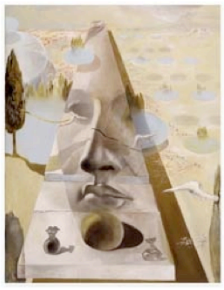
Lecture 6
Stereo Systems
Multi-view geometry

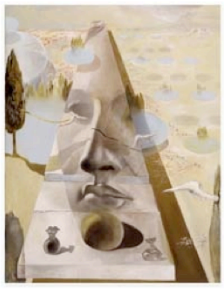


Professor Silvio Savarese
Computational Vision and Geometry Lab

Lecture 6

Stereo Systems

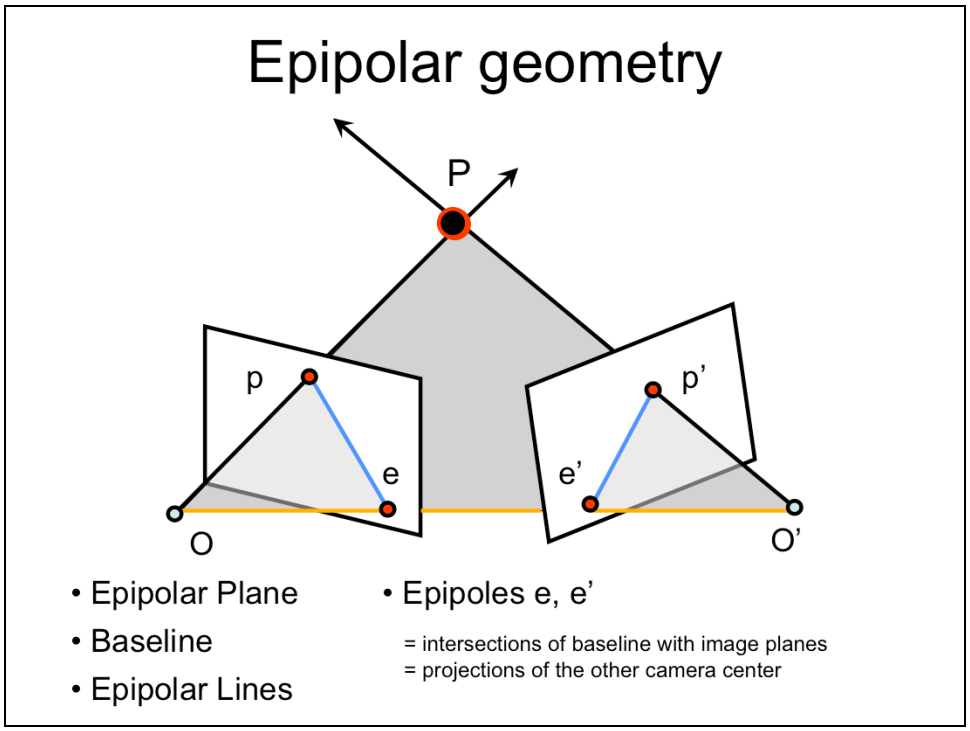
Multi-view geometry



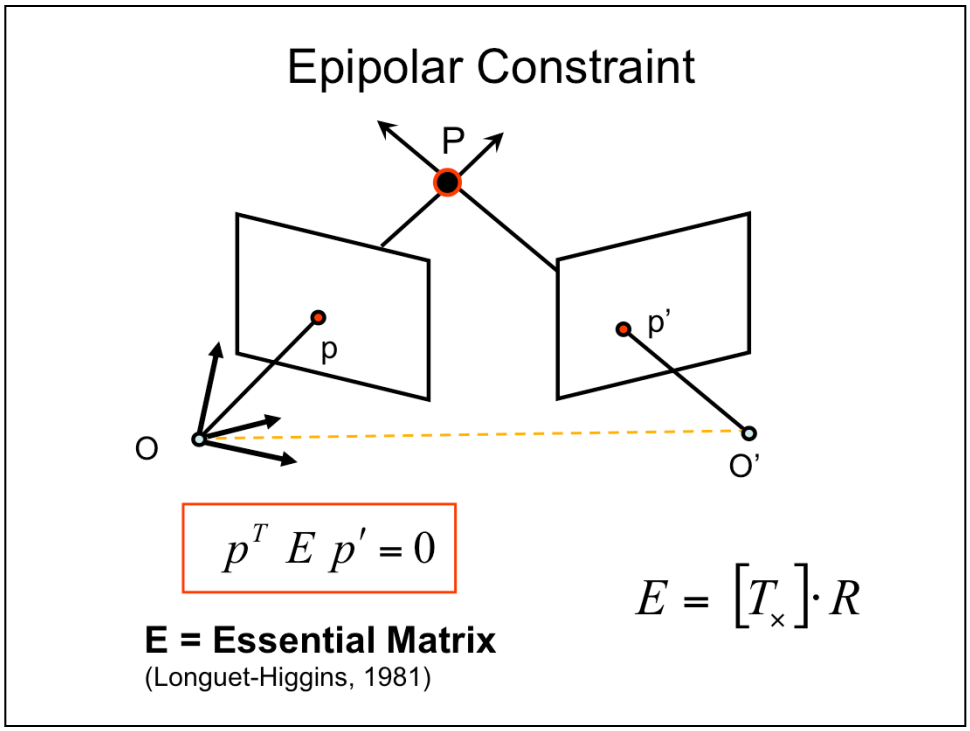
- Stereo systems
  - Rectification
  - Correspondence problem
- Multi-view geometry
  - The SFM problem
  - Affine SFM

Reading:
[AZ] Chapter: 9 "Epip. Geom. and the Fundam. Matrix Transf."
[AZ] Chapter: 18 "N view computational methods"
[FP] Chapters: 7 "Stereopsis"
[FP] Chapters: 8 "Structure from Motion"

In this lecture we will first discuss the correspondence problem in rectified stereo systems. Further, we will discuss two methods for recovering geometry from images. In stereo systems we observe objects from two viewpoints, allowing depth to be determined in much the same way as the human vision system. Structure from motion extends this concept to multiple views, which could be discrete or from a video sequence.

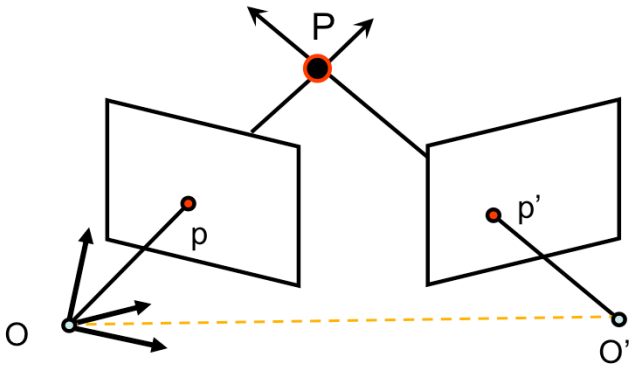


We will begin by reviewing the main concepts of epipolar geometry which we already introduced in lecture 5. Epipolar geometry describes the relationship between images recorded by pairs of cameras. The line between the camera origins is known as the baseline, which intersects the camera planes at the epipoles, e and e'. If a point p is observed in image 1, the corresponding point p' must lie along the corresponding epipolar line in image 2. This line is defined by the intersection of the epipolar plane – which intersects both camera origins and p – and the image plane for camera 2. Therefore, all epipolar lines intersect the image's epipole.



Points in one image correspond to epipolar lines in another – a constraint known as the epipolar constraint. We can concisely describe this constraint using the essential matrix E, under the assumption that the cameras a canonical (i.e., corresponding camera matrices are identity matrices). Points p and p' in camera coordinates that satisfy this equation must correspond to the same world point P. The essential matrix can be computed from the translation and rotation between the cameras, where $[T_x]$ is the matrix representation of the cross product with T.

Epipolar Constraint



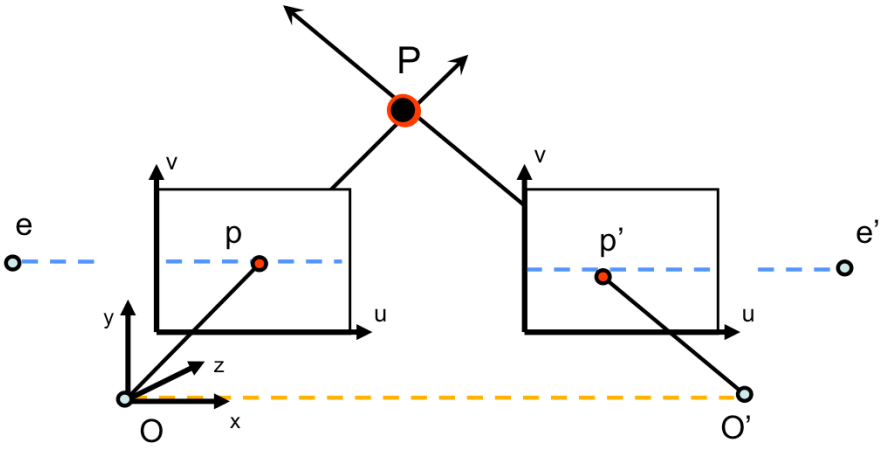
$$p^T F p' = 0$$

$$F = K^{-T} \cdot [T_\times] \cdot R \, K'^{-1}$$

F = Fundamental Matrix
(Faugeras and Luong, 1992)

The fundamental matrix expresses the same constraint, but when two cameras are characterized by general (and unknown) camera matrices K and K', respectively. Therefore the derivation of F includes the camera matrices for both cameras.

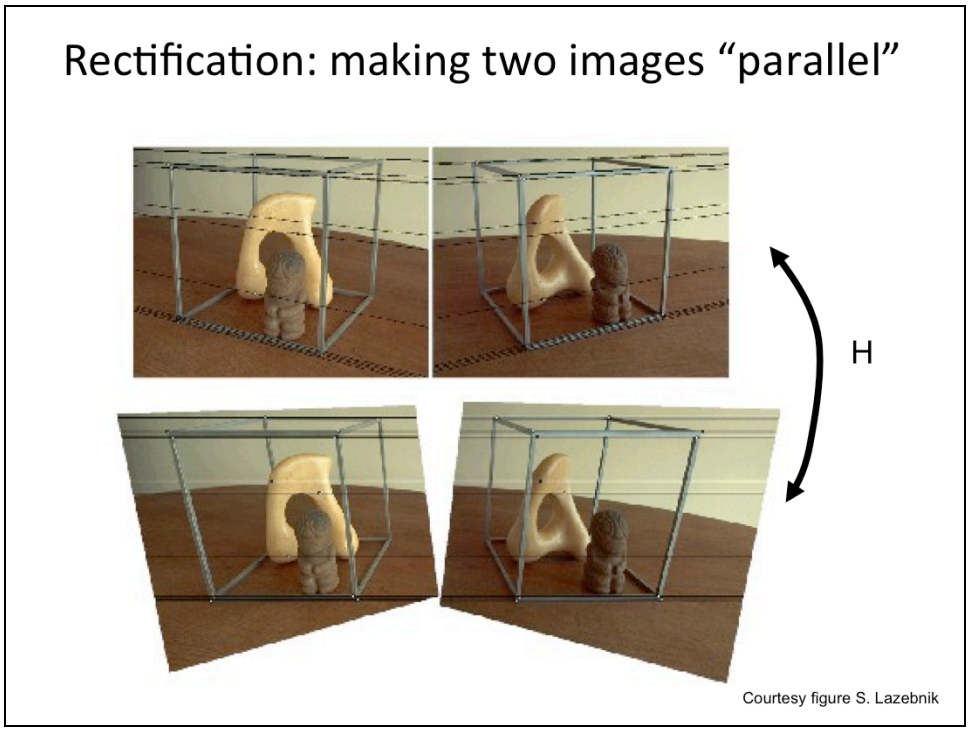
Example: Parallel image planes



- Epipolar lines are horizontal
- Epipoles go to infinity
- v-coordinates are equal

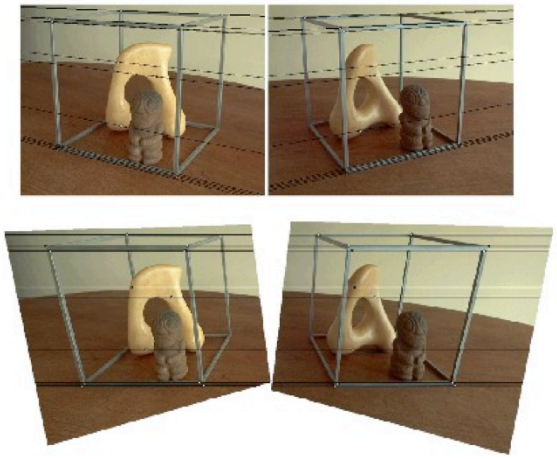
$$p = \begin{bmatrix} p_u \\ p_v \\ 1 \end{bmatrix} \quad p' = \begin{bmatrix} p'_u \\ p_v \\ 1 \end{bmatrix}$$

If the image planes were parallel, then the epipolar lines are necessarily horizontal, and the epipoles lie at infinity. Therefore, the point corresponding to p – that is, p' – must lie along the horizontal line with the same v-coordinate.



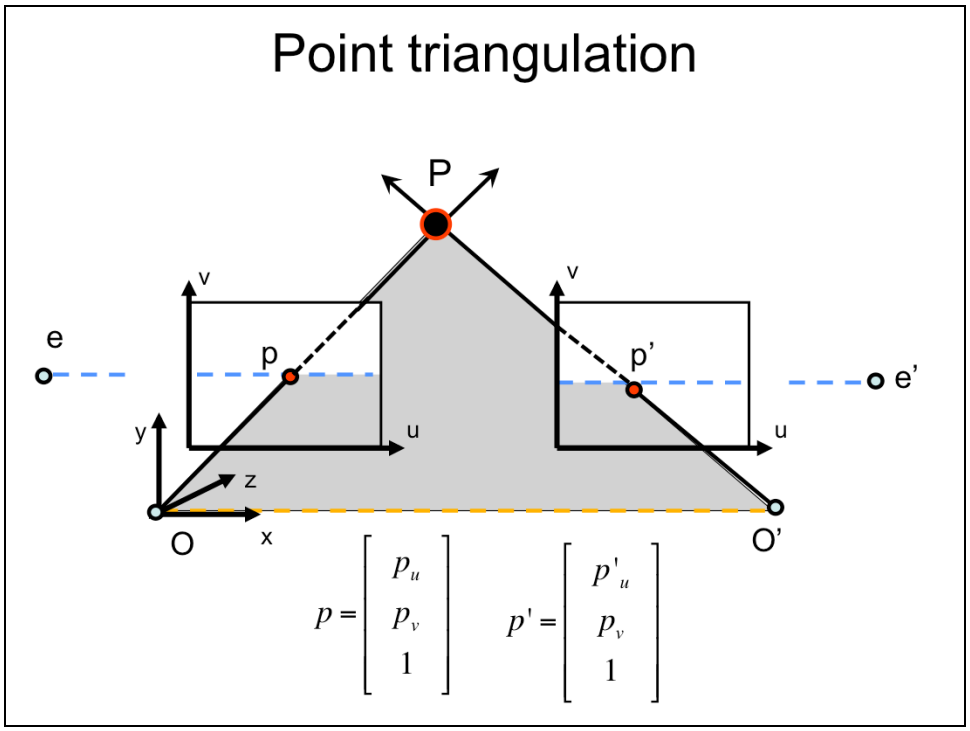
Rectification is the process of warping images to simulate a parallel image plane. This is done by applying am appropriate 2D projective transformation H (which is also called homographic transformation) to each image. Notice that after rectification the epipolar lines become horizontal and parallel.

Why are parallel images useful?



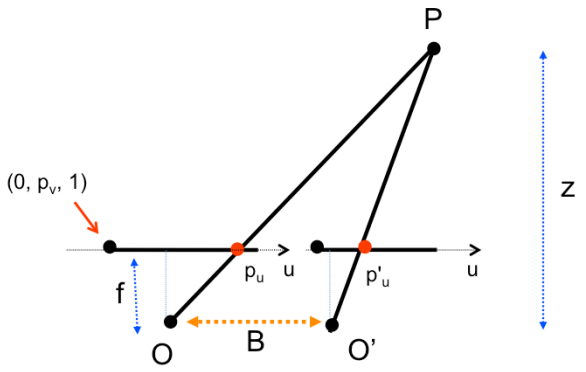
- Makes triangulation easy
- Makes the correspondence problem easier

Why is having parallel images useful? For several reasons— for instance, it makes triangulation easier. Or it helps simplify the problem of finding correspondences across views. Let's start with the first one.



Triangulation is the process of recovering the world coordinates of a point from a pair of corresponding points in stereo images. Because images are parallel, corresponding points will lie on the same v-coordinate for both images (i..e, p and p' share the same v-coordinate value). This allows to simplify our construction and consider a 2D version of the above scheme which is defined by the epipolar plane associated to p, p' and P (gray plane in the figure)

Computing depth



disparity = $p_u - p'_u \propto \frac{B \cdot f}{z}$ [Eq. 1]

Note: Disparity is inversely proportional to depth

Here we see the same geometry as before but mapped over the epipolar plane. Images are shown here as image lines (along u), since we the v-coordinate is the same for both images.

Let's now define the **disparity** as the pixel displacement p_u-p'_u between corresponding points. It's easy to show (by comparing similar triangles) that the **disparity** p_u-p'_u is inversely proportional to the distance z of the point P from the image plane and directly proportional to B and f as described by Eq. 1. Thus, Eq.1 can be used to compute the depth z from the baseline distance B, focal length f and disparity p_u-p'_u for every set of corresponding locations p_u and p'_u in the image.

<experiment in class: draw a line on the board. Students are asked to place one finger 1 foot away from the nose, then to close one eye and then the other. The difference of distances from the line on the board is the disparity. Repeat the same experiment with a finger 2 feet away>

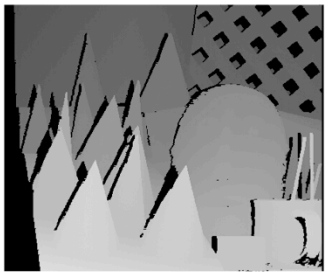
Disparity maps

http://vision.middlebury.edu/stereo/

Stereo pair

$$p_u - p'_u \propto \frac{B \cdot f}{z}$$

[Eq. 1]



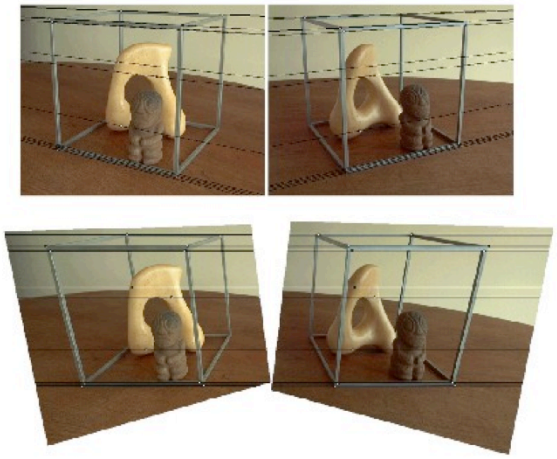
Disparity map / depth map

Suppose that for each point in image 1 we know the corresponding point on image 1 (we will discuss later how to find those). Given this dense set of pair of correspondences p_u and p'_u in the image, we can compute the disparity (or depth) for each of these pairs using Eq. 1 and obtain a disparity map (see image). The brighter the color, the larger is the disparity (and the smaller is the depth).

Do you know why certain regions in the depth map are in black?

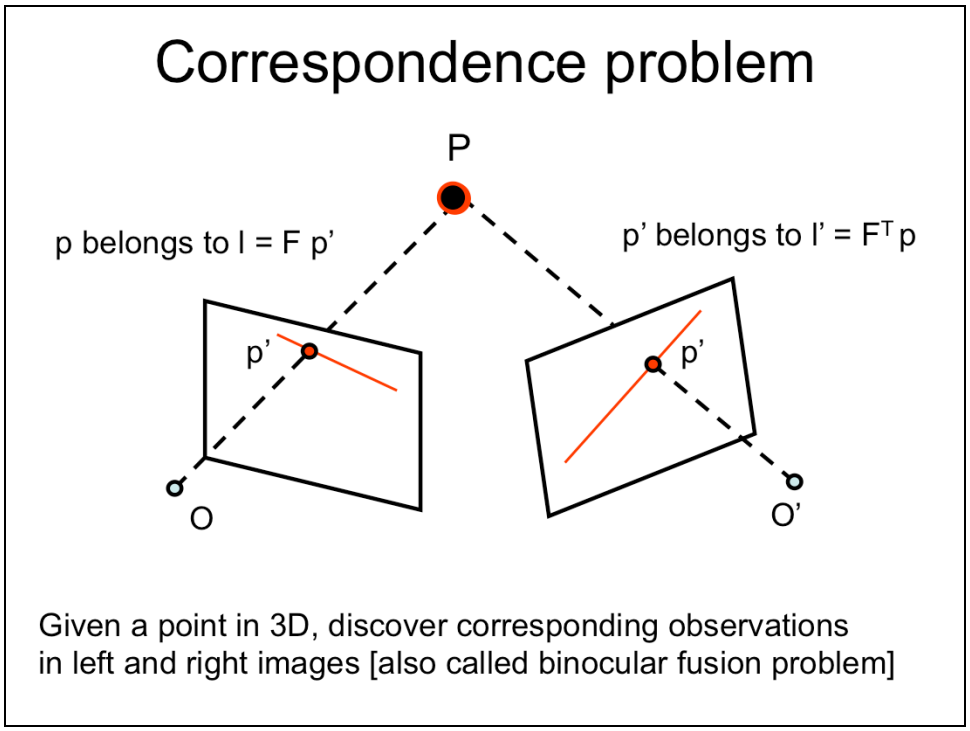
Not all the pixels in one image have a corresponding pixel in the second image. This is because of the presence of occlusions: the same physical point in 3D can be visible from one camera, but not visible (occluded) from the other camera (for example, see the region highlighted by the blue arrow). Occlusions are shown in black in the bottom image.

Why are parallel images useful?



- Makes triangulation easy
- Makes the correspondence problem easier

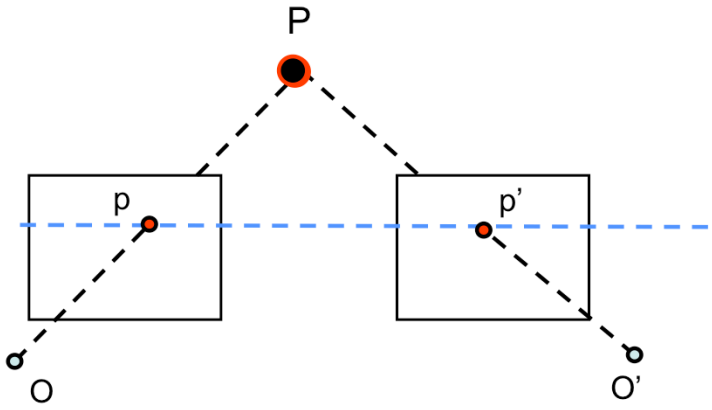
Let's discuss now how to solve the correspondence problem for rectified images.



The correspondence problem (also called binocular fusion problem for stereo systems) seeks to find corresponding observations (say p and p') in both images given a point in 3D (say P). With generic skewed image planes this requires a search over the 2D image space.

If we know the fundamental matrix F associated to the image pair (or we can compute it from at a given number of correspondences), we can search for corresponding points along the epipolar lines (shown in orange).

Correspondence problem



When images are rectified, this problem is much easier!

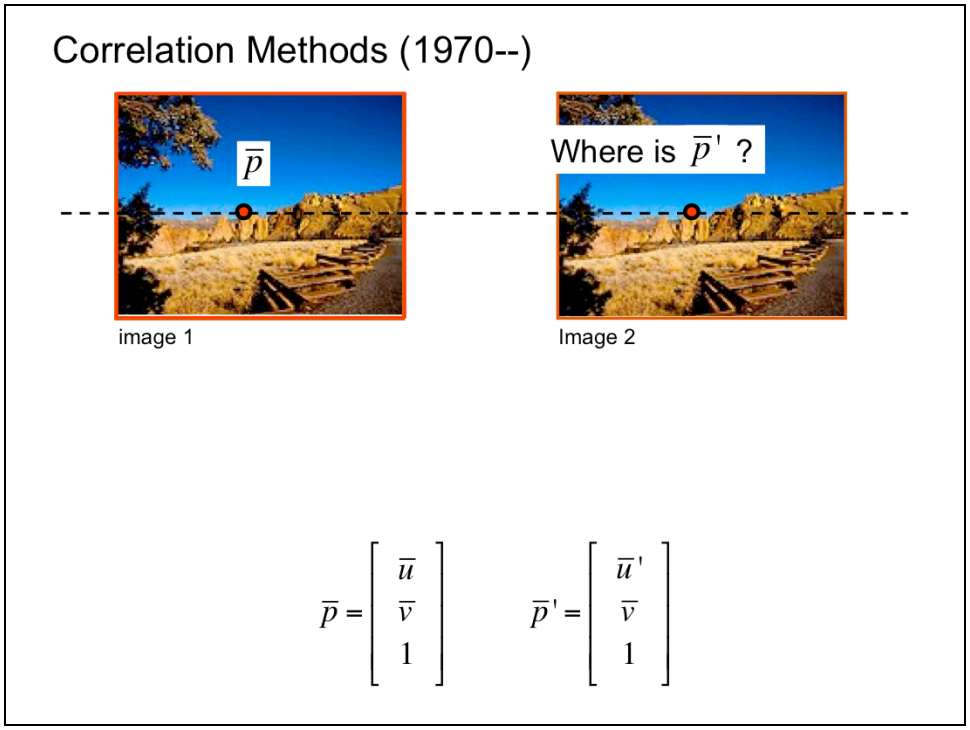
In rectified images corresponding points must lie along horizontal epipolar lines, so solving the correspondence problem is much easier!

Correspondence problem

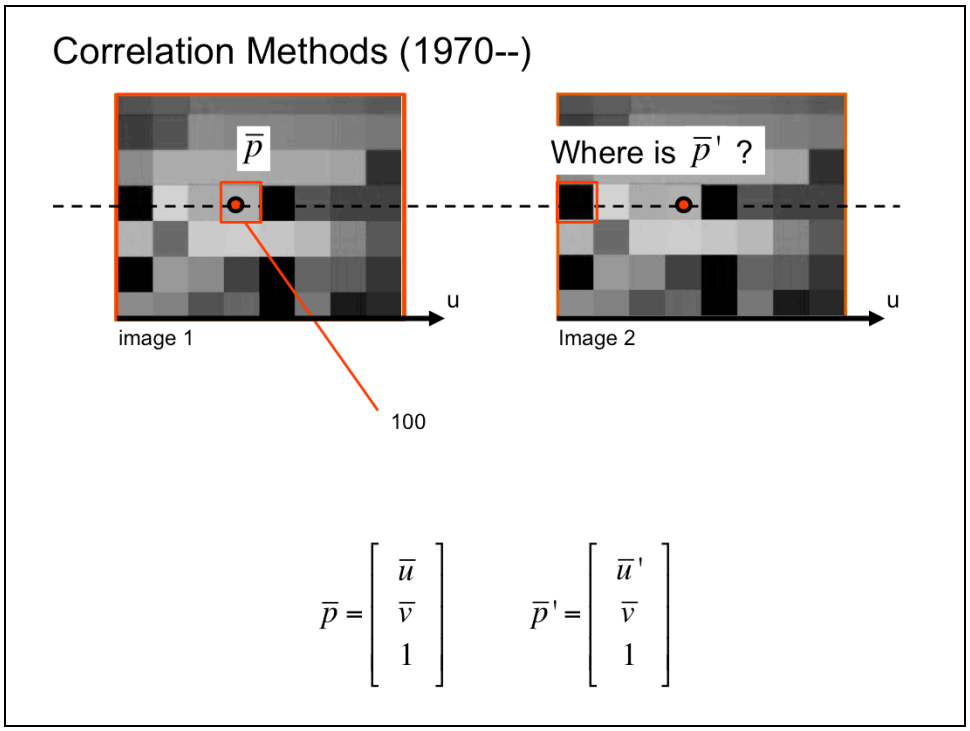
- A Cooperative Model (Marr and Poggio, 1976)
- Correlation Methods (1970--)
- Multi-Scale Edge Matching (Marr, Poggio and Grimson, 1979-81)

[FP] Chapters: 7

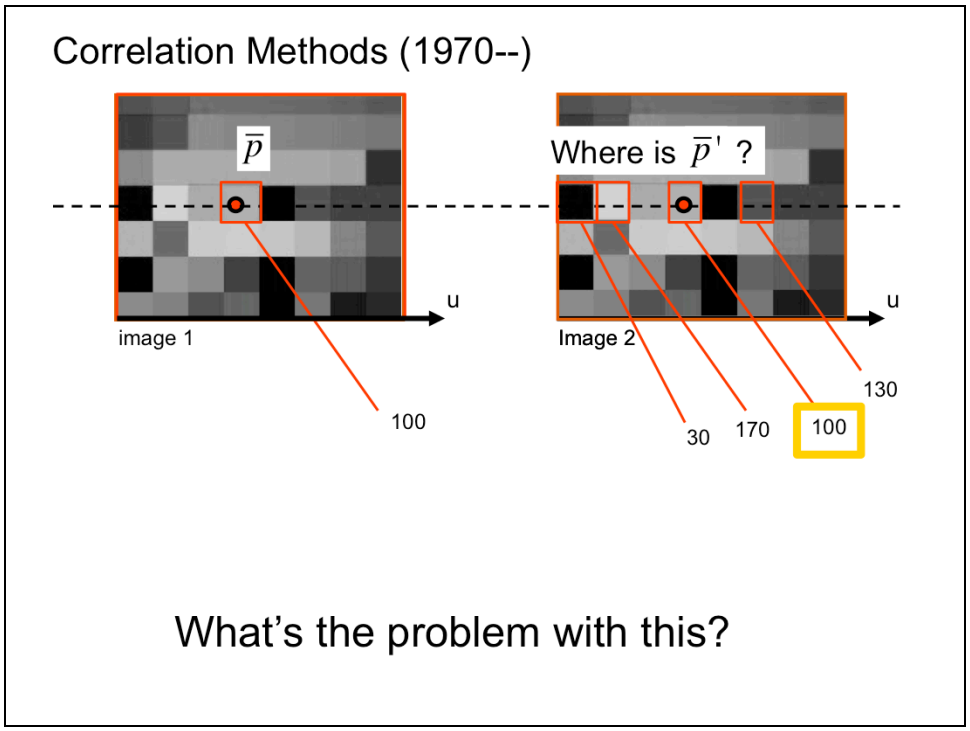
There are several techniques that can be used for solving the correspondence problem in rectified stereo pairs. The slide shows some of these. In this lecture we will focus on the Correlation Methods.



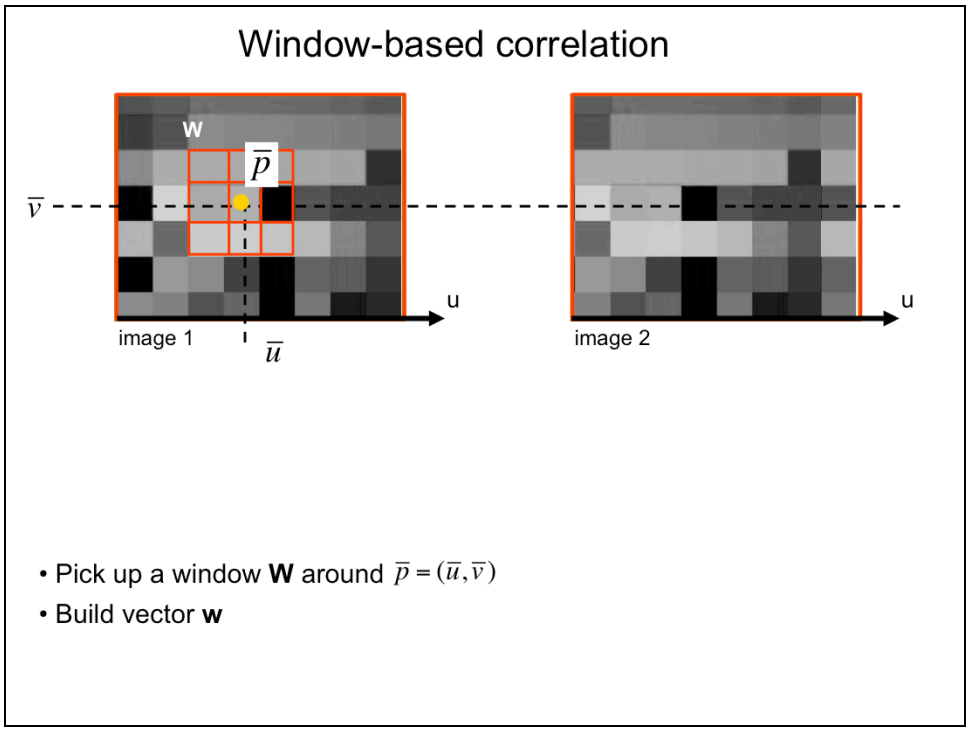
The correspondence problem can be formulated as follows: given a point p^{bar} located at coordinate u^{bar},v^{bar} in the image, find the corresponding point p^{bar} '=(u^{bar}', v^{bar}') in the second image. Because images are parallel, we know that v^{bar}=v^{bar}', thus the problem is to just find u^{bar}'. Correlation methods find corresponding points by comparing the pixel intensity values within windows around the points and choosing the most similar one.



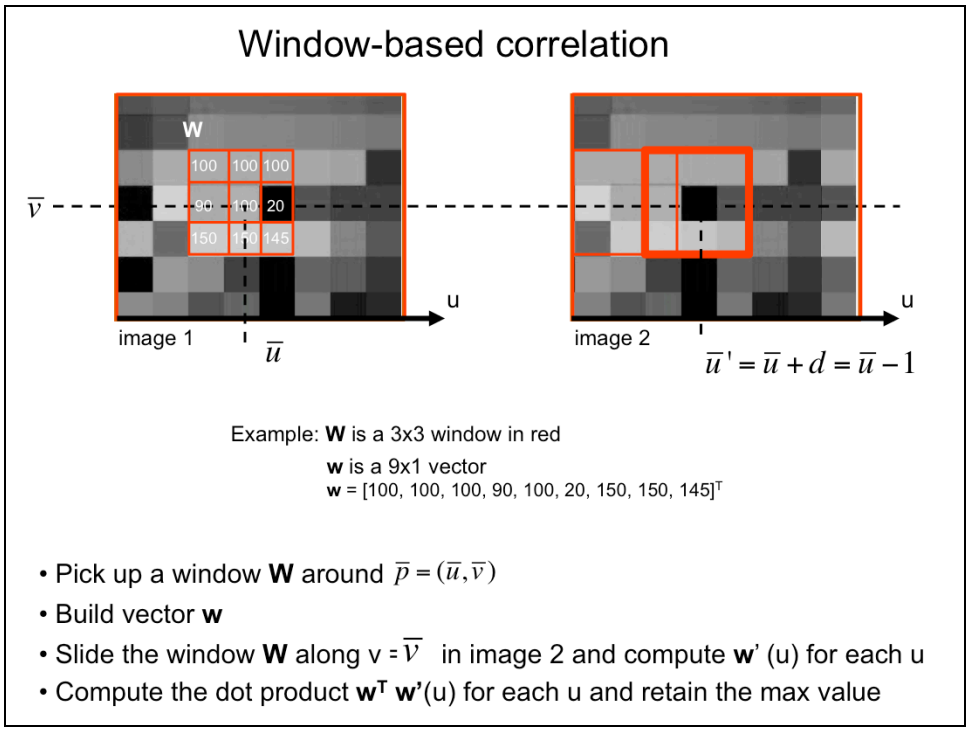
The simplest version of a correlation method is to consider a window that is just 1 pixel big. In the example shown here, the intensity value associated to p^{bar} in the image 1 is 100 (the window is just the pixel where p^{bar} lies);



What the correlation method does is to compare this value with all the values along the line defined by v = v^{bar} '=v^{bar} in the second image (e.g., just to pick up a few: 30, 170, 100, 130), and retain the position u^{bar} ' that returns the highest correlation (agreement) between corresponding values (100 in the case).

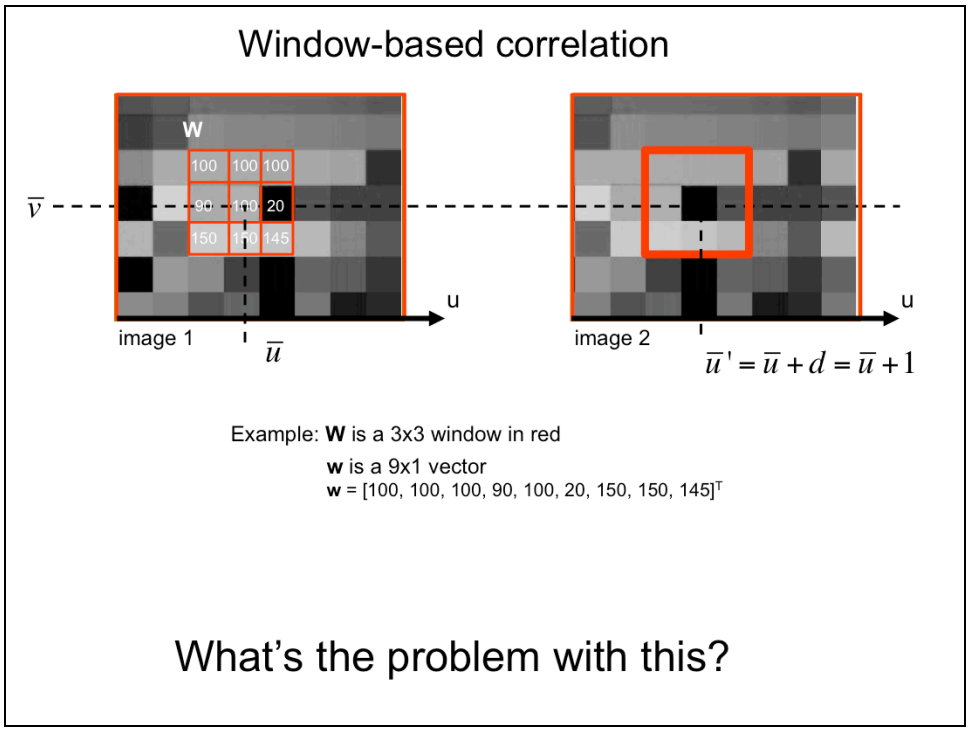


In practice, it is a much better idea to consider a window around the point p^{bar} (for instance, the 3x3 red window around p^{bar} indicated by **W**) and compare the intensity values within this window instead than just the (single) intensity value of a single pixel. Why it is so? Answering this question is an interesting exercise for you.



A basic correlation method creates a column vector **w** from the window **W** by concatenating each row of **W.** Specifically, if $\mathbf{m}_i$ is the i^{th} row of **W**, **w** is obtained by stacking $\mathbf{m}_i^T$ for i=1 to the number of rows of **W**. Thus, if the window **W** is 3x3, **w** will be 9x1. An example is shown in the slide.

After **w** is constructed, the idea is to slide a window **W** along the row $v=v^{bar}$ in image 2 and create a column vector **w'** (similarly to what we did for **w**) for each value of u. To show the dependency of **w'** on u, we refer to this vector as to **w'**(u).Then, we can compute the dot product **w w'**(u) for each u and retain the maximum value (max correlation). Ideally, the u corresponding to such a maximum value, is the estimated u location of the desired (corresponding) point **p'** in the image 2 which is u^{bar} + d (where d=-1, in this example). Notice that d is the disparity.



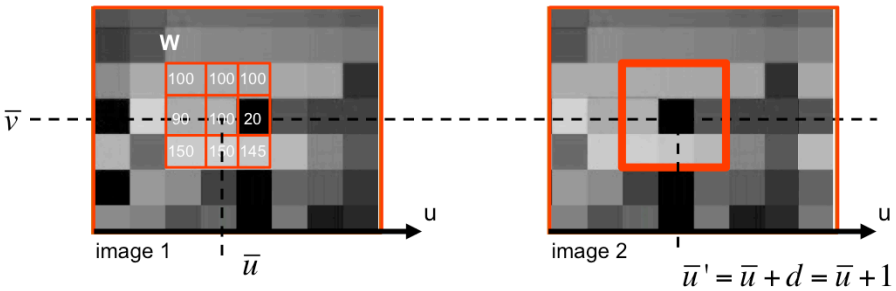
In practice, however, for a number of reasons that we will discuss later (occlusions, foreshortening, illumination conditions, exposure etc..), the maximum value may not necessarily occur at u^{bar} + d but at different u coordinate, which makes the correlation method to fail. In this case we say that correlation method fails to find the right match (corresponding point) and produces a **mismatch**. We will discuss later these effects in a few more details. Here we want address one common case that takes place when the mean and the variance of intensity values in corresponding windows change. This is often due to differences in the camera exposure conditions in image 1 and 2, for instance. A possible way for partially undoing these effects is to normalize the image windows as we compute the correlation – this is called the **normalized cross correlation**.

Changes of brightness/exposure

Changes in the mean and the variance of intensity values in corresponding windows!

In this example, changes of brightness/exposure between the two images, imply changes in the mean and the variance of intensity values in corresponding windows!

Normalized cross-correlation



Find u that maximizes:

$$\frac{(w-\bar{w})^T (w'(u)-\bar{w}')}{\|(w-\bar{w})\| \, \|(w'(u)-\bar{w}')\|}$$

[Eq. 2]

$\bar{w}$ = mean value within **W** located at u^{bar} in image 1

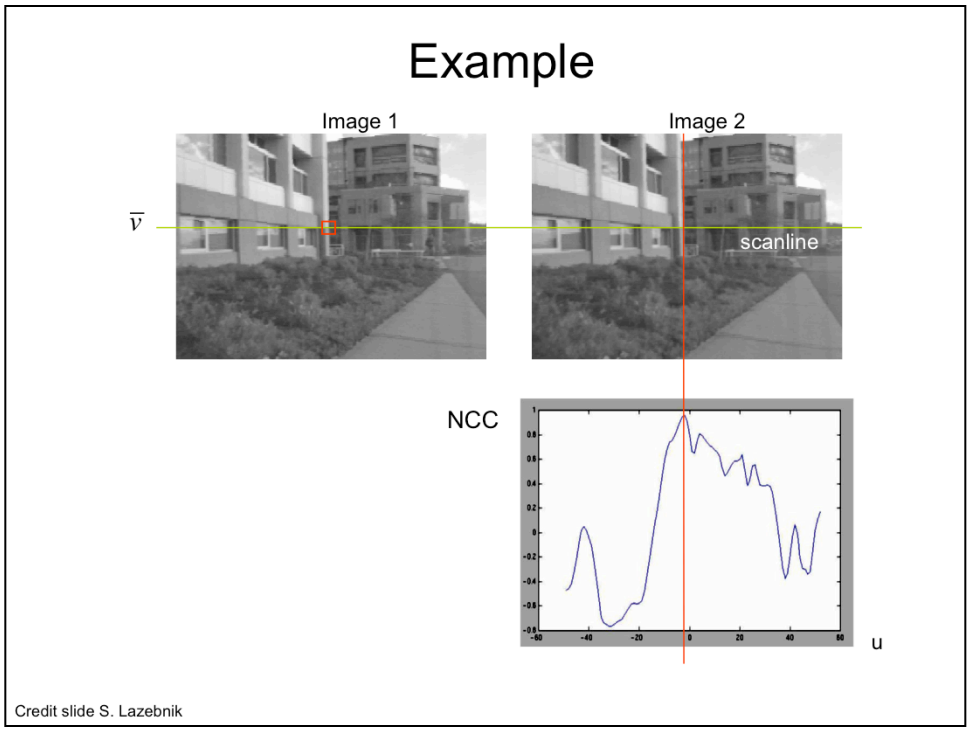
$\bar{w}'(u)$ = mean value within **W** located at u in image 2

In the normalized cross-correlation, instead of just computing the dot product **w w'**(u)**,** one can normalize the intensity values within each window by:

- Removing from **w** and **w'**(u) their mean values $\mathbf{w}^{bar}$ and $\mathbf{w'}^{bar}$(u) respectively. $\mathbf{w}^{bar}$ is the mean value within **W** located at u^{bar} in image 1. $\mathbf{w'}^{bar}$(u) is the mean value within **W** located at u in image 2.

- Dividing by the norms of (**w** – $\mathbf{w}^{bar}$) and (**w'**(u) – $\mathbf{w'}^{bar}$(u))

Thus we obtain Eq. 2



Here's an example. Given two rectified images (1 and 2), we compute the normalized cross correlation (NCC) for each u value (in the image 2) along the horizontal scanline (that is, the line v=v^{bar} along which we are supposed to find the corresponding point). The NCC is shown in the bottom image. The u value that corresponds to the max value of NCC is indicated by the red vertical line. In this example, the correlation method works and the right match is found.

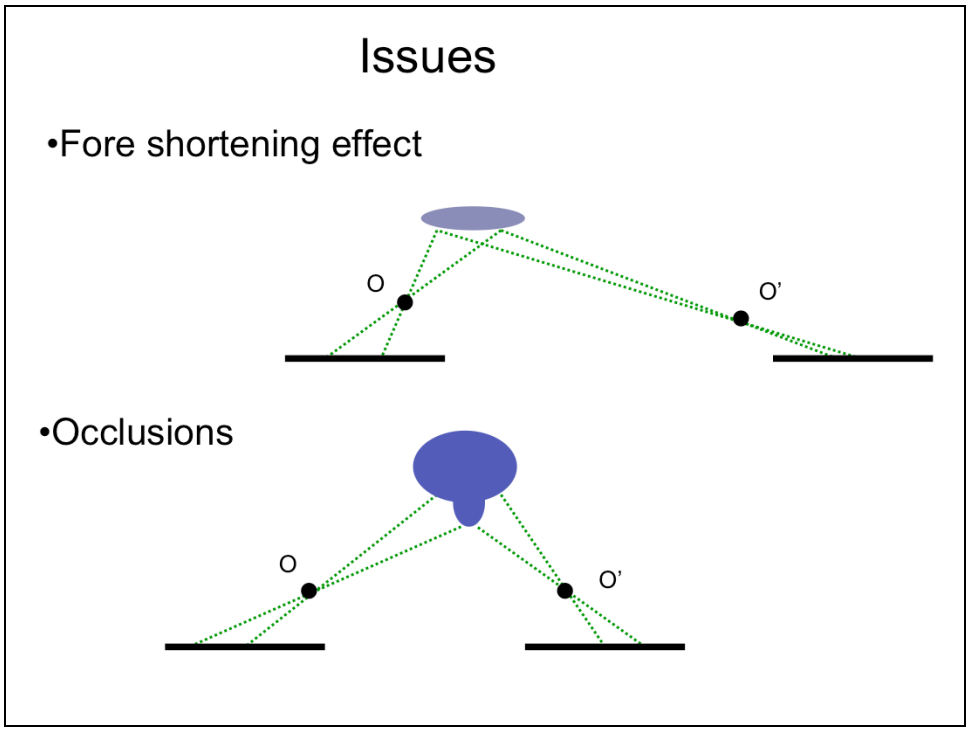
Effect of the window's size

Window size = 3 Window size = 20

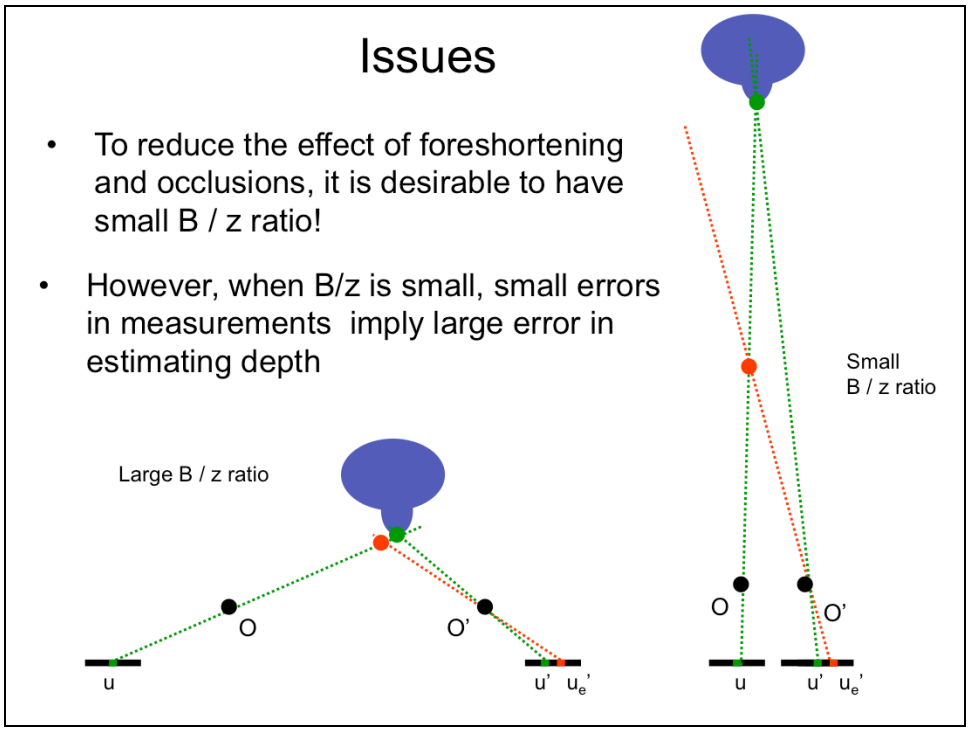
- Smaller window
  - More detail
  - More noise
- Larger window
  - Smoother disparity maps
  - Less prone to noise

Credit slide S. Lazebnik

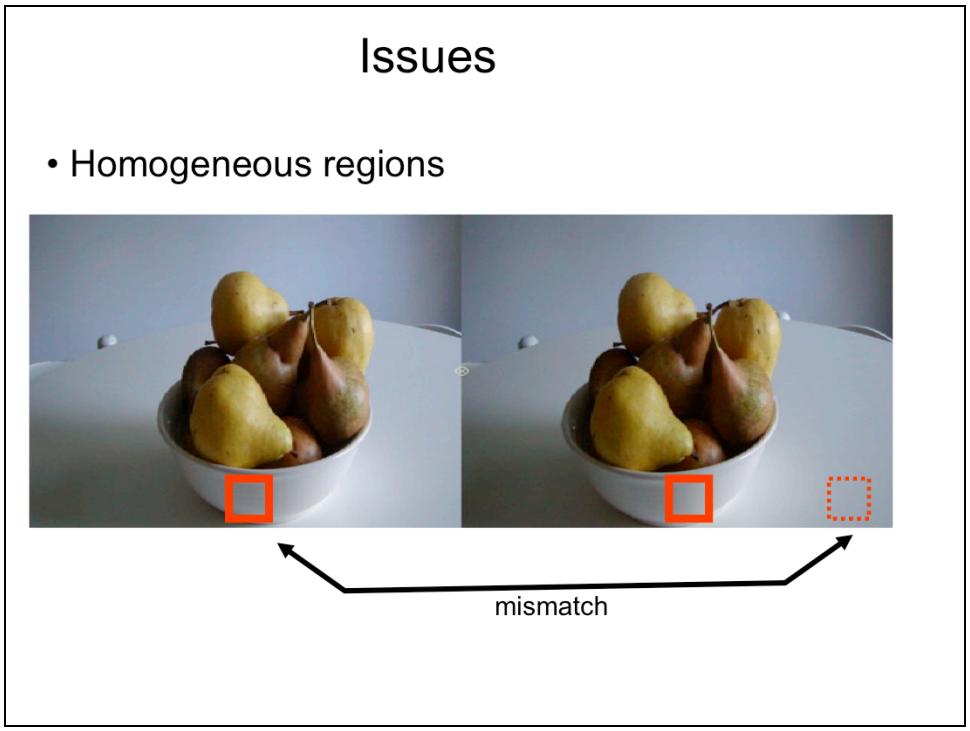
The size of the window plays a critical role in determining the effectiveness of correlation methods. A smaller window results in increasing the ability to resolve small details at the cost of adding more noise, as there is less information to reliably match corresponding points. A larger window allows points to be matched with confidence, but sacrifices detail. In the two top right images we see the disparity maps for a window of size 3 (notice that details of the tree are nicely resolved, but many mismatches occur) and for a window of size 20 (notice that the resolution of the reconstruction is poor, but most of the mismatched have been removed). We can recognize mismatches in those images because mismatched points are estimated with the wrong depth (disparity) and thus are shown with the "wrong" gray code.



A number of common effects can cause problems for correlation methods. Severe foreshortening or occlusions can make the neighborhoods around corresponding points look very different, causing the matching to fail.



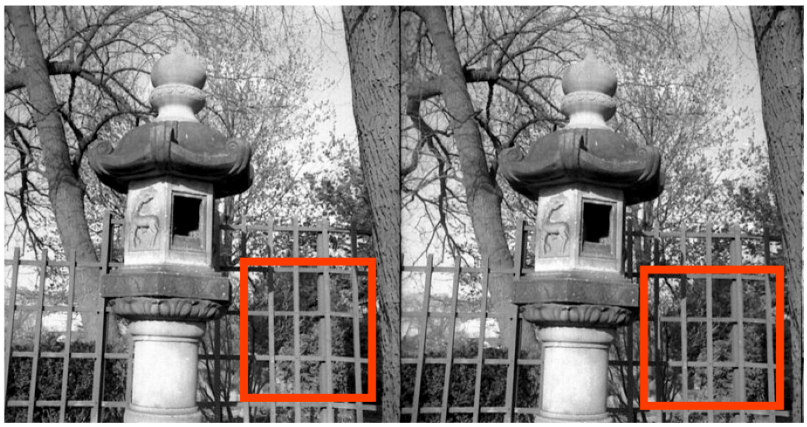
With small baselines or large depths, the matching process is less likely to be affected by foreshortening and occlusions. Thus, it is desirable to have small baseline-depth ratios. However, as such ratio decreases, we also have that the stereo depth estimation process becomes very sensitive to errors in estimating corresponding points. This is because the depth of a point is 3D is obtained by triangulating (intersecting) the two lines of sights corresponding to p and p'. Clearly, if B is small such lines of sights are almost parallel which implies that small errors in estimating the disparity yield large errors in computing the point of intersection of the two lines. As the figures show, when the baseline B is large (left), a small error in estimating the matched point in the second image (e.g., u_e' instead of u') implies that the corresponding estimated 3D point (shown in red) is not too far from the actual one (shown in green). When the baseline is small, a small error in estimating the matched point in the second image (e.g., u_e' instead of u') implies that the corresponding estimated 3D point (shown in red) might be very far from the actual one (shown in green). An interesting exercise is to derive this property from Eq.1.



An other source of problems comes from *homogeneous image regions* – that is, regions in the image that share very similar intensity value distributions. Homogeneous image regions will produce a flat correlation response in which it is very hard to find correspondences. The maxima of the response will be essentially random, so spurious matches will be found, resulting in a noisy disparity map. For instance, the red bounding box (window) on left (that should match the red bounding box on the right), can be mismatched with the window in dashed red on the right because all these windows share very similar intensity distributions.

Issues

•Repetitive patterns



Repetitive patterns also pose a challenge to correlation methods. Here, the similarity response will contain multiple peaks corresponding to each repetition of the pattern, with no guarantee the right repetition will be chosen for any given correspondence.

Correspondence problem is difficult!

- Occlusions
- Fore shortening
- Baseline trade-off
- Homogeneous regions
- Repetitive patterns

Apply non-local constraints to help enforce the correspondences

These problems can be addressed with non-local constraints, using what we know about the geometry of real scenes to constrain the set of possible correspondences.

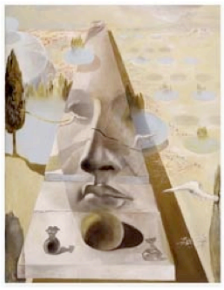
Non-local constraints

- Uniqueness
  - For any point in one image, there should be at most one matching point in the other image
- Ordering
  - Corresponding points should be in the same order in both views
- Smoothness
  - Disparity is typically a smooth function of x (except in occluding boundaries)

Some examples on how non-local constraints can help enforce the correspondences are shown here.

Lecture 6

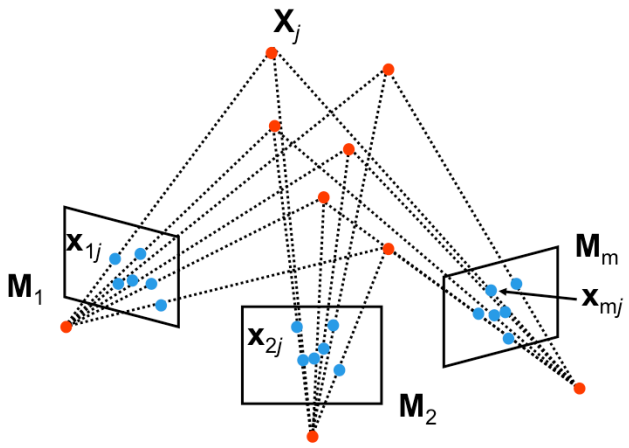
Stereo Systems
Multi-view geometry



- Stereo systems
  - Rectification
  - Correspondence problem
- Multi-view geometry
  - The SFM problem
  - Affine SFM

We'll now explore the extension of the geometry of two cameras to multiple cameras, in what is known as Structure-from-Motion, or SFM. The basic idea is that the geometry of a scene and the parameters of the cameras can be determined by combining observations of points from multiple views.

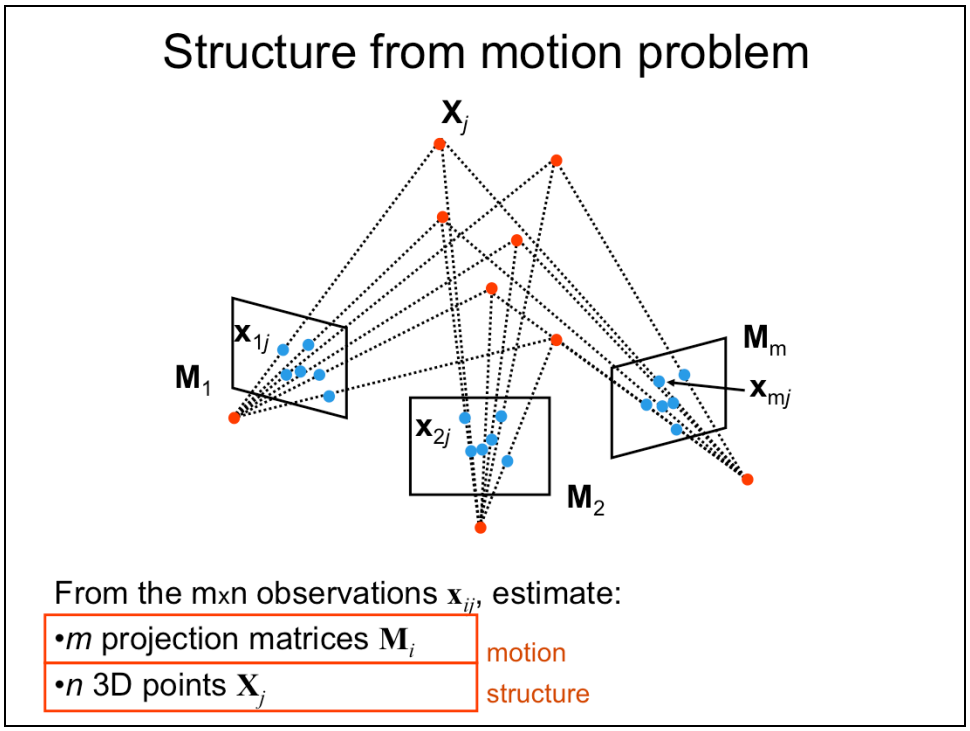
Structure from motion problem



Given *m* images of *n* fixed 3D points

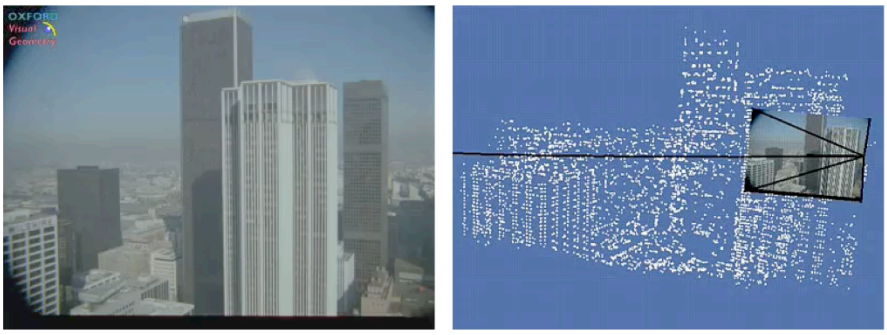
- $\mathbf{x}_{ij} = \mathbf{M}_i \mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$

Here we formulate the structure from motion problem. Suppose we have *m* cameras with camera transformations $\mathbf{M}_i$ encoding both the intrinsic and extrinsic parameters for the cameras. $\mathbf{X}_j$ are 3D points in the scene, usually chosen for their distinctiveness. Suppose there are *n* of such 3D points. Each 3D point may be visible in multiple cameras at the location $\mathbf{x}_{ij}$, where $\mathbf{x}_{ij}$ is the projection of $\mathbf{X}_j$ to the image of the camera i using the projective transformation $\mathbf{M}_i$.



The aim of SFM is to recover both the **structure** of the scene – the n 3D points $\mathbf{X}_j$ – and the **motion** of the cameras – the m projection matrices $\mathbf{M}_i$ – from all the observations $\mathbf{x}_{ij}$.

Structure from motion problem

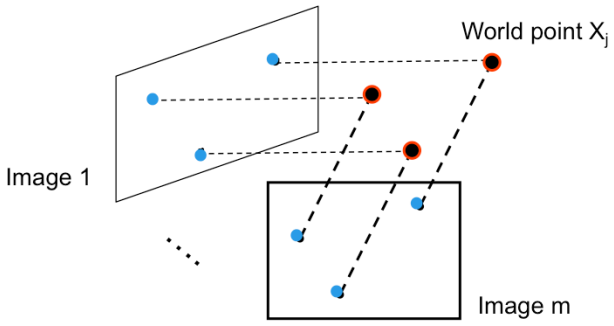


Courtesy of Oxford **Visual Geometry Group**

An example is shown here.

Affine structure from motion

(simpler problem)



From the mxn observations $\mathbf{x}_{ij}$, estimate:

- *m* projection matrices $\mathbf{M}_i$ (affine cameras)
- *n* 3D points $\mathbf{X}_j$

We start with a simpler problem, assuming the cameras are affine or weak perspective. See lecture 3 for details. The result is not a perspective projection – the lack of the perspective scaling operation makes the mathematical derivation easier for our purposes.

Perspective

$$\mathbf{x} = M\,\mathbf{X} = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} \mathbf{X} = \begin{bmatrix} \mathbf{m}_1\mathbf{X} \\ \mathbf{m}_2\mathbf{X} \\ \mathbf{m}_3\mathbf{X} \end{bmatrix} \qquad M = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{v} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix}$$

$$\mathbf{x}^E = \left(\frac{\mathbf{m}_1\mathbf{X}}{\mathbf{m}_3\mathbf{X}}, \frac{\mathbf{m}_2\mathbf{X}}{\mathbf{m}_3\mathbf{X}}\right)$$

Affine

$$\mathbf{x} = M\,\mathbf{X} = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} \mathbf{X} = \begin{bmatrix} \mathbf{m}_1\mathbf{X} \\ \mathbf{m}_2\mathbf{X} \\ 1 \end{bmatrix} \qquad M = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{2x3} & \mathbf{b}_{2x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ 0\ 0\ 0\ 1 \end{bmatrix}$$

$$\mathbf{x}^E = (\mathbf{m}_1\mathbf{X}, \mathbf{m}_2\mathbf{X}) = \begin{bmatrix} \mathbf{A} & \mathbf{b} \end{bmatrix} \mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{A}\mathbf{X}^E + \mathbf{b} \qquad \mathbf{X}^E = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

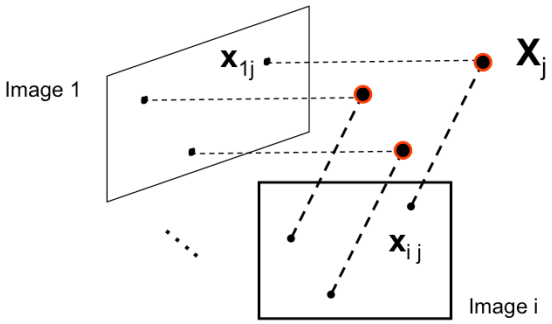
magnification

[Eq. 3]

From lecture 3, we derived the above equations for perspective and weak perspective cases.

Remember that in the full perspective model, m_3 is equal to [v 1], where v is some non-zero 1x3 vector. On the other hand, for the weak perspective model m_3 = [0 0 0 1] which implies that the denominator term m_3 X is 1. As result, the non linearity of the projective transformation disappears (as we move from homogenous to Euclidean coordinates) and the weak perspective transformation acts as a mere magnifier. Equation 3 shows different ways for writing x – the projection of X in the image in Euclidean coordinates.

Affine cameras



Camera matrix $\mathbf{M}^a$ for the affine case (in Euclidean space)

$$\mathbf{x}_{ij} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i = \mathbf{M}_i^a \begin{bmatrix} \mathbf{X}_j \\ 1 \end{bmatrix}; \qquad \mathbf{M}^a = \begin{bmatrix} \mathbf{A} & \mathbf{b} \end{bmatrix}$$

[Eq. 4]

Thus, we now use the affine camera model to express the relationship from a point X_j in 3D and the corresponding observations in each affine camera (for instance, $\mathbf{x}_{ij}$ in camera i). Notice this expression is in Euclidean coordinates and we are omitting the superscript ^{E} in order to simplify the notation. Also, notice that M^a has been defined as [A b] (which is a 2x4 matrix as opposed to M which is 3x4 matrix - see previous slide).

The Affine Structure-from-Motion Problem

Given *m* images of *n* fixed points $\mathbf{X}_j$ we can write

$$\mathbf{x}_{ij} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i$$ for i = 1, …, m and j = 1, …, n

N. of cameras N. of points

Problem: estimate m matrices A_i, m matrices b_i and the n positions $\mathbf{X}_j$ from the m×n observations $\mathbf{x}_{ij}$.

How many equations and how many unknown?

2m × n equations in 8m+3n unknowns

With m=2 cameras, I need at least n=16 3D points
With m=3 cameras, I need at least n=8 3D points

Two approaches:

- Algebraic approach (affine epipolar geometry; estimate F; cameras; points)
- Factorization method

Returning to the SFM problem, we need to estimate m 2x3 matrices A_i, m 2x1 matrices b_i, and the n world coordinate 3X1 vectors X_j, for a total of 8m+3n unknowns, from m x n observations. Each observation creates 2 constraints per camera, so there are 2m x n equations in 8m+3n unknowns. It's easy to see that with m=2 cameras, I need to have at least n=16 points in 3D; With m=3 cameras, I need to have at least n=16 points in 3D;

There are several ways for solving this problem. One is to use an Algebraic approach which takes advantage of the properties of the affine epipolar geometry. Another approach is the factorization method which we will describe in details next.

A factorization method –
Tomasi & Kanade algorithm

C. Tomasi and T. Kanade
Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

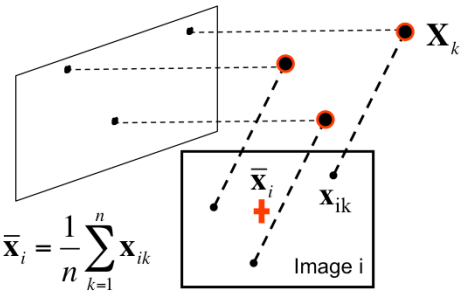
- Data centering
- Factorization

The factorization method is also referred as the Tomasi and Kanade's factorization method for solving the affine SFM problem. This method consists of two major steps: the data centering step and the actual factorization step.

A factorization method - Centering the data

Centering: subtract the centroid of the image points

[Eq. 6] $\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik}$ $\quad \bar{\mathbf{x}}_i$



[Eq. 5] $\bar{\mathbf{x}}_i = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik}$

Let's begin with the data centering step. In this step the main idea is to remove the centroid x_i^{bar} of the image points from each image point x_{ij} in image i, for j=1 to n. The centroid x_i^{bar} is defined in Eq. 5 and it is illustrated by the red cross in image i in the bottom part of the slide. After this centering (normalization) step, normalized observed image points are redefined as x_{ij}^{hat} as indicated by Eq. 6.

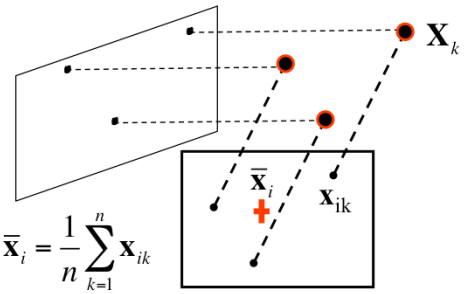
A factorization method - Centering the data

Centering: subtract the centroid of the image points

[Eq. 6] $$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\mathbf{A}_i\mathbf{X}_k - \frac{1}{n}\sum_{k=1}^{n}\mathbf{b}_i$$

$$\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_k + \mathbf{b}_i$$

[Eq. 4]



[Eq. 5] $$\bar{\mathbf{x}}_i = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik}$$

Let us now replace the expression of x_{ik} using Eq. 4.

A factorization method - Centering the data

Centering: subtract the centroid of the image points

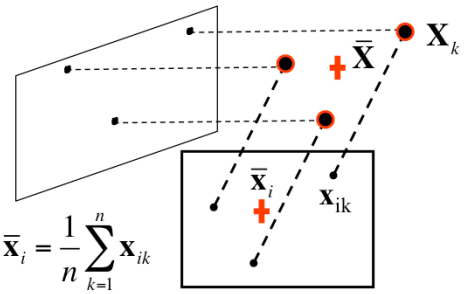
$$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\mathbf{A}_i\mathbf{X}_k - \frac{1}{n}\sum_{k=1}^{n}\mathbf{b}_i$$

$$= \mathbf{A}_i\left(\mathbf{X}_j - \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k\right) = \mathbf{A}_i\left(\mathbf{X}_j - \bar{\mathbf{X}}\right)$$

$$= \mathbf{A}_i\hat{\mathbf{X}}_j \quad \text{[Eq. 8]}$$

$$\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_k + \mathbf{b}_i$$

[Eq. 4]



$$\bar{\mathbf{x}}_i = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik}$$

$$\bar{\mathbf{X}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k \quad \text{[Eq. 7]}$$

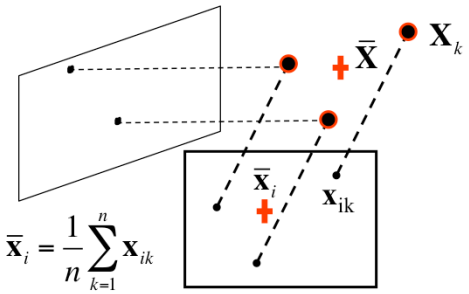
Centroid of 3D points

Further manipulations lead to Eq. 8, where X^{bar} is the centroid of the 3D points (defined by Eq. 7) and X^{hat}_i are the centered (normalized) 3D points with respect to X^{bar}.

A factorization method - Normalizing the data

Thus, after centering, each **normalize**d observed point is related to the 3D point by

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i \hat{\mathbf{X}}_j \quad \text{[Eq. 8]}$$



$$\bar{\mathbf{x}}_i = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

$$\bar{\mathbf{X}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{X}_k \quad \text{[Eq. 7]}$$

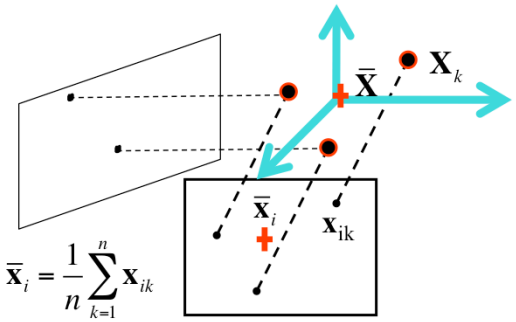
Centroid of 3D points

Thus, after centering, each normalized observed point x_{ij}^{hat} is related to the normalized 3D point X^{hat}_j by the relationship in Eq.8 which only includes the 2x3 matrix A_i

A factorization method - Normalizing the data

If the centroid of points in 3D = center of the world reference system

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i \hat{\mathbf{X}}_j = \mathbf{A}_i \mathbf{X}_j \quad \text{[Eq. 9]}$$



$$\bar{\mathbf{x}}_i = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

$$\bar{\mathbf{X}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{X}_k \quad \text{[Eq. 7]}$$

Centroid of 3D points

If we locate the center of the world reference system at the centroid X^{bar}(defined as in Eq. 7]), Eq. 8 becomes Eq. 9 which relates each normalized observed point x_{ij}^{hat} directly to the corresponding 3D point and X_j via A_i.

Thus, to summarize, by centering (normalizing) the observations x_{ij} in each image (an operation that we can easily implement since x_{ij}'s are all measurable quantities in pixel in the image), and by assuming that the world reference system is located at X^{bar} (the location of the world reference system is arbitrary), we obtain a very compact expression that relates (centered) observations and 3D points (Eq. 9).

A factorization method - factorization

Let's create a 2m × n data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}$$

cameras ($2m$)

points (n)

Each $\hat{\mathbf{x}}_{ij}$ entry is a 2x1 vector!

Using all the observations for all the cameras, we build the measurement matrix D, made up of n observations in the m cameras (remember that each x^{hat}_{ij} entry is a 2x1 vector)

A factorization method - factorization

Let's create a 2m × n data (measurement) matrix:

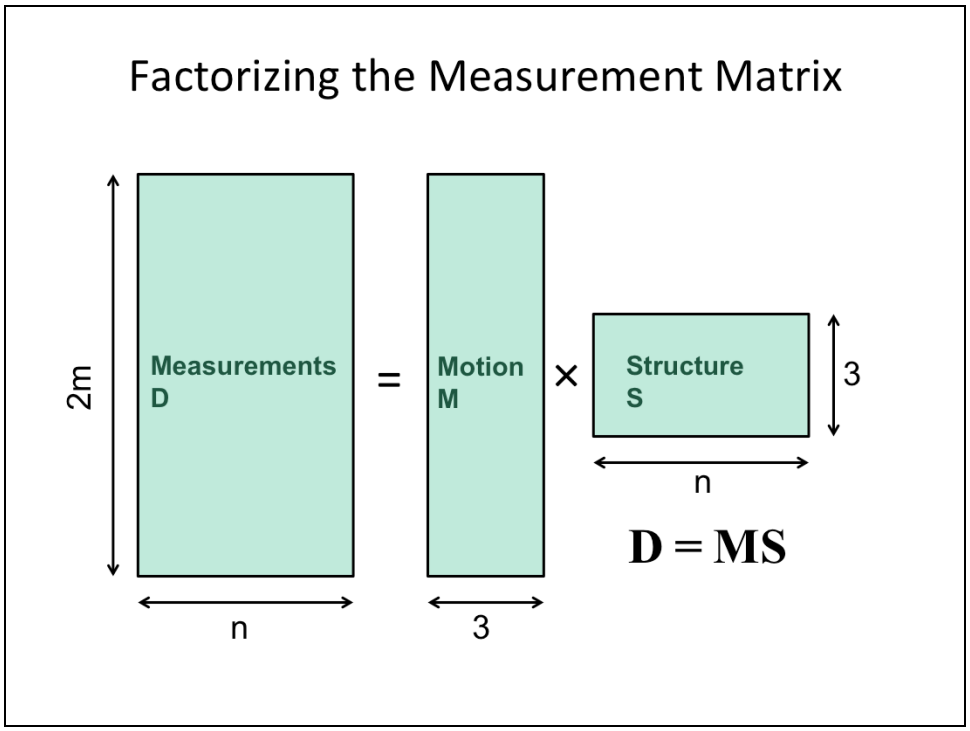
$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

(2*m* × n) — cameras (2*m* × 3) — M — points (3 × *n*) — S — [Eq. 10]

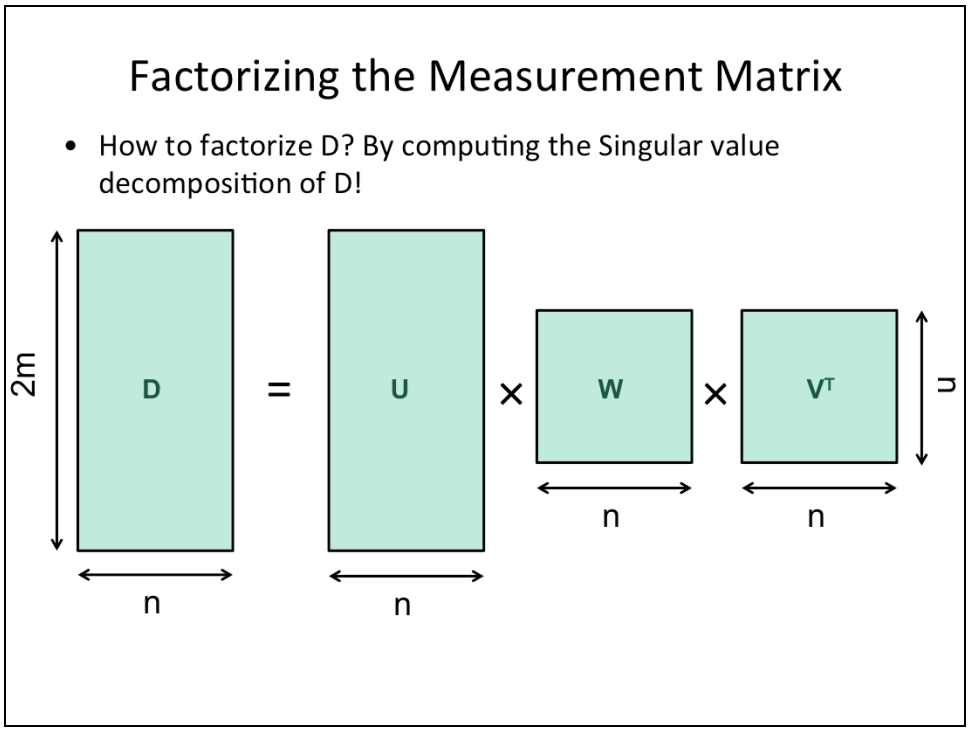
Each $\hat{\mathbf{x}}_{ij}$ entry is a 2x1 vector!
$\mathbf{A}_i$ is 2x3 and $\mathbf{X}_j$ is 3x1

The measurement matrix **D = M S** has rank 3
(it's a product of a 2mx3 matrix and 3xn matrix)

It is easy to see that D can be expressed as the product of the 2m×3 matrix M (which comprises the camera matrices $A_1 \ldots A_m$) and the 3×n matrix S (which comprises the 3D points $X_1, \ldots X_n$). The matrix D has rank three since it is the product of 2 matrices whose max dimension is 3.



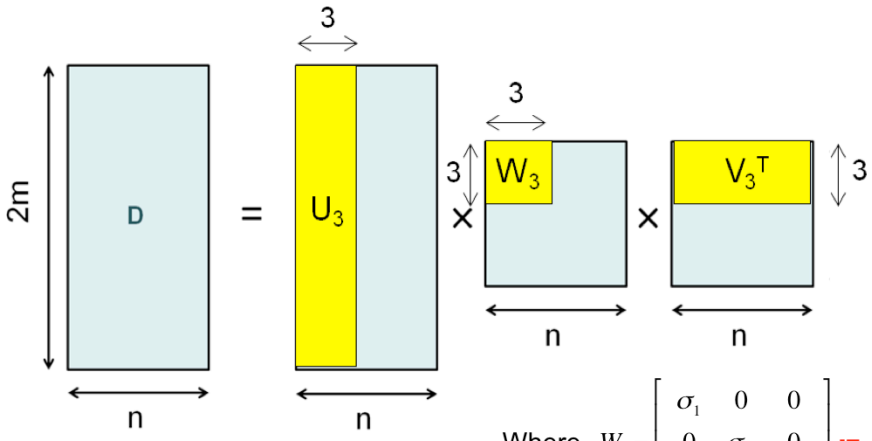
Our aim is to factorize D into these two unknown matrices M and S – the motion and structure parameters. The slide illustrates a visualization of D=M S, where the dimensions and the meaning of D, M and S are highlighted.



How to factorize D into M and S? Using the Singular Value Decomposition U W V^T, where W is the diagonal matrix that contains all the singular values of the decomposition.

Factorizing the Measurement Matrix

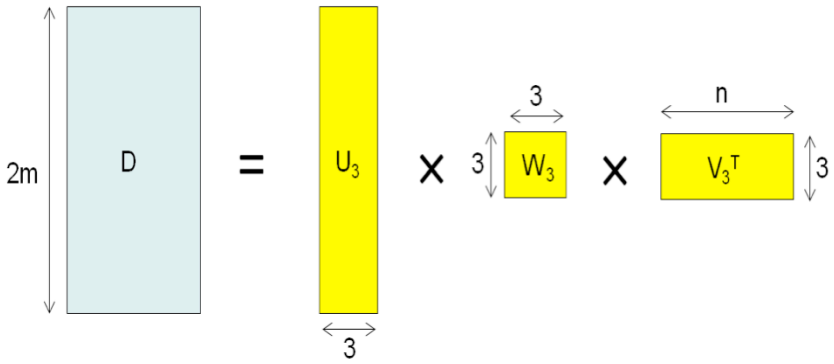
Since rank (D)=3, there are only 3 non-zero singular values σ_1 , σ_2 and σ_3



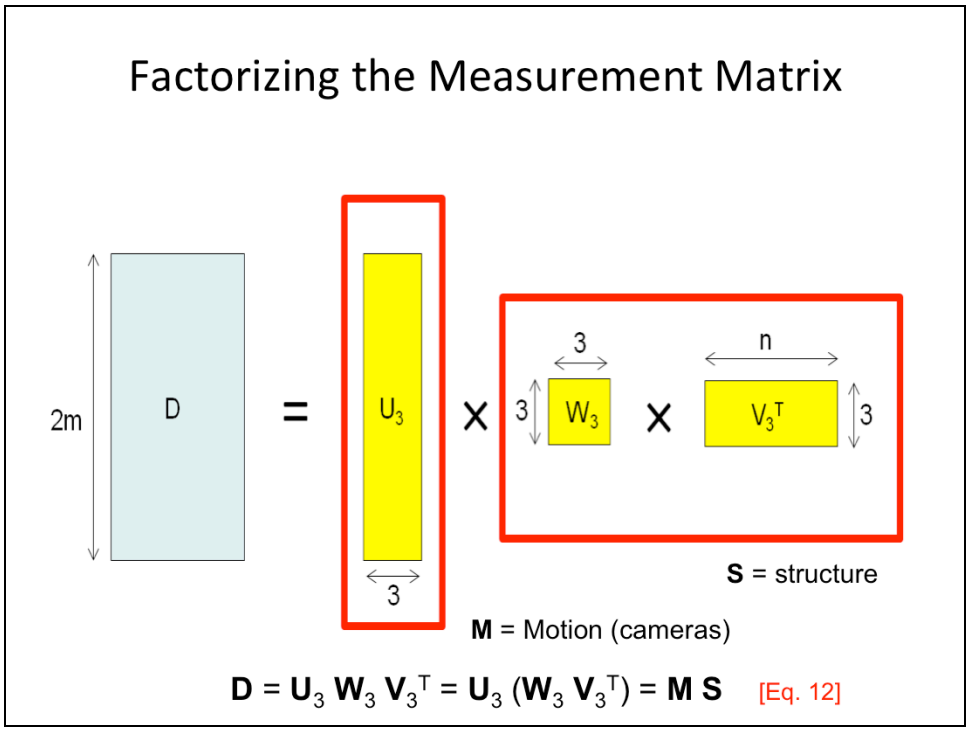
Where $W_3 = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}$ [Eq. 11]

We know the rank of D is 3, so there will only be 3 non-zero singular values σ_1 , σ_2 and σ_3 in W, leading to the matrix W_3 as defined in Eq. 11. Taking the corresponding columns of U and rows of V gives us matrices U_3 and V_3.

Factorizing the Measurement Matrix



Thus, by removing all the zero components of U, W and V, we obtain the following decomposition: $D = U_3 W_3 V_3^T$.



We combine W_3 and V_3^T to make the structure matrix S, and U_3 becomes the motion matrix M. This leads to Eq. 12. While this way of associating the components of the SVD decomposition to M and S leads to a physically and geometrical plausible solution of the affine SFM problem, this choice is not unique. We could also use V_3 to make the structure matrix S, and combine W_3 and U_3 to make the structure matrix M, since in either cases the observation matrix D is the same. We will discuss these ambiguities in the next lecture.

Factorizing the Measurement Matrix

$$\mathbf{D} = \mathbf{U}_3 \mathbf{W}_3 \mathbf{V}_3^T = \mathbf{U}_3 (\mathbf{W}_3 \mathbf{V}_3^T) = \mathbf{M}\,\mathbf{S} \quad \text{[Eq. 12]}$$

What is the issue here? **D** has rank>3 because of:

- measurement noise
- affine approximation

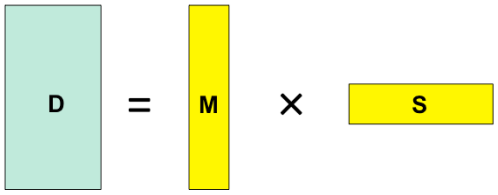
Theorem: When $\mathbf{D}$ has a rank greater than 3, $\mathbf{U}_3\mathbf{W}_3\mathbf{V}_3^T$ is the best possible rank-3 approximation of **D** in the sense of the Frobenius norm.

$$\mathbf{D} = \mathbf{U}_3\mathbf{W}_3\mathbf{V}_3^T \quad \begin{cases} \mathbf{M} \approx \mathbf{U}_3 \\ \mathbf{S} \approx \mathbf{W}_3\mathbf{V}_3^T \end{cases}$$

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}}\sigma_i^2}$$

Unfortunately, in practice, D has a rank greater than 3 because of measurement noise and the affine camera approximation we've used. It can be shown when D has a rank greater than 3, $U_3\ W_3\ V_3^T$ is the best possible rank-3 approximation of D in the sense of the Frobenius norm. Thus, the solutions for M and S can be approximated to U_3 and $W_3\ V_3^T$, respectively.

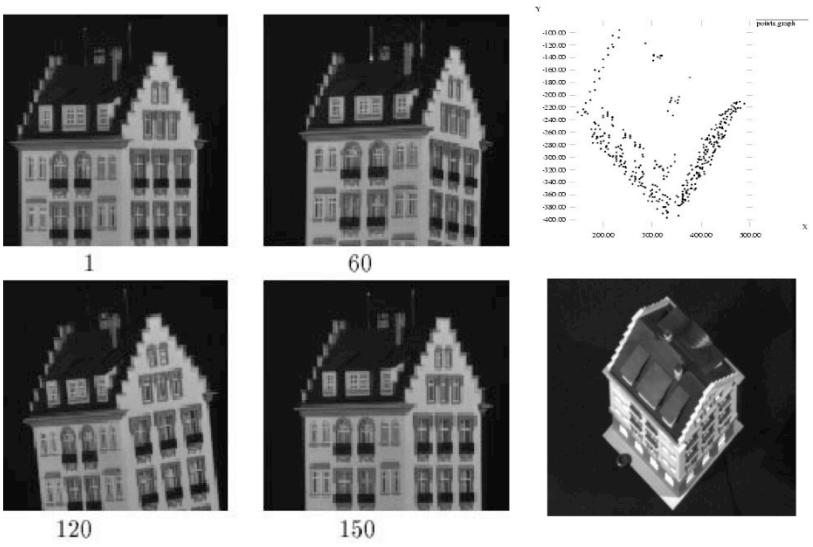
Affine Ambiguity



- The decomposition is not unique. We get the same **D** by using any 3×3 matrix **C** and applying the transformations **M** → **MC**, S →C^{-1}S
- We can enforce some Euclidean constraints to resolve this ambiguity (more on next lecture!)

Notice that we can also arbitrarily transform the motion matrix by a 3x3 transform C, as long as we also transform the structure matrix by the inverse transformation C^{-1}: The resulting observation matrix D will still be the same. Therefore, our solution is underdetermined, and requires extra constraints to resolve this ambiguity. We will discuss this in the next lecture.

Reconstruction results



C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Here's a result presented by Tomasi and Kanade in their paper. The 4 images on the left are 4 views of the same house. Notice that these views are generated using an affine projective transformations (or an approximation of it).

For each of the images we assume that a number of points observations are available along with their correspondences across images. The top right figure shows the reconstruction result (almost top view) obtained using the factorization method applied to the matrix D (which is obtained from all the available observations). Dots are the reconstructed 3D points associated to the observations. The bottom right figure shows a "texture mapping" rendering of the same reconstruction.

Next lecture

Multiple view geometry

Perspective structure from Motion

In the next lecture we'll remove the assumption of affine cameras, and explore the geometric constraints generated by multiple perspective cameras.