Computer Vision (CS763)

Teaching cameras to "see"

Image Alignment

Arjun Jain

Most slides courtesy Ajit Rajwade

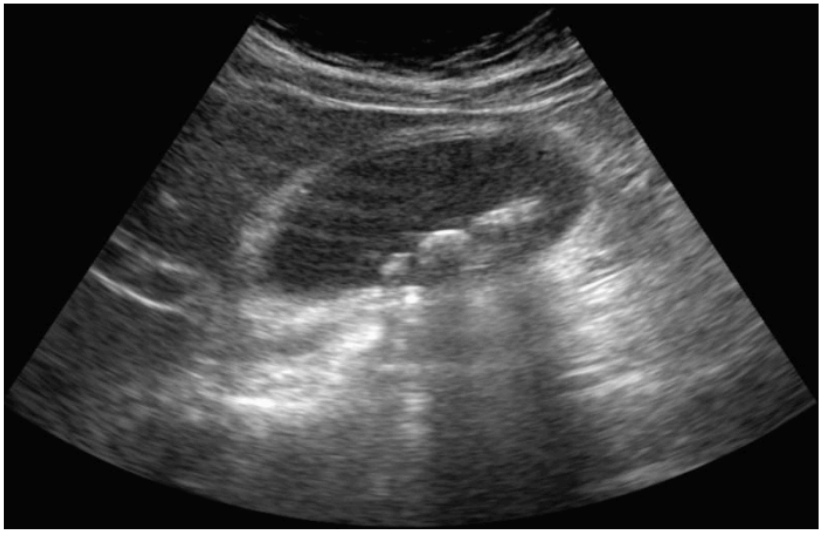
https://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2017/ImageAlignment.pdf

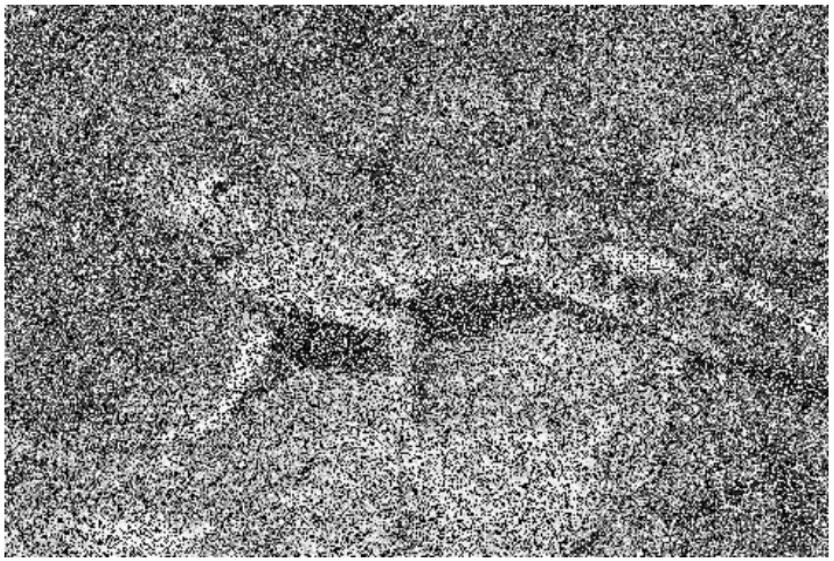
Non-Control Point based Image Alignment
(Using Image Similarity/Correlation based Measures)

Control Points are NOT always available

- In some scenes, good control points may not be available
- Or they cannot sometimes be reliably matched from one image to another
- Example: some modalities such as ultrasound, or images that are heavily blurred or noisy. (e.g. on next slide)

Control Points are NOT always available





Alignment with Mean-Squared-Error

- Mean squared error is given by:

$$MSSD = \frac{1}{N} \sum_{x,y \in \Omega} (I_1(x,y) - I_2(x,y))^2$$

- Find motion parameters as follows:

$$\mathbf{T}^* = \arg\min MSSD_{\mathbf{T}}(I_2(\mathbf{v}), I_1(\mathbf{T}\mathbf{v}))$$

$$\mathbf{T} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{v} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Find transformation matrix **T** which produces the least value of MSSD

I_2 = called the fixed (or reference) image
I_1 = called the moving image

Alignment with Mean-Squared-Error

- For simplicity, assume there was only rotation and translation.
- Then we have

$$\mathbf{T}^* = \arg\min MSSD_{\mathbf{T}}(I_2(\mathbf{v}), I_1(\mathbf{T}\mathbf{v}))$$

$$\mathbf{T} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{v} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Alignment with Mean-Squared-Error

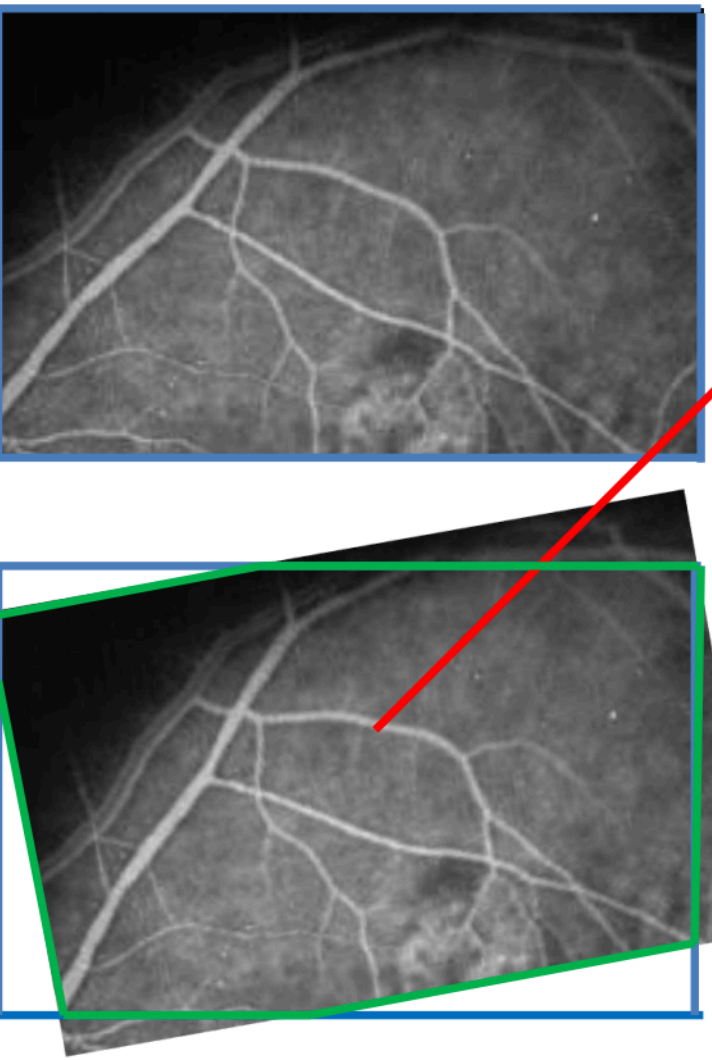
- There are many ways to do this minimization. The simplest but inefficient way is to do a brute-force search.
- Sample θ, t_x and t_y uniformly from a certain range (example: θ from -45 to +45, t_x or t_y from -30 to +30).
- Apply this motion to I_1 keeping I_2 fixed, and compute the MSSD.
- Each time, compute the MSSD. Pick the parameter values (i.e. θ, t_x and t_y) corresponding to minimum MSSD.

Alignment with Mean-Squared-Error

- In the ideal case, the MSSD between two perfectly aligned images is 0. In practice, it will have some small non-zero value even under more or less perfect alignment due to sensor noise or slight mismatch in pixel grids.

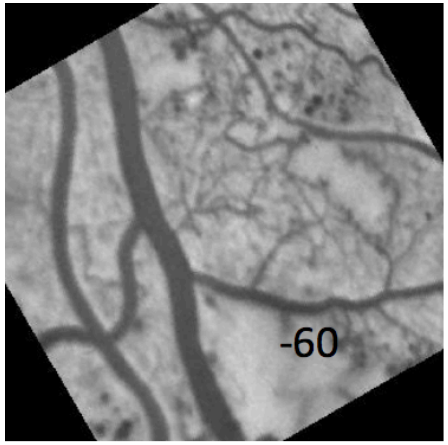
Careful: field of view issues!

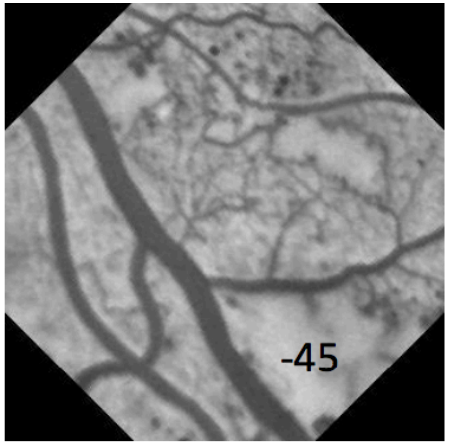
Fixed image

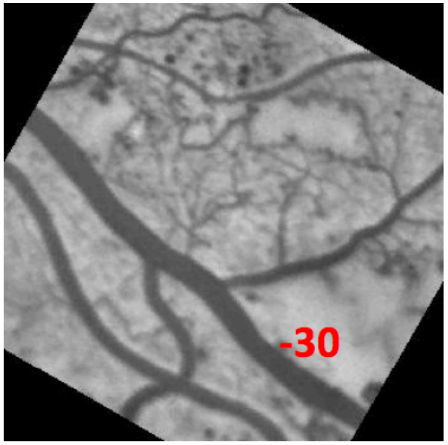


Region of overlap when the moving image is warped

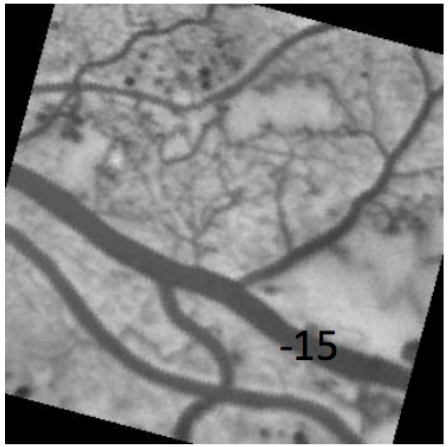
Note: compute MSSD only over region of overlap.

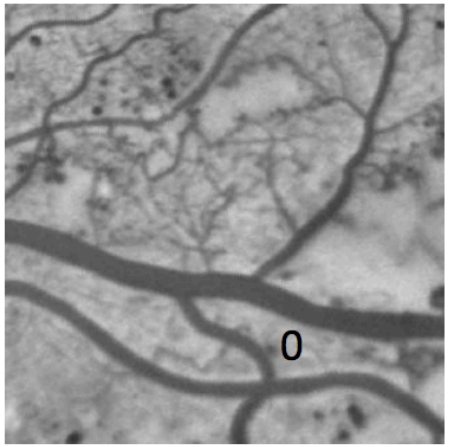


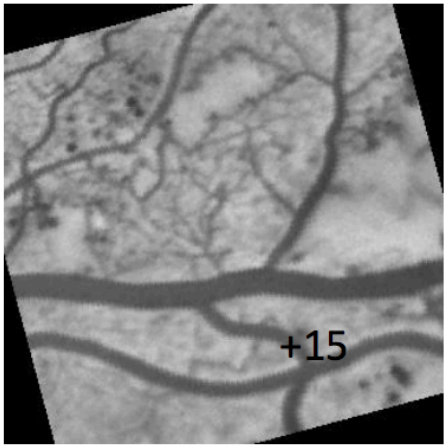


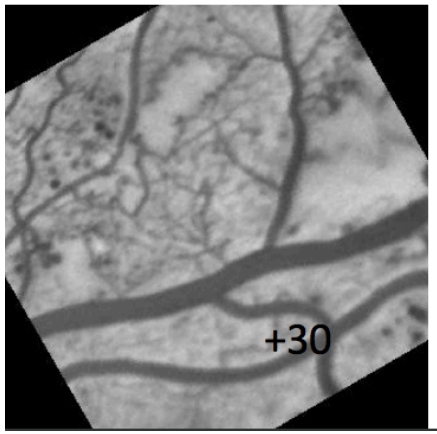


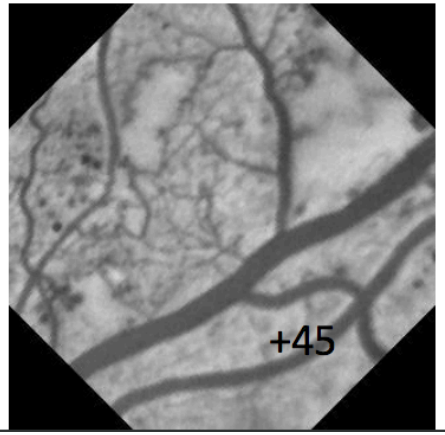
Change in region of overlap, as the moving image is warped

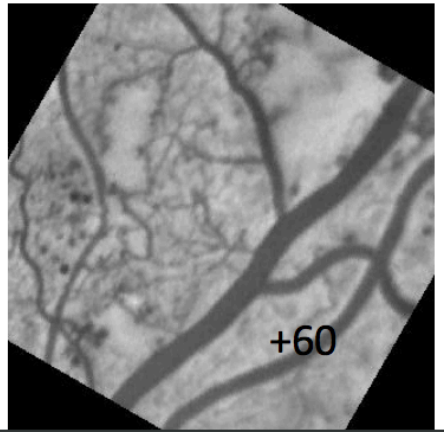


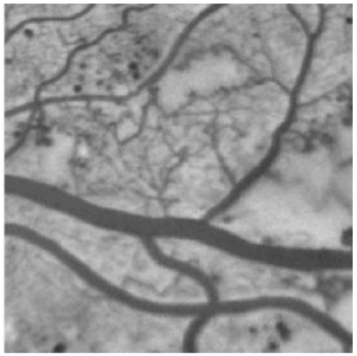












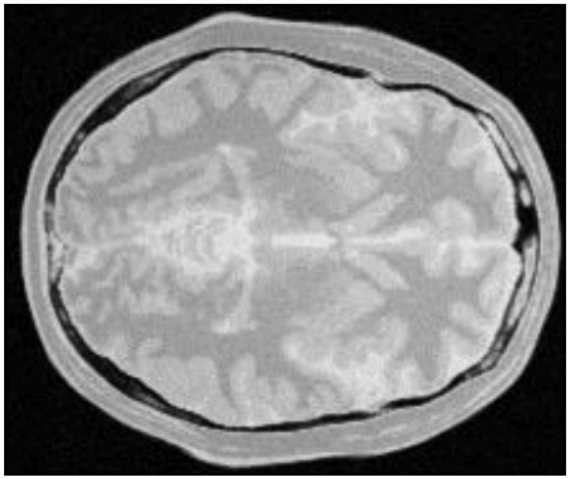
Alignment with Mean-Squared-Error

- MSSD is one example of an "image similarity measure".

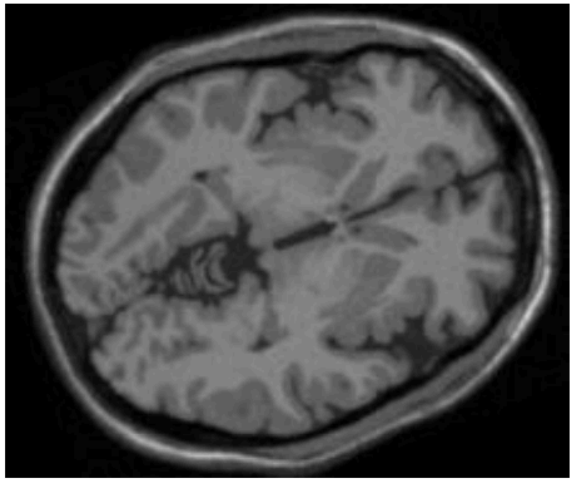
- MAJOR ASSUMPTION: Physically corresponding pixels have the same intensity, i.e. they are acquired by similar cameras and under the same lighting condition (this is often called as **mono-modal image registration**).

Image Alignment: Intensity changes in Images

- Images acquired by different sensors (MR and CT, different types of MR, camera with and without flash, etc.)
- Changes in lighting condition
- This is called as **multimodal image registration**.



MR-PD



MR-T1

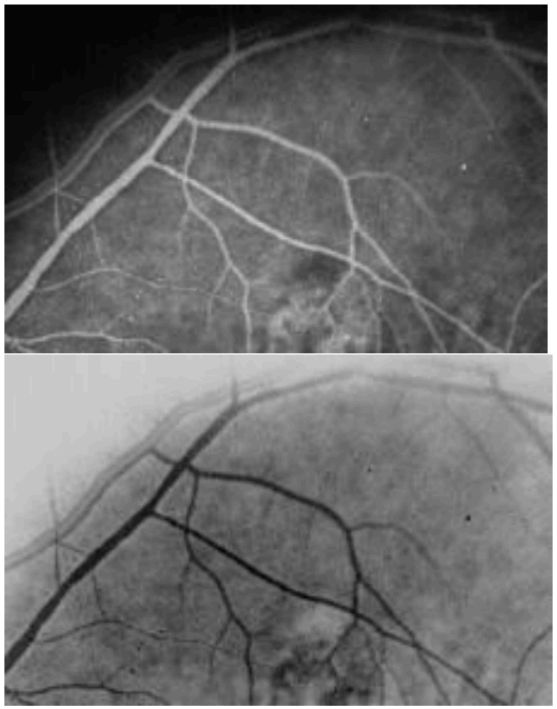


Image Alignment: Intensity changes in Images

- If the following relationship exists and we **knew** the exact functional form (say g), the solution is easy:

$$I_2(x,y) = g\big(I_1(x,y)\big) \quad \forall (x,y) \in \Omega$$

$$E = \frac{1}{N} \sum_{(x,y)\in\Omega} \left(g\big(I_1(x,y)\big) - I_2(x,y)\right)$$

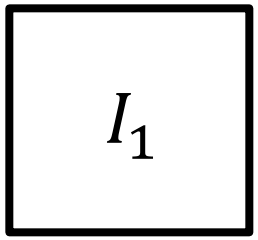
Image Alignment: Intensity changes in Images

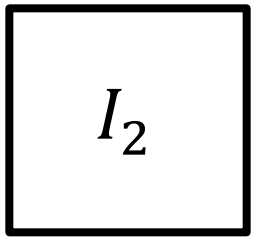
- What if we assumed the change in intensity to be linear (do not know a, b)

$$I_2(x, y) = aI_1(x, y) + \mathrm{b} \quad \forall (x, y) \in \Omega$$

Use Normalized Cross-Correlation

Image Alignment: Intensity changes in Images





intensity normalization

$$\widehat{I_1}(x,y) = \frac{I_1(x,y) - \overline{I_1}}{\sqrt{(I_1(x,y) - \overline{I_1})^2}}$$

$$\widehat{I_2}(x,y) = \frac{I_2(x,y) - \overline{I_2}}{\sqrt{(I_2(x,y) - \overline{I_2})^2}}$$

$\overline{I_1}$, $\overline{I_2}$: average values of images I_1, I_2

Range?

$$NCC(I_1, I_2) = \left| \sum_{(x,y)\in\Omega} \widehat{I_1}(x,y)\widehat{I_2}(x,y) \right|$$

Image Alignment: Intensity changes in Images

$$\widehat{I_1}(x,y) = \frac{I_1(x,y) - \overline{I_1}}{\sqrt{(I_1(x,y) - \overline{I_1})^2}} \qquad \widehat{I_2}(x,y) = \frac{I_2(x,y) - \overline{I_2}}{\sqrt{(I_2(x,y) - \overline{I_2})^2}}$$

$\overline{I_1}$, $\overline{I_2}$: average values of images I_1, I_2

$$NCC(I_1, I_2) = \left| \sum_{(x,y)\in\Omega} \widehat{I_1}(x,y)\widehat{I_2}(x,y) \right|$$

$$\mathbf{T}^* = \arg\max_{\mathbf{T}} NCC(I_2(\mathbf{v}), I_1(\mathbf{T}\mathbf{v}))$$

$$\mathbf{T} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{v} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

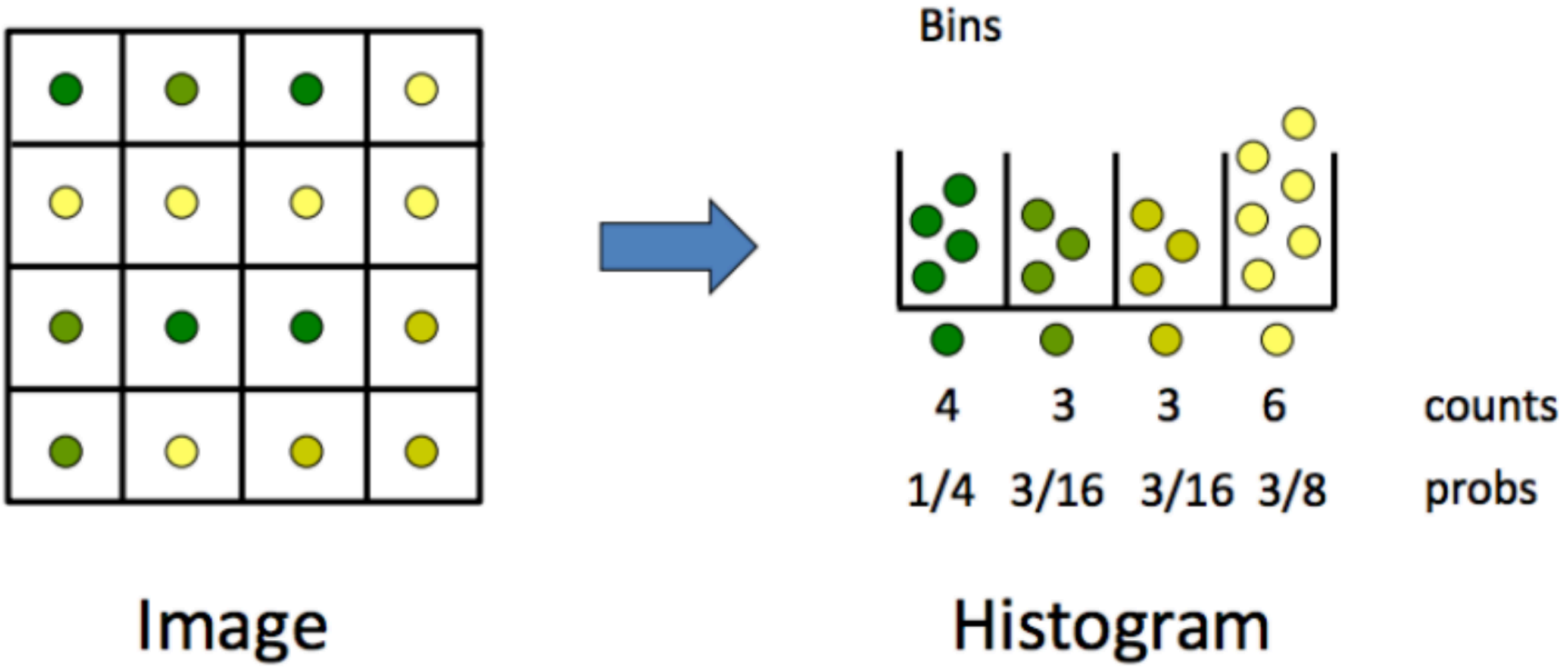
Image Alignment: Intensity changes in Images

- Assume there exists a functional relationship between intensities at physically corresponding locations in the two images
- But suppose we didn't know it (most practical scenario) and couldn't find it out
- We will need image representations: e.g. image histograms (simplest)

Image Histogram

- In a typical digital image, the intensity levels lie in the range [0,L-1].
- The histogram of the image is a discrete function of the form $P(r_k) = n_k/HW$, where r_k is the k-th intensity value, and n_k is the number of pixels with that intensity.
- Sometimes, we may consider a range of intensity values for one entry in the histogram, in which case $r_k = [r^{min}_k, r^{max}_k]$ represents an intensity bin, and n_k is the number of pixels whose intensity falls within this bin.
- Note $P(r_k) >= 0$ always, and all the $P(r_k)$ values sum up to 1.

Image Histogram Calculation



http://www.cs.ucf.edu/~bagci/teaching/mic16/lec16.pdf

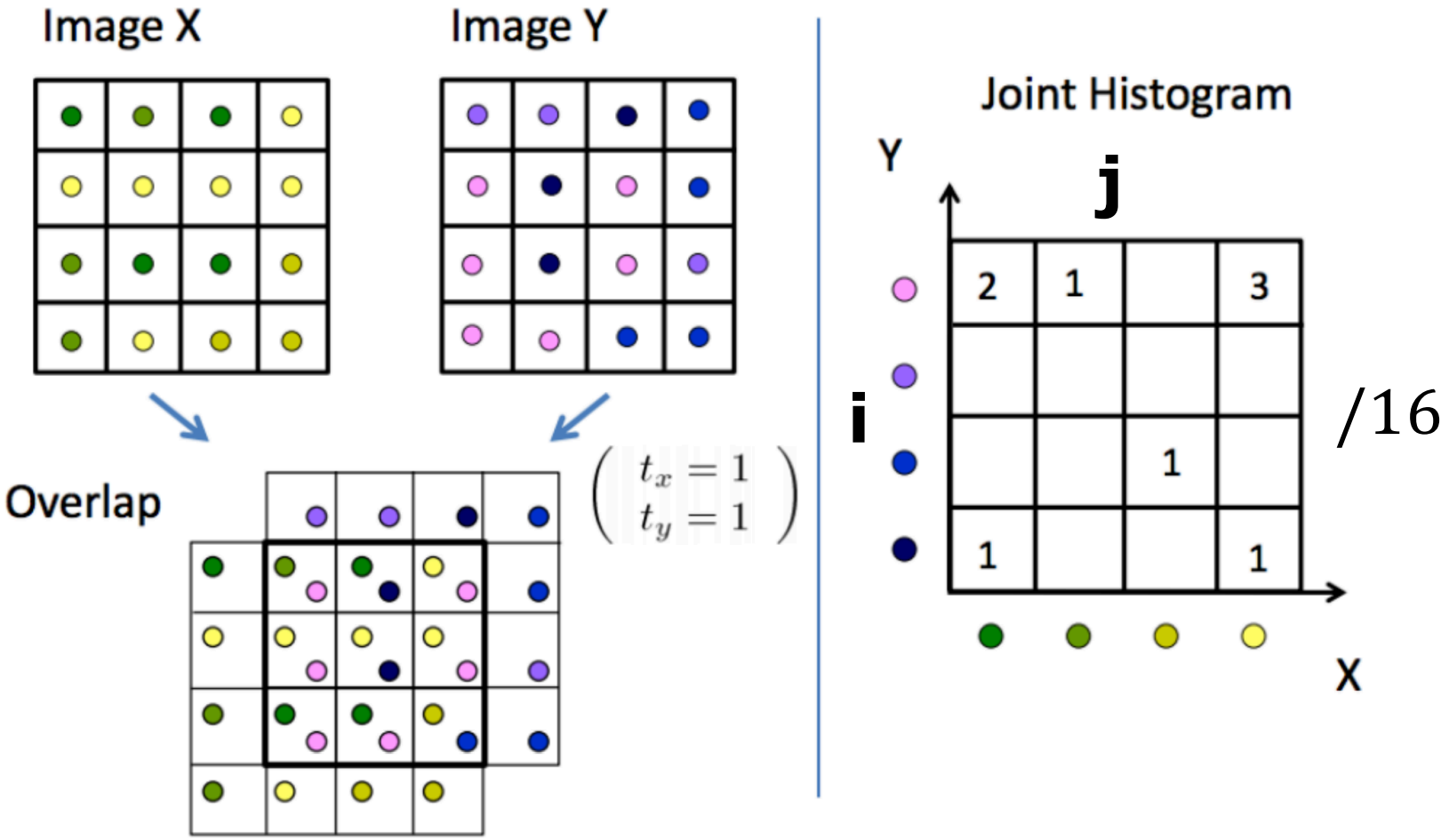
Joint Image Histogram

- Function of the form $P(r_{k1}, r_{k2})$ where r_{k1} and r_{k2} represent intensity bins from the two images I_1 and I_2 respectively.

- $P(r_{k1}, r_{k2})$ = number of pixels (x,y) such that $I_1(x,y)$ and $I_2(x,y)$ lie in bins r_{k1} and r_{k2} respectively, divided by HW.

Joint Histogram Calculation

http://www.cs.ucf.edu/~bagci/teaching/mic16/lec16.pdf

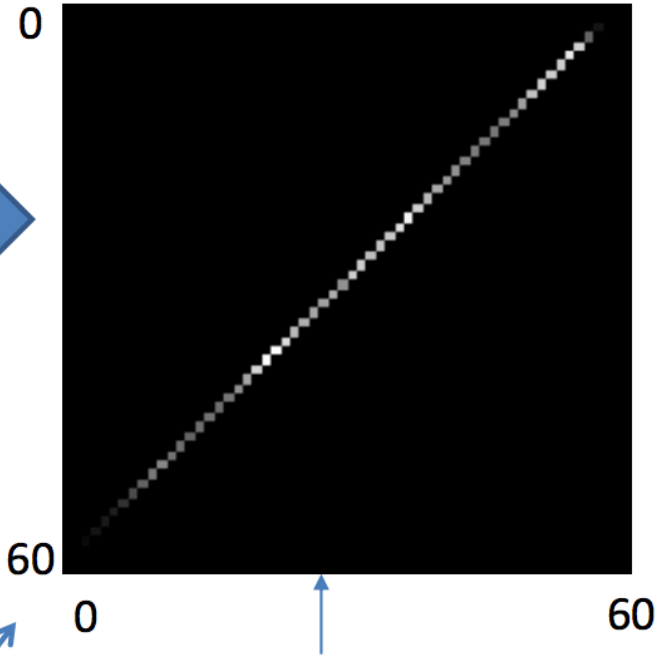


Location (**i,j**) in the joint histogram tells us how many intensity values we have encountered that have intensity **i** in the 1st image and intensity **j** in the 2nd image.

I_1

I_2

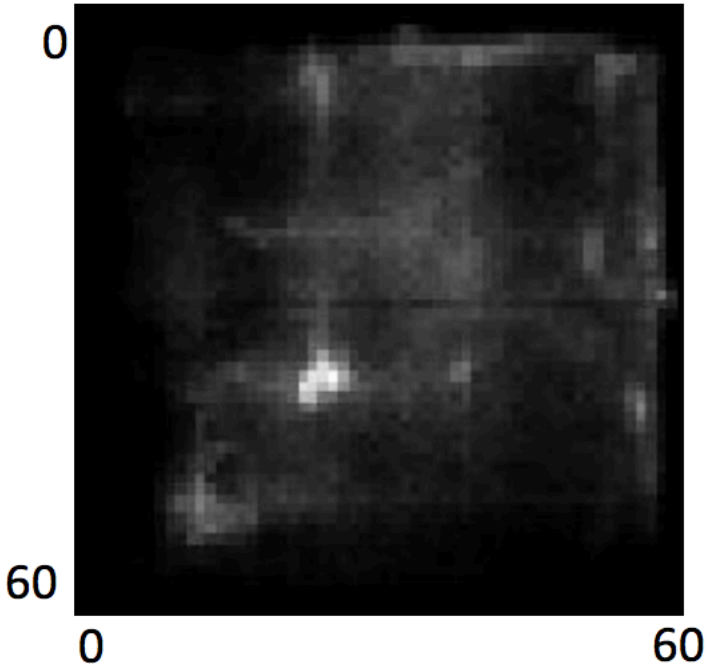
Values in I_2 from 0 to 60



Registered images: joint histogram plot looks "streamlined"

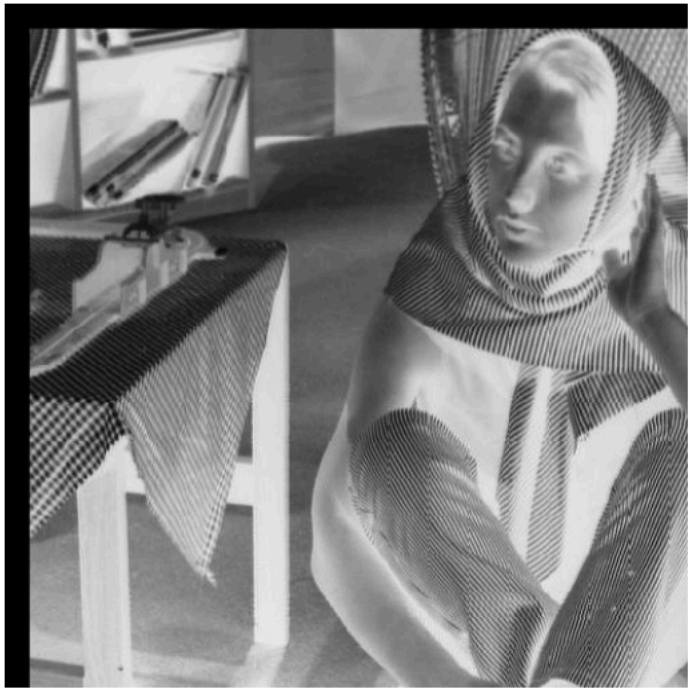
How was this plot generated? The joint histogram is plotted as a grayscale image. Brighter points in this image indicate larger probabilities and darker points indicate lower probabilities.

Values in I_2 from 0 to 60

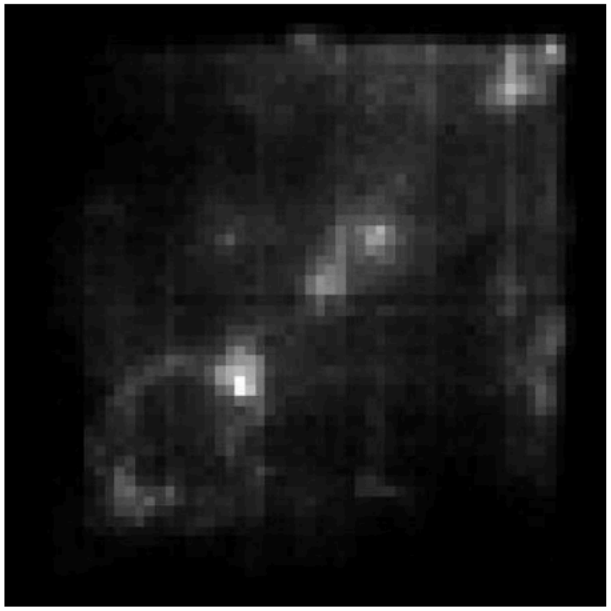


Misaligned images: joint histogram plot looks "dispersed"

We need a method to quantify how dispersed a joint histogram actually is.



Values in I2 (0 to 60, top to bottom)



Values in I1 (0 to 60)

Misaligned images: the joint histogram plot appears dispersed.

We need a method to quantify how dispersed a joint histogram actually is.

Measure of Dispersion

- Consider a discrete random variable *X* with normalized histogram P(*X*=*x*) [also called the probability mass function].
- The entropy of *X* is a measure of the **uncertainty** in *X*, given by the following formula:

$$H(X) = -\sum_{x \in DX} P(X = x) \log_2 P(X = x)$$

$$DX = \text{discrete set of values that } X \text{ can take}$$

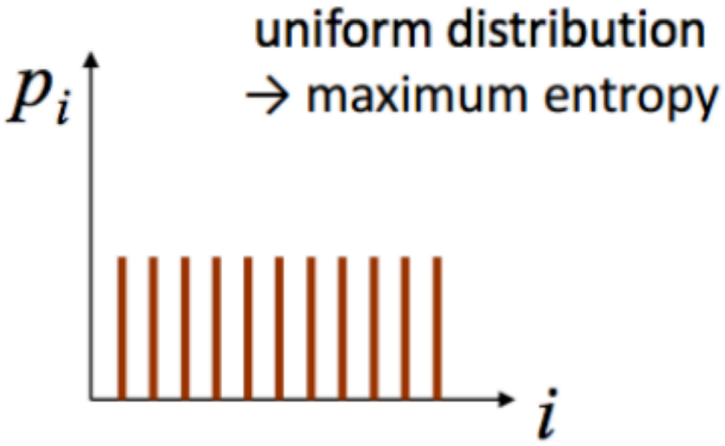
- Entropy is a function of the *histogram* of *X, i.e. of* P(*X*).
- It is not a function of the *actual values* of *X.*
- The entropy is always non-negative.

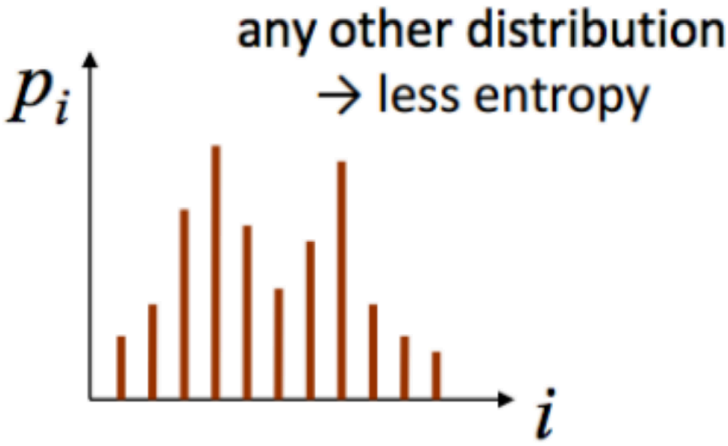
Entropy

- The entropy is maximum if *X* has a discrete uniform distribution, i.e. *P(X=x1) = P(X=x2)* for all values x_1 and x_2 in DX. The maximum entropy value is log(|DX|).

- The entropy is minimum (zero) if the normalized histogram of *X* is a Kronecker delta function, i.e. $P(X=x_1) = 1$ for some x_1, and $P(X=x_2) = 0$ for all $x_2 \neq x_1$.

Entropy





http://www.cs.ucf.edu/~bagci/teaching/mic16/lec16.pdf

Joint Entropy

- The joint entropy of two random variables *X* and *Y* is given as follows:

$$H(X,Y) = -\sum_{x \in DX} \sum_{y \in DY} P(X = x, Y = y) \log_2 P(X = x, Y = y)$$

- Maximum entropy:

?

- Minimum entropy:

?

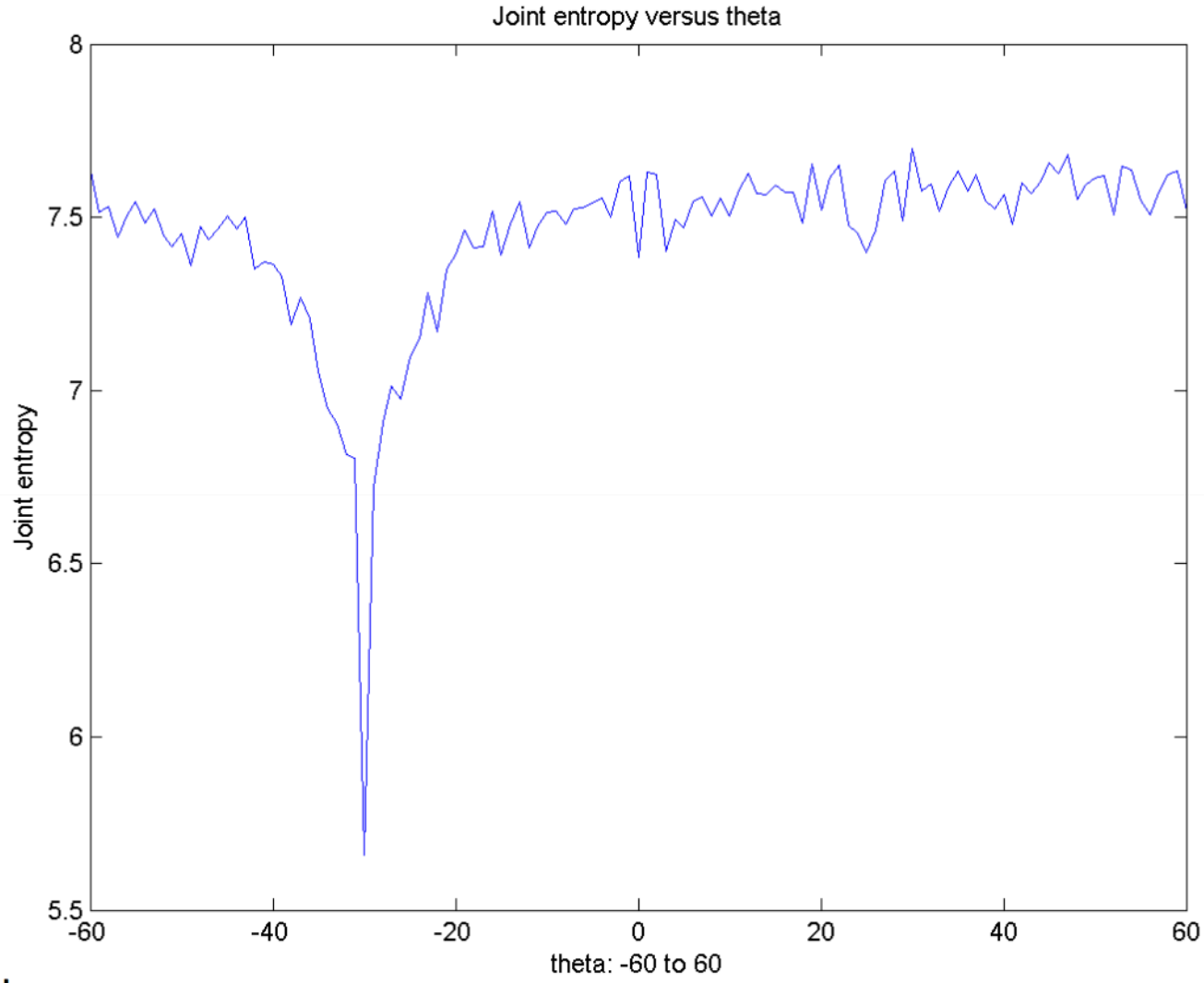
Joint Entropy

- Minimizing joint entropy is one method of aligning two images with different intensity profiles.

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} H(I_2(\mathbf{v}), I_1(\mathbf{T}\mathbf{v}))$$

$$\mathbf{T} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{v} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

I_2



I_1: obtained by squaring the intensities of I_2, and rotating I_2 anticlockwise by 30 degrees.

I_1 treated as moving image, I_2 treated as fixed image. Joint entropy minimum occurs at -30 degrees.

Summary until now:

- First Step: Motion Estimation
  1. Feature point based
  2. Direct pixel value based

- Second Step: Image Warping (bilinear, etc.) (last lecture)

Applications:
Image Alignment

Image Alignment: Applications, Related Problems

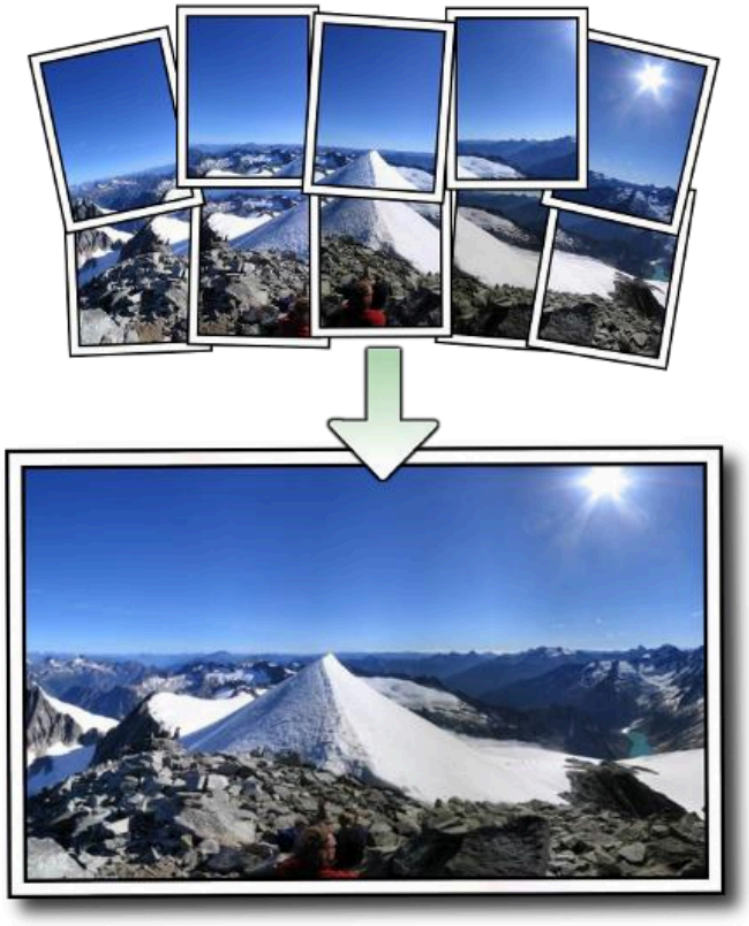
1. Template matching
2. Image Panoramas
3. De-noising image-bursts
4. Collecting photos of paintings
5. Face recognition
6. 3D-2D image registration
7. …

1. Template Matching

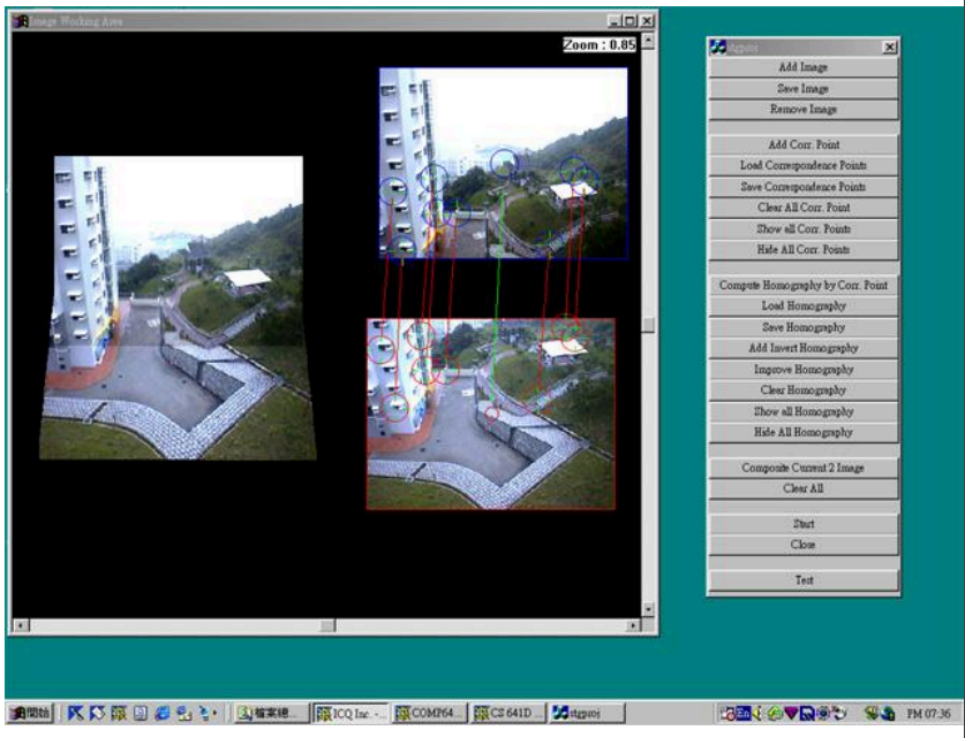
- Look for the occurrence of a template (a smaller image) inside a larger image.
- Example: eyes within face image

Templates

2. Image Panoramas



http://cs.bath.ac.uk/brown/autostitch/autostitch.html



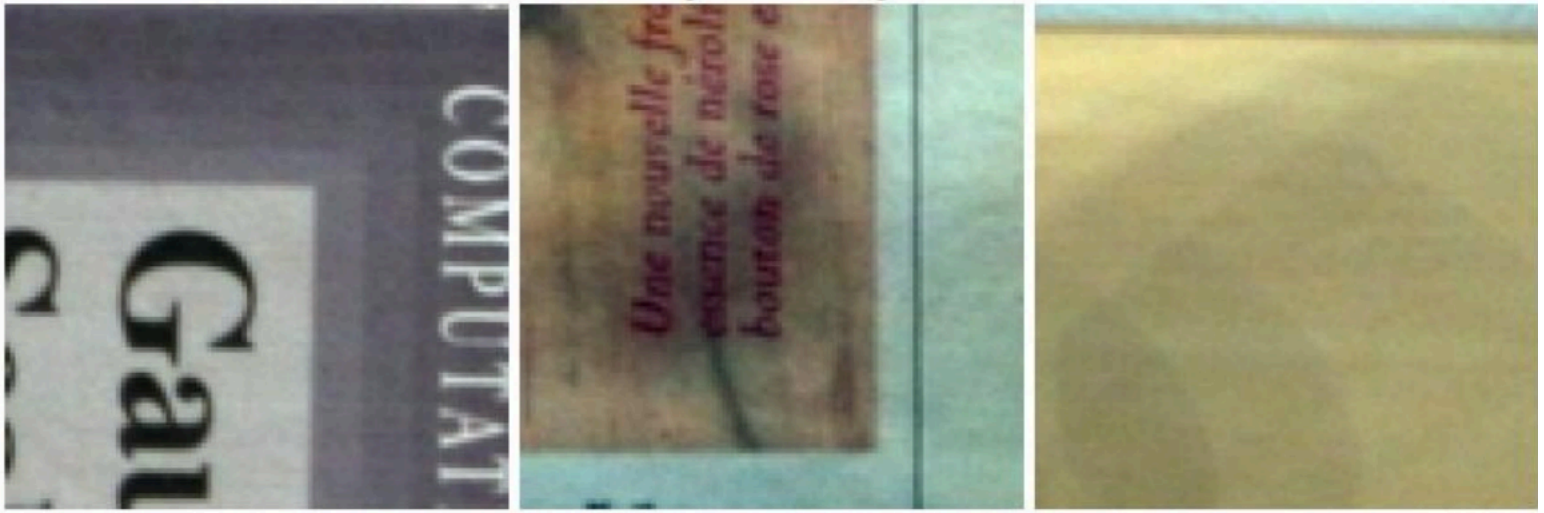
3. Denoising Image Bursts

- An image burst is a collection of photos (of the same scene) taken in quick succession, each with very short exposure time.
- Each image is sharp but usually quite noisy (even more so if the lighting was poor).
- Due to camera motion during burst acquisition, the images can be slightly misaligned.
- You can align the images (say using SIFT for the control points and assuming a homography motion model) and then simply average the images after alignment to remove the noise.

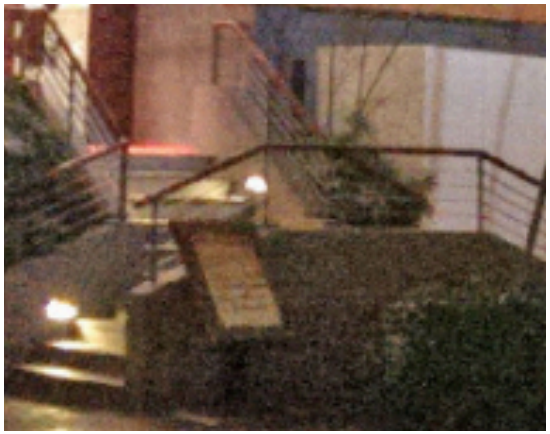
ftp://ftp.math.ucla.edu/pub/camreport/cam09-62.pdf ,Buades et al, "A note on multi-image denoising"
http://research.microsoft.com/en-us/um/people/luyuan/paper/FastBurstDenoising_SIGGRAPHASIA14.pdf

3. Denoising Image Bursts

The average after registration



3. Denoising Image Bursts



4. Photographing Paintings

- The Google Art Project (https://www.google.com/culturalinstitute/user-galleries?projectId=art-project) sought to acquire high-resolution photographs of paintings from famous museums.
- Acquiring such photographs requires very specialized equipment, controlled illumination and intensive post-processing.
- The major problem in photographing a painting is that different portions of the painting exhibit a glare, depending on the viewpoint in which the picture was taken.
- If the painting is behind a glass/plastic frame, one sometimes sees the reflection of the observer in it.
- These glares and reflections change their location, if you change the viewpoint in which the picture was taken.

4. Photographing Paintings

Figure 3: Two sorts of highlights appearing in the photograph of paintings : on the left, the middle up part has a diffuse highlight, creating a bright speckle. On the right the glass covering the painting reflects the observer and the rest of the room. Both perturbations are impossible to remove from a single view.

See:
http://www.siam.org/publicawareness/art.php
http://epubs.siam.org/doi/abs/10.1137/120873923

4. Photographing Paintings

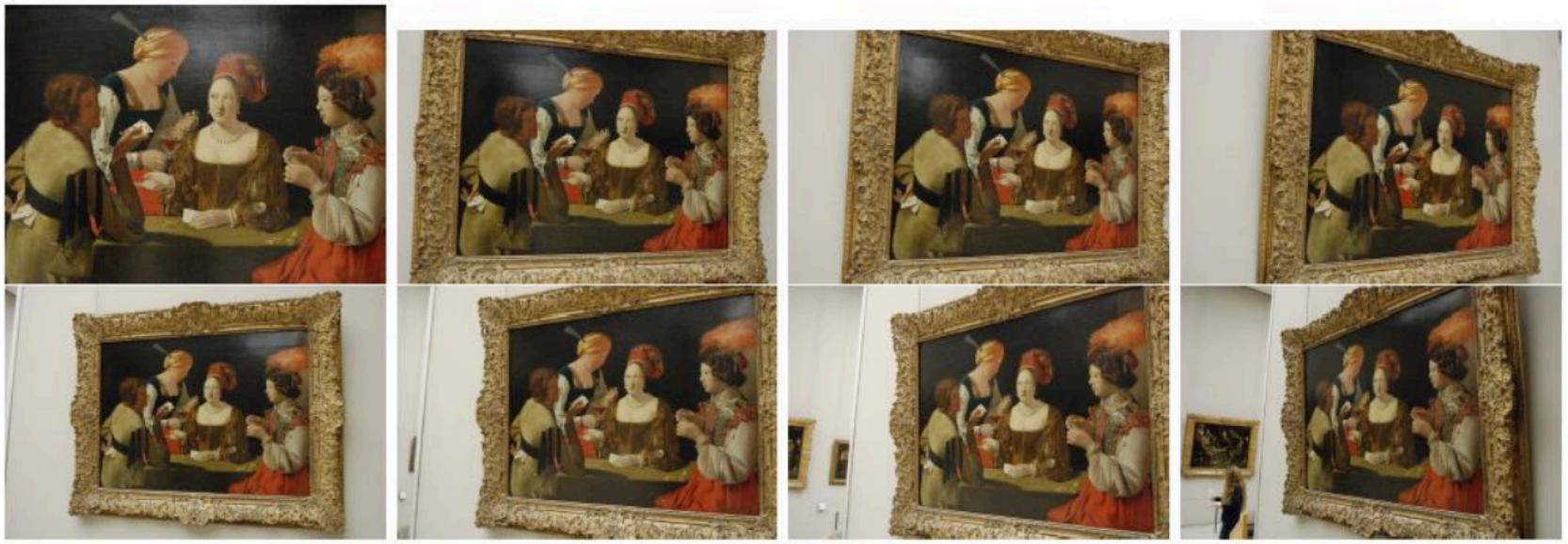


Figure 4: Example of burst detection: each image is taken from one of the eight bursts detected in the original set of 87 images. Each burst corresponds to a different point of view. Notice the moving highlights, covering large regions.

See:
http://www.siam.org/publicawareness/art.php
http://epubs.siam.org/doi/abs/10.1137/120873923

4. Photographing Paintings

- In an approach proposed by Haro, Buades and Morel (http://epubs.siam.org/doi/abs/10.1137/120873923), one takes several bursts of pictures of the painting from different viewpoints.
- All these images are aligned together using SIFT+homography.
- This is followed by image fusion, i.e. computing an average or median of all aligned images (the paper actually does this differently, but that is not so important in the present context of image alignment).

4. Photographing Paintings

Figure 14: Latour example. Left: one of the input images. Right: result of median of gradients after denoising plus sharpening with two iterations. Second row: zoom views of the images in the first row.

5. Face Recognition

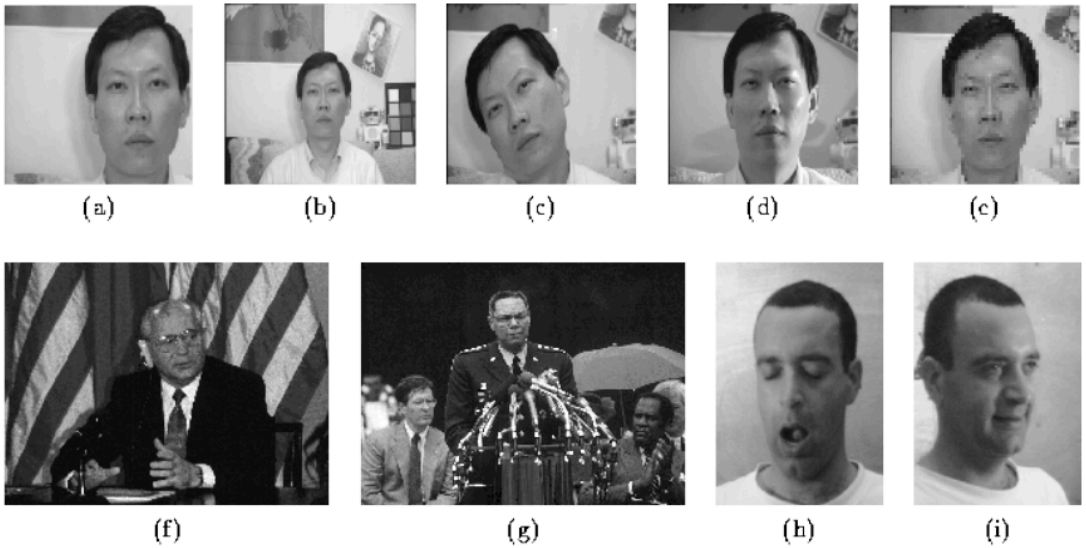
- In a face recognition application, you first store one or more images of each person (say students/staff at IITB) in a database. These are called **gallery images**.
- Given an image of a person at some later point of time, the task is to match the image to the set of gallery images – in order to determine identity.
- This is called the **probe image**.
- The probe image can be in a **different pose** than the gallery image of the same person.

5. Face Recognition

- If the gallery and pose image had only in-plane motion relative to each other, we could use one of the earlier discussed methods to find the unknown motion. Example below:

5. Face Recognition

- But this does not handle the much more realistic issue of out-of-plane motion. What does one do then?



http://www.cs.columbia.edu/~jebara/htmlpapers/UTHESIS/node30.html

Figure 3.1: Variations in faces that require appropriate compensation. (a) Change in position. (b) Change in size or scale. (c) In-plane rotation of the face. (d) Shading and illumination effects. (e) Variation in image quality or resolution. (f) Clutter in the image background. (g) Multiple faces in the image. (h) Changes in facial expression. (i) Out-of-plane rotation.

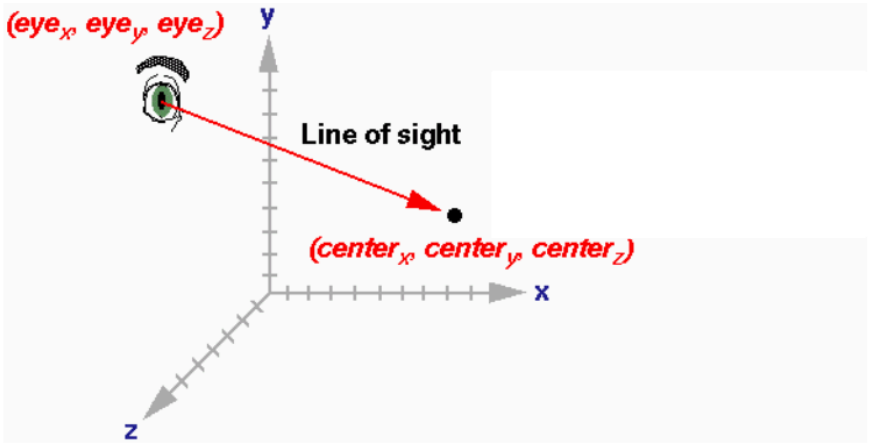
6. 3D-2D Registration

- This motivates the use of 3D face scans or 3D models – which can be acquired by 3D cameras such as stereo-cameras or structured light sensors.
- A 3D face scan consists of a set of vertices in 3D and a set of polygons (in 3D) linking those vertices.

http://www.jonasavrin.com/2011/01/15/free-3d-ir-head-scan-release-smart-hdr-ibl-vray-2-0/

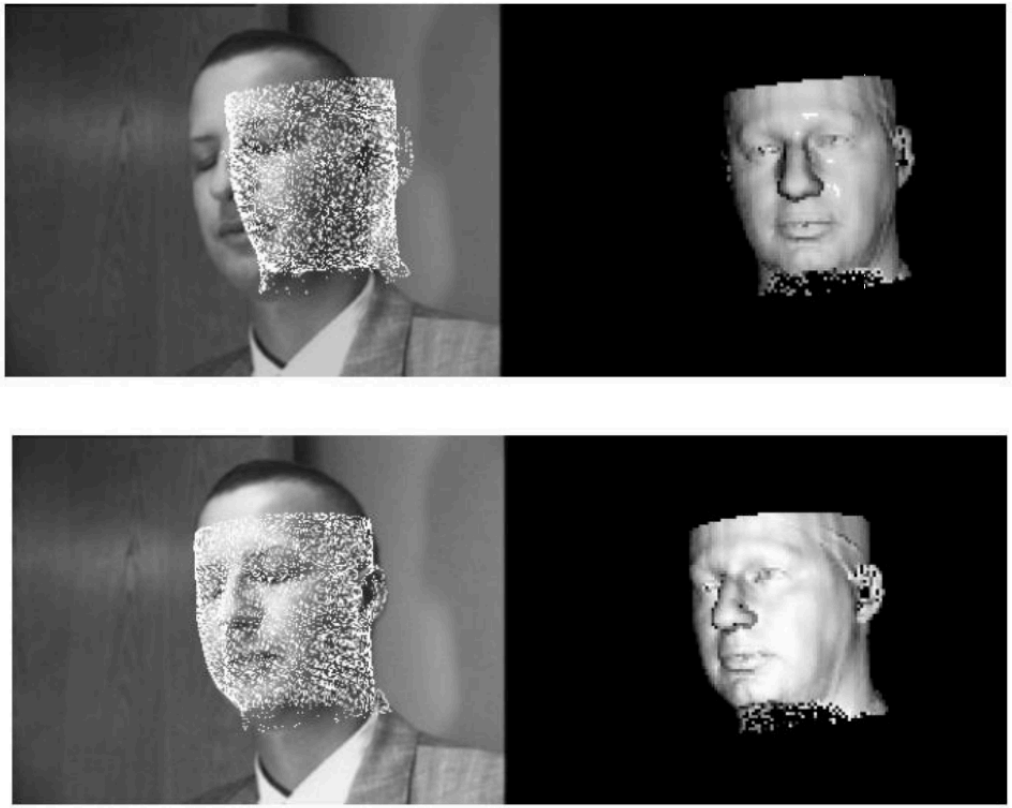
6. 3D-2D Registration

- Given the 3D model of a person, you need to match the probe image to the model.
- How? You create images by projecting the 3D model in different viewing directions, and then match each image with the probe image.
- How many DoF involved?



http://www.alpcentauri.info/glulookat1.html

6. 3D-2D Registration



Source: PhD thesis of Paul Viola at MIT (1995)
http://research.microsoft.com/pubs/66721/phd-thesis.pdf
The method employed was optimization of "mutual information" (closely related to the joint entropy technique) we studied in class.

What We Learnt

- Motion models
- Forward and reverse image warping
- Field of view during image alignment
- Measures for Image alignment: sum of squared differences, normalized cross-correlation, joint entropy
- Registration using control points

What We Didn't Learnt

- Complicated motion models: higher degree polynomials, non-rigid models (example: motion of an amoeba, motion of the heart during the cardiac cycle, facial expressions, etc.)

- Efficient techniques for optimizing the measure for image alignment

Aspects of Image Registration

- Are the images in 2D or in 3D? (2D-2D, 3D-3D, 3D-2D)
- Is the motion model parametric or nonparametric (non-rigid)?
- Do the images have equal intensity values at physically corresponding points? (Unimodal or multimodal). This decides the objective function to be optimized.

Slide Information

- The slide template has been created by Cyrill Stachniss (cyrill.stachniss@igg.uni-bonn.de) as part of the photogrammetry and robotics courses.
- A lot of material from Ajit Rajwade's CS763 course
- **I tried to acknowledge all people from whom I used images or videos. In case I made a mistake or missed someone, please let me know.**

Arjun Jain, ajain@cse.iitb.ac.in