Image Alignment

AJIT RAJWADE

Basics

- A digital image – a discrete/sampled version of a visual stimulus
- Can be regarded as a function $I = f(x,y)$ where (x,y) are spatial (integer) coordinates in a domain $\Omega = [0,H\text{-}1] \times [0,W\text{-}1]$.
- Each ordered pair (x,y) is called a pixel.
- The pixel is generally square (sometimes rectangular) in shape.

Basics

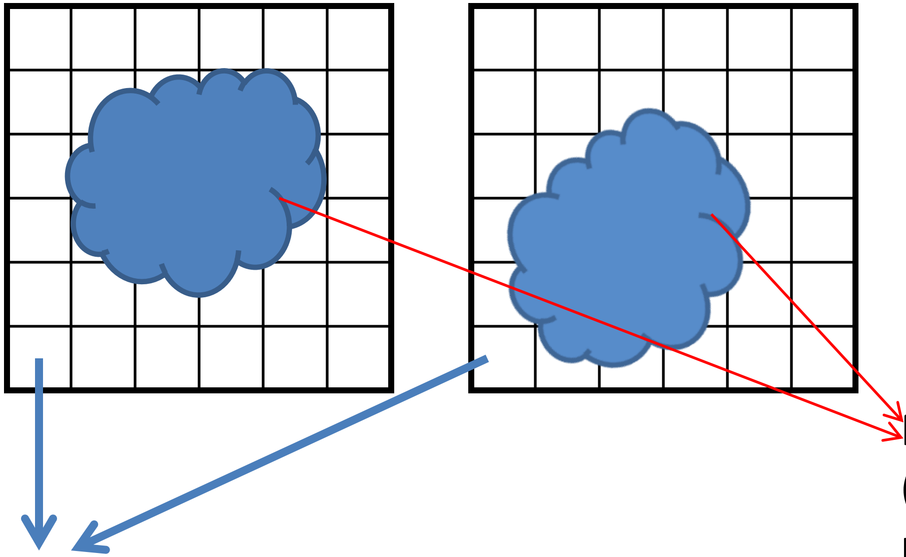
- Pixel dimensions (height/width) relate to the spatial resolution of the sensor in the camera that collects light reflected from a scene.

Basics

- In a typical grayscale image, the intensity values *f(x,y)* lie in the range from 0 to 255 (8 bit integers).
- They are quantized versions continuous values corresponding to the actual light intensity that strikes a light sensor in the camera.

Image Alignment

- Consider images I_1 and I_2 of a scene acquired through different viewpoints.



Pixels in **digital correspondence** (same coordinate values in the image domain Ω, not necessarily containing measurements of the same physical point)

Pixels in **physical correspondence** (containing measurements of the same physical point, but not necessarily the same coordinate values in the image domain Ω)

Image Alignment

- I_1 and I_2 are said to be aligned if for every *(x,y)* in the domain Ω, the pixels in I_1 and I_2 are in physical correspondence.

- Image **alignment** (also called **registration**) is the process of correcting for the relative motion between I_1 and I_2.

Image alignment: steps

- First step: motion estimation
- Second step: image warping

Motion estimation

Motion Models

- Let us denote the coordinates in I_1 as (x_1,y_1) and those in I_2 as (x_2,y_2).
- Translation: $$x_2 = x_1 + t_x \qquad \forall (x_1, y_1) \in \Omega$$ $$y_2 = y_1 + t_y$$
- Rotation about point *(0,0)* anti-clockwise through angle θ

$$x_2 = x_1 \cos\theta - y_1 \sin\theta$$

$$y_2 = x_1 \sin\theta + y_1 \cos\theta$$

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

2D Rotation matrix
(orthonormal matrix)

Motion Models

- Rotation about point *(x_c,y_c)* anti-clockwise through angle θ

$$x_2 = (x_1 - x_c)\cos\theta - (y_1 - y_c)\sin\theta + x_c$$

$$y_2 = (x_1 - x_c)\sin\theta + (y_1 - y_c)\cos\theta + y_c$$

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & x_c \\ \sin\theta & \cos\theta & y_c \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 - x_c \\ y_1 - y_c \\ 1 \end{pmatrix}$$

-Perform translation such that (x_c,y_c) coincides with the origin.
-Rotate about the new origin.
-Translate back.

Motion Models

- Rotation and translation:

$$x_2 = (x_1 - x_c)\cos\theta - (y_1 - y_c)\sin\theta + t_x$$

$$y_2 = (x_1 - x_c)\sin\theta + (y_1 - y_c)\cos\theta + t_y$$

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 - x_c \\ y_1 - y_c \\ 1 \end{pmatrix}$$

- Affine transformation: rotation, translation, scaling and shearing

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

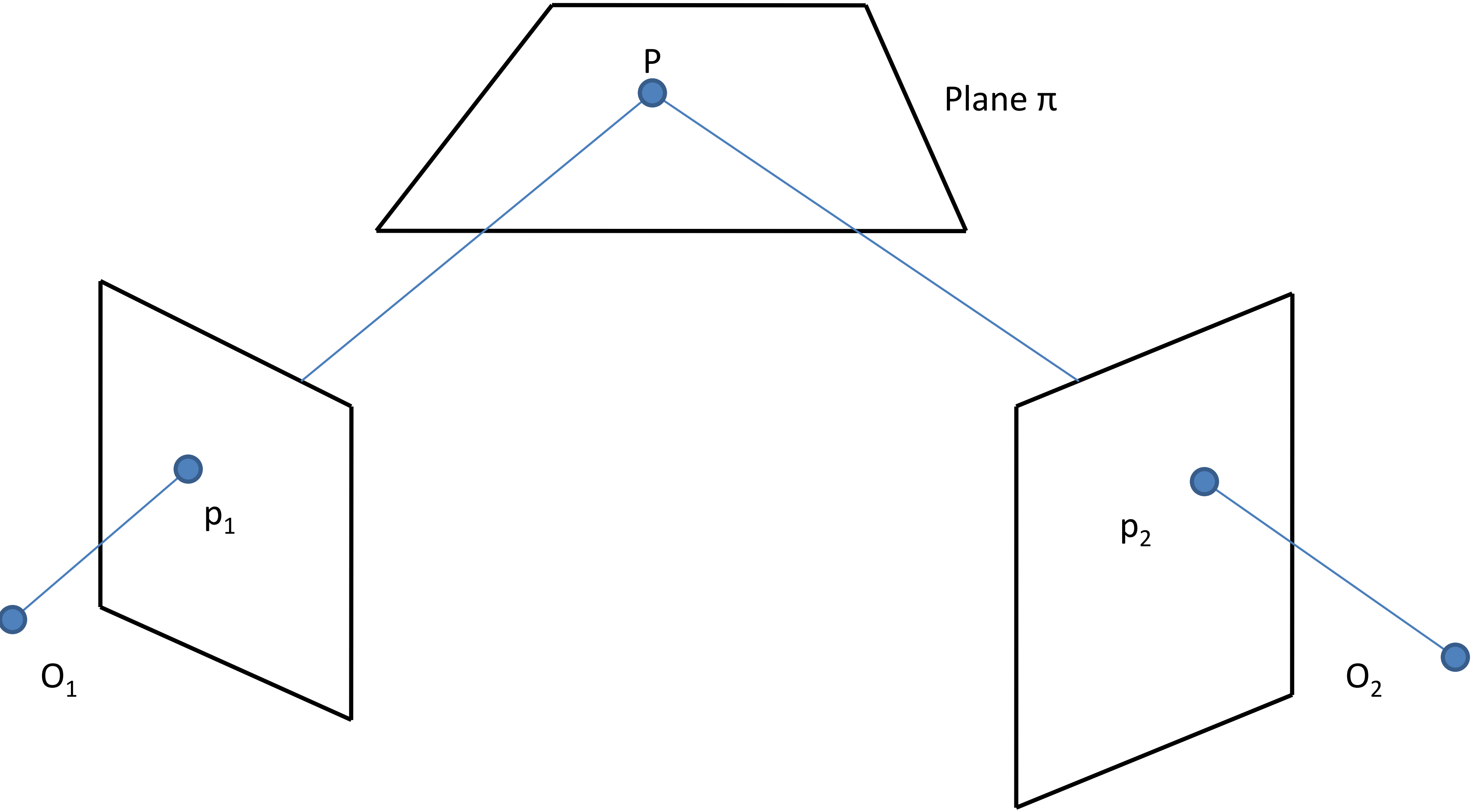
Assumption: the 2 x 2 sub-matrix **A** is NOT rank deficient, otherwise it will transform two-dimensional figures into a line or a point

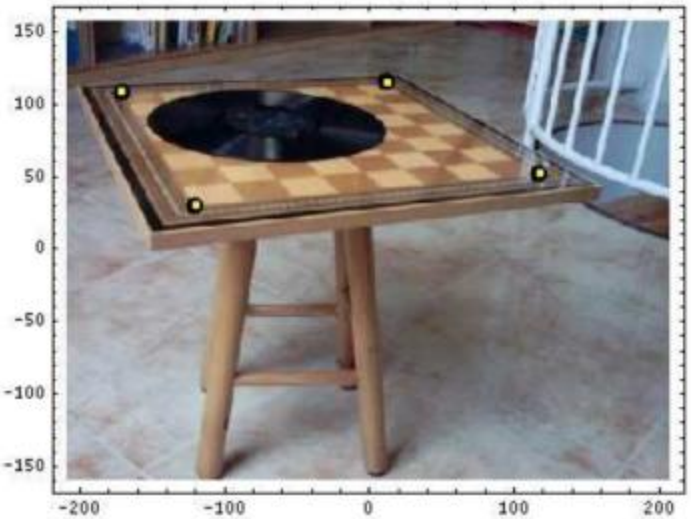
Motion Models

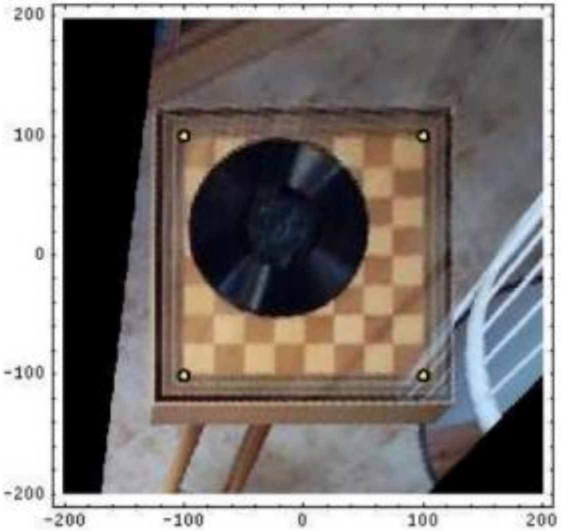
- The 2D affine motion model (including translation in X and Y direction) includes 6 degrees of freedom.
- Note: this motion model accounts for in-plane motion only (example: not an appropriate model for “head profile view versus head frontal view”)

Motion Models

- Consider two images (perspective projection) of a plane in 3D space.
- These images are said to be related by a motion model called as a planar homography – which we studied last class.


P
Plane π
p₁
O₁
p₂
O₂


150
100
50
0
-50
-100
-150
-200
-100
0
100
200


200
100
0
-100
-200
-200
-100
0
100
200

Whole Grain Guaranteed
Cheerios
send Cheer to military families
thank you!
Toasted Whole Grain Oat Cereal
NET WT 14 OZ (396g)

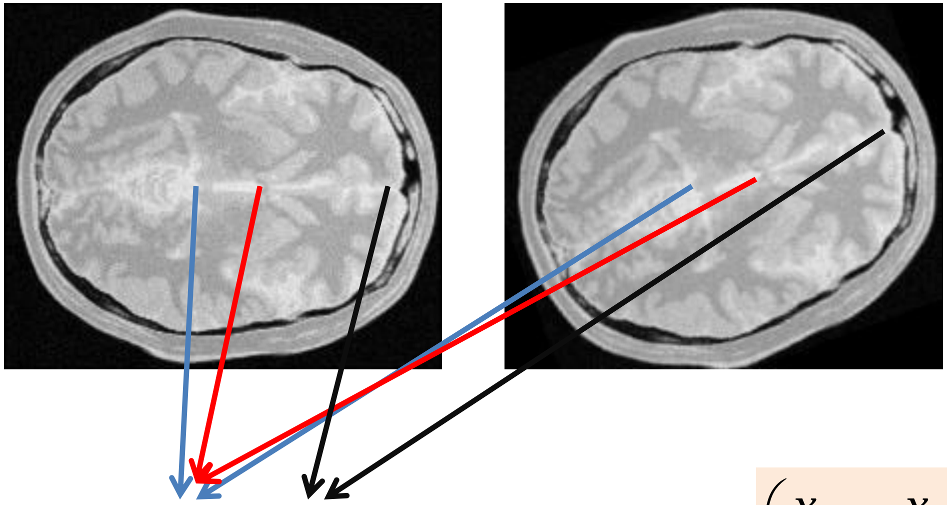
Whole Grain Guaranteed
Cheerios
send Cheer to military families
Toasted Whole Grain Oat Cereal
NET WT 14 OZ (396g)
Cheerios

Motion Models summary

Parametric models

Motion Model	DOF
2D rigid	2+1=3
2D similarity (rigid+ uniform scaling)	3+1=4
2D affine	6
Homography	8
3D rigid	6
3D similarity	6+1=7
3D affine	12
2D (or 3D) non-parametric (also called “non-rigid” or “deformable”)	2 x number of pixels (or 3 x number of voxels)

Alignment with control points



Solve for unknown parameters using least-squares framework (i.e. pseudo-inverse)

Apply the motion based on these parameters to the first image

Control points: pairs of physically corresponding points – maybe marked out manually, or automatically using geometric properties of the image.

Number of control points *N* MUST be **>=** *u*/2, where *u* = number of unknown parameters in the motion model (each point has two coordinates – x and y)

$$\begin{pmatrix} x_{21} & x_{22} & . & . & x_{2k} \\ y_{21} & y_{22} & . & . & y_{2k} \\ 1 & 1 & . & . & 1 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & . & . & x_{1k} \\ y_{11} & y_{12} & . & . & y_{1k} \\ 1 & 1 & . & . & 1 \end{pmatrix}$$

Least squares motion estimation: affine

- For the affine model, we can write the earlier equation as follows:

$$\mathbf{P}_2 = \mathbf{A}\mathbf{P}_1 + noise$$

$$\mathbf{A} = \mathbf{P}_2\mathbf{P}_1^{\mathsf{T}}(\mathbf{P}_1\mathbf{P}_1^{\mathsf{T}})^{-1} = \arg\min_{A^*}\left\|\mathbf{P}_2 - \mathbf{A}^*\mathbf{P}_1\right\|_F^2$$

$$\left\|X\right\|_F^2 = \sum_i\sum_j X_{ij}^2$$

$$\mathbf{A} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$E(\mathbf{A}) = \left\|\mathbf{P}_2 - \mathbf{A}\mathbf{P}_1\right\|_F^2$$

$$\frac{\partial E}{\partial \mathbf{A}} = 2(\mathbf{P}_2 - \mathbf{A}\mathbf{P}_1)\frac{\partial(\mathbf{A}\mathbf{P}_1)}{\partial \mathbf{A}} = 2(\mathbf{P}_2 - \mathbf{A}\mathbf{P}_1)\mathbf{P}_1^{\mathsf{T}} = 0$$

$$\rightarrow \mathbf{A} = \mathbf{P}_2\mathbf{P}_1^{\mathsf{T}}(\mathbf{P}_1\mathbf{P}_1^{\mathsf{T}})^{-1}$$

Least squares motion estimation: homography

- For the homography motion model, we have previously seen that the solution is given as follows:

$$\mathbf{h}^* = \arg\min_{\mathbf{h}} \|\mathbf{A}\mathbf{h}\|^2 \text{ s.t. } \mathbf{h}^t\mathbf{h} = 1$$

$$\mathbf{A} = \begin{pmatrix} -x_{1i} & -y_{1i} & -1 & 0 & 0 & 0 & x_{2i}x_{1i} & x_{2i}y_{1i} & x_{2i} \\ 0 & 0 & 0 & -x_{1i} & -y_{1i} & -1 & y_{2i}x_{1i} & y_{2i}y_{1i} & y_{2i} \\ & & & & & & & & \\ & & & & & & & & \end{pmatrix}; \mathbf{h} = \begin{pmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{pmatrix}$$

$\mathbf{A}$ has size $2N \times 9$, $\mathbf{h}$ has size 9×1

Least squares motion estimation: rigid motion

- For the rigid motion model, we can write the earlier equation as follows:

$$\mathbf{P}_2 = \mathbf{R}\mathbf{P}_1 + \mathbf{t} + noise$$

$$(\mathbf{R}^*, \mathbf{t}^*) = \arg\min_{(\mathbf{R},\mathbf{t})} \|\mathbf{P}_2 - \mathbf{R}\mathbf{P}_1 - \mathbf{t}\|_F^2 \text{ s.t.} \mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{I}$$

This problem cannot be solved correctly by a simple pseudo-inverse because a pseudo-inverse does not impose the fact that **R** is an orthogonal matrix. The correct solution uses the singular value decomposition. It is detailed here below assuming that **t** = **0** (the solution can be extended to the case where **t** is not **0**)

http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2017/procrustes.pdf

Also see:

http://en.wikipedia.org/wiki/Orthogonal_Procrustes_problem

Segway: Singular value Decomposition

- For any *m* x *n* matrix **A**, the following decomposition **always** exists:

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T, \mathbf{A} \in R^{m\times n},$$

$$\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}_m, \mathbf{U} \in R^{m\times m},$$

$$\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}_n, \mathbf{V} \in R^{n\times n},$$

$$\mathbf{S} \in R^{m\times n}$$

Diagonal matrix with non-negative entries on the diagonal – called **singular values.**

Columns of **U** are the eigenvectors of $\mathbf{A}\mathbf{A}^T$ (called the left singular vectors).

Columns of **V** are the eigenvectors of $\mathbf{A}^T\mathbf{A}$ (called the right singular vectors).

The non-zero singular values are the positive square roots of

non-zero eigenvalues of $\mathbf{A}\mathbf{A}^T$ or $\mathbf{A}^T\mathbf{A}$.

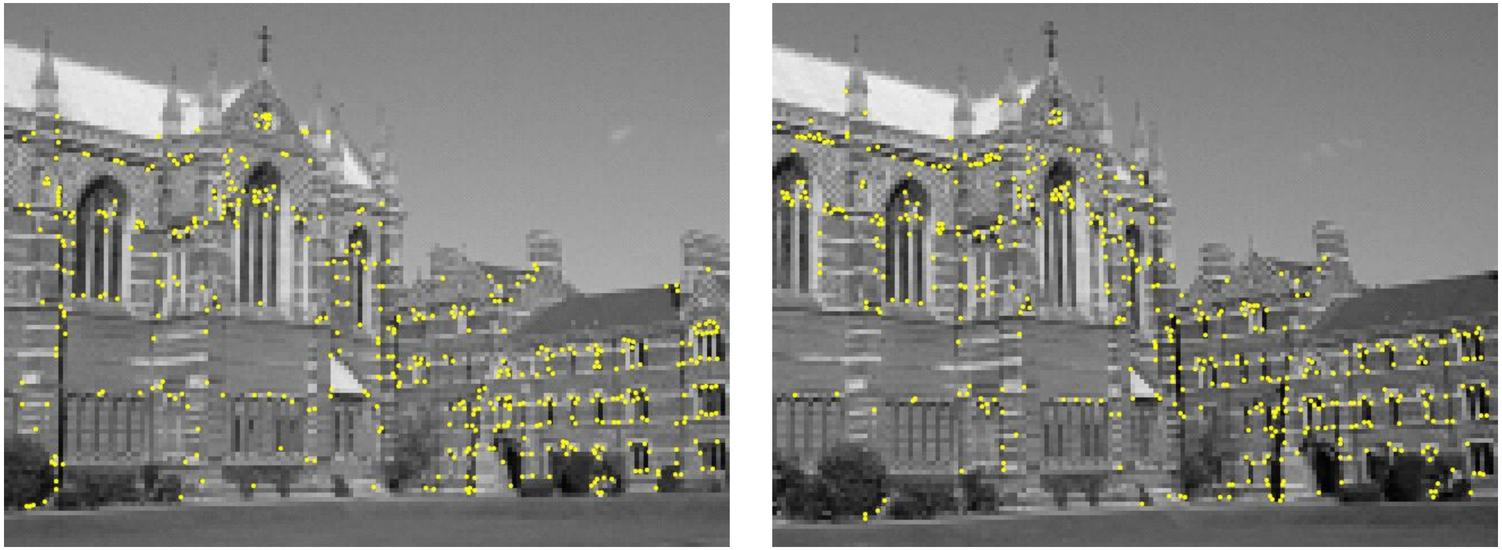
Marking control points

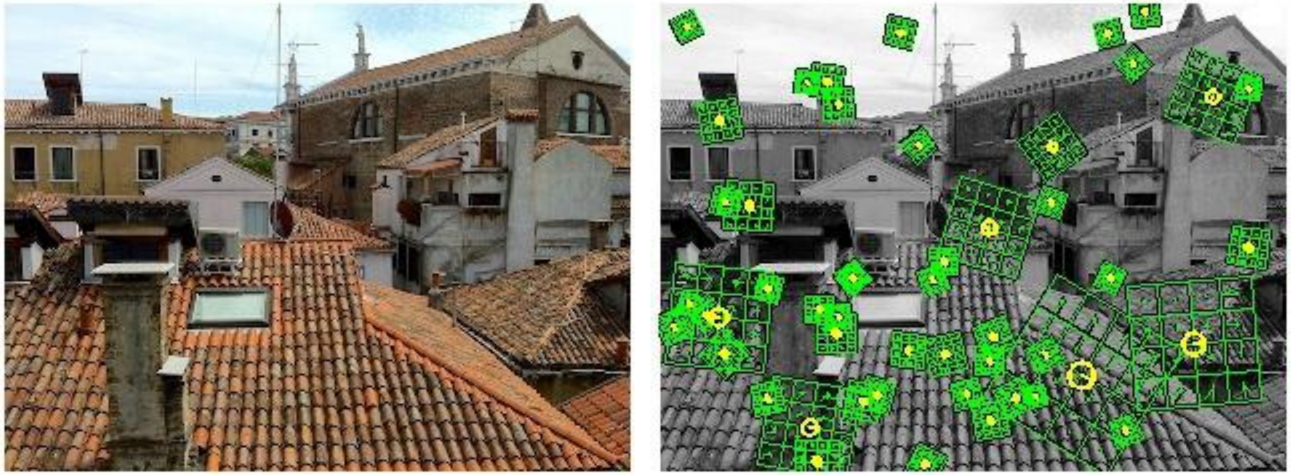
- Control points (also called feature points, salient points, fiducials or salient feature points) can be manually marked – which is error-prone and tedious.
- However this process can also be automated by using salient feature point detectors which are **invariant to geometric transformations**.
- An excellent example of this is SIFT (scale invariant feature transform).

http://www.cs.ubc.ca/~lowe/keypoints/

Marking control points: SIFT

- SIFT does two things: it detect **salient points** in each image and associates each salient point with a **feature vector**.
- The feature vector is a characteristic of a small region of the image around the salient point.
- The feature vector for each salient point is *provably* **invariant** to translation, rotation and reflection – and approximately (or as observed empirically) invariant to affine transformations and even some perspective changes.
- The feature vector is also provably invariant to several illumination changes of the form I' = *a*I + *b*. where I' and I are observed intensities, and *a*,*b* are coefficients (constant for the whole image).
- It is empirically seen to be invariant to many other types of illumination changes.





Marking control points: SIFT

- Due to this invariance, the SIFT technique is able to do a good job at matching salient points from one image to another.
- The motion model can then be estimated from the matched points.

Image warping

Image warping

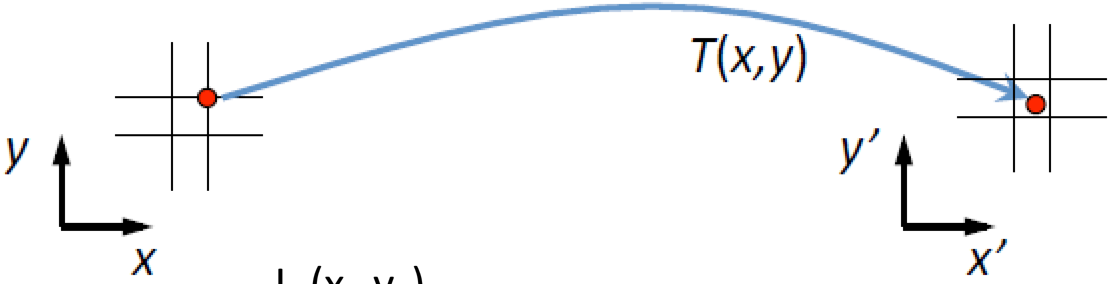
- After motion estimation, the next step is to warp one of the images (say) I_1 using the estimated motion represented by a matrix (say $\mathbf{T}$).

Image Warping

- Forward warping:

  -Apply the motion **Tv** to every coordinate vector **v** = [x_1 y_1 1] in the original image (i.e. I_1).

  -Copy the intensity value from **v** in I_1 to the new location (**Tv**) in the warped image. If the new location is not an integer (most likely), then round off to nearest integer.



I_1 (x_1,y_1)

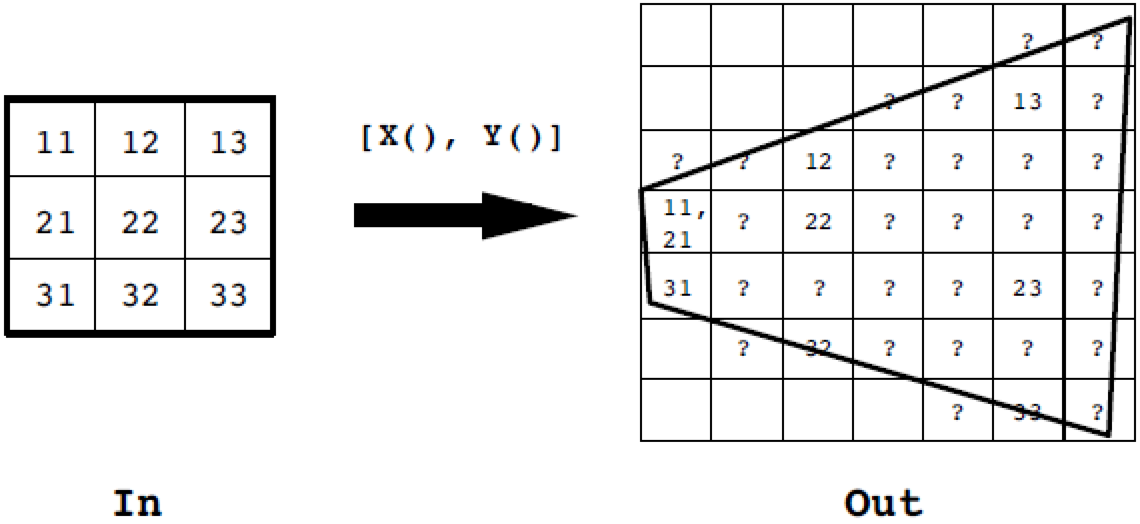
$I_{1\ \text{warped}}$(round(**Tv**))

Image Warping

- Forward warping:

-Can leave the destination image with some holes if you scale up.

-Can lead to multiple answers in one pixel if you scale down.



11	12	13
21	22	23
31	32	33

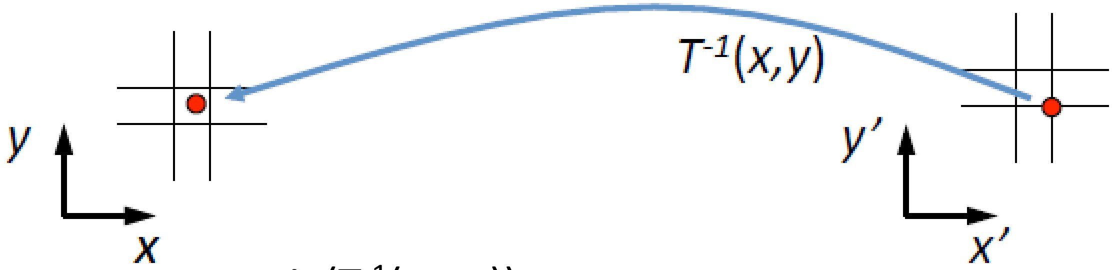
					?	?
			?	?	13	?
?	?	12	?	?	?	?
11, 21	?	22	?	?	?	?
31	?	?	?	?	23	?
	?	32	?	?	?	?
				?	33	?

Image Warping

Note: this assumes that **T** is an invertible transformation, which is guaranteed in the applications we are dealing with. It is **not** guaranteed in case of non-rigid deformations, given the pixelized (i.e. discrete) image coordinate systems.

- Reverse warping:

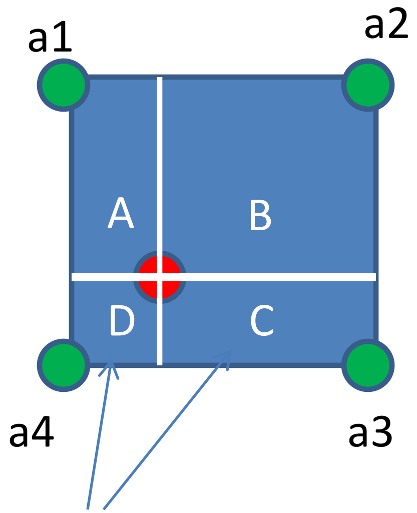
-For every coordinate **v** = [x_2 y_2 1] in the destination image, copy the intensity value from coordinate $\mathbf{T}^{-1}\mathbf{v}$ in the original image. In case of non-integer value, perform interpolation (nearest neighbor or bilinear)



$I_1(\mathbf{T}^{-1}(x_2,y_2))$

$I_{1warped}(x_2,y_2)$

Interpolation



A,B,C,D denotes Areas of these four rectangles

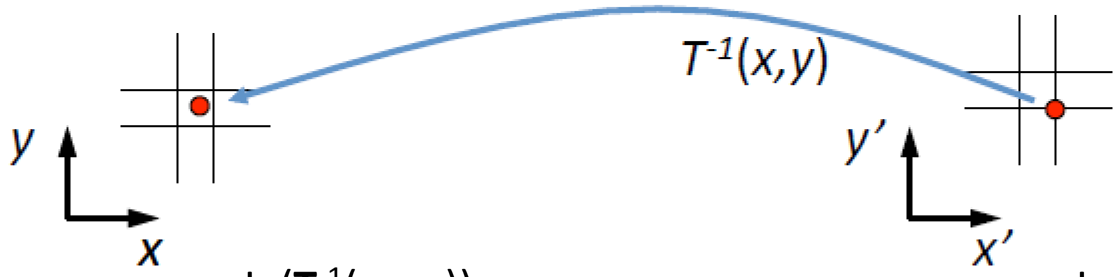
Nearest neighbor method:

- Use value a_4 (as the pixel that is nearest to the red point contains value a_4)

Bilinear method:

- Use the following value, a weighted combination of the four neighboring pixel values, with more weight to nearer values:

$$\frac{Ba_4 + Aa_3 + Ca_1 + Da_2}{(A + B + C + D)} = Ba_4 + Aa_3 + Ca_1 + Da_2$$



$I_1\,(\mathbf{T}^{-1}(x_2,y_2))$

$I_{1warped}\,(x_2,y_2)$

Warping with homographies

- Consider the equation:

$$\mathbf{p}_{2,im} = \begin{pmatrix} u_2 \\ v_2 \\ w_2 \end{pmatrix} = \hat{\mathbf{H}}\mathbf{p}_{1,im} = \begin{pmatrix} \hat{H}_{11} & \hat{H}_{12} & \hat{H}_{13} \\ \hat{H}_{21} & \hat{H}_{22} & \hat{H}_{23} \\ \hat{H}_{31} & \hat{H}_{32} & \hat{H}_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ w_1 \end{pmatrix} = \begin{pmatrix} \hat{H}_{11} & \hat{H}_{12} & \hat{H}_{13} \\ \hat{H}_{21} & \hat{H}_{22} & \hat{H}_{23} \\ \hat{H}_{31} & \hat{H}_{32} & \hat{H}_{33} \end{pmatrix} \begin{pmatrix} x_{1,im} \\ y_{1,im} \\ 1 \end{pmatrix}$$

$$x_{2,im} = \frac{u_2}{w_2} = \frac{\hat{H}_{11}u_1 + \hat{H}_{12}v_1 + \hat{H}_{13}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{11}x_{1,im} + \hat{H}_{12}y_{1,im} + \hat{H}_{13}}{\hat{H}_{31}x_{1,im} + \hat{H}_{32}y_{1,im} + \hat{H}_{33}}, x_{1,im} = \frac{u_1}{w_1}, y_{1,im} = \frac{v_1}{w_1}$$

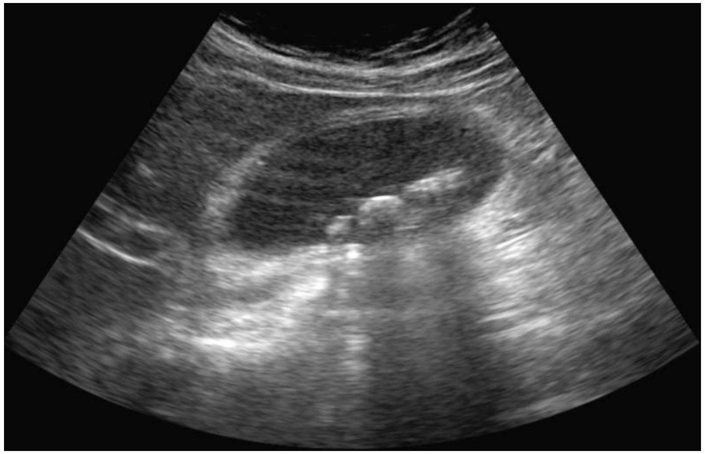
$$y_{2,im} = \frac{v_2}{w_2} = \frac{\hat{H}_{21}u_1 + \hat{H}_{22}v_1 + \hat{H}_{23}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{21}x_{1,im} + \hat{H}_{22}y_{1,im} + \hat{H}_{23}}{\hat{H}_{31}x_{1,im} + \hat{H}_{32}y_{1,im} + \hat{H}_{33}}$$

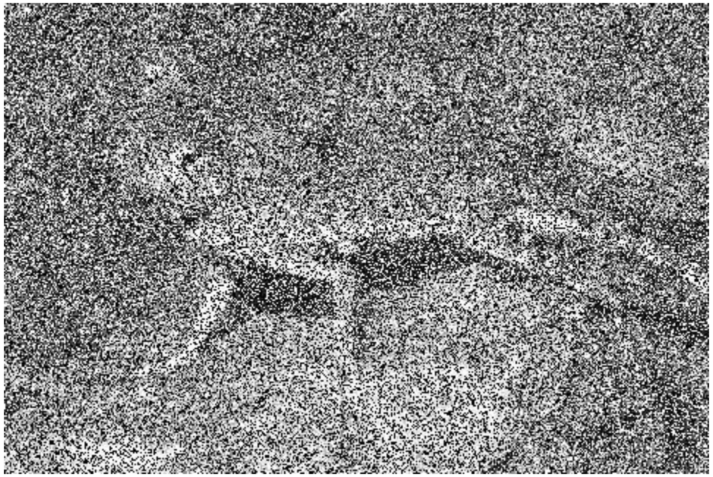
- While warping, note the division by w_2 to yield $x_{2,\mathrm{im}}$ and $y_{2,\mathrm{im}}$. This is different from the affine case!

Image alignment using image similarity measures (without using control points)

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.





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

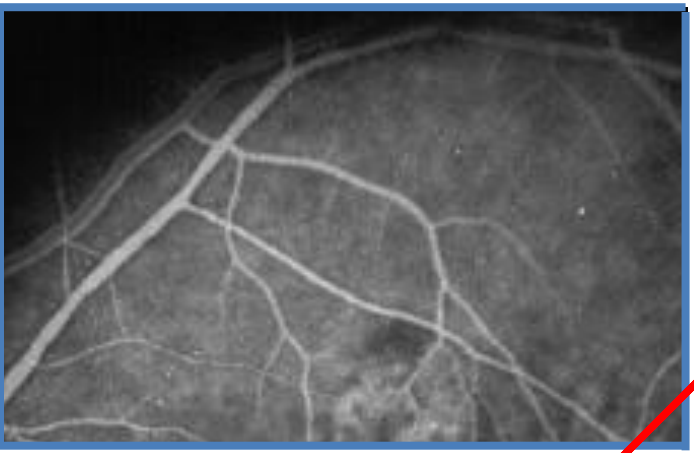
- There are many ways to do this minimization. The simplest but inefficient way is to do a brute-force search.
- Sample θ, t_x and t_y uniformly from a certain range (example: θ from -45 to +45, t_x or t_y from -30 to +30).
- Apply this motion to I_1 keeping I_2 fixed, and compute the MSSD.
- Each time,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

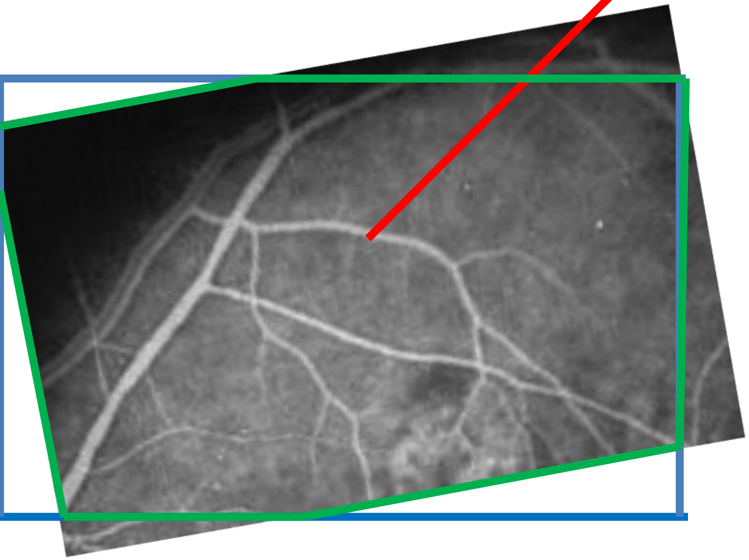
Alignment with mean squared error

- 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

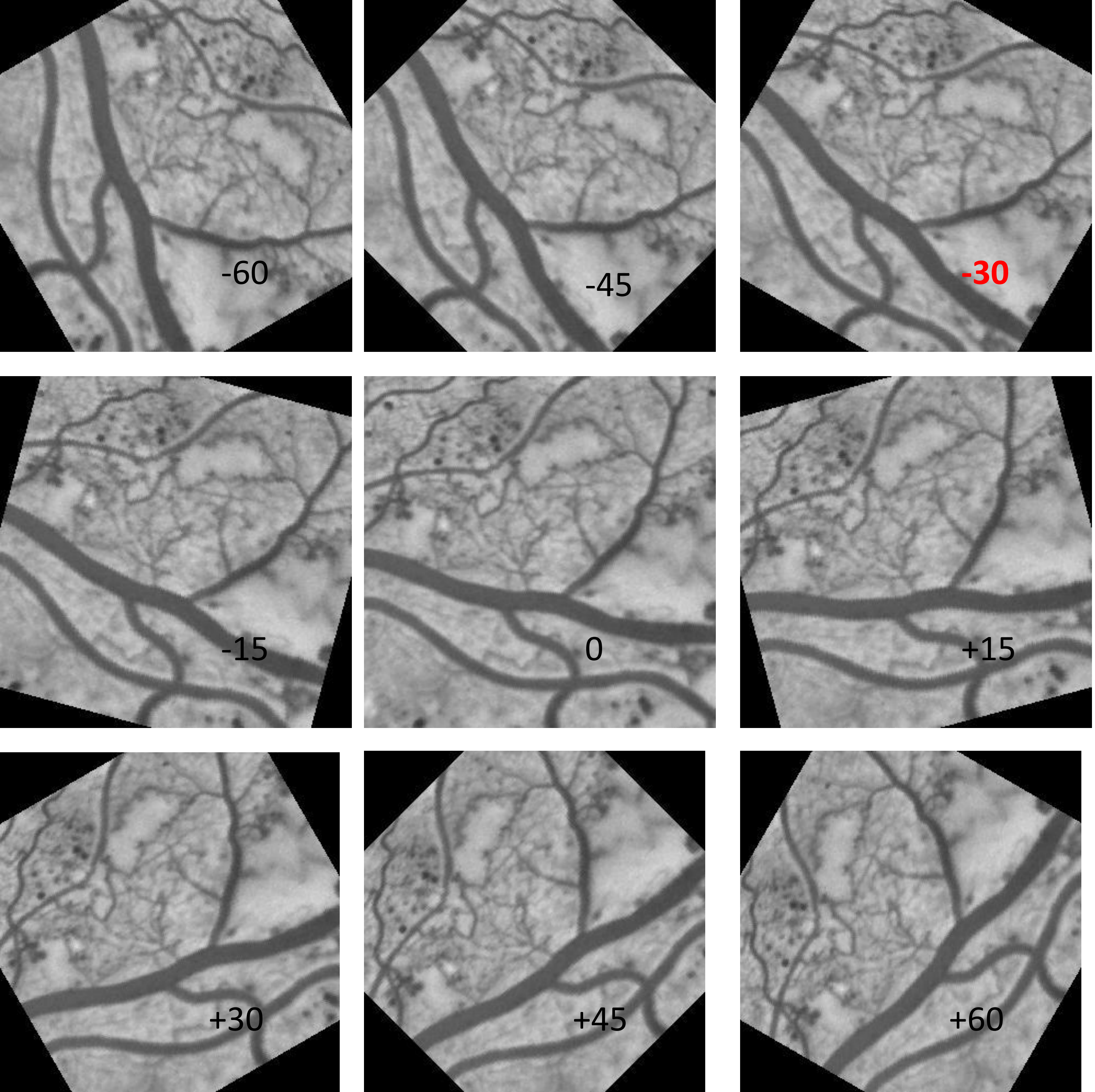


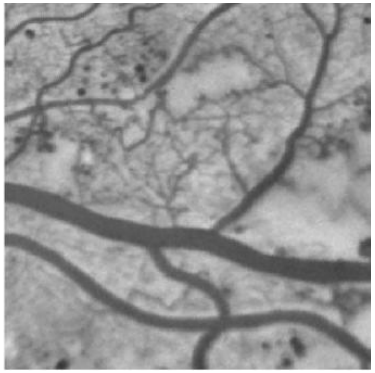


Region of overlap when the moving image is warped

Note: compute MSSD only over region of overlap.

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





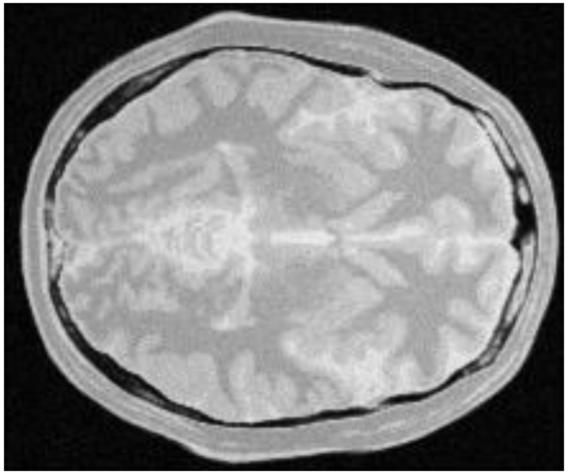
Alignment with mean squared error

- MSSD is one example of an “image similarity measure”.

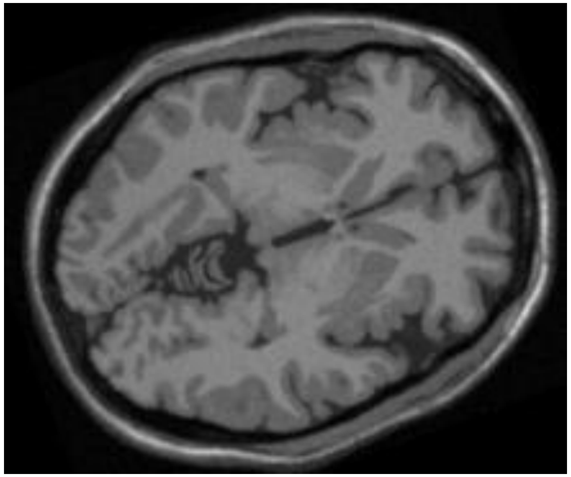
- MAJOR ASSUMPTION: Physically corresponding pixels have the same intensity, i.e. they are acquired by similar cameras and under the same lighting condition (this is often called as **mono-modal image registration**).

Image alignment: Intensity changes in images

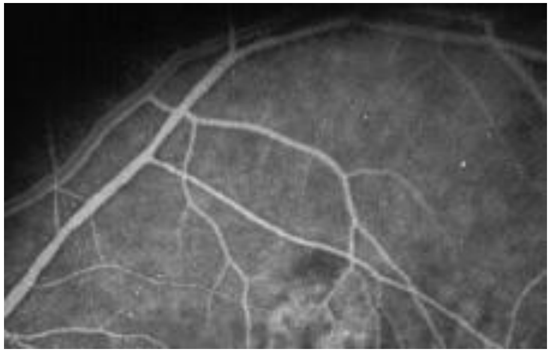
- Images acquired by different sensors (MR and CT, different types of MR, camera with and without flash, etc.)
- Changes in lighting condition
- This is called as **multimodal image registration**.

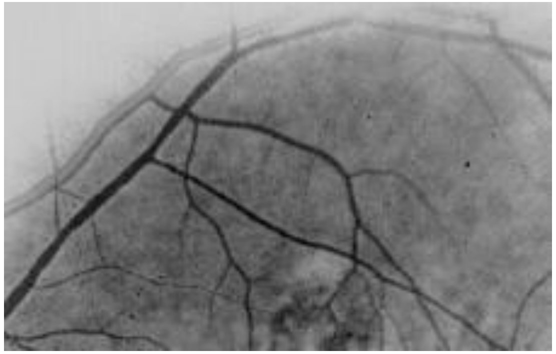


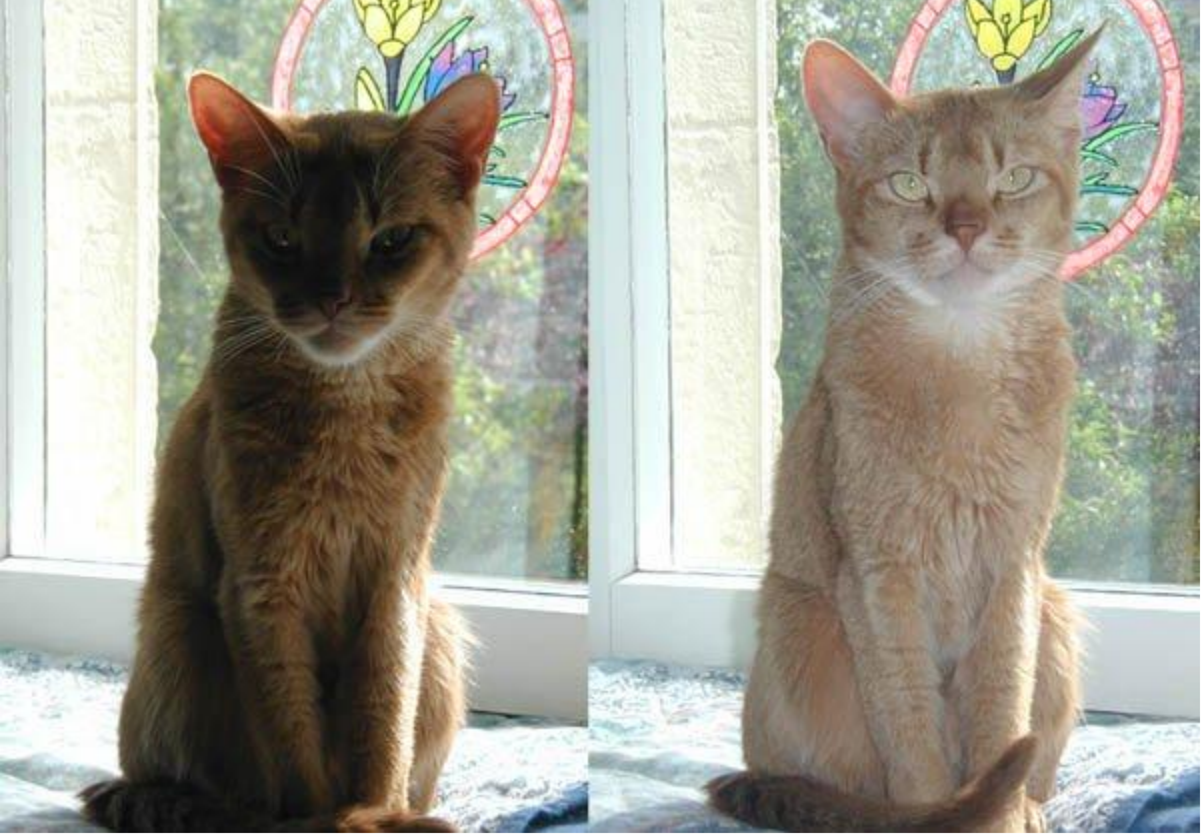
MR-PD



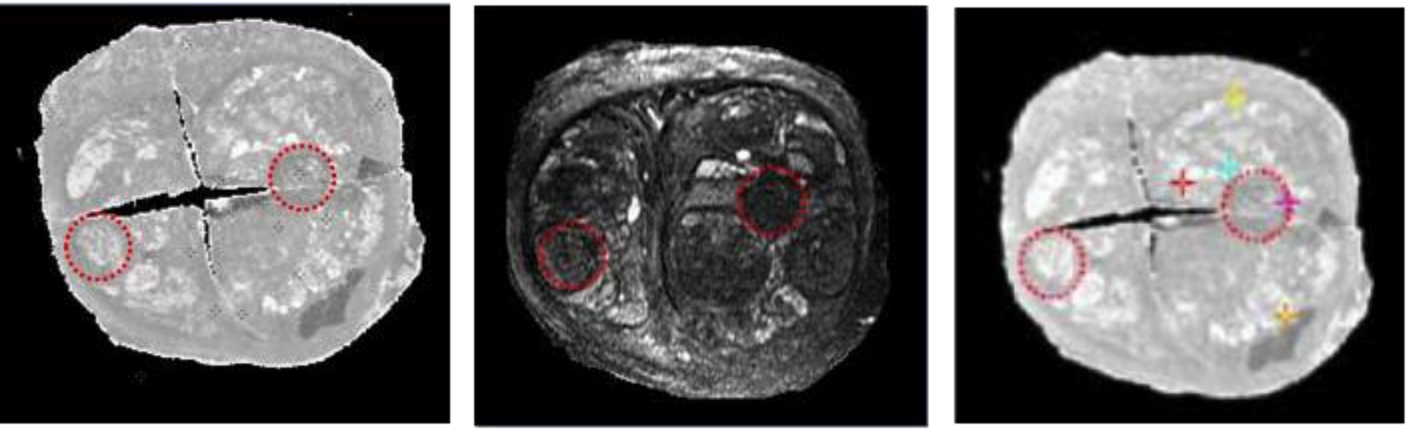
MR-T1







http://en.wikipedia.org/wiki/Flash_%28photography%29#mediaviewer/File:Fill_flash.jpg



a) histology b) MRI c) Hist_to_MRI

http://www.cbica.upenn.edu/sbia/software/dramms/tutorials/ProstateHist2MRI.html

Image alignment: Intensity changes in images

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

$$I_1(x_1, y_1) = g(I_2(x_2, y_2)), \forall (x_1, y_1), (x_2, y_2) \in \Omega$$

Physically corresponding points

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

Image alignment: Intensity changes in images

- What if the relationship exists in the following linear form, but we knew it only partially?

$$I_1(x_1, y_1) = aI_2(x_2, y_2) + b, \forall (x_1, y_1), (x_2, y_2) \in \Omega$$

Physically corresponding points; *a* and *b* are unknown, but we know the relationship is linear

$$NCC = \left| \frac{\sum_{(x,y)\in\Omega} (I_1(x,y) - \bar{I}_1)(I_2(x,y) - \bar{I}_2)}{\sqrt{\sum_{(x,y)\in\Omega} (I_1(x,y) - \bar{I}_1)^2 \sum_{(x,y)\in\Omega} (I_2(x,y) - \bar{I}_2)^2}} \right|$$

$\bar{I}_1, \bar{I}_2$: average value of images I_1, I_2

Normalized cross-correlation, also called correlation-coefficient – observe its relation to a noramlized vector dot product.

We are taking the absolute value here, to take care of cases where one image has positive values and the other has negative values

Image alignment: Intensity changes in images

$$NCC = \left| \frac{\sum_{(x,y)\in\Omega} (I_1(x,y) - \bar{I}_1)(I_2(x,y) - \bar{I}_2)}{\sqrt{\sum_{(x,y)\in\Omega} (I_1(x,y) - \bar{I}_1)^2 \sum_{(x,y)\in\Omega} (I_2(x,y) - \bar{I}_2)^2}} \right|$$

$\bar{I}_1, \bar{I}_2$: average value of images I_1, I_2

Normalized cross-correlation, also called correlation-coefficient

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

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

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 use image histograms!

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.

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.

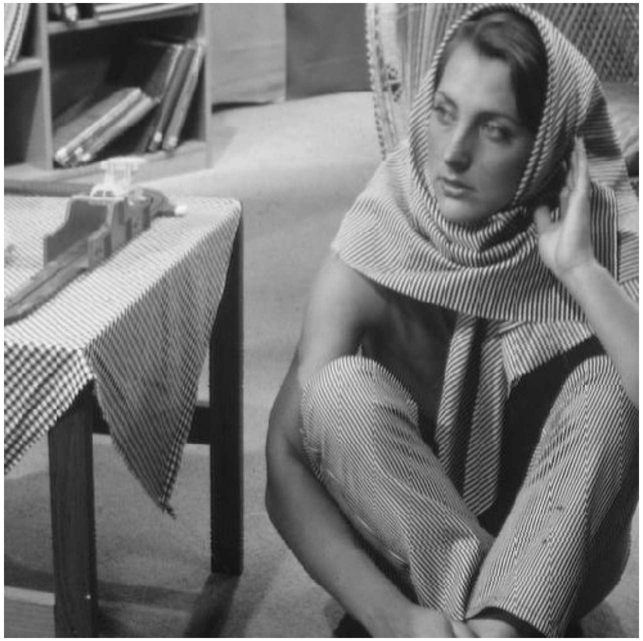
- $P(r_{k1},r_{k2})$ = number of pixels (x,y) such that $I_1(x,y)$ and $I_2(x,y)$ lie in bins r_{k1} and r_{k2} respectively, divided by *HW*.

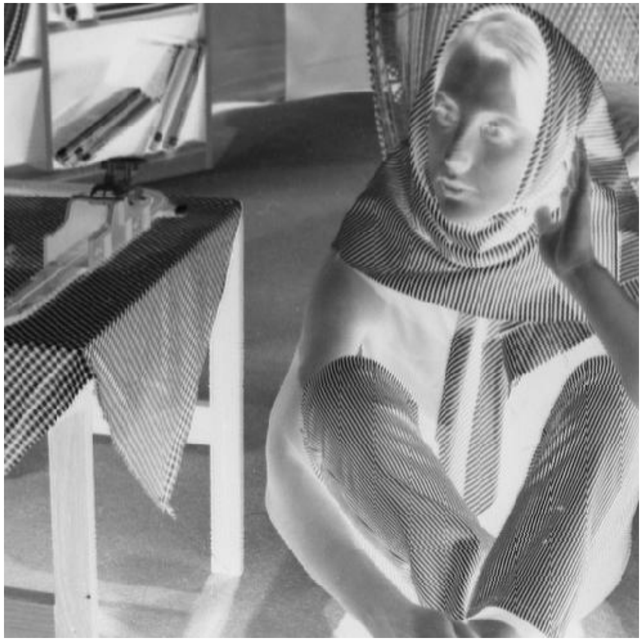
4	4	1
3	3	3
6	1	4

1	1	1
6	6	6
3	4	2

Consider two 3 x 3 images above for example sake.
Their joint histogram is as follows:
P(4,1) = 2/9
P(1,1) = 1/9
P(3,6) = 3/9
P(6,3) = 1/9
P(1,4) = 1/9
P(4,2) = 1/9

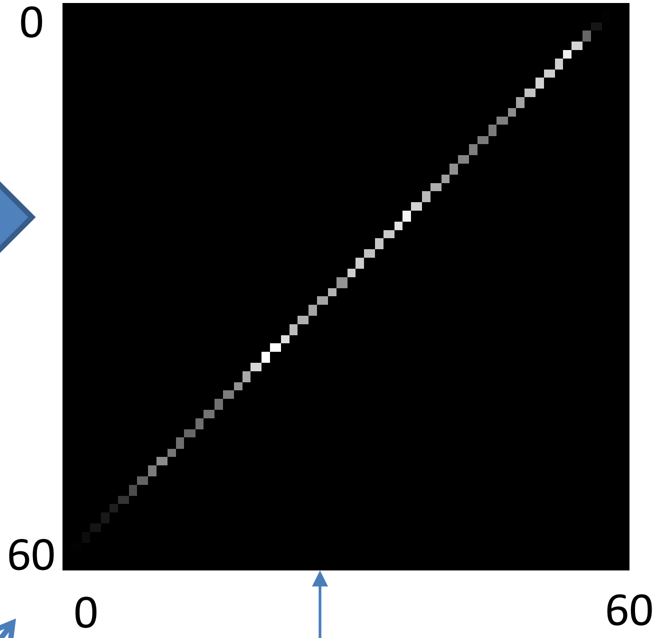
I_1





I_2

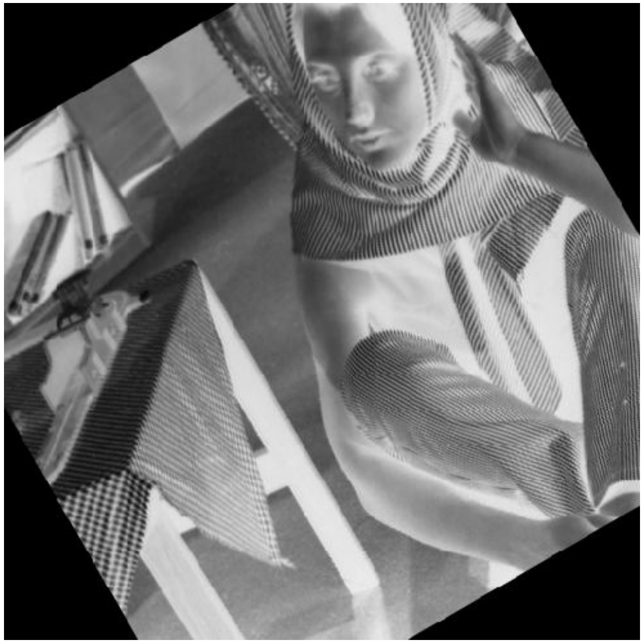
Values in I_2 from 0 to 60

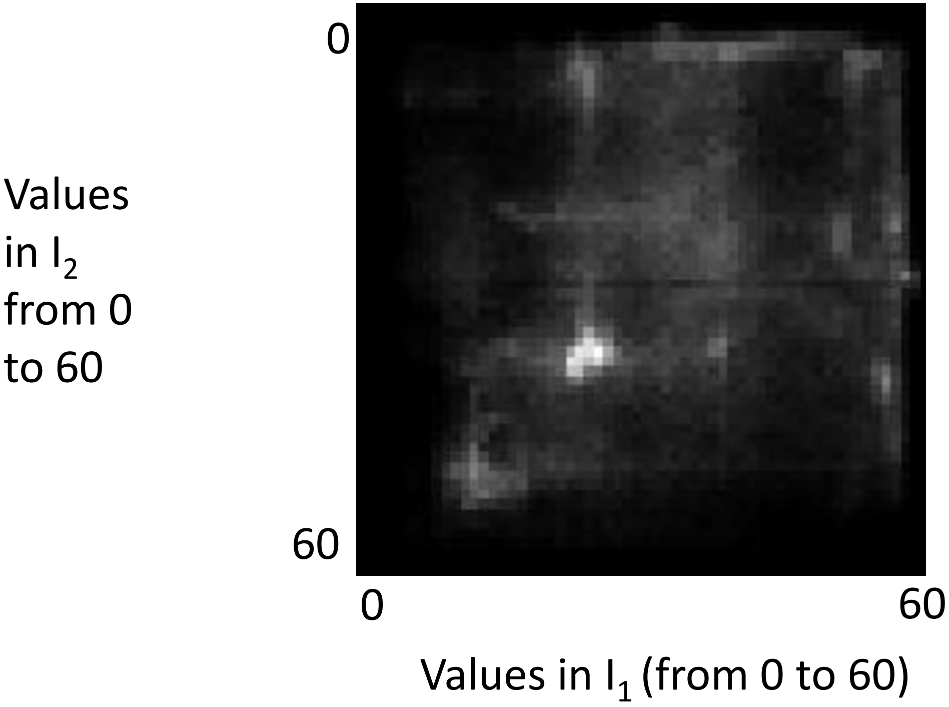


Values in I_1 (from 0 to 60)

Registered images: joint histogram plot looks "streamlined"

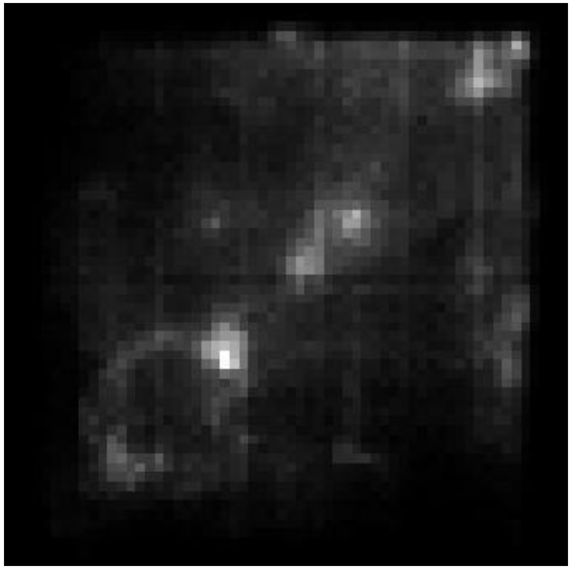
How was this plot generated? The joint histogram is plotted as a grayscale image. Brighter points in this image indicate larger probabilities and darker points indicate lower probabilities.





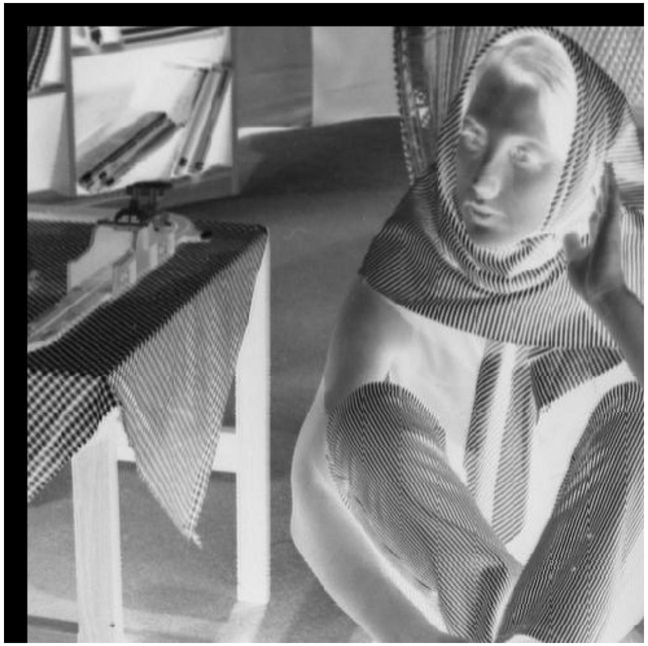
Misaligned images: joint histogram plot looks “dispersed”

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:

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

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

- 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 $x_2 \neq x_1$.

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:

-Uniform distribution on *X* and *Y*: entropy value log(|DX||DY|)

- Minimum entropy:

-Kronecker delta, i.e. P(X=x_1,Y=y_1) = 1 for some x_1,y_1 and $P(X$=x_2,Y=$y_2)$ = 0 for all $x_2 \neq x_1$ *or* $y_2 \neq y_1$.

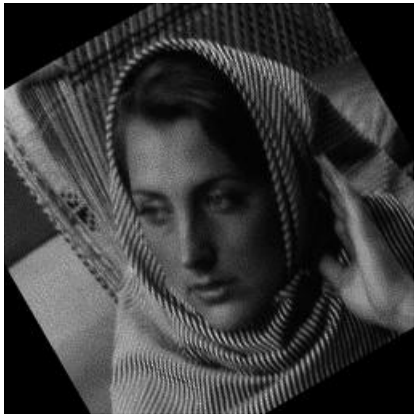
Joint entropy

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

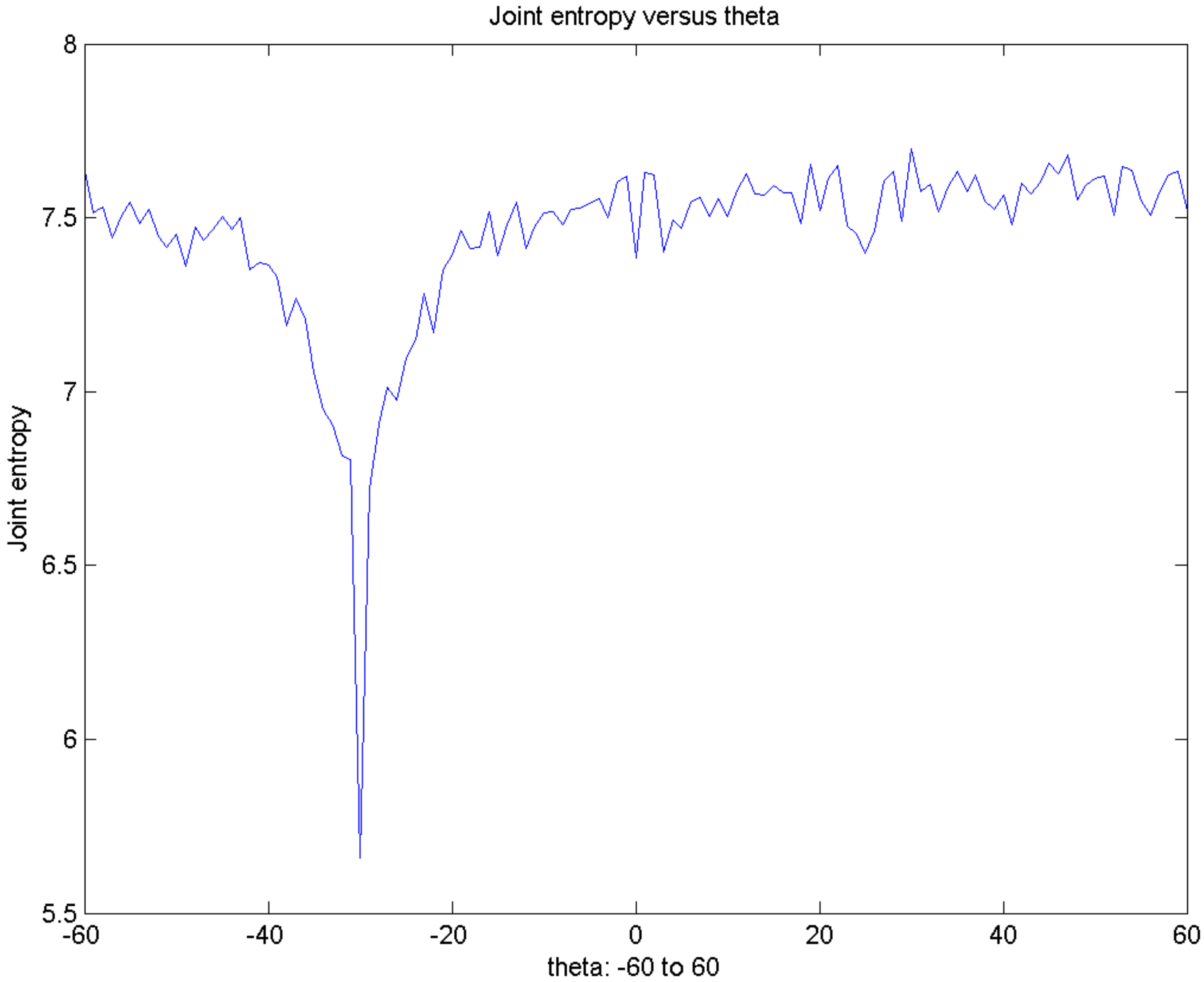
$$\mathbf{T}^* = \arg\min_{\mathbf{T}} H(I_2(\mathbf{v}), I_1(\mathbf{T}\mathbf{v}))$$

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

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.

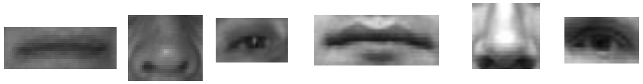
Image Alignment: Application Scenarios

Image Alignment: applications, related problems

- Template matching
- Image Panoramas
- Denoising image-bursts
- Collecting photos of paintings
- Face recognition
- 3D-2D image registration

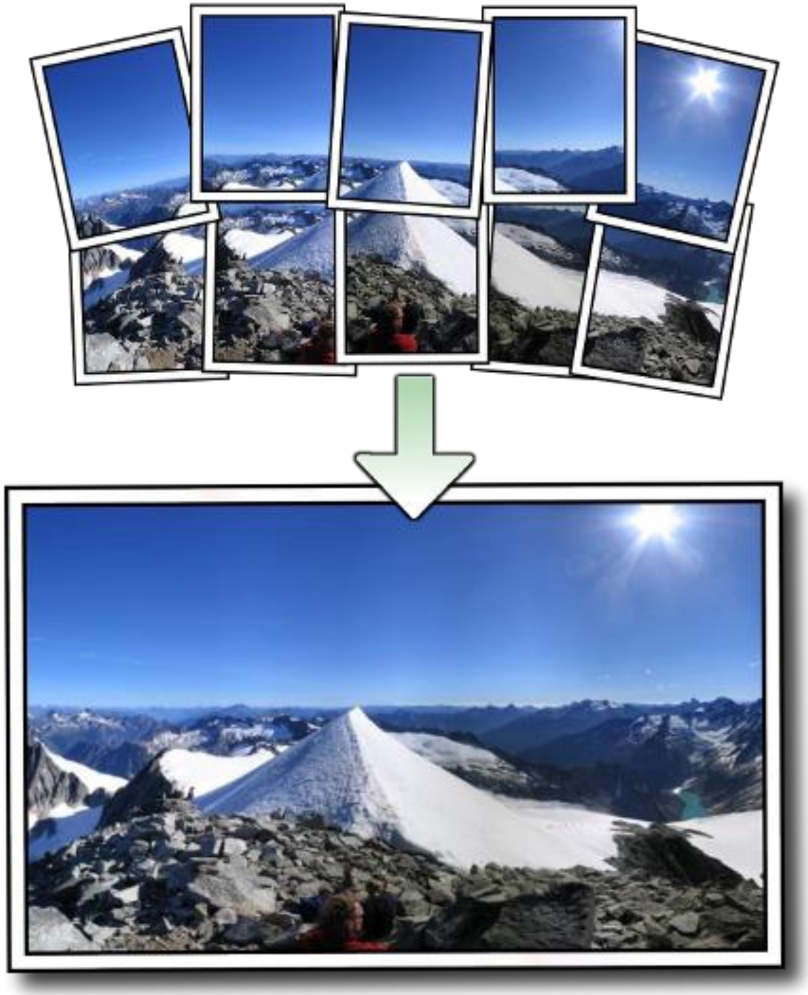
(1) Template Matching

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

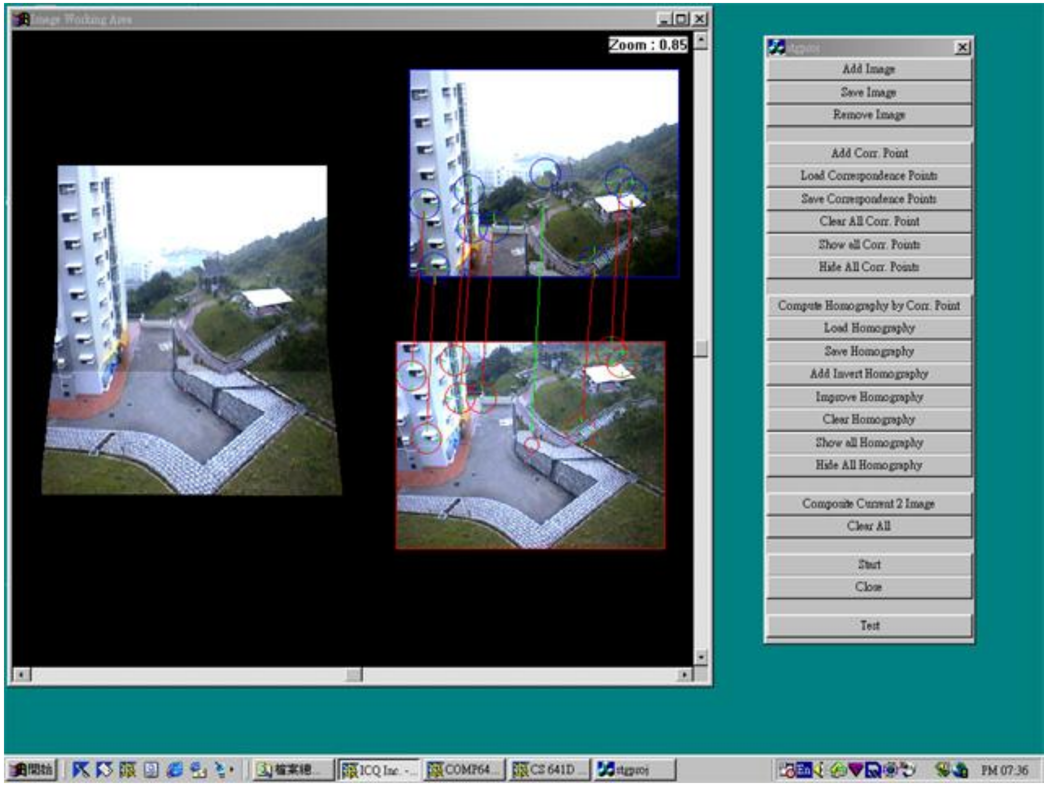


Templates

(2) Image Panoramas



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



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

- 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.

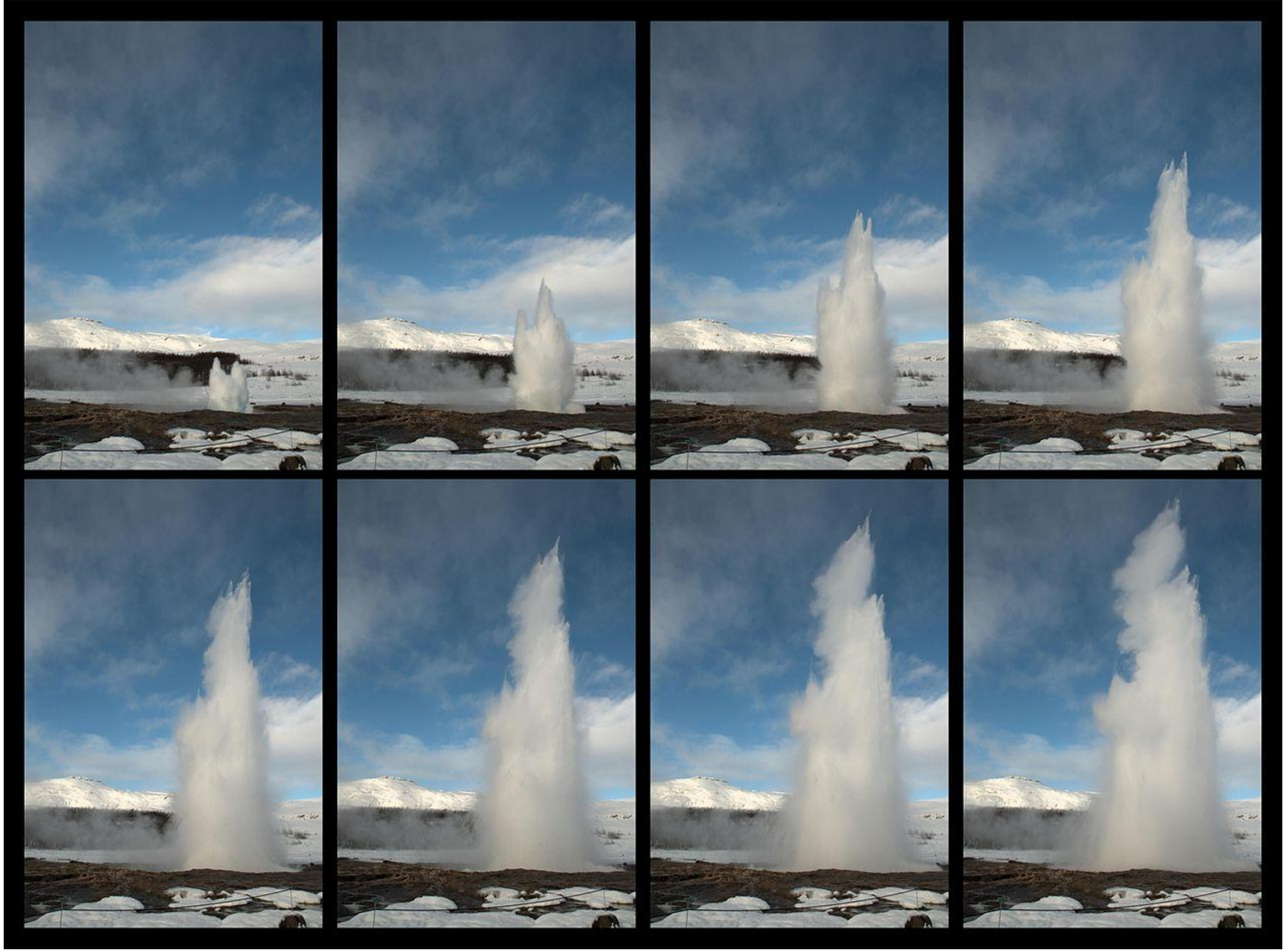
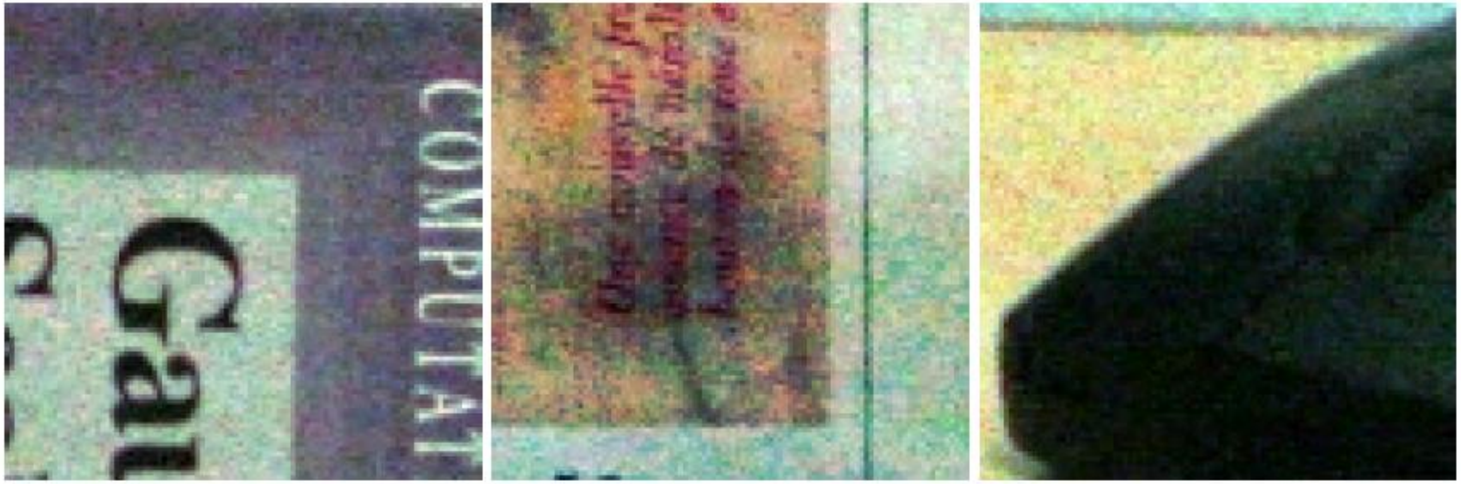
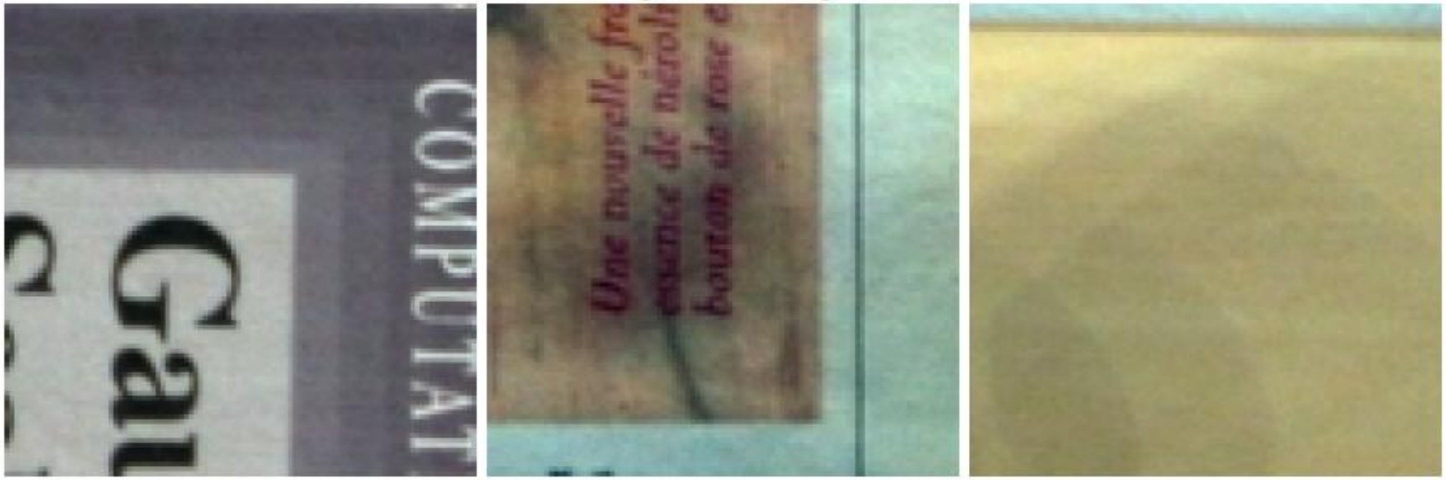


Image bursts are typically used for photographing fast moving objects (water fountains, sports, children). Here, however, we use it for photography under poor lighting conditions.

http://en.wikipedia.org/wiki/Burst_mode_%28photography%29 noisy data

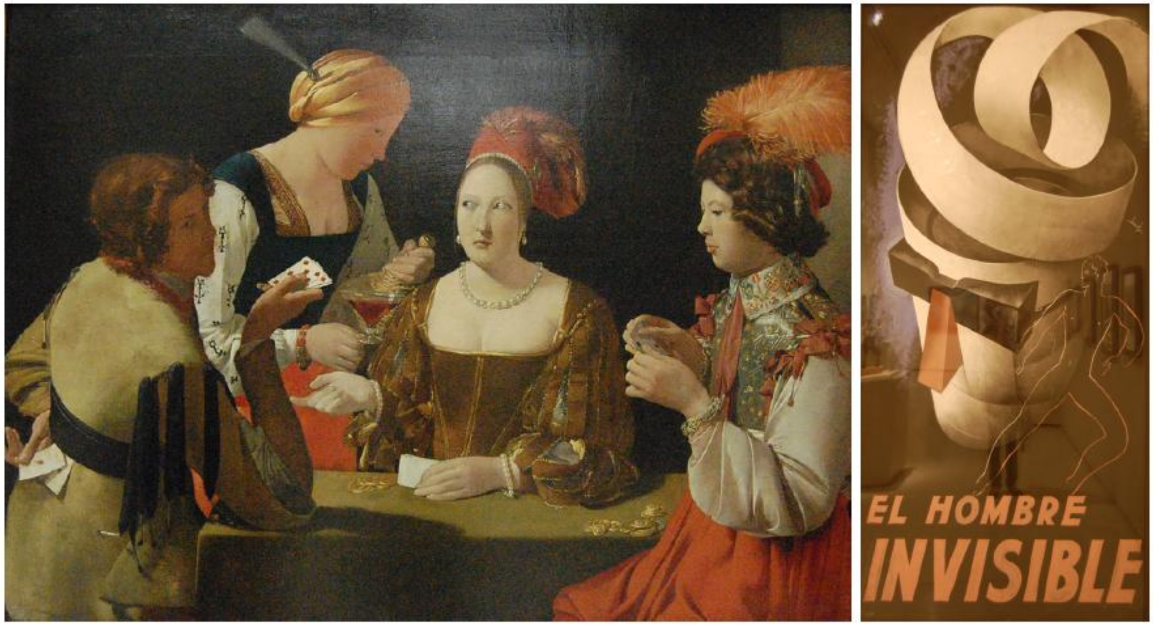


The average after registration



(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.



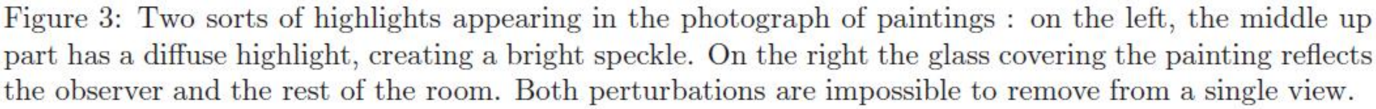


Figure 3: Two sorts of highlights appearing in the photograph of paintings : on the left, the middle up part has a diffuse highlight, creating a bright speckle. On the right the glass covering the painting reflects the observer and the rest of the room. Both perturbations are impossible to remove from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

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

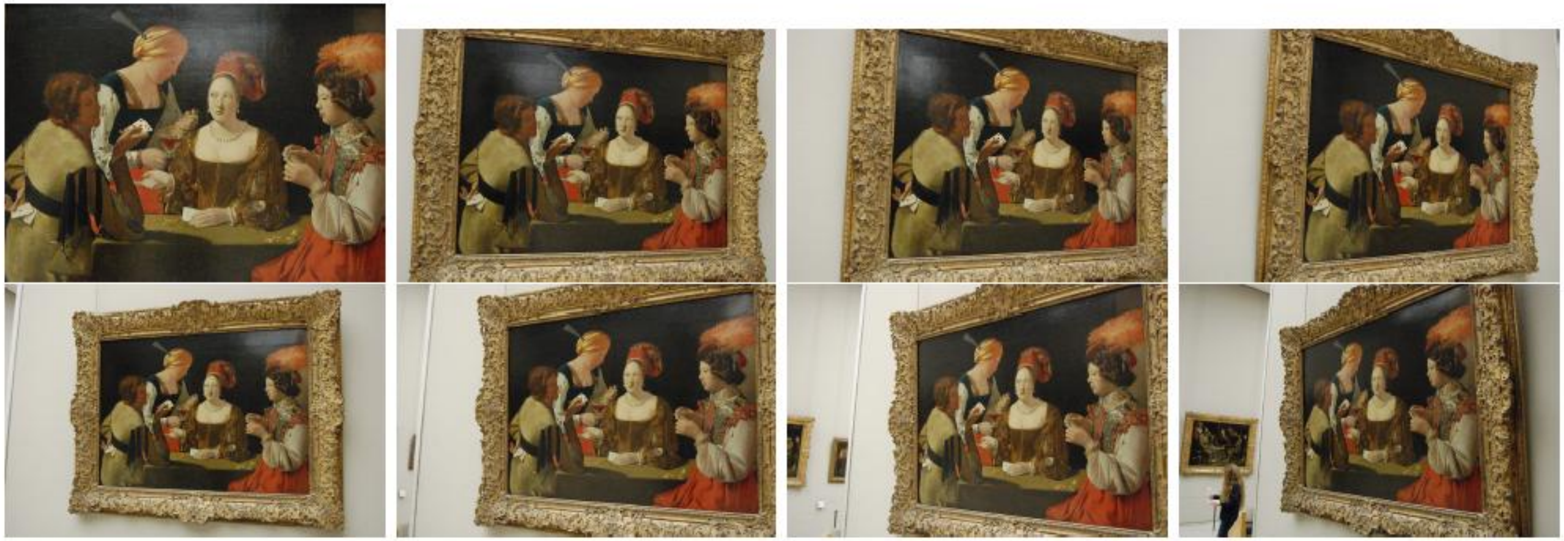


Figure 4: Example of burst detection: each image is taken from one of the eight bursts detected in the original set of 87 images. Each burst corresponds to a different point of view. Notice the moving highlights, covering large regions.

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).

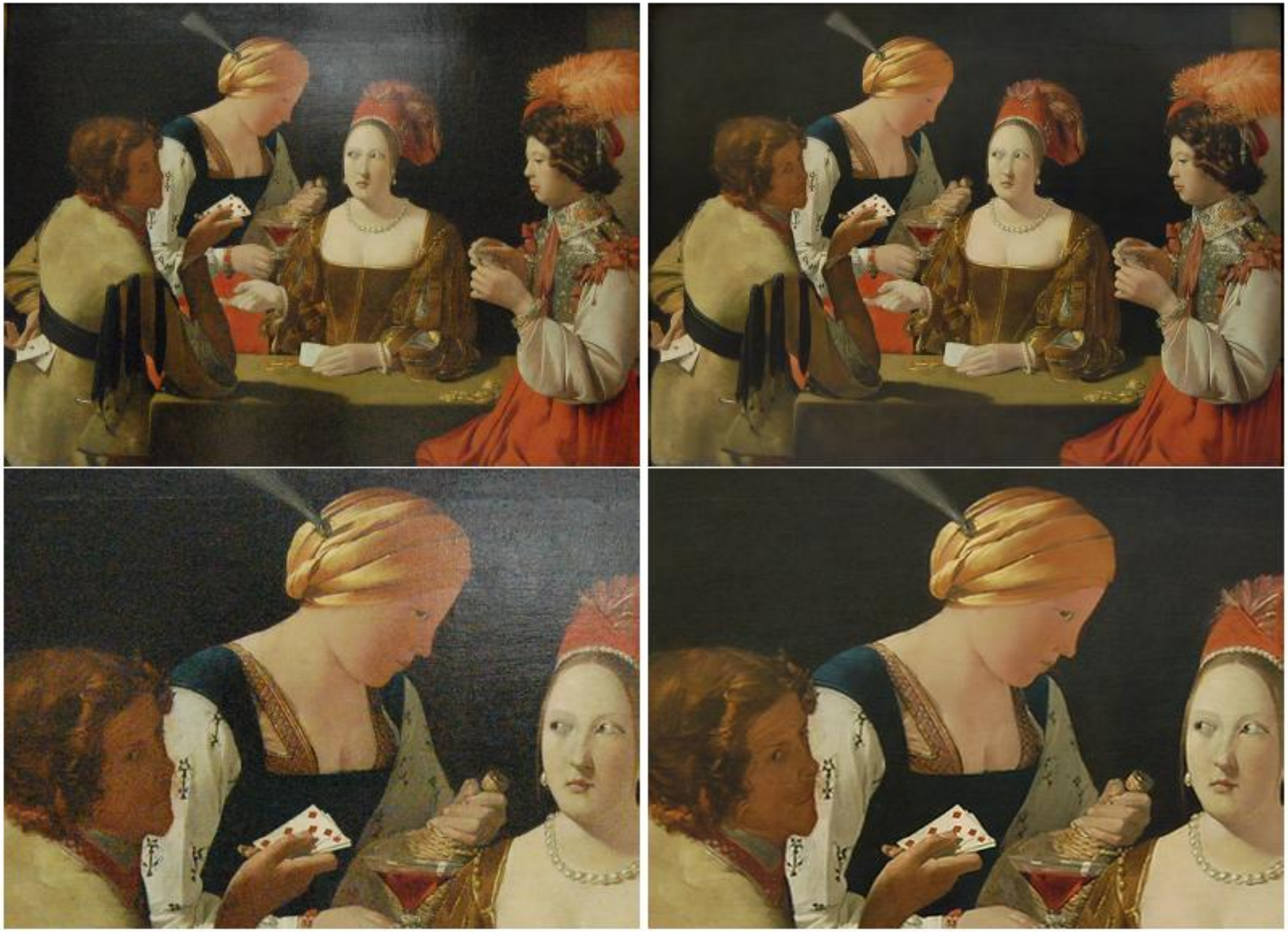


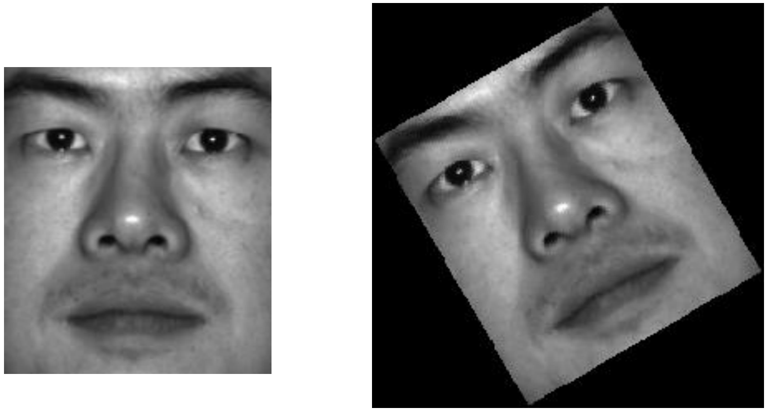
Figure 14: Latour example. Left: one of the input images. Right: result of median of gradients after denoising plus sharpening with two iterations. Second row: zoom views of the images in the first row.

(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?

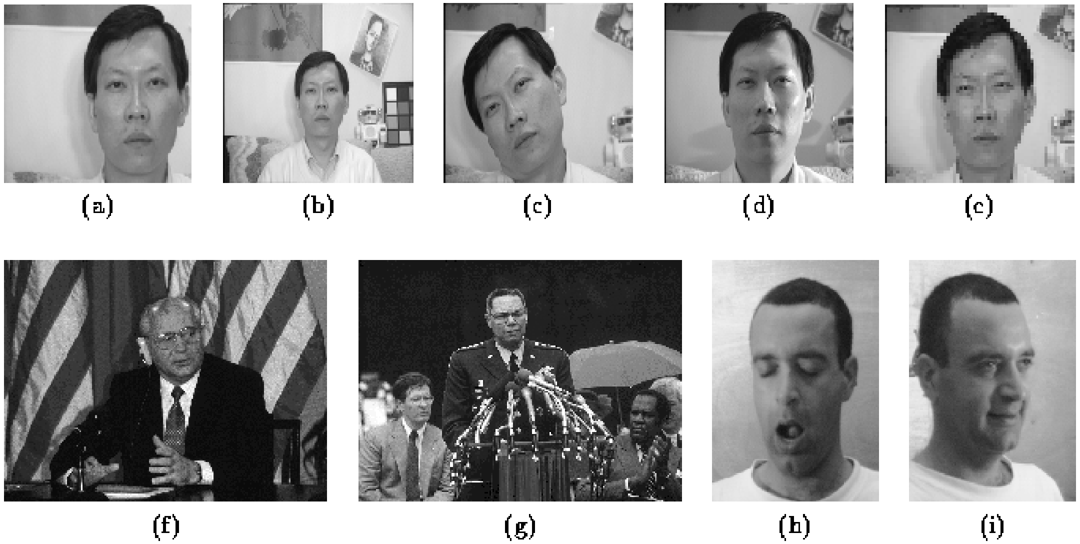
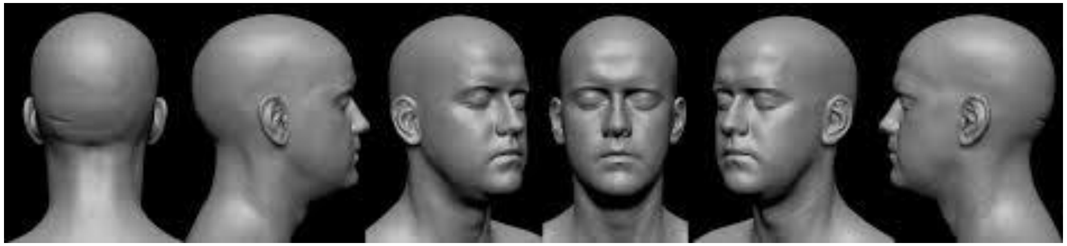


Figure 3.1: Variations in faces that require appropriate compensation. (a) Change in position. (b) Change in size or scale. (c) In-plane rotation of the face. (d) Shading and illumination effects. (e) Variation in image quality or resolution. (f) Clutter in the image background. (g) Multiple faces in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

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

(6) 3D-2D registration

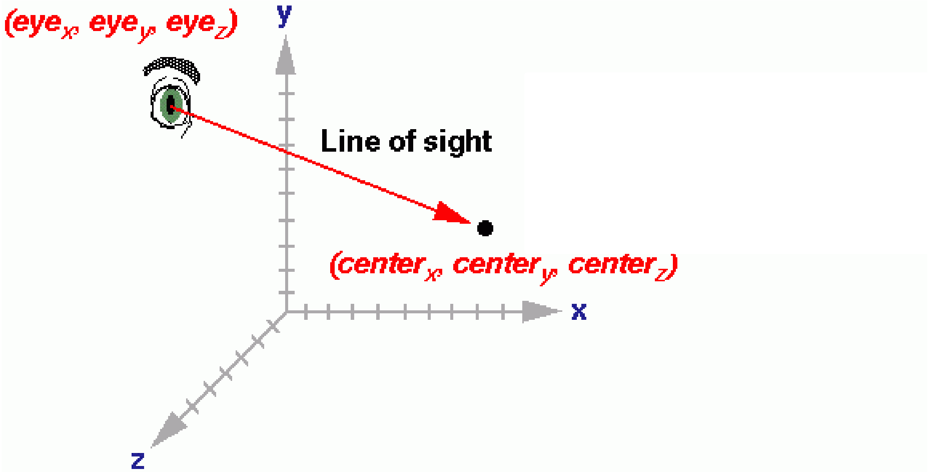
- This motivates the use of 3D face scans or 3D models – which can be acquired by 3D cameras such as stereo-cameras or structured light sensors.
- A 3D face scan consists of a set of vertices in 3D and a set of polygons (in 3D) linking those vertices.



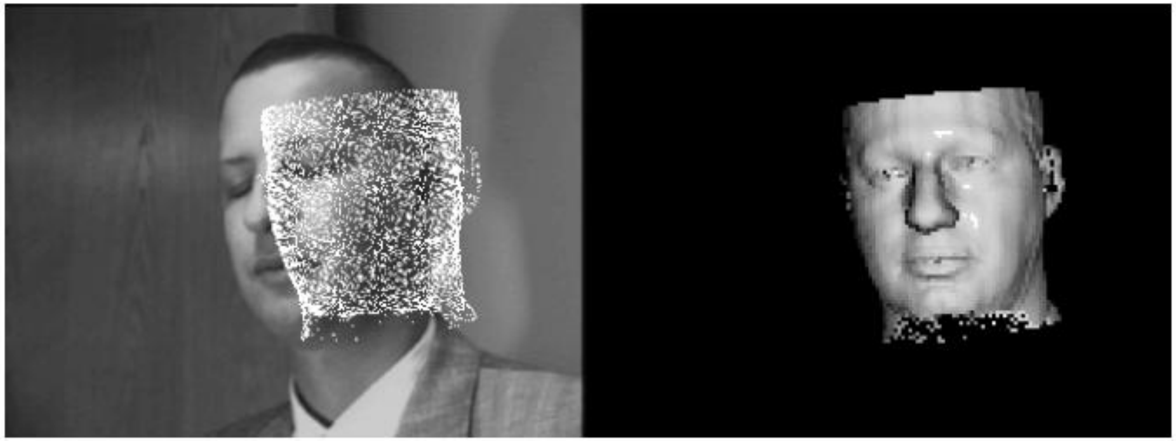
http://www.jonasavrin.com/2011/01/15/free-3d-ir-head-scan-release-smart-hdr-ibl-vray-2-0/

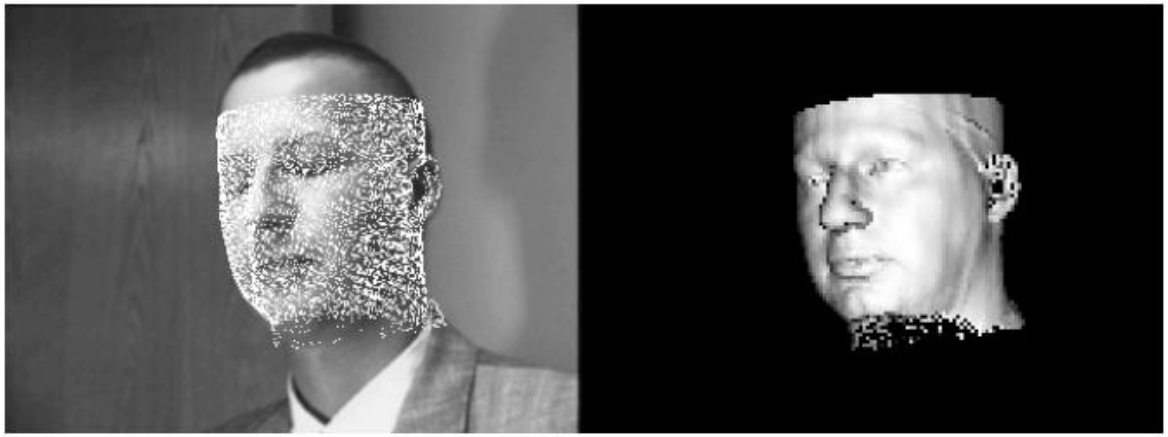
(6) 3D-2D registration

- 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?
- Six: direction of viewing (2), distance of viewpoint along the viewing direction (1), direction measured to which offset point in 3D (another 3).



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





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..

- Affine motion model
- 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 learn

- 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 non-parametric (non-rigid)?
- Do the images have equal intensity values at physically corresponding points? (Unimodal or multimodal). This decides the objective function to be optimized.