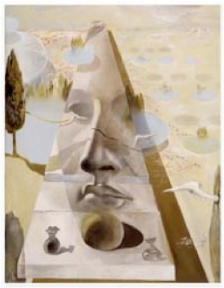
Lecture 7
Multi-view geometry



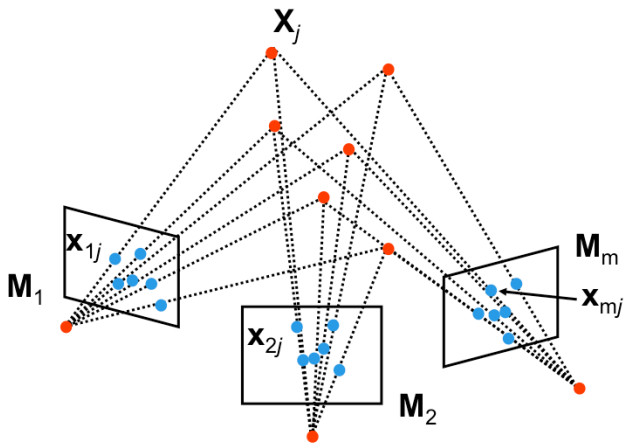
- The SFM problem
- Affine SFM
- Perspective SFM
- Self-calibration
- Applications

Reading:

[HZ] Chapter 10 "3D reconstruction of cameras and structure"
Chapter 18 "N-view computational methods"
Chapter 19 "Auto-calibration"
[FP] Chapter 13 "projective structure from motion"
[Szelisky] Chapter 7 "Structure from motion"

Today we'll continue studying the structure from motion problem. First, we'll recap the affine SFM problem, then we'll explore doing SFM with a more realistic assumption of perspective cameras and discuss the intrinsic ambiguities of SFM. Then, we'll discuss an algorithm for solving SFM with perspective cameras and we will briefly discuss the self-calibration method. We'll conclude with some applications of SFM.

Structure from motion problem

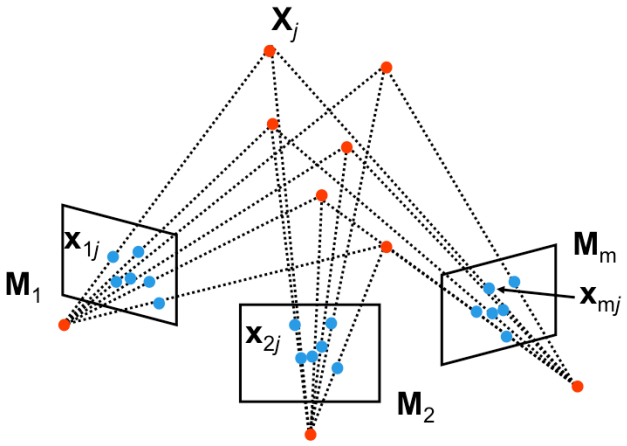


Given *m* images of *n* fixed 3D points

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

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 $\mathbf{M}_i$.

Structure from motion problem

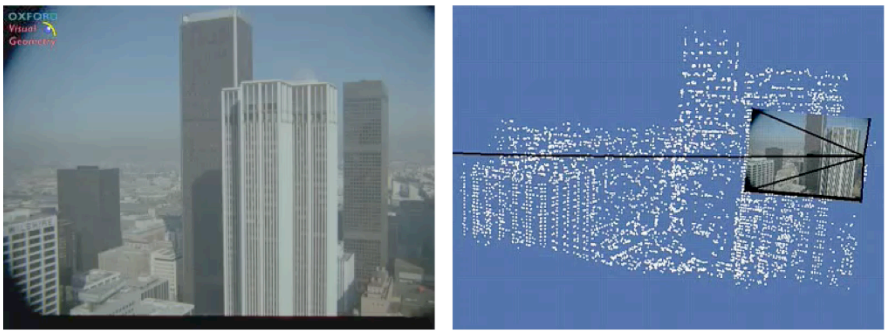


From the mxn observations $\mathbf{x}_{ij}$, estimate:

- *m* projection matrices $\mathbf{M}_i$ — motion
- *n* 3D points $\mathbf{X}_j$ — structure

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

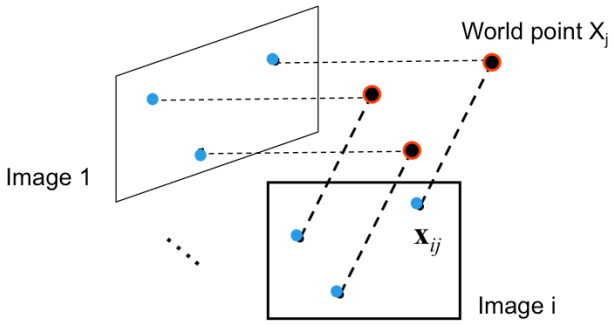


Courtesy of Oxford **Visual Geometry Group**

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

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

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

Affine

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

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

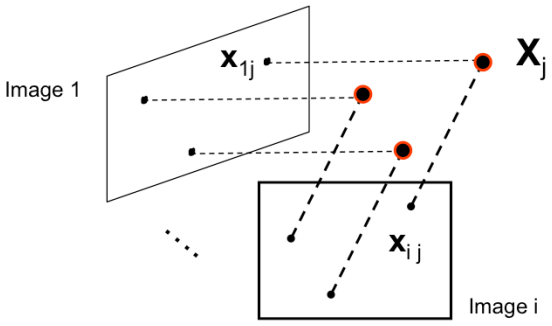
magnification

[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



For the affine case (in Euclidean space)

$$\mathbf{x}_{ij} = \mathbf{A}_i \mathbf{X}_j + \mathbf{b}_i \qquad \text{[Eq. 4]}$$

2x1 2x3 3x1 2x1

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, $\mathbf{x}_{i\,j}$ in camera i) wich is expressed by Eq.4.

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

m: N. of cameras; n: 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

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;

The Affine Structure-from-Motion Problem

Two approaches:

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

- Factorization method

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.

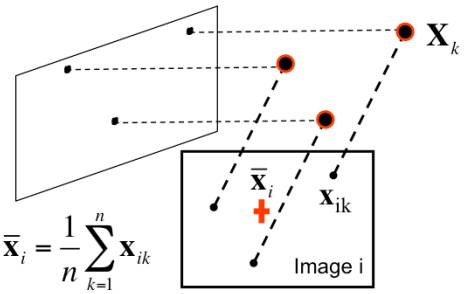
- Data centering
- Factorization

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.

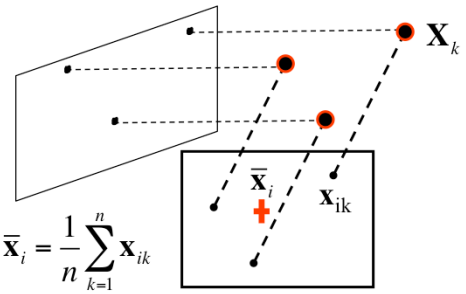
A factorization method - Centering the data

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

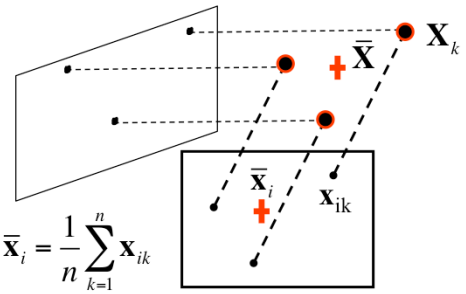
$$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\mathbf{A}_i\mathbf{X}_k - \frac{1}{n}\sum_{k=1}^{n}\mathbf{b}_i$$

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

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

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

[Eq. 4]



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

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

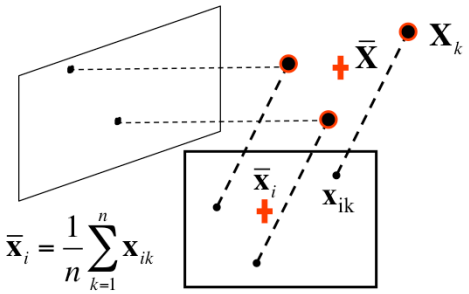
Centroid of 3D points

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 - Centering the data

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

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



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

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

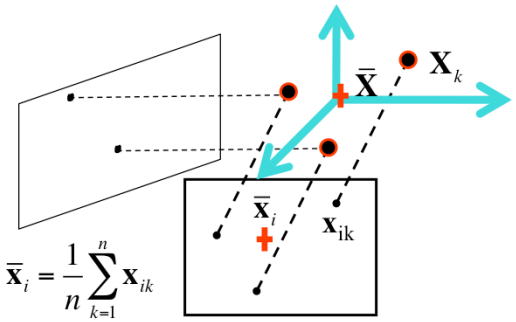
Centroid of 3D points

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 - Centering 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:

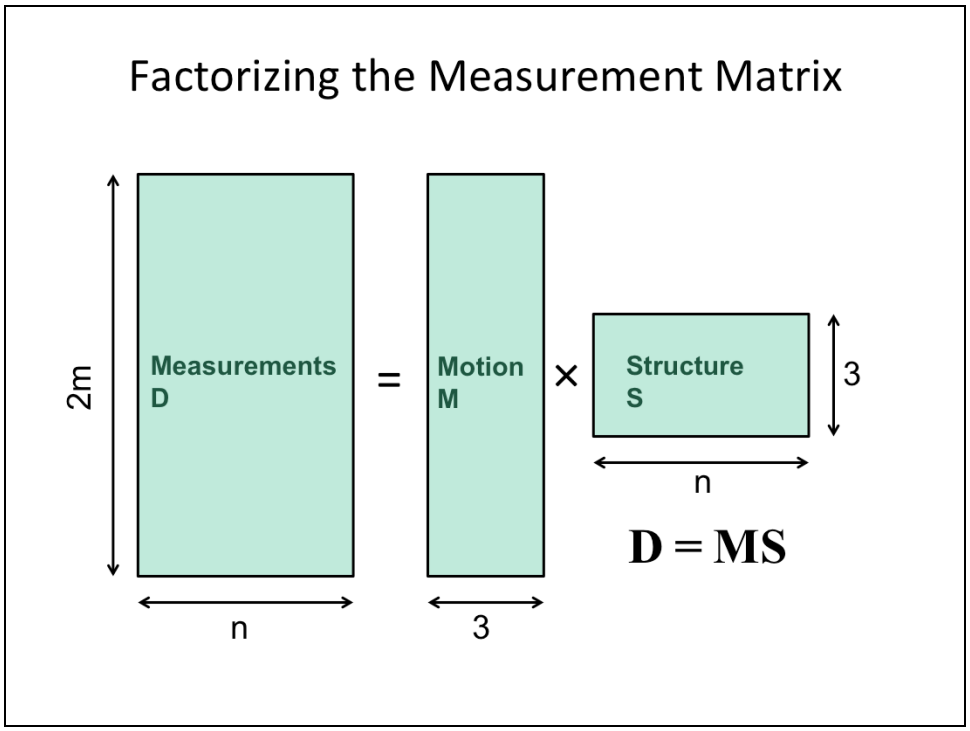
$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}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

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

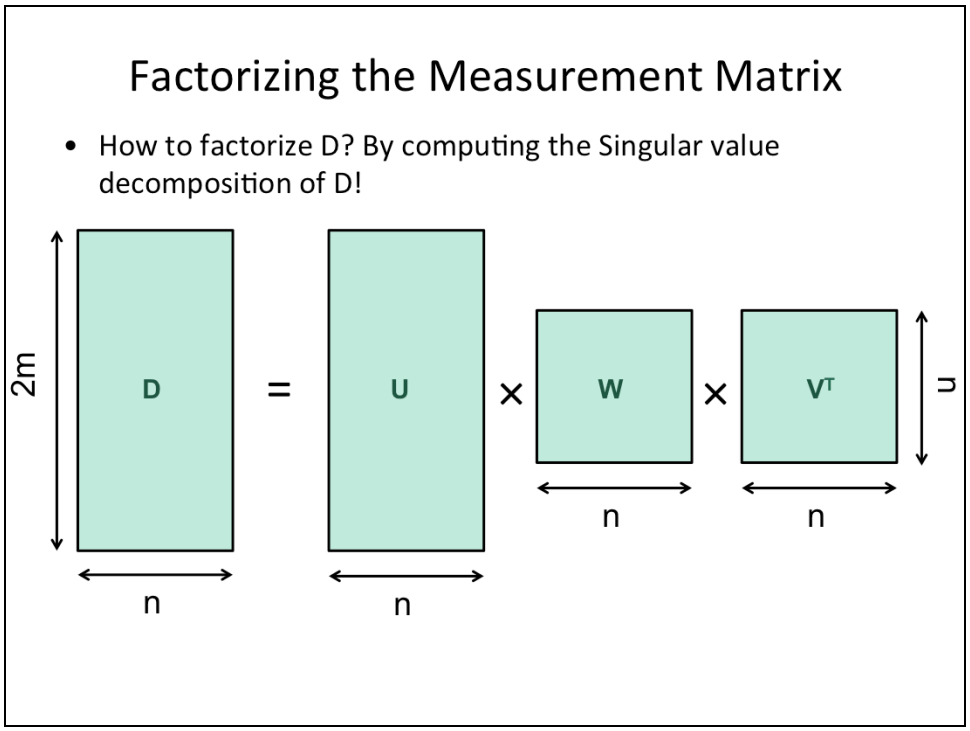
Each $\hat{\mathbf{x}}_{ij}$ entry is a 2x1 vector!
$\mathbf{A}_i$ is 2x3 and $\mathbf{X}_j$ is 3x1

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.



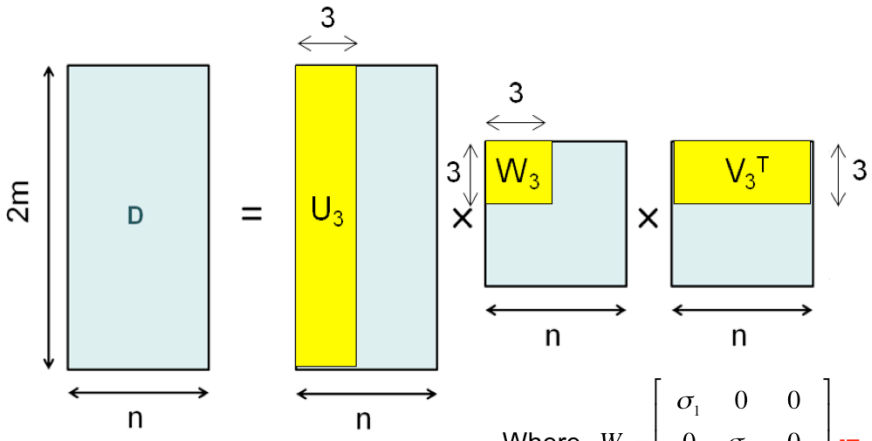
Our aim is to factorize D into these two unknown matrices M and S – the motion and structure parameters. The slide illustrates a visualization of D=M S, where the dimensions and the meaning of D, M and S are highlighted.



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

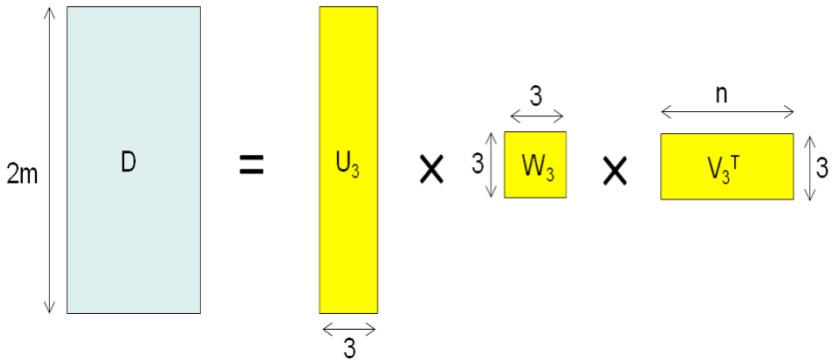
Since rank (D)=3, there are only 3 non-zero singular values σ_1 , σ_2 and σ_3



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

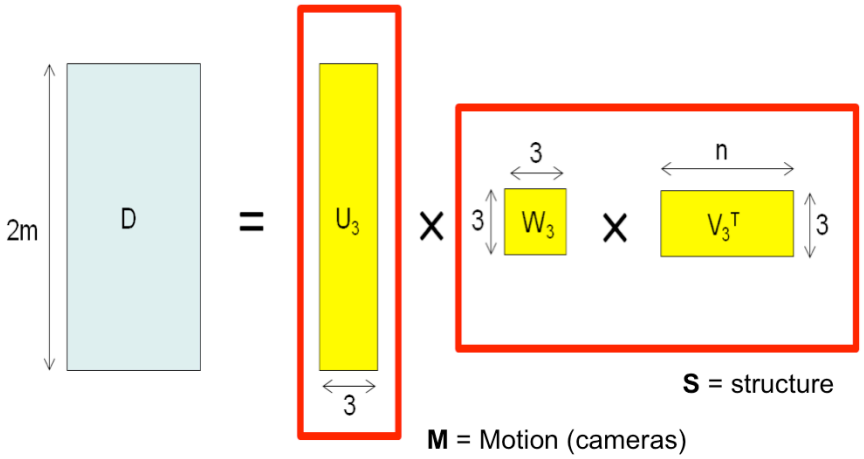
We know the rank of D is 3, so there will only be 3 non-zero singular values σ_1 , σ_2 and σ_3 in W, leading to the matrix W_3 as defined in Eq. 11. Taking the corresponding columns of U and rows of V gives us matrices U_3 and V_3.

Factorizing the Measurement Matrix



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

Factorizing the Measurement Matrix



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

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.

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

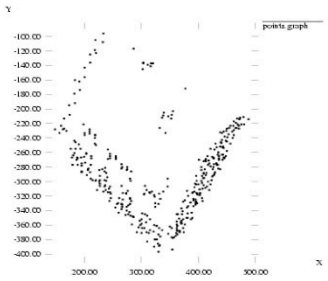
Theorem: When $\mathbf{D}$ has a rank greater than 3, $\mathbf{U}_3\mathbf{W}_3\mathbf{V}_3^T$ is the best possible rank-3 approximation of **D** in the sense of the Frobenius norm.

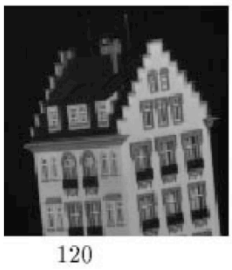
$$\mathbf{D} = \mathbf{U}_3\mathbf{W}_3\mathbf{V}_3^T \quad \begin{cases} \mathbf{M} \approx \mathbf{U}_3 \\ \mathbf{S} \approx \mathbf{W}_3\mathbf{V}_3^T \end{cases}$$

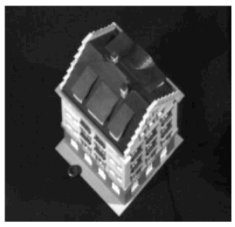
$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}}\sigma_i^2}$$

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.

Reconstruction results





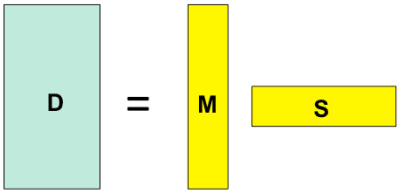


C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Here's a result presented by Tomasi and Kanade in their paper. The 2 images on the left are 2 views of the same house. Notice that these views are generated using an affine projective transformations (or an approximation of it).

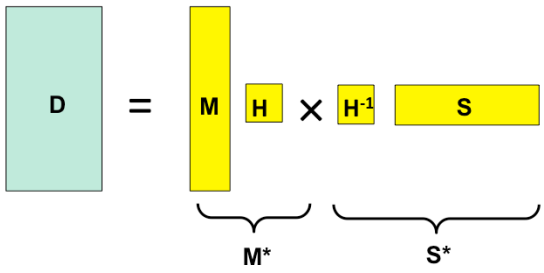
For each of the images we assume that a number of points observations are available along with their correspondences across images. The top right figure shows the reconstruction result (almost top view) obtained using the factorization method applied to the matrix D (which is obtained from all the available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

Affine Ambiguity



Given the decomposition D=M S,

Affine Ambiguity



- The decomposition is not unique. We get the same **D** by applying the transformations:

$$\mathbf{M}^* = \mathbf{M}\,\mathbf{H}$$

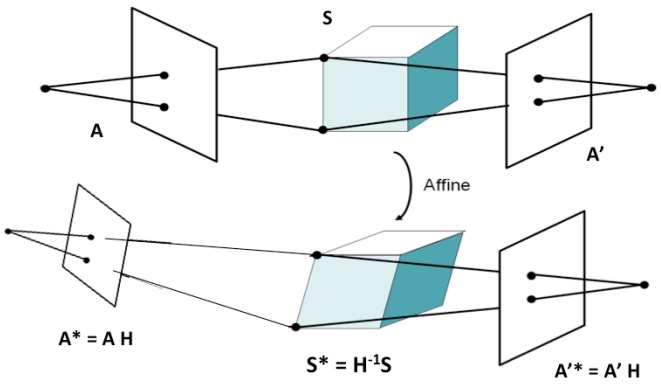
$$\mathbf{S}^* = \mathbf{H}^{-1}\mathbf{S}$$

where **H** is an arbitrary 3x3 matrix describing an affine transformation

- Additional constraints must be enforced to resolve this ambiguity

…we can also arbitrarily transform the motion matrix by a 3x3 affine transform H, as long as we also transform the structure matrix by the inverse transformation HA^{-1}: The resulting observation matrix D will still be the same. Therefore, our solution requires extra constraints to resolve this ambiguity.

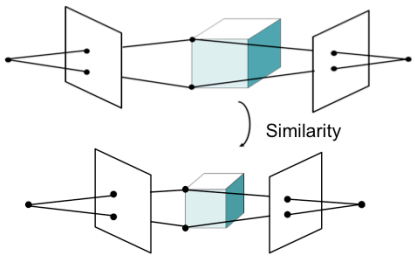
Affine Ambiguity



The slide shows an example of an affine ambiguity for a two-camera system: if the structure is transformed into $\mathbf{H^{-1}S}$, we can always transform the cameras into **A H** and **A' H** respectively, such that the observations won't change.

Similarity Ambiguity

- The scene is determined by the images only up a similarity transformation (rotation, translation and scaling)
- This is called **metric reconstruction**



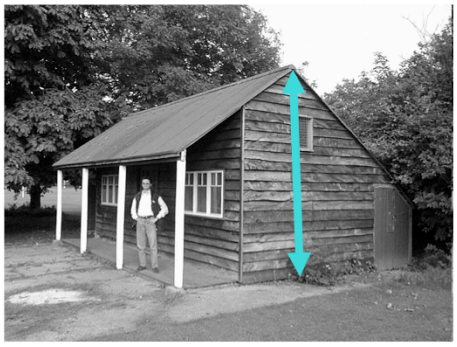
- The ambiguity exists even for (intrinsically) calibrated cameras
- For calibrated cameras, the similarity ambiguity is the only ambiguity

[Longuet-Higgins '81]

An important class of ambiguities is the similarity ambiguity – that is a reconstruction that is correct up to a similarity transform (rotation, translation and scaling). A reconstruction with only similarity ambiguity is known as a **metric reconstruction**. This ambiguity exists even when the camera are (intrinsically) calibrated. The good news is that for calibrated cameras, the similarity ambiguity is the only ambiguity [Longuet-Higgins '81].

Similarity Ambiguity

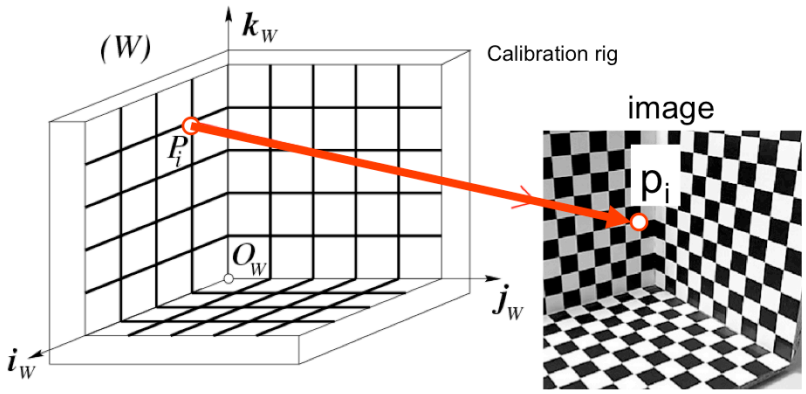
- It is impossible, based on the images alone, to estimate the absolute scale of the scene (i.e. house height)



http://www.robots.ox.ac.uk/~vgg/projects/SingleView/models/hut/hutme.wrl

This is obvious when you think about it – there is no way to recover the absolute scale of a scene from images. An object's scale, absolute position and canonical orientation will always be unknown unless we make further assumptions (e.g, we know the height of the house in the figure) or incorporate more data.

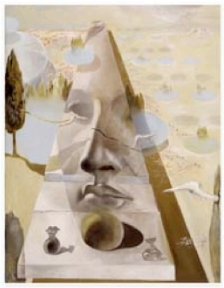
Resolving the similarity ambiguity



While calibrating a camera, we make assumptions about the geometry of the world

For instance, in the camera calibration problem we made the assumption that we know the location of the calibration points with respect to the world reference system (eg. we know the size of the squares of the checker board that we used to build the calibration rig).

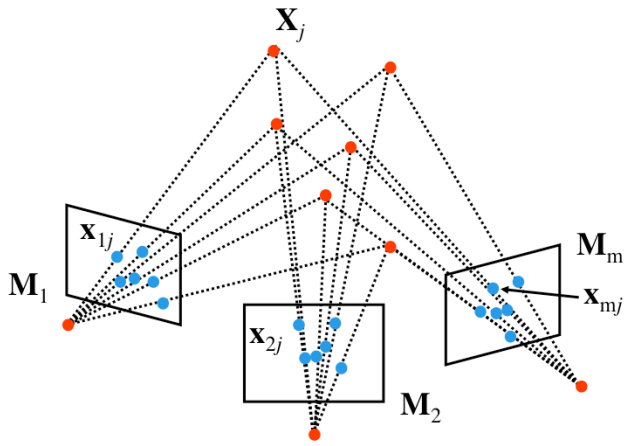
Lecture 7
Multi-view geometry



- The SFM problem
- Affine SFM
- Perspective SFM
- Self-calibration
- Applications

Now we move to the structure from motion problem for perspective cameras.

Structure from motion problem

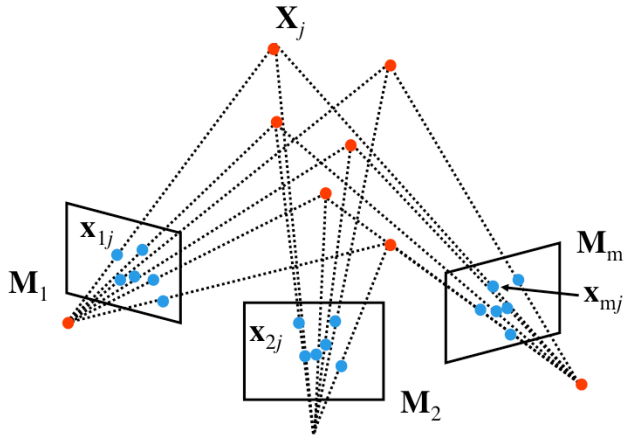


From the mxn observations $\mathbf{x}_{ij}$, estimate:

- *m* projection matrices $\mathbf{M}_i$ = motion
- *n* 3D points $\mathbf{X}_j$ = structure

Let's consider the structure from motion problem for the general case of projective cameras $\mathbf{M}_i$

Structure from motion problem

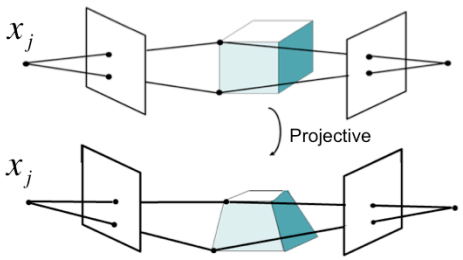


m cameras $M_1 \ldots M_m$

$$M_i = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & 1 \end{bmatrix}$$

In the general case of a structure from motion problem for projective cameras, we need to consider the most general camera matrix $\mathbf{M}_i$, with 11 degrees of freedom (remember the affine camera matrix only had 8). Also, note that $\mathbf{M}_i$ has 11 (and not 12) degrees of freedom as it is defined up to scale.

Structure from Motion Ambiguities



- In the general case (nothing is known) the ambiguity is expressed by an arbitrary affine or projective transformation

$$x_j = M_i X_j \qquad M_i = K_i \begin{bmatrix} R_i & T_i \end{bmatrix}$$

$$H X_j \qquad M_i H^{-1}$$

$$x_j = M_i X_j = \left(M_i H^{-1}\right)\left(H X_j\right)$$

Moreover, similarly to the affine case where the solution can be found up to a affine transformation, solutions for structure and motion can be determined up a projective transformation: we can always arbitrarily transform the motion matrix by a 4x4 transform H, as long as we also transform the structure matrix by the inverse transformation H^{-1}. The resulting observations will still be the same.

The Structure-from-Motion Problem

Given *m* images of *n* fixed points X_j we can write

$$x_{ij} = M_i X_j \qquad \text{for } i = 1, \ldots, m \text{ and } j = 1, \ldots, n$$

N. of cameras N. of points

Problem: estimate m 3×4 matrices M_i and n positions X_j from m×n obvesrvations x_{ij} .

- If the cameras are not calibrated, cameras and points can only be recovered up to a 4x4 projective (16 parameters)
- Given two cameras, how many points are needed?
- How many equations and how many unknown?

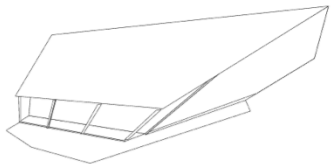
2m × n equations in 11m+3n – 16 unknowns

Similarly for the affine case, we can set up the SFM problem as the one of estimating m 3×4 matrices M_i and n positions X_j from m×n observations x_{ij}.

Notice that, as we discussed in the previous slide, cameras and points can only be recovered up to a 4x4 projective transformation (16 parameters). Therefore we have 11m+3n -16 unknowns in 2m x n equations. From this we can set up a counting argument for the number of views and observations that are required to solve for the unknowns.

Projective Ambiguity

S =

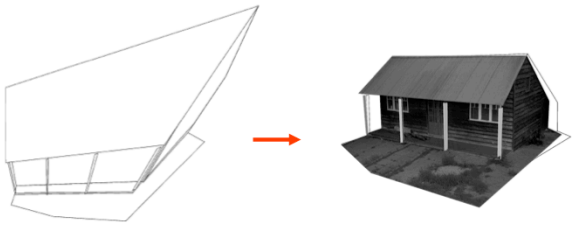


R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd edition, 2003

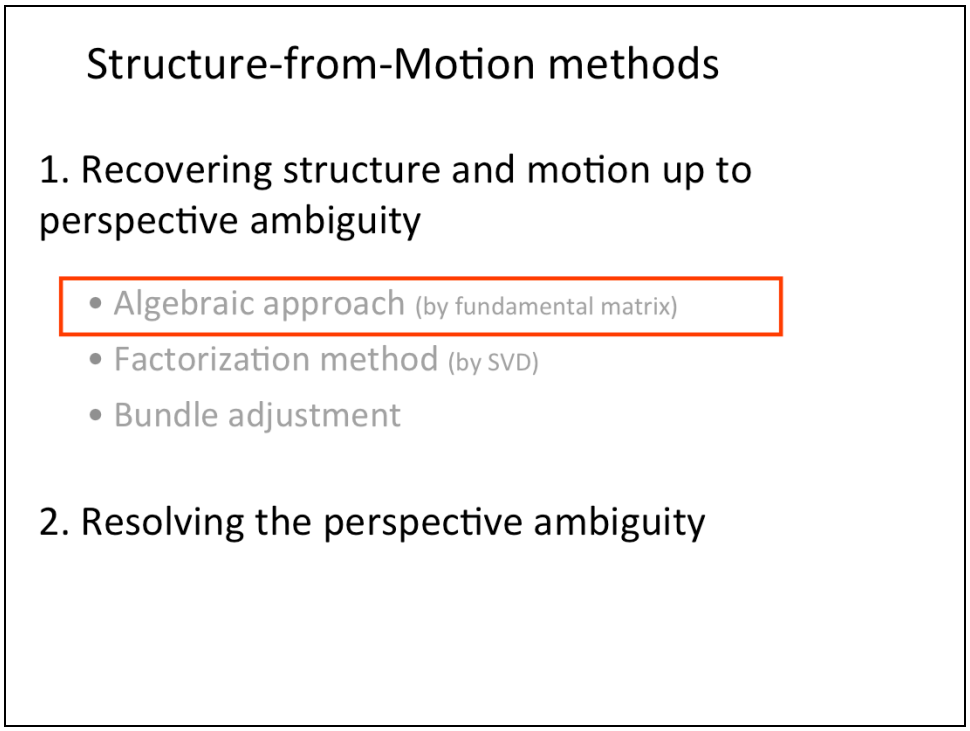
This is an example of a solution of the SFM problem obtained using the two top images. This solution is correct up a projective transformation. The reconstructed S (shown in the figure) and M produce observations which are consistent with the images on top.

Metric reconstruction (upgrade)

- The problem of recovering the metric reconstruction from the perspective one is called self-calibration



The problem of recovering the metric reconstruction from the perspective one is called **self-calibration** (we will discuss this in more details later). It is called metric **upgrade** in that it consists of "upgrading" a reconstruction from perspective to metric.



Thus we can set up a generic SFM problem as the one of: 1) Recovering structure and motion up to perspective ambiguity; 2) Resolving the perspective ambiguity. There are several techniques for step 1:

- Algebraic approach (by fundamental matrix)
- Factorization method (by SVD)
- Bundle adjustment

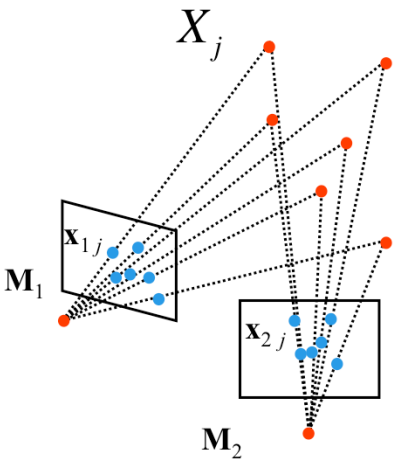
We'll discuss the first and last one in this lecture.

Algebraic approach (2-view case)

1. Compute the fundamental matrix F from two views
2. Use F to estimate projective cameras
3. Use these cameras to triangulate and estimate points in 3D

The algebraic approach leverages the concept of fundamental matrix F for solving the SFM problem for two cameras. This is the outline of the algebraic approach for the 2-camera case. First, we compute the fundamental matrix relating the points in each view. We then use F to estimate the projective camera matrices. Finally, we use these matrices to determine the world coordinates of the points (which is known up to a perspective transformation).

Algebraic approach (2-view case)



$$x_{1j} = M_1 X_j$$
$$x_{2j} = M_2 X_j$$

For j = 1, …,n

N. of points

From at least 8 point correspondences, compute F associated to camera 1 and 2

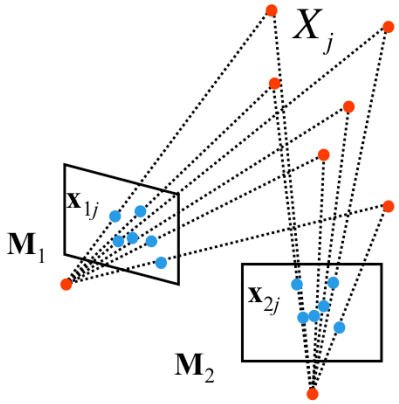
Let's consider a camera pair as in the figure above. Let $\mathbf{M_1}$ and $\mathbf{M_2}$ the camera matrices associated to these two cameras respectively. By assuming that at least 8 correspondences are available, and by following the 8-point algorithm in lecture 5, we estimate F.

Algebraic approach (2-view case)

1. Compute the fundamental matrix F from two views (eg. 8 point algorithm)
2. Use F to estimate projective cameras
3. Use these cameras to triangulate and estimate points in 3D

Let's now focus on step 2.

Algebraic approach (2-view case)



$$x_{1j} = M_1 X_j$$

$$x_{2j} = M_2 X_j$$

For j = 1, …,n

N. of points

Because of the projective ambiguity, we can always apply a projective transformation H such that:

$$\mathrm{M}_1 \mathrm{H}^{-1} = \begin{bmatrix} \mathrm{I} & 0 \end{bmatrix}$$

[Eq. 3]

Canonical perspective camera

$$\mathrm{M}_2 \mathrm{H}^{-1} = \begin{bmatrix} \mathrm{A} & \mathrm{b} \end{bmatrix}$$

[Eq. 4]

Let's consider the camera pair as in the figure above. Since each M_i can only be computed up a perspective transformation H, we can always consider a H such that the first camera projection matrix M_1 H^{-1} is canonical perspective (Eq. 3), that is, it is equal to [**I** 0]. Of course, the same transformation must also be applied to the second camera which lead to the form shown in Eq. 4.

Algebraic approach (2-view case)

- Call **X** a generic 3D point $\mathbf{X}_{ij}$
- Call **x** and **x'** the corresponding observations to camera 1 and respectively

[Eqs. 5]

$$\begin{cases} \tilde{M}_1 = M_1 H^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \\ \tilde{M}_2 = M_2 H^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \end{bmatrix} \\ \tilde{\mathbf{X}} = \mathrm{H}\,\mathbf{X} \end{cases}$$

$$\mathbf{x} = M_1 \mathbf{X} = M_1 H^{-1} H \mathbf{X} = [\mathbf{I}\,|\,\mathbf{0}]\tilde{\mathbf{X}} \quad \text{[Eq. 6]}$$

$$\mathbf{x}' = M_2 \mathbf{X} = M_2 H^{-1} H \mathbf{X} = [\mathbf{A}|\mathbf{b}]\tilde{\mathbf{X}}$$

$$\mathbf{x}' = [\mathbf{A}|\mathbf{b}]\tilde{\mathbf{X}} = [\mathbf{A}\,|\,\mathbf{b}] \begin{bmatrix} \tilde{X}_1 \\ \tilde{X}_2 \\ \tilde{X}_3 \\ 1 \end{bmatrix} = \mathbf{A}[\mathbf{I}|\mathbf{0}] \begin{bmatrix} \tilde{X}_1 \\ \tilde{X}_2 \\ \tilde{X}_3 \\ 1 \end{bmatrix} + \mathbf{b} = \mathbf{A}[\mathbf{I}|\mathbf{0}]\,\tilde{\mathbf{X}} + \mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

?

[Eq. 7]

$$\mathbf{x}' \times \mathbf{b} = (\mathbf{A}\mathbf{x} + \mathbf{b}) \times \mathbf{b} = \mathbf{A}\mathbf{x} \times \mathbf{b} \quad \text{[Eq. 8]}$$

$$\mathbf{x}'^{T} \cdot (\mathbf{x}' \times \mathbf{b}) = \mathbf{x}'^{T} \cdot (\mathbf{A}\mathbf{x} \times \mathbf{b}) = 0 \quad \text{[Eq. 9]}$$

$$\mathbf{x}'^{T} (\mathbf{b} \times \mathbf{A}\mathbf{x}) = 0 \quad \text{[Eq. 10]}$$

To simplify our notation, from this point on, let's call **X** a generic 3D point $\mathbf{X}_{ij}$ and let's call **x** and **x'** the corresponding observations to camera 1 and camera 2, respectively. Since we have applied H^{-1} to both camera projection matrices, we must apply H to the structure, that is, computer H X which leads to the transformed point X^{tilde}. The equation that describes the transformed camera projections matrices and 3D point are summarized in Eqs. 5. Using these expressions we can rewrite the observations x and x' as described in Eq. 6 (by construction, such observations won't change as we apply H to structure and motion). x' can be further manipulated so as to obtain the expression in Eq.7. Notice that we have replaced the expression of $x = [I\ 0]X^{tilde}$ in the last term of the derivation of Eq.7. Using this equation we can write the cross product between x' and b as shown in Eq. 8. Now, by the definition of the cross product, x' cross b is perpendicular to x', so it's dot product with x' is zero (Eq. 9). Therefore, we can determine the constraint between pairs of corresponding points shown in Eq. 10.

Se also, pag 254 HZ.

Cross product as matrix multiplication

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_\times]\mathbf{b}$$

Remember that we can represent the cross product as a matrix multiplication with the skew symmetric matrix shown in the slide. We use the square bracket and subscript x notation to denote this cross product matrix operator.

Algebraic approach (2-view case)

[Eqs. 5]

$$\tilde{M}_1 = M_1 H^{-1} = \begin{bmatrix} I & 0 \end{bmatrix} \qquad \mathbf{x} = M_1 H^{-1} H \mathbf{X} = [\mathbf{I} \mid \mathbf{0}]\tilde{\mathbf{X}}$$

$$\tilde{M}_2 = M_2 H^{-1} = \begin{bmatrix} A & b \end{bmatrix} \qquad \mathbf{x}' = M_2 H^{-1} H \mathbf{X} = [\mathbf{A} \mid \mathbf{b}]\tilde{\mathbf{X}}$$

$$\tilde{\mathbf{X}} = \mathrm{H}\,\mathbf{X}$$

[Eq. 6]

$$\mathbf{x}'^{T}(\mathbf{b} \times \mathbf{A}\mathbf{x}) = 0 \quad \text{[Eq. 10]}$$

$$\mathbf{x}'^{\mathrm{T}}[\mathbf{b}_\times]\mathbf{A}\,\mathbf{x} = 0$$

is this familiar?

$$\mathbf{F} = [\mathbf{b}_\times]\mathbf{A}$$

fundamental matrix!

$$\mathbf{x}'^{\mathrm{T}}\mathrm{F}\,\mathbf{x} = 0$$

Have we seen this constraint before? Yes – It's the fundamental matrix constraint!

Compute cameras

$$\mathbf{x}'^{\mathrm{T}}\mathrm{F}\,\mathbf{x} = 0 \qquad \mathbf{F} = [\mathbf{b}_{\times}]\mathbf{A} = \mathbf{b} \times \mathbf{A}$$ [Eq. 11]

Compute **b**:

- Let's consider the product **F b**

$$\mathbf{F} \cdot \mathbf{b} = [\mathbf{b}_{\times}]\mathbf{A} \cdot \mathbf{b} = \mathbf{b} \times \mathbf{A} \cdot \mathbf{b} = 0$$ [Eq. 12]

- Since **F** is singular, we can compute **b** as least sq. solution of **F b** = 0, with |**b**|=1 using SVD

- Using a similar derivation, we have that $\mathbf{b}^{\mathrm{T}}\,\mathbf{F} = 0$ [Eq. 12-bis]

Using Eq. 11, we can decompose F and estimate A and b. Let's start by computing b. Let's consider the product F b. By using Eq. 11, we can express F b as b × A • b which must be zero (Eq. 12). Since F is singular, b can be computed as a least square solution of F b = 0, with |b|=1, using SVD.

Notice that using a similar derivation, we have that $\mathbf{b}^{\mathrm{T}}\,\mathbf{F} = 0$ (Eq. 12-bis).

Compute cameras

$$\mathbf{x}'^{\mathrm{T}} \mathrm{F}\, \mathbf{x} = 0 \qquad \mathbf{F} = [\mathbf{b}_\times]\mathbf{A} \quad \text{[Eq. 11]} \qquad \begin{cases} \mathbf{F}\,\mathbf{b} = 0 & \text{[Eq. 12]} \\ \mathbf{b}^{\mathrm{T}}\,\mathbf{F} = 0 & \text{[Eq. 12-bis]} \end{cases}$$

Compute $\mathbf{A}$:

- Define: $\mathbf{A'} = -[\mathbf{b}_\times]\,\mathbf{F}$
- Let's verify that $[\mathbf{b}_\times]\mathbf{A}'$ is equal to $\mathbf{F}$:

Indeed: $[\mathbf{b}_\times]\mathbf{A}' = -[\mathbf{b}_\times][\mathbf{b}_\times]\,\mathbf{F} = (\mathbf{b}\,\mathbf{b}^T - |\mathbf{b}|^2\mathbf{I})\,\mathbf{F} = \mathbf{b}\,\mathbf{b}^T\mathbf{F} + |\mathbf{b}|^2\mathbf{F} = 0 + 1\cdot\mathbf{F} = \mathbf{F}$ [Eq. 13]

- Thus, $\mathbf{A} = \mathbf{A'} = -[\mathbf{b}_\times]\,\mathbf{F}$

[Eqs. 14] $\tilde{M}_1 = \begin{bmatrix} I & 0 \end{bmatrix} \qquad \tilde{M}_2 = \begin{bmatrix} -[\mathbf{b}_x]\mathbf{F} & \mathbf{b} \end{bmatrix}$

Once b is known, we can now compute A. Let's define A' = - $[b_x]$ F and verify that A' does satisfy Eq. 11. Indeed, this is demonstrated in a number of steps in Eq. 13. In order to do so we use the property that b^T F = 0 and that |b|=1.

Thus, since F = $[b_x]$ A (Eq 11) and F = $[b_x]$ A' (Eq. 13), we conclude that A=A' = - $[b_x]$ F. From this we have the two expressions for the camera projections matrices in Eqs. 14.

Interpretation of **b**

$$\mathbf{x}'^{\mathrm{T}} \mathrm{F}\, \mathbf{x} = 0 \qquad \mathbf{F} = [\mathbf{b}_{\times}]\mathbf{A}$$

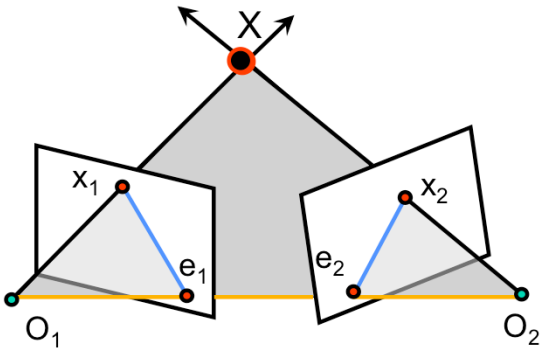
[Eq. 11]

$$\begin{cases} \mathbf{F}\,\mathbf{b} = 0 & \text{[Eq. 12]} \\ \mathbf{b}^{\mathrm{T}}\,\mathbf{F} = 0 & \text{[Eq. 12-bis]} \end{cases}$$

What's **b**??

Before we conclude this section, we want give a geometrical interpretation for b. We know b satisfies Eq. 12.

Epipolar Constraint [lecture 5]



$F x_2$ is the epipolar line associated with x_2 ($l_1 = F x_2$)

$F^T x_1$ is the epipolar line associated with x_1 ($l_2 = F^T x_1$)

F is singular (rank two)

$F e_2 = 0$ and $F^T e_1 = 0$

F is 3x3 matrix; 7 DOF

Remember the epipolar constraints we saw in lecture 5. The epipoles in an image were the points that mapped to zero when transformed by the fundamental matrix.

Interpretation of **b**

$$\mathbf{x}'^{\mathrm{T}} \mathrm{F}\, \mathbf{x} = 0 \qquad \mathbf{F} = [\mathbf{b}_{\times}]\mathbf{A}$$

[Eq. 11]

$$\begin{cases} \mathbf{F}\,\mathbf{b} = 0 \\ \mathbf{b}^{\mathrm{T}}\,\mathbf{F} = 0 \end{cases}$$

b is an epipole!

$$\tilde{M}_1 = \begin{bmatrix} I & 0 \end{bmatrix} \qquad \tilde{M}_2 = \begin{bmatrix} -[\mathbf{b}_x]\mathbf{F} & \mathbf{b} \end{bmatrix}$$

⇩ ⇩

$$\tilde{M}_1 = \begin{bmatrix} I & 0 \end{bmatrix} \qquad \tilde{M}_2 = \begin{bmatrix} -[\mathbf{e}_x]\mathbf{F} & \mathbf{e} \end{bmatrix}$$

[Eq. 15] [Eq. 16]

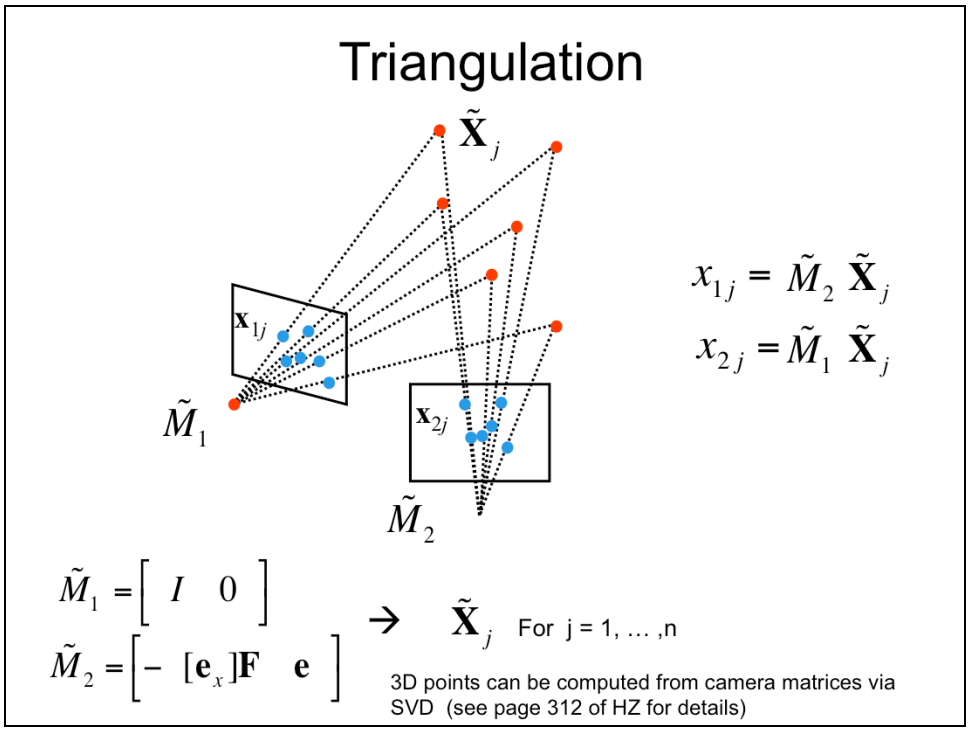
HZ, page 254
PF, page 288

We can see, therefore, that b is an epipole! This provides a new set of equations for the camera projection matrices (Eq. 15 and Eq. 16).

Algebraic approach (2-view case)

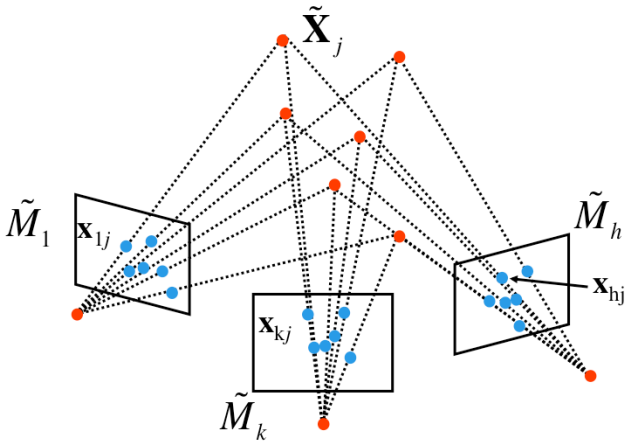
1. Compute the fundamental matrix F from two views (eg. 8 point algorithm)
2. Use F to estimate projective cameras
3. Use these cameras to triangulate and estimate points in 3D

We can now consider the third step of the approach



Using the camera projection matrices M_1^{tilde} and M_2^{tilde} we can compute a points X^{tilde}_j in 3D from any pair of corresponding points in camera 1 and 2. Notice that the estimate of such points will be correct up the perspective transformation H ($X^{tilde}_j = H X_j$). 3D points can be computed from camera matrices via SVD as discussed in more details in page 312 of HZ.

Algebraic approach: the N-views case



- From I_k and I_h $\rightarrow \tilde{M}_k, \tilde{M}_h, \tilde{X}_{[k,h]}$ ← 3D points associated to point correspondences available between I_k and I_h

- Pairwise solutions may be combined together using *bundle adjustment*

The extension to the multi-view case can be done as follows. We can use the algebraic approach to obtain solutions for the camera matrices and the 3D points for any pair of cameras I_k and I_h (provided that there are enough point correspondences to estimate F). The reconstructed 3D points are associated to the point correspondences available between I_k and I_h. Those pairwise solutions may be combined together (optimized) in a approach called *bundle adjustments* as we will see next.

Notice, again, that motion and structure are computed up to a projective transformation.

Structure-from-Motion Algorithms

- Algebraic approach (by fundamental matrix)
- Factorization method (by SVD)
- Bundle adjustment

The Factorization method (by SVD) it's an adaptation from the Tomasi-Kanade method discussed in lecture 6 and follows the same principle. Unfortunately, it makes a strong assumption that relies on having available a rough estimate of distance of the points from the cameras (which may be reasonable in some applications, but not in general).

We'll now look at bundle adjustment method.

Limitations of the approaches so far

- Factorization methods assume all points are visible.

This not true if:

  - occlusions occur
  - failure in establishing correspondences

- Algebraic methods work with 2 views

There are major limitations related to the two methods we have seen so far:

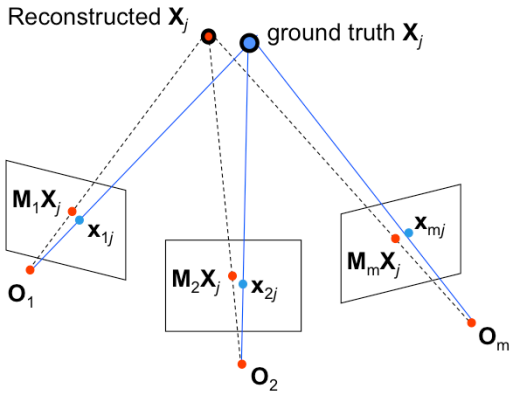
- The factorization method assumes that all points are visible in every image. This very unlikely to happen because of occlusions and failures to find correspondences.
- The algebraic approach produces pairwise solutions but not a coherent optimized reconstruction using all the cameras and 3D points.

The bundle adjustment approach addresses some of these limitations.

Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizes re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



Bundle adjustment is a non-linear method for solving the SFM problem. In the optimization we aim to minimize the re-projection error – the pixel distance between the projection of a reconstructed point into the estimated cameras (i.e., a point $\mathbf{M}_i \mathbf{X}_j$ shown in red) and its corresponding observation (i.e., $\mathbf{x}_{i\ j}$ shown in blue) for all the cameras and for all the points. This optimization problem is very similar to the one we introduced in lecture 5 when we talked about triangulation.

General Calibration Problem

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$

parameters

measurements

D is the nonlinear mapping

- Newton Method
- Levenberg-Marquardt Algorithm
  - Iterative, starts from initial solution
  - May be slow if initial solution far from real solution
  - Estimated solution may be function of the initial solution
  - Newton requires the computation of J, H
  - Levenberg-Marquardt doesn't require the computation of H

As we discussed for the camera calibration problem, a common way to solve this problem is to resort to non-linear optimization techniques. Two common ones include the Newton's method and the Levenberg-Marquardt algorithm. The slide lists some of the advantages and trade-offs of Levenberg-Marquardt. J and H refer to the Jacobian and Hessian, respectively. For more details about optimization methods, please refer to [FP] Sec. 22.2 (page 669-672) or any reference text books.

Bundle adjustment

- Advantages
  - Handle large number of views
  - Handle missing data
- Limitations
  - Large minimization problem (parameters grow with number of views)
  - Requires good initial condition
- Used as the final step of SFM (i.e., after the factorization or algebraic approach)
- Factorization or algebraic approaches provide a initial solution for optimization problem

Bundle adjustment has some important advantages and limitations when compared to the other methods we've surveyed:

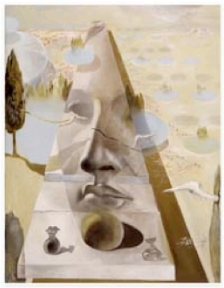
- it handles large number of views
- it handles missing data

However, it also suffers from limitations:

- it's a large minimization problem (parameters grow with number of views)
- it requires good initial condition, which we often find with one of the other methods we've discussed.

For this reason it is often used as final step of SFM (i.e., after the factorization or algebraic approach) in that a factorization or algebraic approach may provide a good initial solution for optimization problem.

Lecture 7
Multi-view geometry

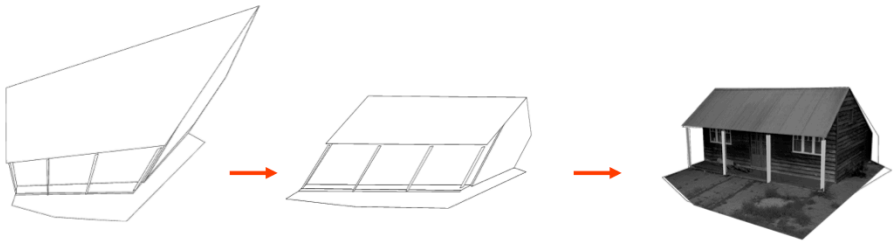


- The SFM problem
- Affine SFM
- Perspective SFM
- Self-calibration
- Applications

Now we move to the self-calibration problem

Self-calibration

- Self-calibration is the problem of recovering the metric reconstruction from the perspective (or affine) reconstruction
- We can self-calibrate the camera by making some assumptions about the cameras



The self-calibration problem is the problem of recovering the metric reconstruction from the perspective (or affine) reconstruction. This problem is solved by assuming that certain knowledge about the cameras is available. For instance, we know the focal lengths or other intrinsic parameters of the cameras; or we know that all the images are acquired with fixed focal length.

Self-calibration

[HZ] Chapters 19 "Auto-calibration"

Several approaches:

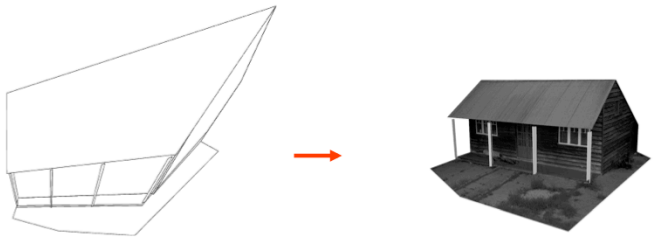
- Use single-view metrology constraints (lecture 4)
- Direct approach (Kruppa Eqs) for 2 views
- Algebraic approach
- Stratified approach

There are several approaches to solve the problem of self-calibration. We have seen in lecture 4, a way of using single-view metrology constraints which can be employed for this problem. There are a few other approaches that we will study in this lecture which include:

- Direct approach (Kruppa Eqs) for 2 views
- Algebraic approach
- Stratified approach

In the appendix we discuss in details the direct approach (Kruppa Eqs) for 2 views.

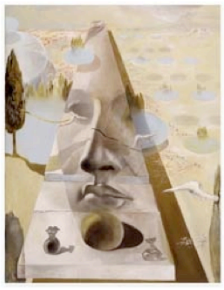
Inject information about the camera during the bundle adjustment optimization



For calibrated cameras, the similarity ambiguity is the only ambiguity [Longuet-Higgins '81]

In practice, in many applications, self-calibration takes places within bundle adjustment by injecting information about the camera. For instance, one common assumption is that the focal length is known, there is no skew, pixels are square and there is no off-set. In this case, the solution is guaranteed to be known up to a similarity transformation as the theorem by Longuet-Higgins '81 states (see slide). Thus, if we run bundle adjustment by making these assumptions on the camera matrices, our solution is a metric reconstruction.

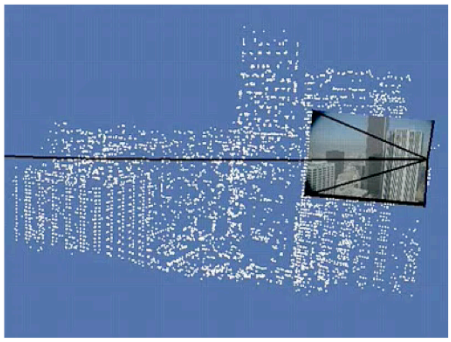
Lecture 7
Multi-view geometry



- The SFM problem
- Affine SFM
- Perspective SFM
- Self-calibration
- Applications

Let's see now some examples and applications

Structure from motion problem



Courtesy of Oxford **Visual Geometry Group**

Lucas & Kanade, 81
Chen & Medioni, 92
Debevec et al., 96
Levoy & Hanrahan, 96
Fitzgibbon & Zisserman, 98
Triggs et al., 99
Pollefeys et al., 99
Kutulakos & Seitz, 99

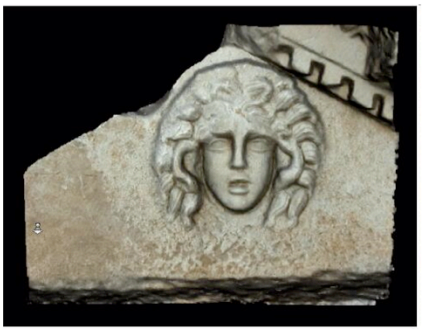
Levoy et al., 00
Hartley & Zisserman, 00
Dellaert et al., 00
Rusinkiewic et al., 02
Nistér, 04
Brown & Lowe, 04
Schindler et al, 04
Lourakis & Argyros, 04
Colombo et al. 05

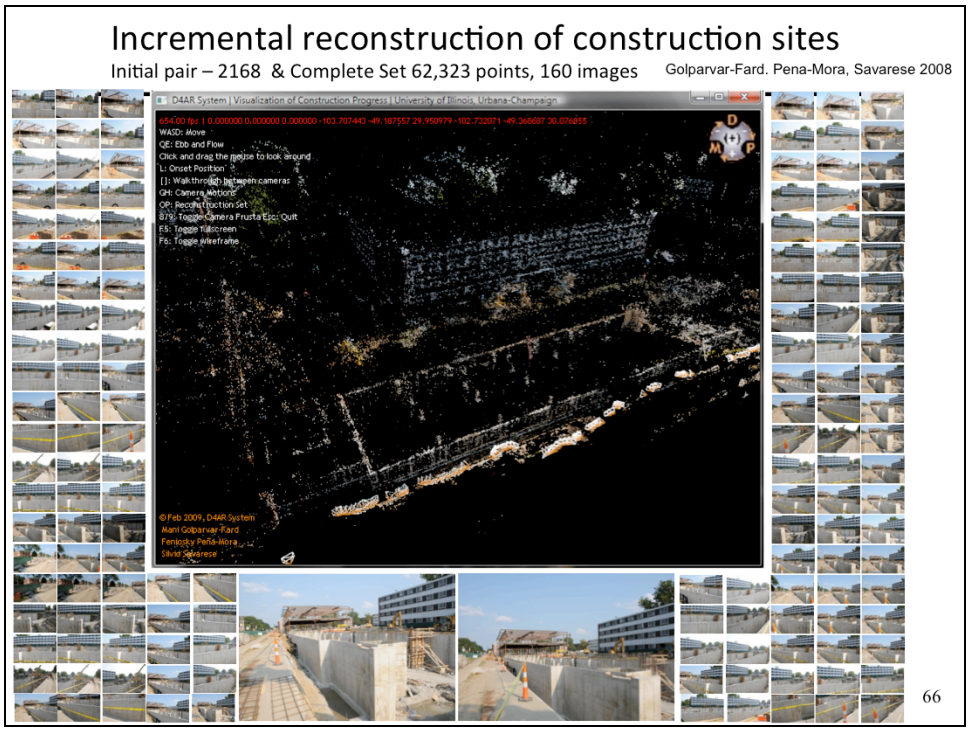
Golparvar-Fard, et al. JAEI 10
Pandey et al. IFAC , 2010
Pandey et al. ICRA 2011
Microsoft's PhotoSynth
Snavely et al., 06-08
Schindler et al., 08
Agarwal et al., 09
Frahm et al., 10

Here is we show some results from some the existing SFM pipelines.

Reconstruction and texture mapping

M. Pollefeys et al 98---

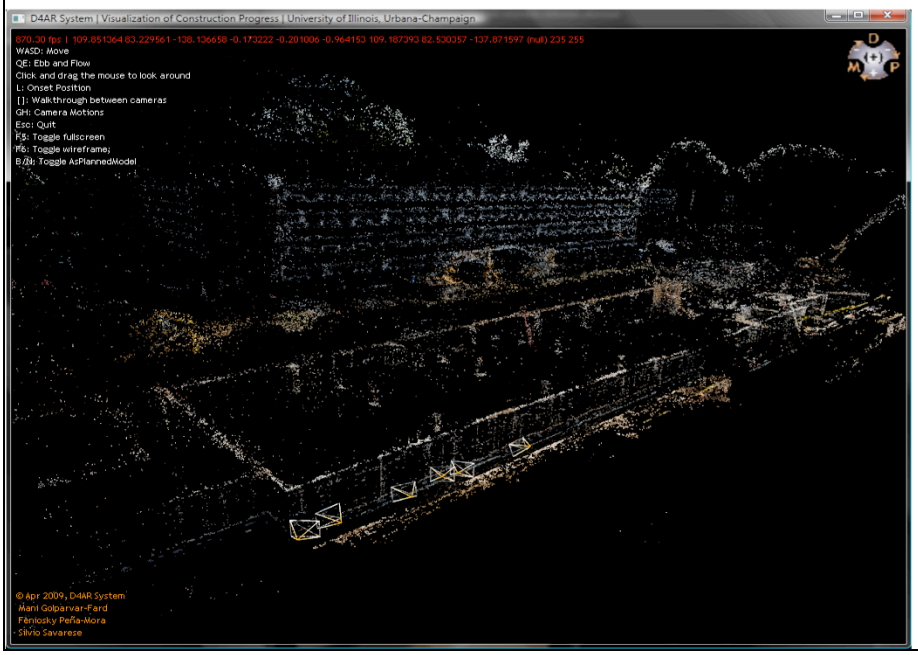




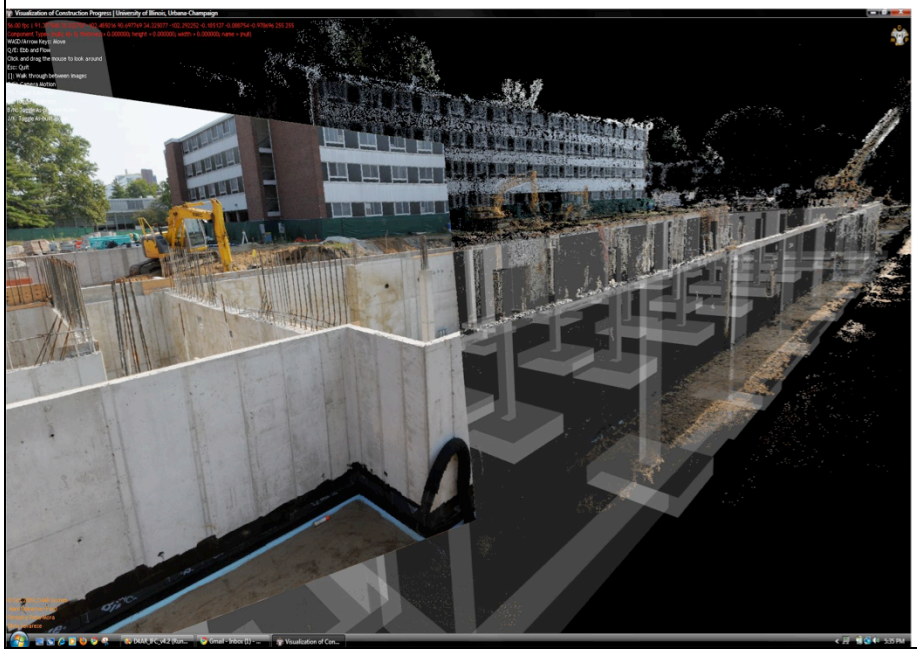
This slide animates the reconstruction. As seen the structure and motion start from 2 cameras and 2168 points and eventually it reaches to the full image dataset (160 images) and 62,323 points.

The photos are all unordered. They are taken for to monitor the progress of the construction, so there's enough photographs to recover the building's shape.

Reconstructed scene + Site photos



Reconstructed scene + Site photos



Results and applications

Noah Snavely, Steven M. Seitz, Richard Szeliski, "Photo tourism: Exploring photo collections in 3D," ACM Transactions on Graphics (SIGGRAPH Proceedings),2006,

One particularly interest application used Flickr images from tourists to reconstruct famous landmarks such as the Coliseum.

Next lecture

- Active Stereo & Volumetric Stereo

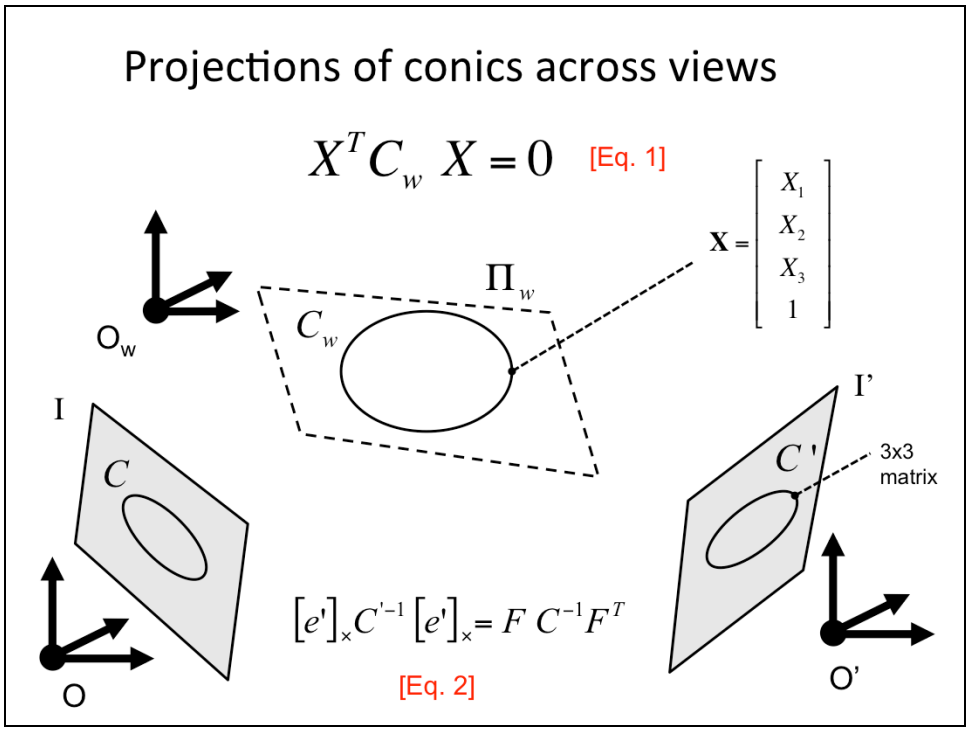
Appendix

Direct approach

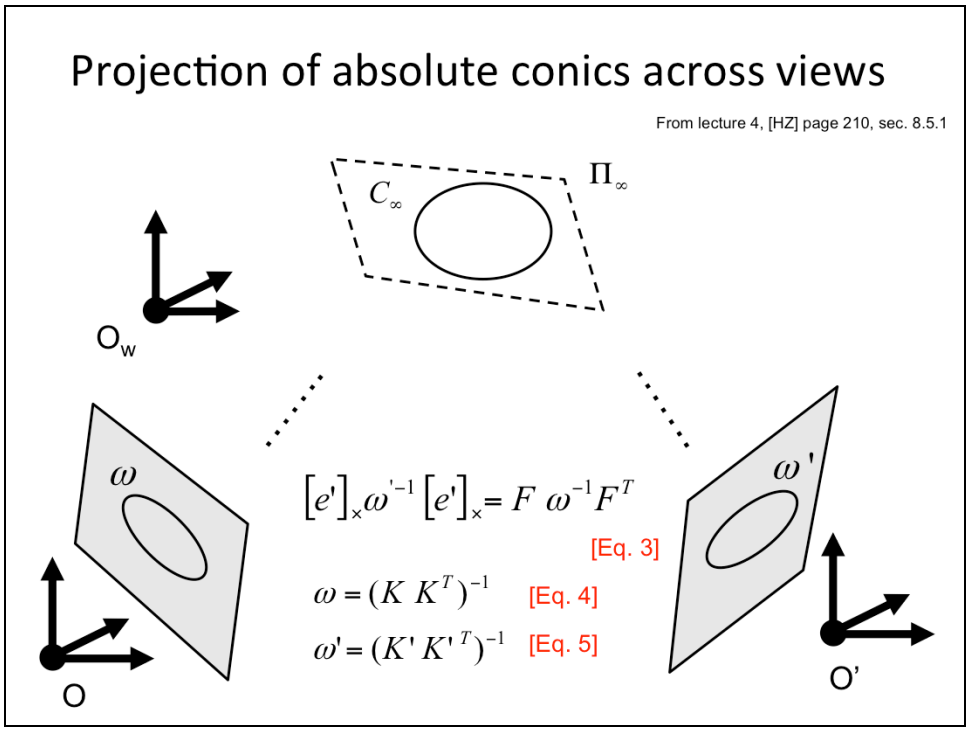
We use the following results:

1. A relationship that maps conics across views
2. Concept of absolute conic and its relationship to K
3. The Kruppa equations

In the direct approach, we use information about how a conic's image is related across different views. We introduce the notion of absolute conic and use its image (image of the absolute conic or IAC). These will lead us to the so call *Kruppa* equations.



Let's start by considering the conic C_w which is described by a matrix C_w, such that points belonging to it satisfy $x^T C_w x = 0$ (which is a general 2nd degree equation that describes a conic) (Eq. 1). X is a point in 3D in homogeneous coordinate systems. C_w is imaged in two cameras (with centers at O and O') as C and C'. Eq.2 describes a relationship between C and C' through the fundamental matrix F, defined for the camera pair I and I', and the epipole e' (in second camera).



Now, remember that an absolute conic is a particular conic in the plane at infinity (lecture 5). In this slide, we consider the absolute conic and its image in the cameras with centers at O and O'. We apply Eq. 2 for the case of the absolute conic to arrive at Equation 3. From Equation 30 in Lecture 4, we have Equation 4 and Equation 5. K and K' are the camera matrices for camera 1 and 2 respectively.

Kruppa equations

[Faugeras et al. 92] From [HZ] page 471

$$\begin{pmatrix} u_2^T K' K'^T u_2 \\ -u_1^T K' K'^T u_2 \\ u_1^T K' K'^T u_1 \end{pmatrix} \times \begin{pmatrix} \sigma_1^2 v_1^T K K^T v_1 \\ \sigma_1 \sigma_2 v_1^T K K^T v_2 \\ \sigma_2^2 v_2^T K K^T v_2 \end{pmatrix} = 0 \qquad \text{[Eq. 6]}$$

where u_i, v_i and σ_i are the columns and singular values of SVD of F

These give us two independent constraints in the elements of K and K'

If we perform a Singular Value Decomposition of F, as $U \Sigma V^T$, we can expand Equation 3 and arrive at Equation 6 (refer to HZ page 471 for the detailed derivation). In this equation, u_1 and u_2 are the left singular vectors of F, while v_1 and v_2 are the right singular vectors, and σ_1 and σ_2 are the singular values. Equation 6 gives us two independent constraints in the elements of K and K' which are called the Kruppa Equations.

Kruppa equations

[Faugeras et al. 92]

$$\begin{pmatrix} u_2^T K' K'^T u_2 \\ -u_1^T K' K'^T u_2 \\ u_1^T K' K'^T u_1 \end{pmatrix} \times \begin{pmatrix} \sigma_1^2 v_1^T K K^T v_1 \\ \sigma_1 \sigma_2 v_1^T K K^T v_2 \\ \sigma_2^2 v_2^T K K^T v_2 \end{pmatrix} = 0$$

$$\frac{u_2^T K K^T u_2}{\sigma_1^2 v_1^T K K^T v_1} = \frac{-u_1^T K K^T u_2}{\sigma_1 \sigma_2 v_1^T K K^T v_2} = \frac{u_1^T K K^T u_1}{\sigma_2^2 v_2^T K K^T v_2} \quad \text{[Eq. 7]}$$

- Let's make the following assumption: $K' = K = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}$ [Eq. 8]

[Eq. 9] $\alpha f^2 + \beta f + \gamma = 0 \longrightarrow f$

In the case where K and K' are equal, the Kruppa equations can be further simplified to the form in Equation 7. Moreover, if the camera matrices K and K' have the form in Equation 8, then the Kruppa equations reduce to a quadratic equation in the focal length 'f', as shown in Equation 9, from which we can compute f – this allows to estimate the intrinsic parameters of the cameras and, thus, remove the projective ambiguity.

Kruppa equations

[Faugeras et al. 92]

- Powerful if we want to self-calibrate 2 cameras with unknown focal length
- Limitations:
  - Work on a camera pair
  - Don't work if R=0

[Eq. 10] $[e']_\times \omega^{-1} [e']_\times = F\ \omega^{-1} F^T$ becomes trivial

Since: $F = [e']_\times$

The Kruppa equations are useful when calibrating two cameras whose focal lengths are the same but are unknown (and all the other parameters are either zero or known). Some of the drawbacks are that this method can be employed only on a pair of cameras. Also, this method will not work if the relative rotation R between the cameras is 0 (pure translation motion between the two cameras). This is because Equation 10 anyway holds trivially.

Self-calibration

[HZ] Chapters 19 "Auto-calibration"

Several approaches:

- Use single-view metrology constraints (lecture 4)
- Direct approach (Kruppa Eqs) for 2 views
- Algebraic approach
- Stratified approach

Let's now consider the algebraic approach.

Algebraic approach Multi-view approach

Suppose we have a projective reconstruction $\{\tilde{M}_i, \tilde{X}_j\}$

Let H be a homography such that:

- First perspective camera is canonical: $\tilde{M}_1 = [\ I \quad 0\ ]$ [Eq. 11]
- i^{th} perspective reconstruction of the camera (known): $\tilde{M}_i = [\ A_i \quad b_i\ ]$ [Eq. 12]

[Eq. 13] $\left(A_i - b_i p^T\right) K_1 K_1^T \left(A_i - b_i p^T\right)^T = K_i K_i^T$ i=2…m

[Eq. 14] $H = \begin{bmatrix} K_1 & 0 \\ -p^T K_1 & 1 \end{bmatrix}$

p is an unknown 3x1 vector

$K_1 \ldots K_m$ are unknown

This approach allows working with more than just two cameras, so it is a multi-view approach. Assume that a perspective solution for the structure from motion problem (i.e. $\{M_i^{tilde}, X_j^{tilde}\}$) is available. In this approach, we first use a homographic transformation H to make one of the cameras canonical (Equation 11), similarly to what we did in the algebraic approach for solving the SFM problem (see previous slides). Thus, the other cameras will have perspective projections of the form in Equation 12. These are known in that they can be estimated by using any of the methods discussed before for the SFM problem.

By using the concept of absolute conic and plane at infinity, it is possible to derive a constraint of the form expressed in Equation 13 (see HZ pages 460-461). In such equation we have A_i and b_i (which are known), the camera matrices K_i for i=2…m (which are unknown), the camera matrix K_1 (of the first camera which is also unknown), and a 3x1 vector p (which is also unknown). It can be shown that p is related to H by Equation 14 (see HZ pages 460-461).

Algebraic approach Multi-view approach

Suppose we have a projective reconstruction

Let H be a homography such that:

- First perspective camera is canonical: $\tilde{M}_1 = [\ I \quad 0\ ]$ [Eq. 11]
- i^{th} perspective reconstruction of the camera (known): $\tilde{M}_i = [\ A_i \quad b_i\ ]$ [Eq. 12]

[Eq. 13] $\left(A_i - b_i p^T\right) K_1 K_1^T \left(A_i - b_i p^T\right)^T = K_i K_i^T$ i=2…m

How many unknowns?
- 3 from p
- 5 m from $K_1 \ldots K_m$

How many equations? 5 independent equations [per view]

Thus, if we consider m cameras, we have in total 5×m + 3 unknowns (3 of which are in p, and 5×m are in K_i matrices for i=1…m). The constraint in Eq. 13 provides 9 equations ($K_i K_i^T$ is a 3x3 matrix). However, because $K_i K_i^T$ is symmetric and defined up to scale (i.e., it has 5 independent elements, see lecture 4), it produces only 5 independent constraints. From this we can set up a counting argument for the number of views that are required to solve for the unknowns.

Once we solve our system, p and K_1 become known and we can compute the homographic transformation H (Eq. 14) which allows to "upgrade" the perspective solution (i.e. $\{M_i^{tilde}, X_j^{tilde}\}$) into a metric one.

Algebraic approach Multi-view approach

Suppose we have a projective reconstruction

Let H be a homography such that:

- First perspective camera is canonical: $\tilde{M}_1 = [\ I \quad 0\ ]$ [Eq. 11]
- i^{th} perspective reconstruction of the camera (known): $\tilde{M}_i = [\ A_i \quad b_i\ ]$ [Eq. 12]

Assume all camera matrices are identical: $K_1 = K_2 \ldots = K_m$

[Eq. 15] $\left(A_i - b_i p^T\right) K\ K^T \left(A_i - b_i p^T\right)^T = K\ K^T$ i=2…m

How many unknowns?
- 3 from p
- 5 from K

How many equations? 5 independent equations [per view]

We need at least 3 views to solve the self-calibration problem

An interesting case is the one when all the camera matrices are identical: $K_1 = K_2 \ldots = K_m = K$ and, thus, Eq. 13 becomes Eq. 15. In this case the we have 8 unknown (3 from p and 5 from K). Thus, we need at least 3 views to solve the self-calibration problem; Indeed the number of equations (5(m-1)) should be greater of equal to the number of unknowns (8).

Algebraic approach

Art of self-calibration:

Use assumptions on Ks to generate enough equations on the unknowns

Condition	*N. Views*
• Constant internal parameters	3
• Aspect ratio and skew known • Focal length and offset vary	4
• Skew =0, all other parameters vary	8

Issue: the larger is the number of view,
the harder is the correspondence problem

Bundle adjustment helps!

This last example shows the general way to go about self-calibration; that is, to use various constraints on the K matrices so that we have enough equations to evaluate each of the unknowns. The table enlists the number of views needed to perform self-calibration for different conditions listed in the first column. Notice, however, as the number of views is increasing, it is difficult to establish correspondences. As we discussed earlier, in this case bundle adjustment can help.

SFM problem - summary

1. Estimate structure and motion up perspective transformation
    1. Algebraic
    2. factorization method
    3. bundle adjustment
2. Convert from perspective to metric (self-calibration)
3. Bundle adjustment

** or **

1. Bundle adjustment with self-calibration constraints

The slide summarizes what we have discussed so far. Notice that it is always recommended to run bundle adjustment at the end of any reconstruction approach (even after the self-calibration step), to make sure that a coherent and globally optimal solution is obtained.

Another common strategy, which avoids the multiple-step process needed to recover a metric reconstruction, is to performing bundle adjustment with added self-calibration constraints.