Recap: A General Point

- Equations of the form
  \[ A x = 0 \]
- How do we solve them? (always!)
  - Apply SVD
    \[ A = U D V^T = \begin{bmatrix} d_1 & \cdots & d_m \end{bmatrix} \begin{bmatrix} \alpha_1 & \cdots & \alpha_p \end{bmatrix} \begin{bmatrix} v_1 & \cdots & v_p \end{bmatrix} \]
    - Singular values \( \alpha_i \) of \( A \)
    - Singular vectors \( v_i \) of \( A \)
    - In the sense of the Frobenius norm
      \[ \| A \|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\sum_{i=1}^{p} \sigma_i^2} \]
- Partial reconstruction property of SVD
  - Let \( \alpha_i = \sigma_i e_i \) be the singular values of \( A \).
  - Let \( A_p = U_p D_p V_p^T \) be the reconstruction of \( A \) when we set \( \sigma_{p+1}, \ldots, \sigma_p \) 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
  \[ P = K [ R \mid t ] \]
  - 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 \( P = P_{\text{int}} P_{\text{ext}} \)
Recap: Camera Calibration (DLT Algorithm)

\[
\begin{bmatrix}
0^T & X_1^T - y_1X_1^T \\
X_1^T & 0^T - x_1X_1^T (P_1) \\
\vdots & \vdots \\
0^T & X_1^T - y_2X_2^T (P_2) \\
X_2^T & 0^T - x_2X_2^T
\end{bmatrix}
\]

- 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 \( \frac{1}{2} \) correspondences needed for a minimal solution.

Recap: Triangulation – Linear Algebraic Approach

\[
\lambda_i \mathbf{x}_i = P_i \mathbf{x}, \quad \mathbf{x}_1 \times P_1 \mathbf{x} = 0, \quad [\mathbf{x}_1]P_1 \mathbf{x} = 0
\]

- Two independent equations each in terms of three unknown entries of \( \mathbf{x} \).
- Stack equations and solve with SVD.
- This approach nicely generalizes to multiple cameras.

Recap: Epipolar Geometry - Calibrated Case

\[
x \cdot [t \times (R'x')] = 0 \quad \Rightarrow \quad x^T E' x' = 0 \\
E = [t_1]R
\]

\[\text{Essential Matrix} \] (Longuet-Higgins, 1981)

Recap: Epipolar Geometry - Uncalibrated Case

\[
\tilde{x}' \cdot E \tilde{x}' = 0 \quad \Rightarrow \quad x^T F' x' = 0 \\
F = K^{-1} E K^{-1}
\]

\[\text{Fundamental Matrix} \] (Faugeras and Luong, 1992)

Recap: The Eight-Point Algorithm

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

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

\[
\begin{bmatrix}
u_1
\end{bmatrix}
\begin{bmatrix}
u_2
\end{bmatrix}
\begin{bmatrix}
u_3
\end{bmatrix}
\]

This minimizes:

\[
\sum_{i=0}^{N} (x_i^T F x_i')^2
\]
Recap: Normalized Eight-Point Algorithm

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute $F$ from the normalized points.
3. Enforce the rank-2 constraint using SVD,
   $$ F = UDV^T \begin{bmatrix} d_{11} & \cdots & d_{13} \\ \vdots & \ddots & \vdots \\ d_{31} & \cdots & d_{33} \end{bmatrix} $$
   Set $d_{33}$ to zero and reconstruct $F$.
4. Transform fundamental matrix back to original units: if $T$ and $T'$ are the normalizing transformations in the two images, the fundamental matrix in original coordinates is $T' FT$.

Active Stereo with Structured Light

- Idea: Project “structured” light patterns onto the object
  - simplifies the correspondence problem
  - Allows us to use only one camera

Recall: Optical Triangulation

- Principle: 3D point given by intersection of two rays.
  - Crucial information: point correspondence
  - Most expensive and error-prone step in the pipeline

Microsoft Kinect - How Does It Work?

- Built-in IR projector
- IR camera for depth
- Regular camera for color

Recall: Optical Triangulation
Laser Scanning

- Optical triangulation
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning

Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich

Laser Scanned Models

The Digital Michelangelo Project, Levoy et al.
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
  \[ x_i = P_iX, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n \]

• Problem: estimate \( m \) projection matrices \( P_i \) and \( n \) 3D points \( X \) from the \( mn \) correspondences \( x_{ij} \)

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!

What Can We Use This For?

• E.g. movie special effects

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.

- 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**

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

---

**Reconstruction Ambiguity: Affine**

\[
x = PX = (PQ_A^{-1})QA_X
\]

---

**Reconstruction Ambiguity: Projective**

\[
x = PX = (PQ_P^{-1})QPX
\]

---

**Projective Ambiguity**

---

**From Projective to Affine**

---
From Affine to Similarity

Topics of This Lecture
- Structure from Motion (SfM)
  - Reconstruction
  - 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

Hierarchies of 3D Transformations
- Projective
  - 15 dof
  - Preserves intersection and tangency
- Affine
  - 12 dof
  - Preserves parallelism, volume ratios
- Similarity
  - 7 dof
  - Preserves angles, ratios of length
- Euclidean
  - 6 dof
  - Preserves angles, lengths
- With no constraints on the camera calibration matrix or on the scene, we get a projective reconstruction.
- Need additional information to upgrade the reconstruction to affine, similarity, or Euclidean.

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
  - Projection matrix:
    \[
    \begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
    \end{bmatrix}
    \begin{bmatrix}
    x \\
    y \\
    z \\
    1
    \end{bmatrix} =
    \begin{bmatrix}
    x \\
    y \\
    1
    \end{bmatrix} \Rightarrow (x, y)
    \]

Affine Cameras
- Orthographic Projection
- Parallel Projection
## 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:

\[
P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

### Affine Projection

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

### Affine Structure from Motion

- Given: \( m \) images of \( n \) fixed 3D points:
  - \( x_i = A_i X_i + b_i \), \( i = 1, \ldots, m \)
- Problem: use the \( mn \) correspondences \( x_{ij} \) to estimate \( m \) projection matrices \( A_i \) and translation vectors \( b_i \)
- And \( n \) points \( X_j \)
- The reconstruction is defined up to an arbitrary affine transformation \( Q \) (12 degrees of freedom):

\[
\begin{bmatrix} A \ & b \\ 0 \ & 1 \end{bmatrix} Q^{-1}, \quad X \rightarrow Q X
\]

- We have \( 2mn \) knowns and \( 8m + 3n \) unknowns (minus 12 dof for affine ambiguity).

- Thus, we must have \( 2mn \gg 8m + 3n \cdot 12 \).
- For two views, we need four point correspondences.

## Affine Structure from Motion

- Centering: subtract the centroid of the image points

\[
\hat{x}_i = x_i - \frac{1}{n} \sum_{k=1}^{n} x_k = A X_i + b - \frac{1}{n} \sum_{k=1}^{n} (A X_k + b)
\]

\[
\hat{X}_i = A X_i
\]

- 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 \( X_j \) by

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

### Affine Structure from Motion

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

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

- The measurement matrix \( D = MS \) must have rank 3!
Perceptual and Sensory Augmented Computing

Computer Vision WS 11/12

Factorizing the Measurement Matrix

• Singular value decomposition of D:
  \[ D = U \Sigma V^T \]

Affine Ambiguity

• The decomposition is not unique. We get the same D by using any 3x3 matrix C and applying the transformations 
  \( M \rightarrow MC \), \( S \rightarrow C^{-1}S \).

• 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 3m equations:

  \[ \begin{cases} a_1 \cdot a_2 = 0 \\ a_1 \cdot C a_2 = 0 \\ a_1 \cdot C a_2 = 1 \end{cases} \]
  \[ \begin{cases} a_1 \cdot a_2 = 0 \\ a_1 \cdot C a_2 = 0 \\ a_1 \cdot C a_2 = 1 \end{cases} \]

  for the transformation matrix \( C \) goal: estimate \( C \)
**Estimating the Euclidean Upgrade**

- System of 3m equations:
  \[
  \begin{align*}
  \dot{a}_i \cdot \dot{a}_j &= 0 \\
  |a_i|^2 &= 1 \\
  \dot{a}_i \cdot \dot{a}_j &= 1, \\
  i &= 1, \ldots, m
  \end{align*}
  \]
- Let \(L = CC^T\), \(A_i = \frac{a_i}{|a_i|^2}\), \(i = 1, \ldots, m\)
- Then this translates to 3m equations in \(L\)
  \[
  A_i L A_i^T = I, \quad i = 1, \ldots, m
  \]
  - Solve for \(L\)
  - Recover \(C\) from \(L\) by Cholesky decomposition: \(L = CC^T\)
  - Update \(M\) and \(S\): \(M = MC\), \(S = C^{-1}S\)

**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 \(U_i\) by taking the first 3 columns of \(U\)
  - Create \(V_i\) by taking the first 3 columns of \(V\)
  - Create \(W_i\) by taking the upper left \(3 \times 3\) block of \(W\)
- Create the motion and shape matrices:
  - \(M = U_i W_i^h\) and \(S = W_i^h V_i^T\) (or \(M = U_i\) and \(S = W_i V_i^T\))
- Eliminate affine ambiguity

**Reconstruction Results**

![Reconstruction Results](image-source)

**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
  
(1) Perform factorization on a dense sub-block


**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
  
(1) Perform factorization on a dense sub-block

(2) Solve for a new 3D point visible by at least two known cameras

**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
  1. Perform factorization on a dense sub-block
  2. Solve for a new point visible by at least two known cameras (linear least squares)
  3. 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

**Projective 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 \( x_{ij} \)
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation \( Q \):
  \[ X \rightarrow QX, \ P \rightarrow PQ^{-1} \]
- We can solve for structure and motion when
  \[ 2mn \geq 11m + 3n - 15 \]
- 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: \( [BH]Q^t \)
  - Second camera matrix: \( [AB]Q^t \)
- Let \( \tilde{X} = QX \), then
  \[ x = [I|0] \tilde{X}, \quad \tilde{x}' = [Ab] \tilde{X} \]
- And
  \[ \tilde{x}'x = \tilde{z}Ax + b \]
  \[ \tilde{x}'x = \tilde{z}Ax + b \]
  \[ \tilde{x}'x = \tilde{z}Ax + b \]
- So we have
  \[ \tilde{x}'[b_1]A = 0 \]
- \[ F = [b_1]A \]  \quad \[ b_1 \text{ epipole } (F^tb = 0), \quad A = -[b_1]F \]
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?

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 structure and motion: bundle adjustment

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

Bundle Adjustment

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

$$E(P, X) = \sum_{i=1}^{n} \sum_{j=1}^{m} D(x_j, P X_i)$$
**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&BP 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 $P_i = Q R_i T_i$.
  - 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&Z).
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

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**

Commercial Software Packages

- boujou ([http://www.2d3.com/](http://www.2d3.com/))
- PFTrack ([http://www.thepixelfarm.co.uk/](http://www.thepixelfarm.co.uk/))
- Voodoo Camera Tracker ([http://www.digilab.uni-hannover.de/](http://www.digilab.uni-hannover.de/))

Applications: Matchmoving

- Putting virtual objects into real-world videos

Applications: Large-Scale SfM from Flickr


boujou demo

(We have a license available at I8, so if you want to try it for interesting projects, contact us.)
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

Prentice Hall, 2003