Structure from Motion

Kintan Saha

June 1, 2025

Outline

1. Primitives and Transformations
2. Geometric Image Formation
3. Structure from Motion
   - Two-Frame Structure from Motion
   - Preliminaries
   - Two-frame Structure-from-Motion
   - Factorization Methods
   - Bundle Adjustment

2D Points: Coordinates and Conversion

Inhomogeneous: $\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2$ **Homogeneous:** $\tilde{\mathbf{x}} = \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{pmatrix} \in \mathbb{P}^2$

where $\mathbb{P}^2 = \mathbb{R}^3 \setminus \{(0,0,0)\}$ is projective space.

Conversion:

$$\tilde{\mathbf{x}} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \bar{\mathbf{x}}, \quad \frac{1}{\tilde{w}}\tilde{\mathbf{x}} = \begin{pmatrix} \tilde{x}/\tilde{w} \\ \tilde{y}/\tilde{w} \\ 1 \end{pmatrix} = \bar{\mathbf{x}}$$

Notes:

- Homogeneous vectors are defined only up to scale.
- Vectors differing only by scale are equivalent.
- If $\tilde{w} = 0$, the point is at infinity and not representable in $\mathbb{R}^2$.

2D Lines and Cross Product

Lines in Homogeneous Coordinates: $\tilde{l} = (a, b, c)^\top$
Point $\bar{x}$ lies on $\tilde{l}$ iff $\tilde{l}^\top \bar{x} = 0 \Rightarrow ax + by + c = 0$
Normalization: $\tilde{l} = (n_x, n_y, -d)^\top = (n, -d)^\top$ with $\|n\|_2 = 1$
Line at infinity: $\tilde{l}_\infty = (0, 0, 1)^\top$

Cross Product:

$$a \times b = [a]_\times b, \quad [a]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

$$a \times b = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}$$

2D Line Arithmetic

Intersection of lines: $\tilde{\mathbf{x}} = \tilde{l}_1 \times \tilde{l}_2$
Line through two points: $\tilde{l} = \bar{\mathbf{x}}_1 \times \bar{\mathbf{x}}_2$

Example: Intersection of $\tilde{l}_1 = (1, 0, -1)^\top$ (corresponding to x-1 $= 0$), $\tilde{l}_2 = (0, 1, -1)^\top$ (corresponding to y-1 $= 0$):

$$\tilde{x} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Rightarrow (1, 1)$$

2D Transformations: Overview

- **Translation** (2 DoF): $x' = x + t \Leftrightarrow \bar{x}' = \begin{bmatrix} I & t \\ 0^\top & 1 \end{bmatrix} \bar{x}$
- **Euclidean** (3 DoF): $x' = Rx + t \Leftrightarrow \bar{x}' = \begin{bmatrix} R & t \\ 0^\top & 1 \end{bmatrix} \bar{x}$ ($R \in SO(2)$, preserves distances)
- **Similarity** (4 DoF): $x' = sRx + t \Leftrightarrow \bar{x}' = \begin{bmatrix} sR & t \\ 0^\top & 1 \end{bmatrix} \bar{x}$ (preserves angles)
- **Affine** (6 DoF): $x' = Ax + t \Leftrightarrow \bar{x}' = \begin{bmatrix} A & t \\ 0^\top & 1 \end{bmatrix} \bar{x}$ ($A \in \mathbb{R}^{2\times 2}$, preserves parallelism)
- **Projective** (Homography, 8 DoF): $\tilde{x}' = \tilde{H}\tilde{x}$ ($\tilde{H} \in \mathbb{R}^{3\times 3}$, preserves straight lines)

Transformations form nested set of groups.

Overview of 3D Transformations

Analogous to 2D transformations, using 4×4 matrices for homogeneous coordinates.

- Translation (3 DoF)
- Euclidean (6 DoF, $R \in SO(3)$)
- Similarity (7 DoF)
- Affine (12 DoF, $A \in \mathbb{R}^{3\times3}$)
- Projective (15 DoF, $\tilde{H} \in \mathbb{R}^{4\times4}$)

Transformations preserve properties similarly to the 2D case.

Direct Linear Transform (DLT) for Homography Estimation

Goal: Estimate homography $\tilde{H}$ from N 2D-to-2D correspondences $\{(\tilde{x}_i, \tilde{x}'_i)\}_{i=1}^{N}$ where $\tilde{x}'_i = \tilde{H}\tilde{x}_i$.

- Since vectors are homogeneous, $\tilde{x}'_i \times \tilde{H}\tilde{x}_i = 0$.
- Let h^k be the k-th row of $\tilde{H}$. This gives a linear equation in $h = (h^1, h^2, h^3)^\top$:

$$\begin{bmatrix} 0^\top & -\tilde{w}'_i\tilde{x}_i^\top & \tilde{y}'_i\tilde{x}_i^\top \\ \tilde{w}'_i\tilde{x}_i^\top & 0^\top & -\tilde{x}'_i\tilde{x}_i^\top \\ -\tilde{y}'_i\tilde{x}_i^\top & \tilde{x}'_i\tilde{x}_i^\top & 0^\top \end{bmatrix} \begin{pmatrix} h^1 \\ h^2 \\ h^3 \end{pmatrix} = 0$$

  The last row is linearly dependent and can be dropped, giving 2 equations per correspondence.
- Stacking equations gives $Ah = 0$, a $2N \times 9$ system.
- Solve $\min_h ||Ah||^2$ subject to $||h||^2 = 1$. Solution is the singular vector corresponding to the smallest singular value of A (via SVD).

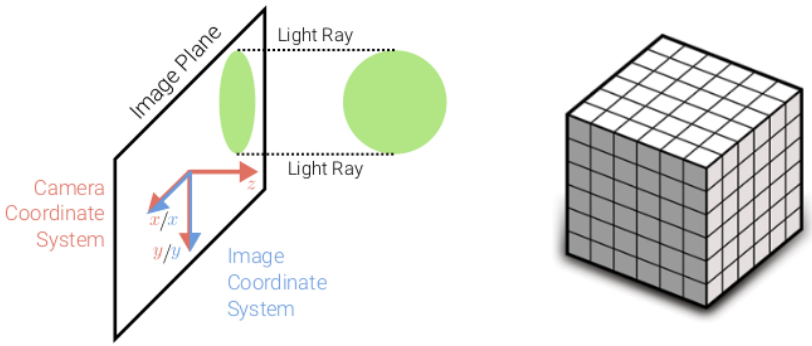
Projection Models

- **Perspective Projection**: Light rays to converge at a focal point. Objects appear smaller with distance. (e.g., standard camera lenses)
- **Orthographic Projection**: Light rays are parallel. Object size remains constant regardless of distance. (Approximation for telecentric or very long telephoto lenses).

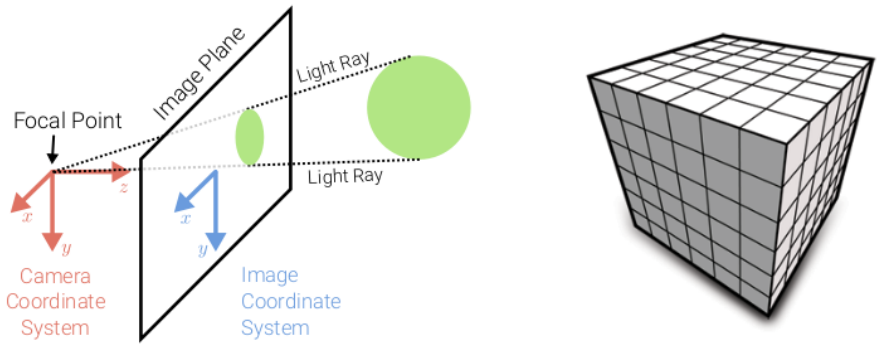
Notation:

- $x_c = (x_c, y_c, z_c)^\top$: a 3D point in camera coordinates
- $x_s = (x_s, y_s)^\top$: 2D projection of the above 3D point as obtained by the camera

Orthographic Projection



Perspective Projection



Orthographic Projection

Projects a 3D point $x_c = (x_c, y_c, z_c)^\top$ in camera coordinates to 2D screen coordinates $x_s = (x_s, y_s)^\top$ by dropping the z component.

$$x_s = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} x_c \Leftrightarrow \bar{x}_s = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{x}_c$$

Scaled Orthographic Projection: Includes a scale factor s (to account for measurement unit disparity).

$$x_s = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \end{pmatrix} x_c \Leftrightarrow \bar{x}_s = \begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{x}_c$$

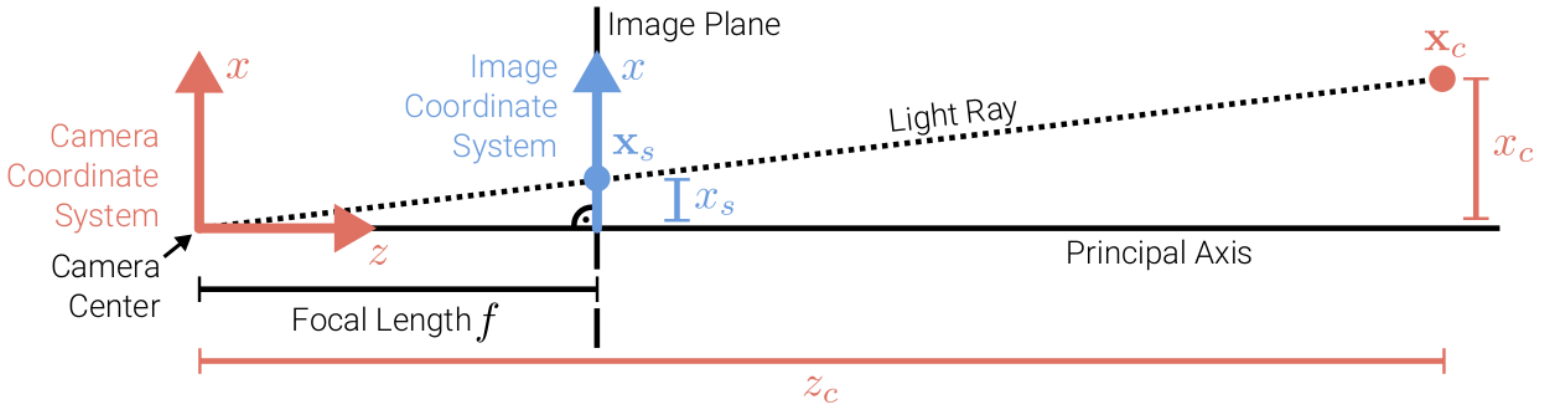
- Distance information is lost. The 3D point can't be retrieved since the matrix is singular.
- Under orthographic projection, structure and motion can be estimated via SVD (more on this later)

Perspective Projection

3D points $x_c = (x_c, y_c, z_c)^\top$ are mapped to the image plane by dividing by their z_c component and multiplying by the focal length f:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} f x_c / z_c \\ f y_c / z_c \end{pmatrix} \Leftrightarrow \tilde{x}_s = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bar{x}_c$$

This projection is linear using homogeneous coordinates. Distance information is lost and the 3D point can't be recovered since the matrix is singular.



$$\frac{x_s}{f} = \frac{x_c}{z_c}$$

Perspective Projection with Principal Point Offset

To ensure positive pixel coordinates, a principal point offset $c = (c_x, c_y)^\top$ is added. The complete perspective projection model with potentially different focal lengths f_x, f_y and skew s is as follows:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} f_x x_c/z_c + s y_c/z_c + c_x \\ f_y y_c/z_c + c_y \end{pmatrix}$$

In homogeneous coordinates:

$$\tilde{x}_s = \begin{bmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bar{x}_c = K[I|0]\bar{x}_c$$

The 3×3 matrix $K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$ is the **calibration matrix** (camera intrinsics). Often, $f_x = f_y$ and $s = 0$.

Chaining Transformations: World to Image

To project a point $\bar{x}_w$ in world coordinates to image coordinates x_s:

1. Transform from world to camera coordinates: $\bar{x}_c = \begin{bmatrix} R & t \\ 0^\top & 1 \end{bmatrix} \bar{x}_w$, where $[R|t]$ are camera extrinsics (pose).
2. Project from camera to image coordinates: $\tilde{x}_s = K[I|0]\bar{x}_c$.

Combined:

$$\tilde{x}_s = K[I|0] \begin{bmatrix} R & t \\ 0^\top & 1 \end{bmatrix} \bar{x}_w = K[R|t]\bar{x}_w = P\bar{x}_w$$

$P = K[R|t]$ is the 3×4 projection matrix.

Chaining Transformations: World to Image

To project a 3D point $\bar{\mathbf{x}}_w$ in world coordinates to 2D image coordinates $\tilde{\mathbf{x}}_s$, we apply the following steps:

1. World to Camera Coordinates (Extrinsics):

$$\bar{\mathbf{x}}_c = \begin{bmatrix} R & t \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{\mathbf{x}}_w \quad \text{where } R \in SO(3),\ t \in \mathbb{R}^3$$

2. Camera to Image Coordinates (Intrinsics):

$$\tilde{\mathbf{x}}_s = K[I \mid 0]\, \bar{\mathbf{x}}_c \quad \text{where } K \text{ is the camera intrinsic matrix}$$

Combined Projection:

$$\tilde{\mathbf{x}}_s = K[I \mid 0] \begin{bmatrix} R & t \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{\mathbf{x}}_w = K[R \mid t]\, \bar{\mathbf{x}}_w = P\, \bar{\mathbf{x}}_w$$

Projection Matrix: $P = K[R \mid t] \in \mathbb{R}^{3\times 4}$

Full Rank Representation

Sometimes a full rank 4×4 projection matrix is used:

$$\tilde{x}_s = \begin{bmatrix} K & 0 \\ 0^\top & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0^\top & 1 \end{bmatrix} \bar{x}_w = \tilde{P} \bar{x}_w$$

Now $\tilde{x}_s$ is a 4D vector. Normalize wrt its 3rd entry z_s for inhomogeneous pixels:

$$\bar{x}_s = \tilde{x}_s / z_s = (x_s/z_s, y_s/z_s, 1, 1/z_s)^\top$$

The 4th component is inverse depth. If known, 3D point can be retrieved from pixel coordinates via $\tilde{x}_w = \tilde{P}^{-1} \bar{x}_s$.

Full-Rank Projection Matrix

Sometimes a full-rank 4×4 projection matrix is used:

$$\tilde{\mathbf{x}}_s = \underbrace{\begin{bmatrix} K & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix}}_{\text{Augmented intrinsics}} \begin{bmatrix} R & t \\ \mathbf{0}^\top & 1 \end{bmatrix} \bar{\mathbf{x}}_w = \tilde{P}\bar{\mathbf{x}}_w$$

Now $\tilde{\mathbf{x}}_s \in \mathbb{R}^4$ is a **homogeneous 4D vector**. To obtain pixel coordinates, normalize by the 3rd component z_s:

$$\bar{\mathbf{x}}_s = \frac{1}{z_s}\tilde{\mathbf{x}}_s = \begin{pmatrix} x_s/z_s \\ y_s/z_s \\ 1 \\ 1/z_s \end{pmatrix}$$

Note: The 4th component $\frac{1}{z_s}$ represents the **inverse depth**
If inverse depth is known, the corresponding 3D point in world coordinates can be recovered.

Problem Statement

An example: Orthographic Projection

Triangulation

Epipolar Geometry

Essential and Fundamental Matrices

Camera Calibration: Introduction

- Camera calibration is the process of determining the intrinsic and extrinsic parameters of a camera.
- **Intrinsic parameters** relate to the camera's internal characteristics (e.g., focal length, principal point, lens distortion).
- **Extrinsic parameters** define the camera's position and orientation in the world.
- Most commonly, a known calibration target (e.g., checkerboard pattern) is used. This target provides known 3D points whose projections onto the 2D image plane can be easily identified.

Camera Calibration: Process Steps

1. **Capture Images:** The known calibration target is captured from various poses (different angles and distances). This ensures that the camera parameters are estimated robustly across different views.
2. **Detect Features:** Salient features (e.g., corners of the checkerboard squares) on the target are detected in the captured images. Accurate feature detection is crucial for precise calibration.
3. **Optimize Parameters:** Camera intrinsics and extrinsics (poses of the target relative to the camera) are jointly optimized.
   - A closed-form solution can initialize most parameters, except for lens distortion.
   - Non-linear optimization is then performed to minimize reprojection errors (the distance between observed feature points and the projected 3D points using the estimated parameters).

Feature Detection and Description: Overview

- **Point features** (or keypoints) capture the appearance of distinctive, repeatable local regions in an image.
- They are fundamental to sparse 3D reconstruction methods such as Structure-from-Motion (SfM), where they enable the estimation of camera poses and 3D scene geometry.
- Ideal features exhibit invariance to:
  - Changes in viewpoint (i.e., affine or projective transformations)
  - Illumination and contrast variations
- The projection of the same 3D point into multiple views should yield similar feature descriptors, enabling robust cross-image matching.
- Feature detection (e.g., SIFT, ORB) and description are followed by feature matching, which identifies 2D correspondences across images.
- These correspondences are used in triangulation to estimate the 3D locations of scene points and in bundle adjustment to refine camera parameters and structure jointly.

SIFT: Detection

SIFT is a widely used algorithm for detecting and describing local features.

1. **Scale-Space Construction:**
   - A scale space is created by iteratively filtering the image with Gaussian kernels of increasing sigma.
   - This simulates viewing the image at different levels of blur or "scale."
2. **Difference of Gaussians (DoG):**
   - Adjacent Gaussian-blurred images in the scale space are subtracted to produce Difference of Gaussian (DoG) images.
   - DoG images are an approximation of the Laplacian of Gaussian, which is effective for blob detection.
3. **Interest Point Detection:**
   - Interest points (often referred to as keypoints or blobs) are detected as local extrema (maxima or minima) in the DoG scale space across both spatial dimensions and scale.

SIFT: Description

Once keypoints are detected, a descriptor is computed for each:

1. **Orientation Assignment:**
   - The descriptor is rotated to align with the dominant gradient orientation in the local neighborhood of the keypoint. This provides invariance to image rotation.
2. **Descriptor Computation:**
   - Gradient magnitudes and orientations are computed for a region around the keypoint (typically 16x16 pixels).
   - This region is divided into sub-regions (e.g., 4x4 sub-regions).
   - For each sub-region, a histogram of gradient orientations (e.g., 8 bins) is computed.
3. **Feature Vector:**
   - All these histograms are concatenated to form a high-dimensional feature vector (e.g., 4x4 sub-regions * 8 bins = 128 dimensions).
   - This vector is then normalized to unit length to provide invariance to affine changes in illumination.

Feature Detection and Description: Summary & Matching

- Many algorithms exist: SIFT, SURF, U-SURF, BRISK, ORB, FAST, and more recently, deep learning-based methods.
- SIFT was seminal due to its robustness and invariance, revolutionizing matching and enabling large-scale SfM. It is still used in modern pipelines like COLMAP.
- **Feature Matching:**
  - Correspondences between features in different images are found using efficient nearest neighbor search in the descriptor space.
  - **Ratio Test for Filtering:** Ambiguous matches are often filtered by comparing the distance to the closest neighbor with the distance to the second closest neighbor. A ratio close to 1 (e.g., > 0.8) indicates an ambiguous match that might be incorrect and is often discarded.

Setup and Questions

Setup:

Let $\bar{\mathbf{x}}_w$ be a 3D point in world coordinates, and let P_1 and P_2 be the projection matrices of two cameras. The projections of this point onto the two image planes are:

$$\tilde{\mathbf{x}}_1 = P_1 \bar{\mathbf{x}}_w \quad \text{and} \quad \tilde{\mathbf{x}}_2 = P_2 \bar{\mathbf{x}}_w$$

Thus, $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_2$ form a point correspondence, as they are projections of the same 3D point.

Questions:

- Given $\tilde{\mathbf{x}}_1$, does it constrain $\tilde{\mathbf{x}}_2$?
  Intuitively, one might expect so, since both are images of the same 3D point.
- Given N point correspondences $\{\mathbf{x}_1^{(i)} \leftrightarrow \mathbf{x}_2^{(i)}\}_{i=1}^{N}$, can we recover the projection matrices P_1 and P_2?
- Given a point correspondence $\tilde{\mathbf{x}}_1 \leftrightarrow \tilde{\mathbf{x}}_2$ and known P_1, P_2, can we recover the 3D point $\bar{\mathbf{x}}_w$?

Epipolar Geometry: Introduction

Goal: Recover camera pose (relative rotation R and translation t) and 3D structure from image correspondences between two views.

- Let **x** be a 3D point.
- **x** is projected to pixel $\bar{x}_1$ in image 1 and $\bar{x}_2$ in image 2.
- The 3D point **x**, the two camera centers (O_1, O_2), span the **epipolar plane**.
- The **baseline** is the line connecting the two camera centers.
- **Epipoles** (e_1, e_2) are the projections of one camera center onto the other camera's image plane.
- The **epipolar line** (e.g., l_2 in image 2) is the intersection of the epipolar plane with the image plane. The corresponding point for $\bar{x}_1$ in image 2, $\bar{x}_2$, must lie on this line l_2.
- All epipolar lines in an image pass through the epipole in that image.

Epipolar Geometry: Setup

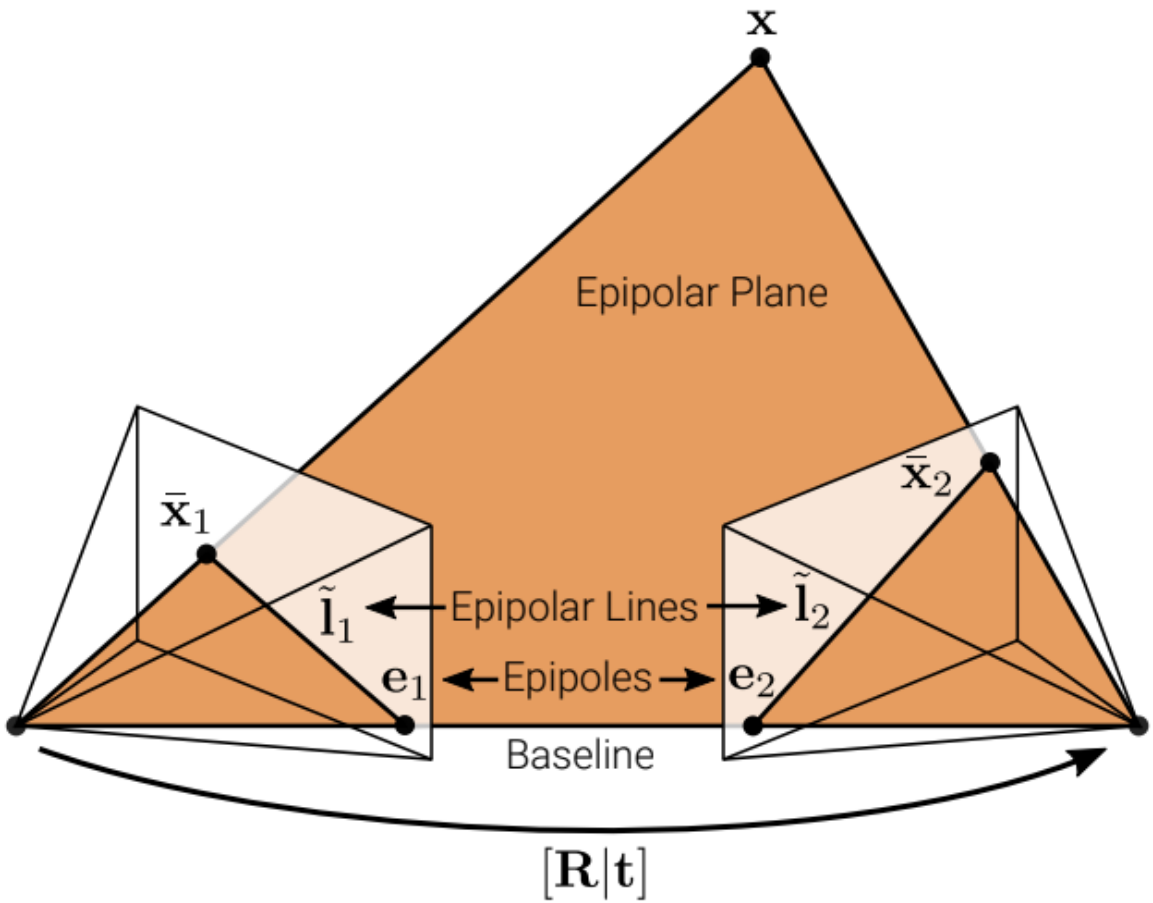


Figure 3: Epipolar Geometry

Epipolar Geometry and Ray Directions

Normalized Coordinates:
Let K_i be the intrinsic matrix of camera i. The normalized image coordinates (unit ray direction) are:

$$\tilde{\mathbf{x}}_i = K_i^{-1}\bar{\mathbf{x}}_i$$

Relative Pose:
Given rotation R and translation t between two calibrated cameras, a 3D point projects to:

$$\tilde{\mathbf{x}}_2 \propto R\tilde{\mathbf{x}}_1 + s\mathbf{t}$$

for some scalar depth s.

Cross Product:
Taking the cross product with $\mathbf{t}$ eliminates s:

$$[\mathbf{t}]_\times \tilde{\mathbf{x}}_2 \propto [\mathbf{t}]_\times R\tilde{\mathbf{x}}_1$$

The Essential Matrix

Epipolar Constraint:
Taking the dot product with $\tilde{\mathbf{x}}_2^\top$:

$$\tilde{\mathbf{x}}_2^\top [\mathbf{t}]_\times R \tilde{\mathbf{x}}_1 = 0 \Rightarrow \tilde{\mathbf{x}}_2^\top \mathbf{E} \tilde{\mathbf{x}}_1 = 0$$

where $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$ is the **Essential Matrix**.

Properties:

- $\mathbf{E}$ encodes relative pose between two calibrated cameras.
- $\mathbf{E}\tilde{\mathbf{x}}_1$ defines the epipolar line $\tilde{\mathbf{l}}_2$ in image 2.
- $\mathbf{E}^\top \tilde{\mathbf{x}}_2$ defines $\tilde{\mathbf{l}}_1$ in image 1.
- $\mathbf{E}$ has 5 DoF (3 for rotation, 2 for translation).

Properties of the Essential Matrix

Epipolar Geometry: Properties of E

- The epipole $\tilde{\mathbf{e}}_2$ in image 2 satisfies:

$$\tilde{\mathbf{e}}_2^\top \tilde{\mathbf{l}}_2 = \tilde{\mathbf{e}}_2^\top \mathbf{E}\tilde{\mathbf{x}}_1 = 0 \quad \text{for all } \tilde{\mathbf{x}}_1$$

- Therefore, $\tilde{\mathbf{e}}_2^\top \mathbf{E} = 0 \implies \tilde{\mathbf{e}}_2$ lies in the **left null space** of $\mathbf{E}$ (left singular vector with singular value 0).
- Similarly, $\mathbf{E}\tilde{\mathbf{e}}_1 = 0 \implies \tilde{\mathbf{e}}_1$ lies in the **right null space** of $\mathbf{E}$.
- The essential matrix is **rank 2** and singular. Its singular values are of the form:

$$(\sigma, \sigma, 0)$$

under ideal, noise-free conditions.

Estimating the Essential Matrix: Linear Setup

Given: $N \geq 8$ normalized point correspondences $\{(\tilde{\mathbf{x}}_1^{(k)}, \tilde{\mathbf{x}}_2^{(k)})\}_{k=1}^{N}$.
Epipolar constraint:

$$\tilde{\mathbf{x}}_2^{(k)\top} \mathbf{E} \tilde{\mathbf{x}}_1^{(k)} = 0$$

Let $\tilde{\mathbf{x}}_1 = (x, y, 1)^\top$, $\tilde{\mathbf{x}}_2 = (x', y', 1)^\top$. Expanding the above yields:

$$x'xe_{11} + x'ye_{12} + x'e_{13} + y'xe_{21} + y'ye_{22} + y'e_{23} + xe_{31} + ye_{32} + e_{33} = 0$$

This gives one linear constraint on the 9 unknowns in E.
Stacking all N correspondences yields a homogeneous linear system:

$$A \cdot \text{vec}(\mathbf{E}) = 0$$

vec($\mathbf{E}$) represents the elements of $\mathbf{E}$ stacked as a single vector in row-major order.
A is an $N \times 9$ matrix, where each row corresponds to one point correspondence and is given by::

$$[x'x, x'y, x', y'x, y'y, y', x, y, 1]$$

Solving the Linear System using SVD

Homogeneous system: $A \cdot \text{vec}(\mathbf{E}) = 0$

- For a solution (up to scale) to exist, A must have rank 8 (or more if noisy).
- The solution vec(**E**) lies in the null space of A.
- Use **SVD**: $A = UDV^\top$.
- vec(**E**) is the last column of V — corresponding to the smallest singular value of A.

Why 8 points?

- Each correspondence gives 1 equation.
- **E** has 9 entries, but only defined up to scale $\Rightarrow$ 8 equations suffice.

In practice:

- More than 8 points $\Rightarrow$ least squares solution.
- Noisy data causes A to have full rank $\Rightarrow$ solve via SVD.

Enforcing the Rank-2 Constraint on **E**

The matrix **E** must be rank 2.

- Solution from SVD of A may yield a full-rank matrix.
- Let $\mathbf{E} = UDV^\top$, where $D = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)$.
- To enforce rank-2: set smallest singular value to 0:

$$\hat{D} = \mathrm{diag}(\sigma_1, \sigma_2, 0)$$

- Then, set:

$$\hat{\mathbf{E}} = U\hat{D}V^\top$$

This projection minimizes $\|\mathbf{E} - \hat{\mathbf{E}}\|_F$ (Frobenius norm) and ensures $\hat{\mathbf{E}}$ satisfies the epipolar constraint.

The Normalized 8-Point Algorithm

Normalization is critical for numerical stability.

- Raw pixel coordinates can result in poor conditioning of A.
- Normalize image coordinates:
  - Translate so mean is 0
  - Scale so average distance from origin is $\sqrt{2}$
- Apply this transformation to both sets of image points.

Algorithm Steps:

1. Normalize points in both images.
2. Build matrix A from normalized correspondences.
3. Solve $A \cdot \text{vec}(\mathbf{E}) = 0$ via SVD.
4. Enforce rank-2 constraint on $\mathbf{E}$.
5. Denormalize $\mathbf{E}$ by applying inverse transforms.

Epipolar Geometry: The Fundamental Matrix **F**

When intrinsics $\mathbf{K}_1$, $\mathbf{K}_2$ are unknown, we use pixel coordinates $\bar{\mathbf{x}}_1$, $\bar{\mathbf{x}}_2$.
Calibrated constraint:

$$\tilde{\mathbf{x}}_2^\top \mathbf{E} \tilde{\mathbf{x}}_1 = 0 \quad \Rightarrow \quad \bar{\mathbf{x}}_2^\top \mathbf{K}_2^{-\top} \mathbf{E} \mathbf{K}_1^{-1} \bar{\mathbf{x}}_1 = 0$$

Define: $\mathbf{F} = \mathbf{K}_2^{-\top} \mathbf{E} \mathbf{K}_1^{-1}$
Resulting constraint:

$$\bar{\mathbf{x}}_2^\top \mathbf{F} \bar{\mathbf{x}}_1 = 0$$

Properties of F:

- Encodes epipolar geometry between uncalibrated views.
- Estimated from pixel-level correspondences.
- 3×3 matrix with rank 2 (ideally).
- Epipoles lie in null spaces of **F**.
- Enables only **projective reconstruction**.

Epipolar Geometry: Decomposing the Essential Matrix $\mathbf{E}$

Once the essential matrix $\mathbf{E}$ is estimated from calibrated correspondences:

- $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$ contains both rotation ($\mathbf{R}$) and translation direction ($\hat{\mathbf{t}}$) between the two cameras.
- Using the **SVD** of $\mathbf{E} = \mathbf{U}\Sigma\mathbf{V}^\top$ (with $\Sigma = \text{diag}(\sigma, \sigma, 0)$):
  - The direction of translation $\hat{\mathbf{t}}$ is given by the third column of $\mathbf{U}$ (or $\mathbf{V}$).
  - Rotation matrices can be computed using:

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top, \quad \mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top$$

$$\text{where } \mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- There are **four possible solutions** for $(\mathbf{R}, \mathbf{t})$.
  - The correct one is determined by checking which configuration places triangulated 3D points in front of both cameras (positive depth).

This decomposition allows recovery of the relative pose (up to scale) between two calibrated cameras.

Estimating **F** from 7 Points

Given 7 point correspondences $\{(\bar{\mathbf{x}}_1^{(k)}, \bar{\mathbf{x}}_2^{(k)})\}_{k=1}^{7}$:

- Each correspondence gives one linear constraint: $\bar{\mathbf{x}}_2^\top \mathbf{F} \bar{\mathbf{x}}_1 = 0$
- 7 constraints on 9 unknowns $\Rightarrow$ 2D null space of A
- General solution: $\mathbf{F} = \alpha \mathbf{F}_1 + (1 - \alpha)\mathbf{F}_2$

To enforce $\mathrm{rank}(\mathbf{F}) = 2$, solve:

$$\det(\alpha \mathbf{F}_1 + (1 - \alpha)\mathbf{F}_2) = 0$$

- This yields a cubic in α with up to 3 real roots
- Each real root $\Rightarrow$ one valid fundamental matrix
- Disambiguation done via cheirality check or geometric validation

Triangulation: Recovering 3D Structure

Goal: Given camera intrinsics, extrinsics (poses P_1, P_2), and corresponding 2D image points $(\bar{x}_1, \bar{x}_2)$, find the 3D point x_w.

- In theory, the rays from camera centers through $\bar{x}_1$ and $\bar{x}_2$ should intersect at x_w.
- Due to noise in image measurements and camera parameters, these rays might not perfectly intersect in 3D.
- We aim to find the 3D point x_w that is "closest" to both rays.

Linear Triangulation (DLT-based):

- Let $\tilde{x}_i^s = \tilde{P}_i \tilde{x}_w$ be the projection of world point $\tilde{x}_w$ onto camera i.
- Since they are homogeneous, $\tilde{x}_i^s \times (\tilde{P}_i \tilde{x}_w) = 0$.
- This gives two linear equations in the coordinates of $\tilde{x}_w$ per camera view.
- For $N \geq 2$ views, stack equations to form $A\tilde{x}_w = 0$.
- The solution is the right singular vector of A corresponding to the smallest singular value. This is the Direct Linear Transform (DLT).

Triangulation: Reprojection Error Minimization

- While DLT is simple, it's not always optimal as it doesn't directly minimize a physically meaningful error in image space and is not invariant to projective transformations.
- **Gold Standard:** Minimize the sum of squared reprojection errors using numerical methods:

$$\bar{x}_w^* = \arg\min_{\bar{x}_w} \sum_{i=1}^{N} \|\bar{x}_i^s(\bar{x}_w) - \bar{x}_i^o\|_2^2$$

where $\bar{x}_i^s(\bar{x}_w)$ is the projection of $\bar{x}_w$ into image i, and $\bar{x}_i^o$ is the observed feature.

Triangulation Uncertainty:

- Accuracy depends on the relative camera pose. Uncertainty increases as rays become more parallel (small baseline or distant point).
- There's a tradeoff: feature matching is easier for nearby views (small baseline), but triangulation is less accurate.

Factorization

How can we use more than two views for structure-from-motion?

Orthographic Factorization: Problem Setup

Goal: Given P feature points tracked across N frames under orthographic projection, recover:

- 3D structure of the scene (shape)
- Camera motion (rotation)

Projection model: Let $\mathbf{x}_p \in \mathbb{R}^3$ be the 3D point, and (x_{ip}, y_{ip}) its projection in frame i.
Under orthographic projection:

$$x_{ip} = \mathbf{u}_i^\top (\mathbf{x}_p - \mathbf{t}_i), \quad y_{ip} = \mathbf{v}_i^\top (\mathbf{x}_p - \mathbf{t}_i)$$

where $\mathbf{u}_i, \mathbf{v}_i \in \mathbb{R}^3$ are orthonormal rows of the rotation matrix $\mathbf{R}_i$, and $\mathbf{t}_i$ is the 2D translation.
Assumption: Center the world coordinate frame at the centroid of 3D points:

$$\frac{1}{P} \sum_{p=1}^{P} \mathbf{x}_p = \mathbf{0}$$

Orthographic Factorization: Measurement Matrix

Step 1: Center image points per frame to eliminate translation:

$$\tilde{x}_{ip} = x_{ip} - \frac{1}{P}\sum_{q=1}^{P} x_{iq}, \quad \tilde{y}_{ip} = y_{ip} - \frac{1}{P}\sum_{q=1}^{P} y_{iq}$$

Using the centroid assumption:

$$\tilde{x}_{ip} = \mathbf{u}_i^\top \mathbf{x}_p, \quad \tilde{y}_{ip} = \mathbf{v}_i^\top \mathbf{x}_p$$

Step 2: Build the $2N \times P$ measurement matrix $\tilde{\mathbf{W}}$:

$$\tilde{\mathbf{W}} = \begin{bmatrix} \tilde{x}_{11} & \cdots & \tilde{x}_{1P} \\ \vdots & \ddots & \vdots \\ \tilde{x}_{N1} & \cdots & \tilde{x}_{NP} \\ \tilde{y}_{11} & \cdots & \tilde{y}_{1P} \\ \vdots & \ddots & \vdots \\ \tilde{y}_{N1} & \cdots & \tilde{y}_{NP} \end{bmatrix} = \mathbf{RX}$$

Where:

- $\mathbf{R} \in \mathbb{R}^{2N \times 3}$ is the motion matrix (stacked $\mathbf{u}_i^\top$, $\mathbf{v}_i^\top$)

Orthographic Factorization: SVD Solution

In practice, $\tilde{\mathbf{W}}$ is full rank due to noise. But ideally $\mathrm{rank}(\tilde{\mathbf{W}}) \leq 3$.

Step 3: Low-rank approximation via SVD:

$$\tilde{\mathbf{W}} \approx \hat{\mathbf{W}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

Keep top 3 singular values: $\mathbf{U}_3 \in \mathbb{R}^{2N\times 3}$, $\boldsymbol{\Sigma}_3 \in \mathbb{R}^{3\times 3}$, $\mathbf{V}_3^\top \in \mathbb{R}^{3\times P}$

Step 4: Initial estimates

$$\hat{\mathbf{R}} = \mathbf{U}_3\boldsymbol{\Sigma}_3^{1/2}, \quad \hat{\mathbf{X}} = \boldsymbol{\Sigma}_3^{1/2}\mathbf{V}_3^\top$$

Note: Factorization is only unique up to an invertible 3×3 matrix $\mathbf{Q}$:

$$\hat{\mathbf{W}} = (\hat{\mathbf{R}}\mathbf{Q})(\mathbf{Q}^{-1}\hat{\mathbf{X}})$$

Orthographic Factorization: Metric Upgrade

To resolve ambiguity, enforce metric constraints on $\mathbf{R}_{\text{true}} = \hat{\mathbf{R}}\mathbf{Q}$.
For each frame i, rows $\mathbf{u}_i^\top \mathbf{Q}$ and $\mathbf{v}_i^\top \mathbf{Q}$ must be:

- Unit norm:

$$\mathbf{u}_i^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{u}_i = 1, \quad \mathbf{v}_i^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{v}_i = 1$$

- Orthogonal:

$$\mathbf{u}_i^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{v}_i = 0$$

Let $\mathbf{G} = \mathbf{Q}\mathbf{Q}^\top$. Solve for $\mathbf{G}$ via least squares over $i = 1, \ldots, N$.
Recover $\mathbf{Q}$ from $\mathbf{G}$ via Cholesky decomposition, then:

$$\mathbf{R} = \hat{\mathbf{R}}\mathbf{Q}, \quad \mathbf{X} = \mathbf{Q}^{-1}\hat{\mathbf{X}}$$

Summary:

1. Form $\tilde{\mathbf{W}}$ from centered 2D tracks
2. Do rank-3 SVD: $\hat{\mathbf{W}} = \hat{\mathbf{R}}\hat{\mathbf{X}}$
3. Recover $\mathbf{Q}$ from metric constraints
4. Compute final $\mathbf{R}$ and $\mathbf{X}$

Perspective Factorization: Extensions

- Tomasi-Kanade assumes orthography, which is a strong simplification.
- **Iterative Approaches for Perspective:**
  - Christy and Horaud (1996): Initial orthographic reconstruction, then iterative correction for perspective effects.
  - Triggs (1996): Performs projective factorization, iteratively updating depths.
- Factorization methods can provide a good initialization for more accurate iterative techniques like Bundle Adjustment.
- However, modern SfM pipelines (e.g., COLMAP) often prefer incremental bundle adjustment, starting with a robust two-view reconstruction and adding views iteratively.

Bundle Adjustment: The Gold Standard

Goal: Simultaneously refine all camera parameters (intrinsic and extrinsic) and 3D point coordinates to minimize the total reprojection error. It's a large, non-linear optimization problem. Let $\Pi = \{\pi_i\}$ be the set of N cameras (parameters). Let $X_w = \{\mathbf{x}_p^w\}$ be the set of P 3D points. Let $X_s = \{\mathbf{x}_{ip}^s\}$ be the 2D image observations of point p in camera i.
Minimize:

$$\Pi^*, X_w^* = \arg\min_{\Pi, X_w} \sum_{i=1}^{N} \sum_{p=1}^{P} w_{ip} \|\mathbf{x}_{ip}^s - \pi_i(\mathbf{x}_p^w)\|_2^2$$

- w_{ip} is a visibility term (1 if point p is observed in image i, 0 otherwise).
- $\pi_i(\mathbf{x}_p^w)$ is the projection of 3D point $\mathbf{x}_p^w$ onto the image plane of camera i.

$$\pi_i(\mathbf{x}_p^w) = \begin{pmatrix} \tilde{x}_p^s / \tilde{w}_p^s \\ \tilde{y}_p^s / \tilde{w}_p^s \end{pmatrix} \text{ with } \begin{pmatrix} \tilde{x}_p^s \\ \tilde{y}_p^s \\ \tilde{w}_p^s \end{pmatrix} = K_i(R_i \mathbf{x}_p^w + \mathbf{t}_i)$$

Bundle Adjustment: Challenges and Solutions

- **Initialization:**
  - The energy landscape is highly non-convex.
  - A good initial guess for camera poses and 3D points is crucial to avoid poor local minima.
  - Initializing all parameters jointly is difficult due to occlusions, viewpoint changes, and matching outliers.
  - **Incremental SfM:** Start with a robust two-view reconstruction, then incrementally add new images/cameras and perform BA. This is a common strategy.
- **Optimization:**
  - Can involve millions of features and thousands of cameras, leading to a very large number of parameters.
  - Computational complexity can be high (e.g., cubic in the number of unknowns for dense methods).
  - **Sparsity:** The problem is typically sparse because not all 3D points are visible in every camera. The Jacobian matrix of the error function is sparse.
  - Efficient sparse solvers (e.g. Ceres Solver) are used in practice.

Incremental SfM: Feature Matching and Initialization

Incremental SfM (used in systems like COLMAP) is a pipeline that builds the scene step-by-step:

Step 1: Correspondence Search

- **Feature Extraction:** Detect keypoints (e.g., SIFT) in all input images.
- **Feature Matching:** Match features between image pairs using descriptors.
- **Geometric Verification:** Eliminate incorrect matches using RANSAC with:
  - Fundamental Matrix (uncalibrated cameras), or
  - Essential Matrix (calibrated cameras).

  This ensures only geometrically consistent matches are kept.

Step 2: Initialization

- Choose an image pair with:
  - High number of inlier matches
  - Large baseline (to reduce depth ambiguity)
- Estimate relative pose between the two views

Incremental SfM: Registration, Triangulation, and Optimization

Step 3: Incremental Image Registration

- Iteratively add new images to the model.
- For each new image:
  - Match its 2D features to existing 3D points.
  - Use **PnP + RANSAC** to estimate the camera pose.

Step 4: Triangulation of New Points

- Triangulate additional 3D points using new matches between registered images.
- Handle outliers using robust methods (e.g., RANSAC, multi-view checks).

Step 5: Bundle Adjustment (BA)

- Jointly refine 3D point positions and camera poses to minimize reprojection error.
- Use:
  - **Local BA** for speed (on recent additions)

Incremental SfM: Registration, Triangulation, and Optimization

Step 3: Incremental Image Registration

- Iteratively add new images to the model.
- For each new image:
  - Match its 2D features to existing 3D points.
  - Use **PnP + RANSAC** to estimate the camera pose.

Step 4: Triangulation of New Points

- Triangulate additional 3D points using new matches between registered images.
- Handle outliers using robust methods (e.g., RANSAC, multi-view checks).

Step 5: Bundle Adjustment (BA)

- Jointly refine 3D point positions and camera poses to minimize reprojection error.
- Use:
  - **Local BA** for speed (on recent additions)

Incremental SfM: Image Registration Details

Given a set of N 3D-to-2D correspondences $\{\mathbf{x}_i^w, \bar{x}_i^s\}$, where $\bar{x}_i^s = P\mathbf{x}_i^w$.

- This is a Perspective-n-Point (PnP) problem.
- Since $\bar{x}_i^s$ and $P\mathbf{x}_i^w$ are homogeneous and should have the same direction, their cross product is zero: $\bar{x}_i^s \times (P\mathbf{x}_i^w) = \mathbf{0}$.
- This provides linear equations in the 12 entries of the projection matrix P.
- With enough correspondences (at least 6 for general P, fewer for specific cases), P can be solved using DLT (SVD).
- $P = K[R|\mathbf{t}]$. K and R can be recovered from the 3×3 front submatrix of P using RQ factorization.
- If intrinsics K are known, pose can be estimated from only 3 points (P3P algorithm).
- **RANSAC** (RANdom SAmple Consensus) is crucial for robustness against outlier correspondences.

Structure-from-Motion: Summary

- SfM recovers 3D structure and camera motion from a collection of 2D images.
- Key components include:
  - Feature detection and matching (e.g., SIFT).
  - Two-view geometry estimation (Essential/Fundamental matrix).
  - Triangulation of 3D points.
  - Multi-view reconstruction methods (Factorization, Incremental SfM).
  - Bundle Adjustment for global optimization.
- SfM has enabled numerous applications in 3D modeling, robotics, augmented reality, and visual effects.
- Challenges remain in handling dynamic scenes, textureless regions, and achieving real-time performance for very large-scale scenarios.

Our Results