Recap: A General Point

- Equations of the form
  \[ Ax = 0 \]

- How do we solve them? (always!)
  - Apply SVD
    \[ A = UDV^T = U \begin{bmatrix} d_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_m \end{bmatrix} V^T \]

  - Singular values: square roots of eigenvalues of \( A^T A \).
  - The solution of \( Ax = 0 \) is the nullspace vector of \( A \).
  - This corresponds to the smallest singular vector of \( A \).

Recap: Properties of SVD

- Frobenius norm
  - Generalization of the Euclidean norm to matrices
    \[ \| A \|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\sum_{i=1}^{m} \sigma_i^2} \]

- Partial reconstruction property of SVD
  - Let \( \sigma_i = 1, \ldots, N \) be the singular values of \( A \).
  - Let \( A_p = U_p D_p V_p^T \) be the reconstruction of \( A \) when we set \( \sigma_{p+1}, \ldots, \sigma_N \) to zero.
  - Then \( A_p = U_p D_p V_p^T \) is the best rank-\( p \) approximation of \( A \) in the sense of the Frobenius norm (i.e. the best least-squares approximation).

Recap: Camera Parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - Skew (non-rectangular pixels)
  - Radial distortion

- Extrinsic parameters
  - Rotation \( R \)
  - Translation \( t \) (both relative to world coordinate system)

- Camera projection matrix
  - \( P = K [ R | t ] \)

  - General pinhole camera: 9 DoF
  - CCD Camera with square pixels: 10 DoF
  - General camera: 11 DoF

Recap: Calibrating a Camera

Goal
- Compute intrinsic and extrinsic parameters using observed camera data.

Main idea
- Place "calibration object" with known geometry in the scene
- Get correspondences
- Solve for mapping from scene to image: estimate \( P = P_{int} P_{ext} \)
Recap: Camera Calibration (DLT Algorithm)

\[
\begin{bmatrix}
0^T & X_i^p & -y_iX_i^p \\
X_i^v & 0^T & -x_iX_i^v \\
\vdots & \vdots & \vdots \\
0^T & X_n^p & -y_nX_n^p \\
& X_n^v & 0^T \\
& & -x_nX_n^v \\
\end{bmatrix}
\begin{bmatrix}
P_1 \\
P_2 \\
\vdots \\
P_n \\
\end{bmatrix} = 0 \quad \text{Ap = 0}
\]

- P has 11 degrees of freedom.
- Two linearly independent equations per independent 2D/3D correspondence.
- Solve with SVD (similar to homography estimation).
- Solution corresponds to smallest singular vector.
- 5 ½ correspondences needed for a minimal solution.

Recap: Triangulation - Linear Algebraic Approach

\[
\lambda_1 x_1 = P_1 X \\
\lambda_2 x_2 = P_2 X
\]

- Two independent equations each in terms of three unknown entries of X.
- Stack equations and solve with SVD.
- This approach nicely generalizes to multiple cameras.

Recap: Epipolar Geometry - Calibrated Case

\[
x - \{r \times (Rr')\} = 0 \quad x^T E x' = 0 \quad \text{with} \quad E = [u_i]R
\]

Essential Matrix
(Longuet-Higgins, 1981)

Recap: Epipolar Geometry - Uncalibrated Case

\[
\hat{x} E \hat{x}' = 0 \quad x = \hat{x} \\
x' = K\hat{x}'
\]

Fundamental Matrix
(Faugeras and Luong, 1992)

Recap: The Eight-Point Algorithm

\[
x = (u, v, 1)^T, \quad x' = (u', v', 1)^T
\]

\[
\begin{bmatrix}
F_{11} & F_{12} & F_{13} \\
F_{21} & F_{22} & F_{23} \\
F_{31} & F_{32} & F_{33}
\end{bmatrix}
\begin{bmatrix}
u' \\
v' \\
1
\end{bmatrix} = 0
\]

\[
[xu \, uu' \, uu' \, uu' \, uu' \, uu' \, uu' \, uu' \, uu' \, uu'] F_{13} = 0
\]

Solve using... SVD!

This minimizes:

\[
\sum_{i=1}^{N} (x_i^T F x_i')^2
\]

Problem with the Eight-Point Algorithm

- In practice, this often looks as follows:

\[
[F_1] =
\begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

Slide adapted from Svetlana Lazebnik
In practice, this often looks as follows:

- Archival videos
  - Enforce the rank
  - Find interest points (e.g. Harris corners)
- Procedure
  - Use the eight
- So, where to start with
  - Center the image data at the origin, and scale it so the
  - Find interest points in both images
  - Compute correspondences
  - Main idea:
    - Transform fundamental matrix back to original units: if
    - Want to estimate world geometry without requiring
    - Refine

Example from Andrew

\[
\begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} = \begin{bmatrix} u_0 & u_1 & u_2 & \cdots \end{bmatrix} F \begin{bmatrix} u_0 & u_1 & u_2 & \cdots \end{bmatrix}^T
\]

\[
U \Sigma V^T = \begin{bmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & U_{32} & U_{33} \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{bmatrix}
\]

Set \( d_3 \) to zero and reconstruct \( F \)

3. Enforce the rank-2 constraint using SVD.

\[
\begin{bmatrix} d_{11} & d_{12} & \cdots \\ d_{21} & d_{22} & \cdots \\ d_{31} & d_{32} & \cdots \end{bmatrix} = U \Sigma V^T
\]

4. Transform fundamental matrix back to original units: if \( T \) and \( T' \) are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is \( T^T F T' \).

3D Reconstruction with Weak Calibration

- Want to estimate world geometry without requiring calibrated cameras.
- Many applications:
  - Archival videos
  - Photos from multiple unrelated users
  - Dynamic camera system
- Main idea:
  - Estimate epipolar geometry from a (redundant) set of point correspondences between two uncalibrated cameras.

Stereo Pipeline with Weak Calibration

- So, where to start with uncalibrated cameras?
  - Need to find fundamental matrix \( F \) and the correspondences (pairs of points \((u',v') \leftrightarrow (u,v))\).
- Procedure
  1. Find interest points in both images
  2. Compute correspondences
  3. Compute epipolar geometry
  4. Refine
Stereo Pipeline with Weak Calibration

2. Match points using only proximity

RANSAC for Robust Estimation of \( F \)

- Select random sample of correspondences
- Compute \( F \) using them
  - This determines epipolar constraint
- Evaluate amount of support - number of inliers within threshold distance of epipolar line
- Choose \( F \) with most support (#inliers)

Putative Matches based on Correlation Search

- Many wrong matches (10-50%), but enough to compute \( F \)

Pruned Matches

- Correspondences consistent with epipolar geometry

Resulting Epipolar Geometry

- Many wrong matches (10-50%), but enough to compute \( F \)
Epipolar Transfer

- Assume the epipolar geometry is known
- Given projections of the same point in two images, how can we compute the projection of that point in a third image?

![Epipolar Transfer Diagram](image1)

Extension: Epipolar Transfer

- Assume the epipolar geometry is known
- Given projections of the same point in two images, how can we compute the projection of that point in a third image?

![Extension: Epipolar Transfer Diagram](image2)

When does epipolar transfer fail?

Topics of This Lecture

- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Structure from Motion

- Given: $m$ images of $n$ fixed 3D points
- Problem: estimate $m$ projection matrices $P_i$ and $n$ 3D points $X_j$ from the $mn$ correspondences $x_{ij}$

![Structure from Motion Diagram](image3)

What Can We Use This For?

- E.g. movie special effects

![What Can We Use This For? Image](image4)

Video

Structure from Motion Ambiguity

- If we scale the entire scene by some factor $k$ and, at the same time, scale the camera matrices by the factor of $1/k$, the projections of the scene points in the image remain exactly the same:

$$x = PX = \left(\frac{1}{k}P\right)(kX)$$

⇒ It is impossible to recover the absolute scale of the scene!
Structure from Motion Ambiguity

- If we scale the entire scene by some factor \( k \) and, at the same time, scale the camera matrices by the factor of \( 1/k \), the projections of the scene points in the image remain exactly the same.
- More generally: if we transform the scene using a transformation \( Q \) and apply the inverse transformation to the camera matrices, then the images do not change.

\[
x = PX = (PQ^{-1})QX
\]

Reconstruction Ambiguity: Similarity

\[
x = PX = (PQ^{-1})_S Q_S X
\]

Reconstruction Ambiguity: Affine

\[
x = PX = (PQ^{-1}_A)Q_A X
\]

Reconstruction Ambiguity: Projective

\[
x = PX = (PQ^{-1}_P)Q_P X
\]

Projective Ambiguity

From Projective to Affine
From Affine to Similarity

Topics of This Lecture
- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Structure from Motion
- Let's start with affine cameras (the math is easier)

Orthographic Projection
- Special case of perspective projection
  - Distance from center of projection to image plane is infinite

Affine Cameras
Affine Cameras

- A general affine camera combines the effects of an affine transformation of the 3D space, orthographic projection, and an affine transformation of the image:

\[
P = \begin{bmatrix} 3 \times 3 \text{ affine} \\ \text{affine} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{bmatrix} = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}
\]

- Affine projection is a linear mapping + translation in inhomogeneous coordinates

\[
x = \begin{bmatrix} x \\ y \\ Z \end{bmatrix} \rightarrow \hat{x} = \begin{bmatrix} x \\ y \end{bmatrix} = AX + b
\]

Projection of world origin

Affine Structure from Motion

- Centering: subtract the centroid of the image points

\[
\hat{x}_i = x_i - \frac{1}{n} \sum x_i = A X_i + b - \frac{1}{n} \sum (A X_i + b) = A \hat{X}_i
\]

- For simplicity, assume that the origin of the world coordinate system is at the centroid of the 3D points.

- After centering, each normalized point \( x_{ij} \) is related to the 3D point \( X_i \) by

\[
\hat{x}_{ij} = \hat{A}_i X_j
\]

Affine Structure from Motion

- Let’s create a 2m x n data (measurement) matrix:

\[
\begin{bmatrix}
\hat{x}_{11} & \hat{x}_{12} & \cdots & \hat{x}_{1n} \\
\hat{x}_{21} & \hat{x}_{22} & \cdots & \hat{x}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\hat{x}_{m1} & \hat{x}_{m2} & \cdots & \hat{x}_{mn}
\end{bmatrix} = \begin{bmatrix}
A_1 \\
A_2 \\
\vdots \\
A_m
\end{bmatrix} \begin{bmatrix}
X_1 \\
X_2 \\
\vdots \\
X_n
\end{bmatrix}
\]

Points (3 x n)

Cameras (2m x 3)

- The measurement matrix \( D = MS \) must have rank 3!
Factorizing the Measurement Matrix

- Singular value decomposition of $D$:

$$D = U \times W \times V^T$$

- Obtaining a factorization from SVD:

$$D = U_3 \times W_3 \times V_3^T$$

Affine Ambiguity

- The decomposition is not unique. We get the same $D$ by using any $3 \times 3$ matrix $C$ and applying the transformations $M \rightarrow MC$, $S \rightarrow CS$.

- That is because we have only an affine transformation and we have not enforced any Euclidean constraints (like forcing the image axes to be perpendicular, for example). We need a *Euclidean upgrade*.

Estimating the Euclidean Upgrade

- Orthographic assumption: image axes are perpendicular and scale is 1.

$$a_1 \cdot a_2 = 0$$

$$|a_1|^2 = |a_2|^2 = 1$$

- This can be converted into a system of 3 $m$ equations:

$$a_i \cdot a_j = 0$$

$$|a_i|^2 = |a_j|^2$$

for the transformation matrix $C$. Goal: estimate $C$.

Possible decomposition:

$$M = U_1 W_1^T \quad S = W_1^T V_1^T$$

This decomposition minimizes $|D - MS|^2$. 

To reduce to rank 3, we just need to set all the singular values to 0 except for the first 3.
Estimating the Euclidean Upgrade

- System of 3m equations:
  \[
  \begin{align*}
  a_1^T a_1 &= 0 \\
  a_1^T a_2 &= 0 \\
  \end{align*}
  \]
  \[
  \begin{align*}
  a_2^T a_1 &= 1, & i = 1, \ldots, m \\
  a_2^T a_2 &= 1
  \end{align*}
  \]
- Let
  \[
  L = CC^T
  \]
- Then this translates to 3m equations in L:
  \[
  A_i L_i^T = I, \quad i = 1, \ldots, m
  \]
  \[
  \begin{align*}
  \text{Solve for } L \\
  \text{Recover } C \text{ from } L \text{ by Cholesky decomposition: } L = CC^T \\
  \text{Update } M \text{ and } S: \quad M = MC, \quad S = C^{-1} S
  \end{align*}
  \]

Algorithm Summary

- Given: m images and n features \( x_{ij} \)
- For each image \( i \), center the feature coordinates.
- Construct a \( 2m \times n \) measurement matrix \( D \):
  - Column \( j \) contains the projection of point \( j \) in all views
  - Row \( i \) contains one coordinate of the projections of all the \( n \) points in image \( i \)
- Factorize \( D \):
  - Compute SVD: \( D = U W V^T \)
  - Create \( U_3 \) by taking the first 3 columns of \( U \)
  - Create \( V_3 \) by taking the first 3 columns of \( V \)
  - Create \( W_3 \) by taking the upper left \( 3 \times 3 \) block of \( W \)
  - Create the motion and shape matrices:
    \[
    M = U_3 W_3^{1/2} \quad \text{and} \quad S = W_3^{1/2} V_3^T
    \]
    \[
    \text{or } M = U_3 \quad \text{and} \quad S = W_3 V_3^T
    \]
  - Eliminate affine ambiguity

Reconstruction Results

Dealing with Missing Data

- So far, we have assumed that all points are visible in all views
- In reality, the measurement matrix typically looks something like this:
- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete
  - Equivalent to finding maximal cliques in a graph
- Incremental bilinear refinement
- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete
  - Equivalent to finding maximal cliques in a graph
- Incremental bilinear refinement
Dealing with Missing Data

- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete (equivalent to finding maximal cliques in a graph)
- Incremental bilinear refinement
  - Perform factorization on a dense sub-block
  - Solve for a new 3D point visible by at least two known cameras (linear least squares)
  - Solve for a new camera that sees at least three known 3D points (linear least squares)


Comments: Affine SfM

- Affine SfM was historically developed first.
- It is valid under the assumption of affine cameras.
  - Which does not hold for real physical cameras...
  - ...but which is still tolerable if the scene points are far away from the camera.
- For good results with real cameras, we typically need projective SfM.
  - Harder problem, more ambiguity
  - Math is a bit more involved...
  - (Here, only basic ideas. If you want to implement it, please look at the H&Z book for details).

Topics of This Lecture

- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Slides adapted from Svetlana Lazebnik

Projective Structure from Motion

- Given: $m$ images of $n$ fixed 3D points
  - $x_0 = P_i X_j$, $i = 1, \ldots, m$, $j = 1, \ldots, n$
- Problem: estimate $m$ projection matrices $P_i$ and $n$ 3D points $X_j$ from the $mn$ correspondences $x_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation $Q$:
  - $X \leftarrow QX$, $P \leftarrow PQ^{-1}$
  - We can solve for structure and motion when $2mn > 11m + 3n - 15$
  - For two cameras, at least 7 points are needed.

Projective SfM: Two-Camera Case

- Assume fundamental matrix $F$ between the two views
  - First camera matrix: $[I\,0]Q^t$
  - Second camera matrix: $[A\,b]Q^t$
- Let $\bar{X} = QX$, then $z x = [I\,0]\bar{X}$, $z' x' = [A\,b]\bar{X}$
- And $z' x' = [A\,b] z x + b = [A\,b] x + b$
- So we have $x^T [b\,A] b = 0$.
- $F = [b\,A]$ b: epipole ($F^T b = 0$), $A = [-b\,F]$. 
Projective SfM: Two-Camera Case

- This means that if we can compute the fundamental matrix between two cameras, we can directly estimate the two projection matrices from $F$.
- Once we have the projection matrices, we can compute the 3D position of any point $X$ by triangulation.
- How can we obtain both kinds of information at the same time?

Projective Factorization

- If we knew the depths $z$, we could factorize $D$ to estimate $M$ and $S$.
- If we knew $M$ and $S$, we could solve for $z$.
- Solution: iterative approach (alternate between above two steps).

Sequential Structure from Motion

- Initialize motion from two images using fundamental matrix
- Initialize structure
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image - calibration

Sequential Structure from Motion

- Initialize motion from two images using fundamental matrix
- Initialize structure
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image - calibration
  - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera - triangulation
- Refine structure and motion: bundle adjustment

Bundle Adjustment

- Non-linear method for refining structure and motion
- Minimizing mean-square reprojection error

$E(P, X) = \sum_{j=1}^{m} \sum_{j=1}^{n} D(x_j, P X_j)$
**Bundle Adjustment**

- Seeks the Maximum Likelihood (ML) solution assuming the measurement noise is Gaussian.
- It involves adjusting the bundle of rays between each camera center and the set of 3D points.
- Bundle adjustment should generally be used as the final step of any multi-view reconstruction algorithm.
  - Considerably improves the results.
  - Allows assignment of individual covariances to each measurement.
- However...
  - It needs a good initialization.
  - It can become an extremely large minimization problem.
- Very efficient algorithms available.

**Projective Ambiguity**

- If we don’t know anything about the camera or the scene, the best we can get with this is a reconstruction up to a projective ambiguity \( Q \).
  - This can already be useful.
  - E.g. we can answer questions like “at what point does a line intersect a plane?”
- If we want to convert this to a “true” reconstruction, we need a **Euclidean upgrade**.
  - Need to put in additional knowledge about the camera (calibration) or about the scene (e.g. from markers).
  - Several methods available (see F&P Chapter 13.5 or H&Z Chapter 19)

**Self-Calibration**

- Self-calibration (auto-calibration) is the process of determining intrinsic camera parameters directly from uncalibrated images.
- For example, when the images are acquired by a single moving camera, we can use the constraint that the intrinsic parameter matrix remains fixed for all the images.
  - Compute initial projective reconstruction and find 3D projective transformation matrix \( Q \) such that all camera matrices are in the form \( P_i = K [R_i | t_i] \).
- Can use constraints on the form of the calibration matrix: square pixels, zero skew, fixed focal length, etc.

**Practical Considerations (1)**

1. Role of the baseline
   - Small baseline: large depth error
   - Large baseline: difficult search problem
2. Solution
   - Track features between frames until baseline is sufficient.

**Practical Considerations (2)**

2. There will still be many outliers
   - Incorrect feature matches
   - Moving objects
   - Apply RANSAC to get robust estimates based on the inlier points.
3. Estimation quality depends on the point configuration
   - Points that are close together in the image produce less stable solutions.
   - Subdivide image into a grid and try to extract about the same number of features per grid cell.

**General Guidelines**

- Use calibrated cameras wherever possible.
  - It makes life so much easier, especially for SfM.
- SfM with 2 cameras is **far** more robust than with a single camera.
  - Triangulate feature points in 3D using stereo.
  - Perform 2D-3D matching to recover the motion.
  - More robust to loss of scale (main problem of 1-camera SfM).
- Any constraint on the setup can be useful
  - E.g. square pixels, zero skew, fixed focal length in each camera
  - E.g. fixed baseline in stereo SfM setup
  - E.g. constrained camera motion on a ground plane
  - Making best use of those constraints may require adapting the algorithms (some known results are described in H&BZ).
Structure-from-Motion: Limitations

- Very difficult to reliably estimate metric SfM unless
  - Large (x or y) motion  
  - Large field-of-view and depth variation
- Camera calibration important for Euclidean reconstruction
- Need good feature tracker

Commercial Software Packages

- boujou  
  (http://www.2d3.com/)
- PFTrack  
  (http://www.thepixelfarm.co.uk/)
- MatchMover  
  (http://www.realviz.com/)
- SynthEyes  
  (http://www.ssontech.com/)
- Icarus  
  (http://aig.cs.man.ac.uk/research/reveal/icarus/)
- Voodoo Camera Tracker  
  (http://www.digilab.uni-hannover.de/)

Applications: Matchmoving

- Putting virtual objects into real-world videos

Applications: Large-Scale SfM from Flickr

References and Further Reading

• A (relatively short) treatment of affine and projective SfM and the basic ideas and algorithms can be found in Chapters 12 and 13 of

  D. Forsyth, J. Ponce,
  Computer Vision - A Modern Approach.
  Prentice Hall, 2003

• More detailed information (if you really want to implement this) and better explanations can be found in Chapters 10, 18 (factorization) and 19 (self-calibration) of

  R. Hartley, A. Zisserman
  Multiple View Geometry in Computer Vision
  2nd Ed., Cambridge Univ. Press, 2004