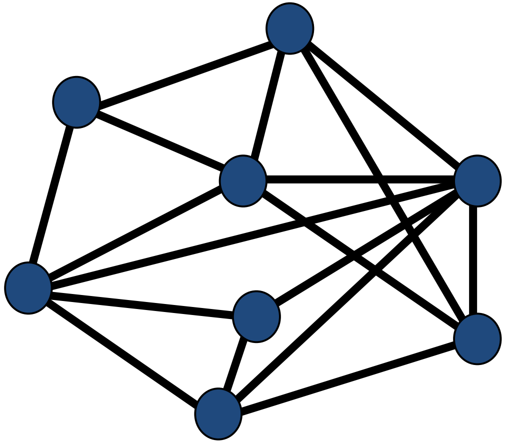
Triangulations

- A *triangulation* of set of points in the plane is a *partition* of the convex hull to triangles whose vertices are the points, and do not contain other points.
- There are an exponential number of triangulations of a point set.



An O(n^3) Triangulation Algorithm

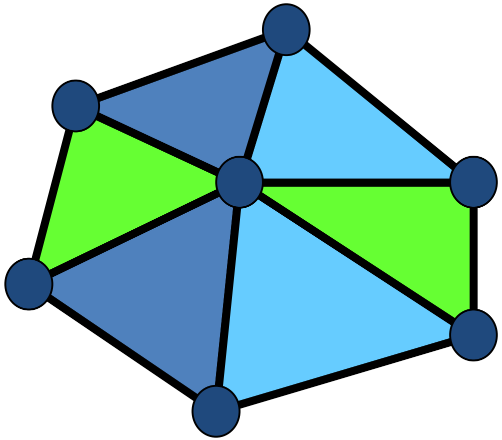
- Repeat until impossible:
  - Select two sites.
  - If the edge connecting them does not intersect previous edges, keep it.



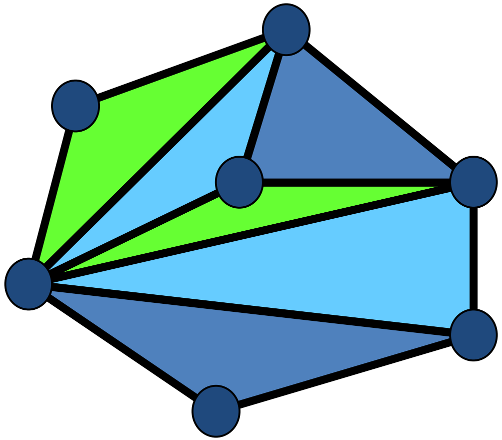
“Quality” Triangulations

- Let $\alpha(T_i) = (\alpha_{i1}, \alpha_{i2}, .., \alpha_{i3})$ be the vector of angles in the triangulation T in increasing order:
- A triangulation T_1 is “better” than T_2 if the smallest angle of T_1 is larger than the smallest angle of T_2
- Delaunay triangulation is the “best” (maximizes the smallest angles)

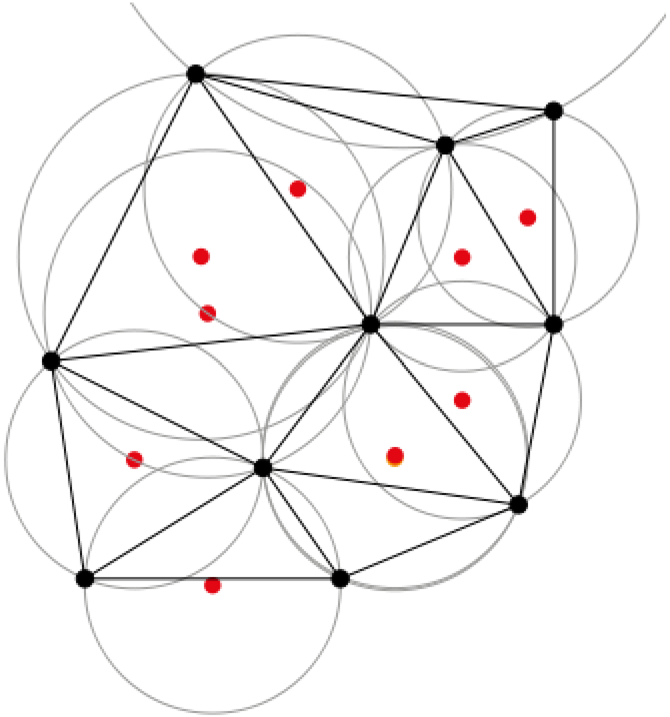
good



bad



Delaunay triangulation



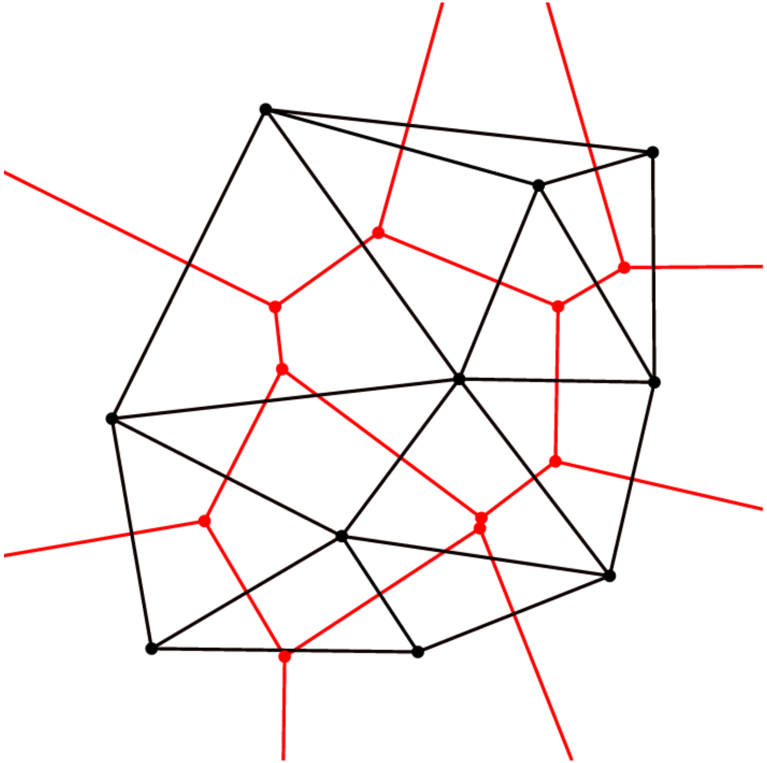
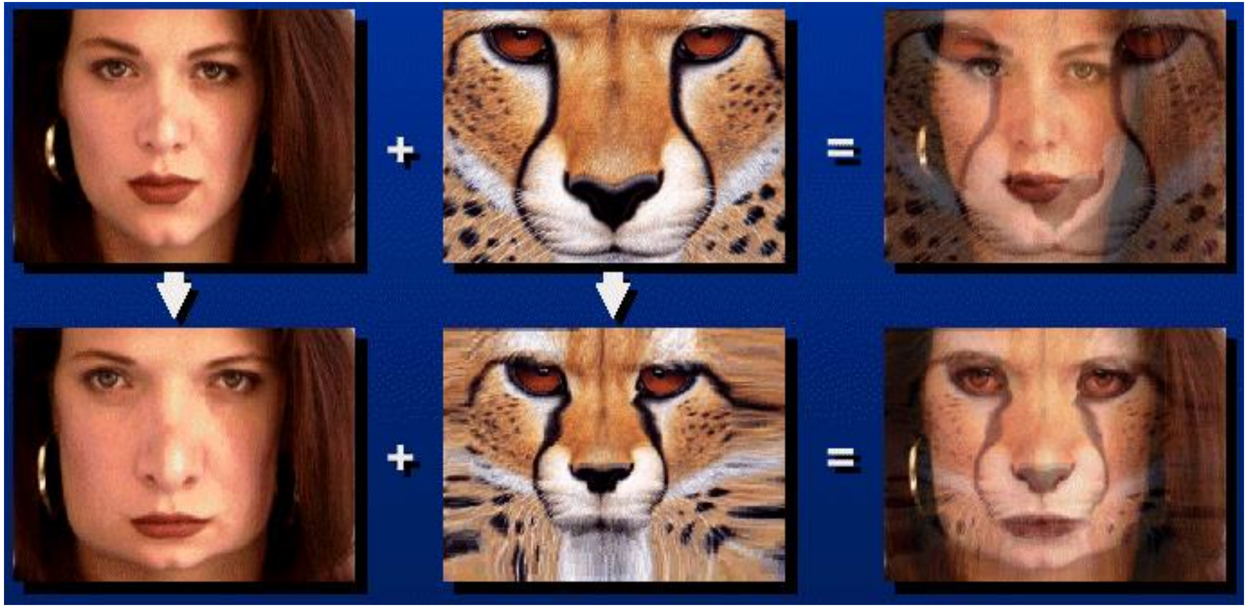


Image Morphing

How do we create a morphing sequence?

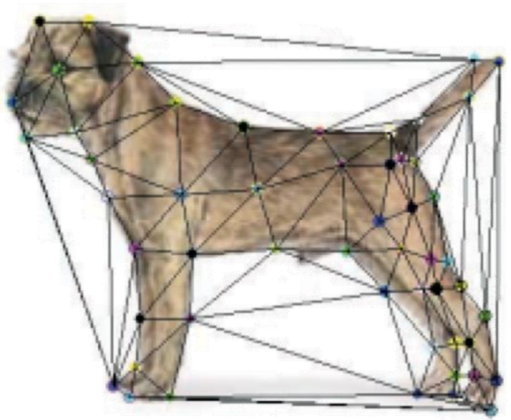
1. Create an intermediate shape (by interpolation)
2. Warp both images towards it
3. Cross-dissolve the colors in the newly warped images

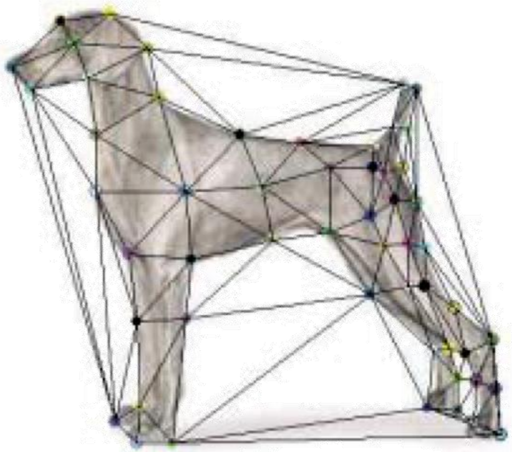


Warp interpolation

How do we create an intermediate shape at time t?

- Assume t = [0,1]
- Simple linear interpolation of each feature pair
  - (1-t)*p1+t*p0 for corresponding features p0 and p1





Summary of Morphing

1. Define corresponding points
2. Define triangulation on points
   – Use same triangulation for both images
3. For each t in 0:step:1
   a. Compute the average shape (weighted average of points)
   b. For each triangle in the average shape
      - Get the affine projection to the corresponding triangles in each image
      - For each pixel in the triangle, find the corresponding points in each image and set value to weighted average (optionally use interpolation)
   c. Save the image as the next frame of the sequence

Black Or White - MJ

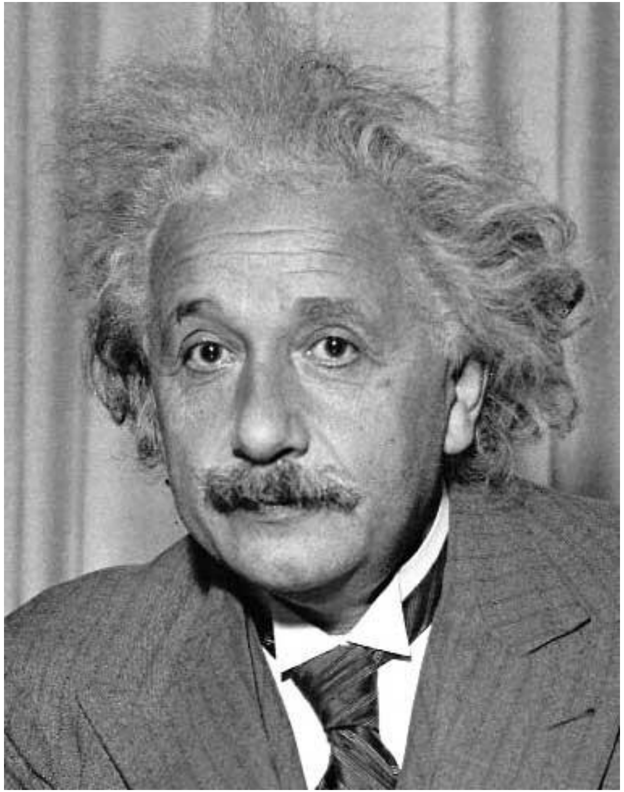
Changing topic now

Template Matching

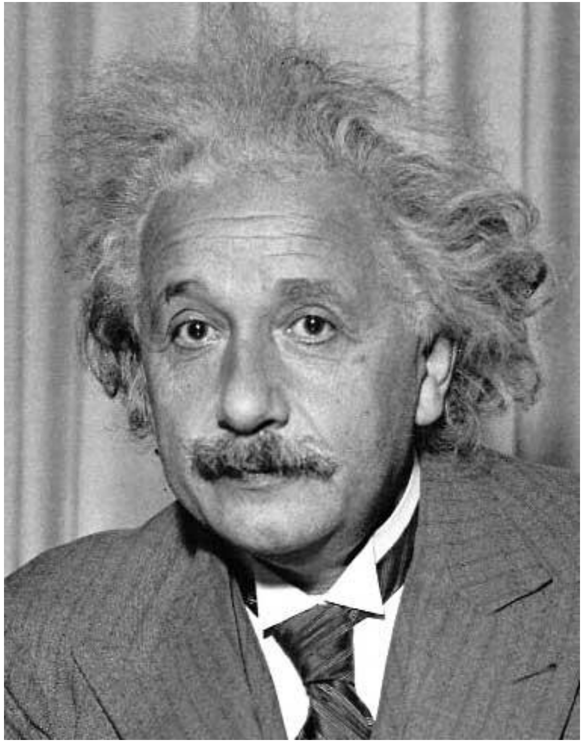
- Template matching

Template Matching

- Goal: find  in image
- Main challenge: What is a good similarity or distance measure between two patches?
  - Correlation
  - Zero-mean correlation
  - Sum Square Difference
  - Normalized Cross Correlation

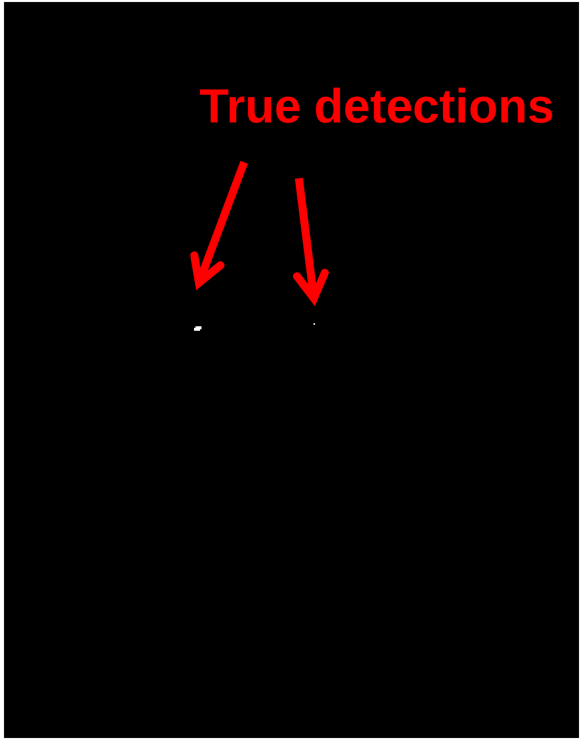


SSD



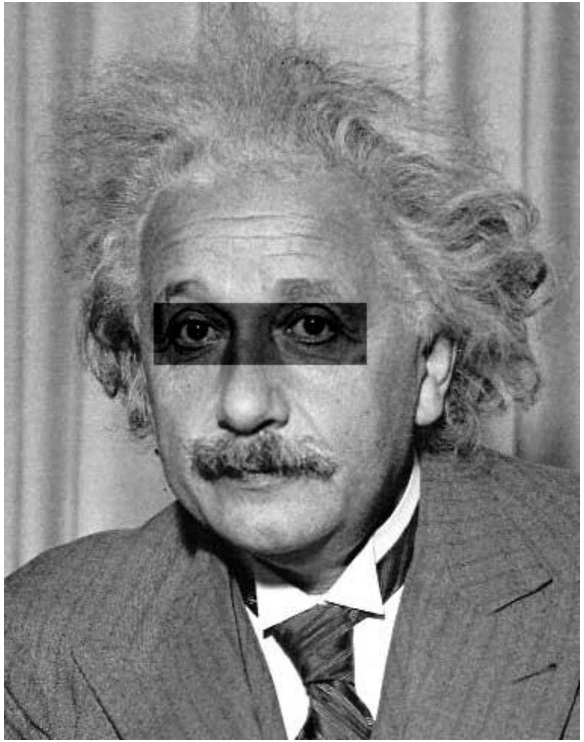
Input

1- sqrt(SSD)



Thresholded Image

SSD



Input

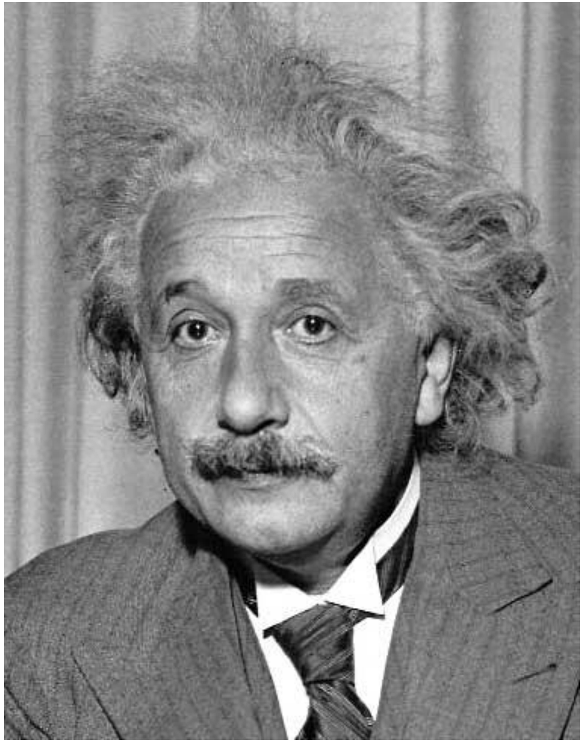
1- sqrt(SSD)

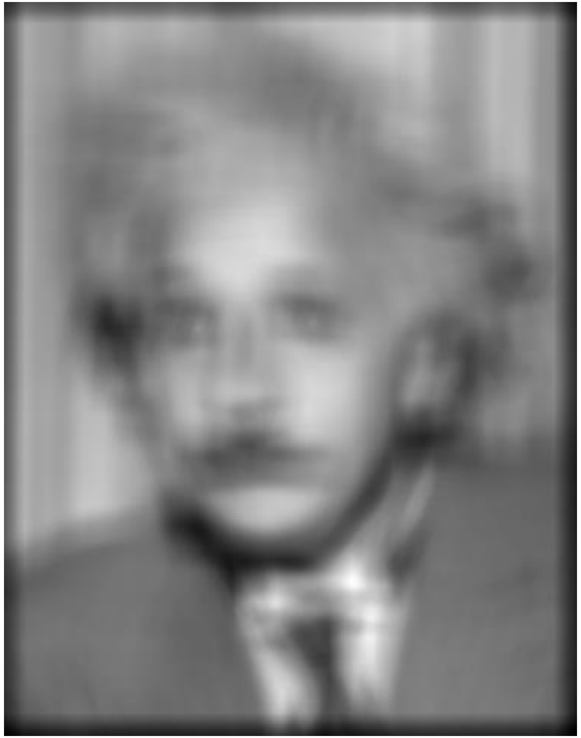
Correlation (filtering)

$$h[m,n]=\sum_{k,l} g[k,l]\ f[m+k,n+l]$$

f = image

g = filter 

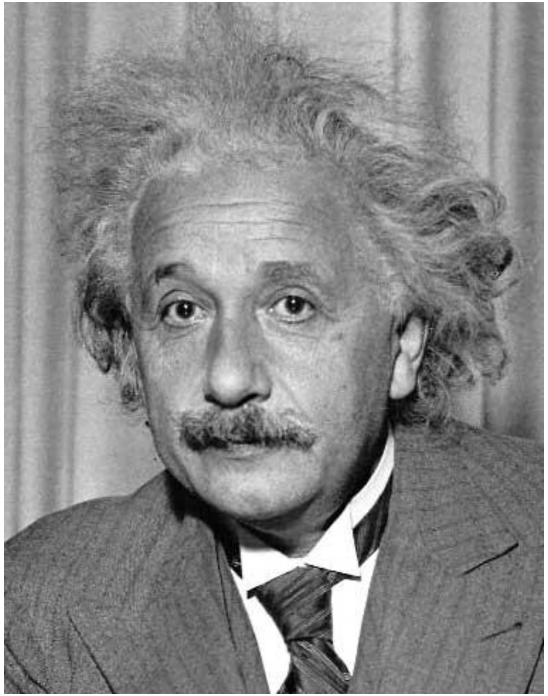




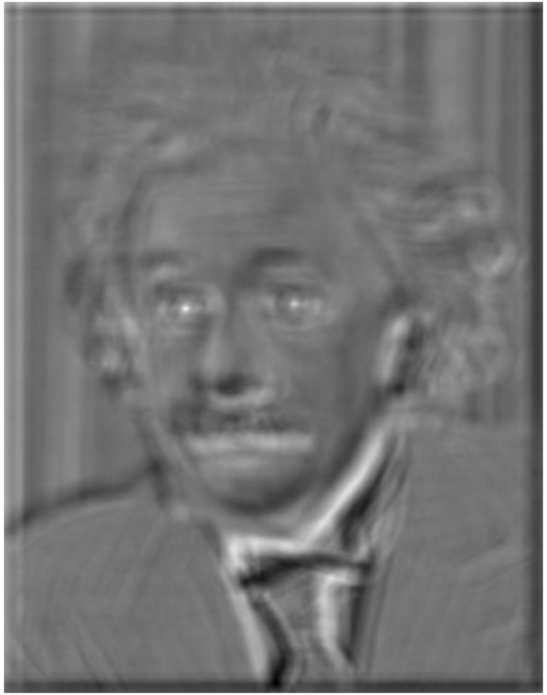
What went wrong?

Correlation (filtering)

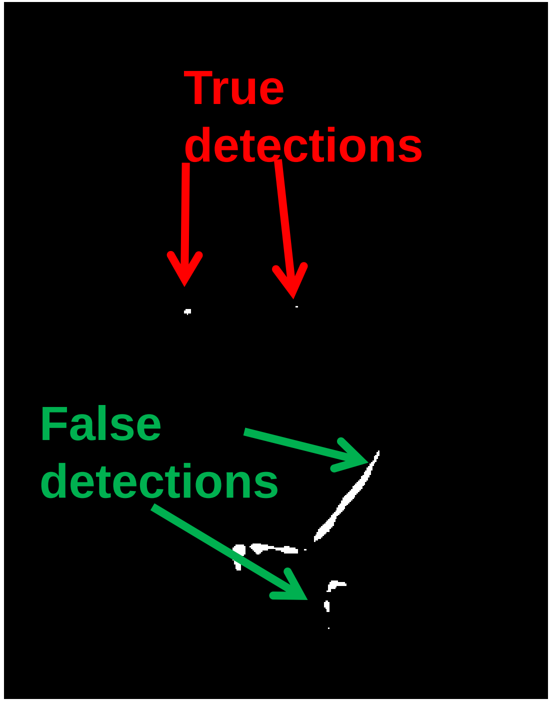
$$h[m,n]=\sum_{k,l}(f[k,l]-\bar{f})\,(g[m+k,n+l])$$



Input



Filtered Image (scaled)



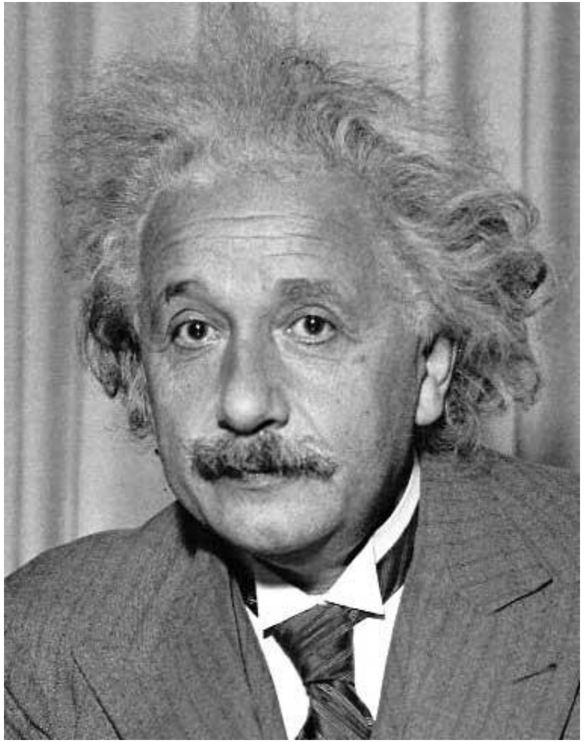
Thresholded Image

Normalized cross correlation

$$h[m,n] = \frac{\sum_{k,l} (g[k,l] - \overset{\text{mean template}}{\overset{\downarrow}{\bar{g}}})(f[m+k,n+l] - \overset{\text{mean image patch}}{\overset{\downarrow}{\bar{f}_{m,n}}})}{\left(\sum_{k,l} (g[k,l] - \bar{g})^2 \sum_{k,l} (f[m+k,n+l] - \bar{f}_{m,n})^2 \right)^{0.5}}$$

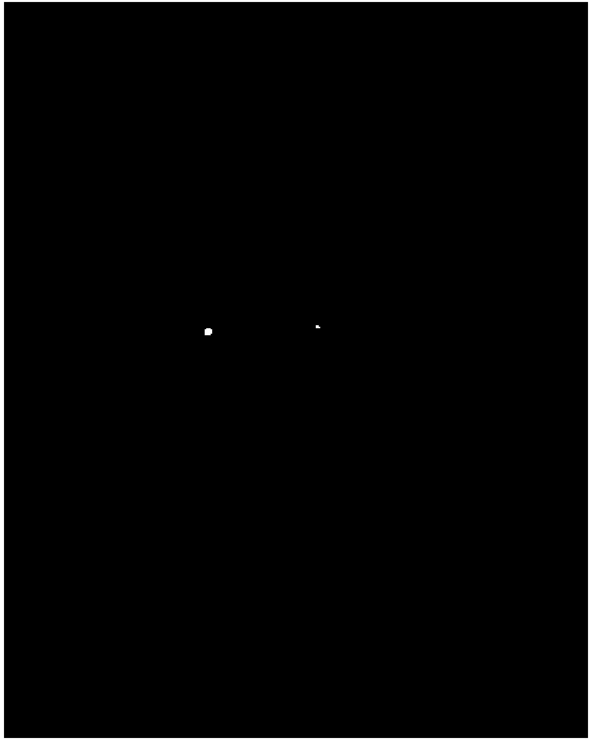
Matlab: `normxcorr2(template, im)`

Normalized Cross Correlation



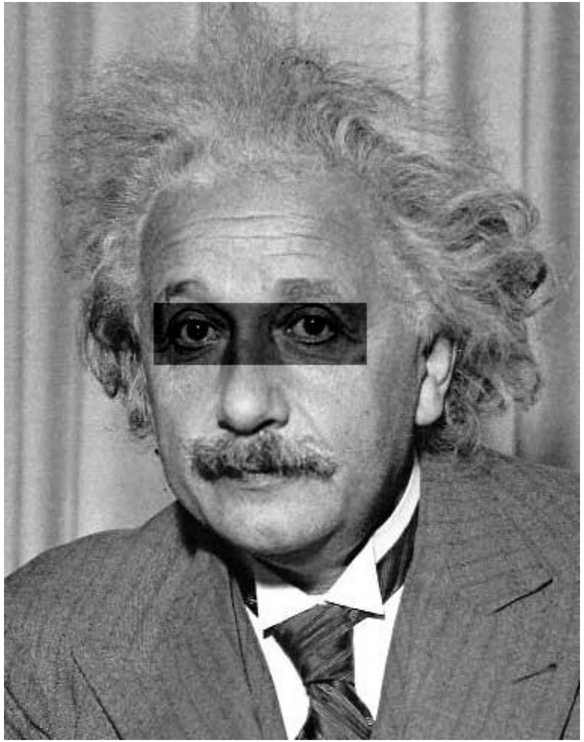
Input

Normalized X-Correlation



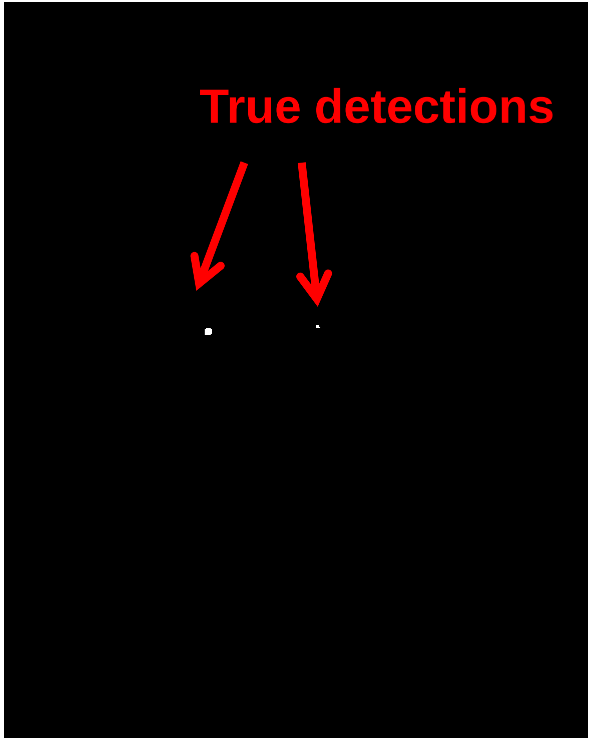
Thresholded Image

Normalized Cross Correlation



Input

Normalized X-Correlation



Thresholded Image