Reprojection, Reconstruction, Realization

Refining Multi-View 3D Geometry through Bundle Adjustment

Soumyajoy Kundu
M.Sc Data Science
Chennai Mathematical Institute

Chandranath Bhattacharya
M.Sc Data Science
Chennai Mathematical Institute

Computer Vision
April 14, 2025

Outline

Projection Camera Model

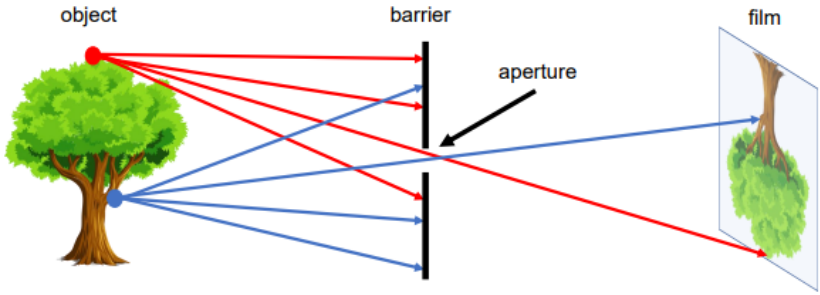


Figure: A simple working camera model

Pinhole Camera

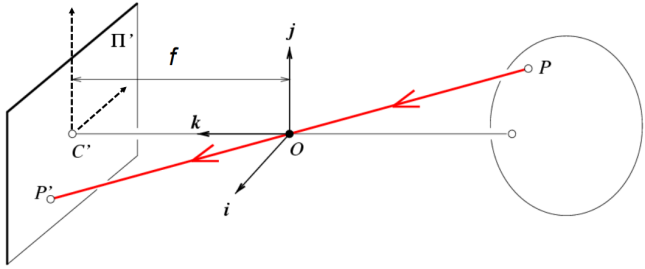


Figure: Pinhole camera model

$$\vec{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \longrightarrow \vec{P'} = \begin{bmatrix} x' \\ y' \end{bmatrix} \qquad \text{similar } \Delta\text{'s} \begin{cases} x' = f\frac{X}{Z} \\ \\ y' = f\frac{Y}{Z} \end{cases}$$

Camera Projection and Pixel Coordinates

- Pixel coordinates $(\mu, \nu)^T$ in the pixel plane.
- Let scaling factors be α (in u) and β (in v).
- Translation to the origin is $(c_x, c_y)^T$.

Thus, the relationship between imaging plane and pixel plane coordinates is:

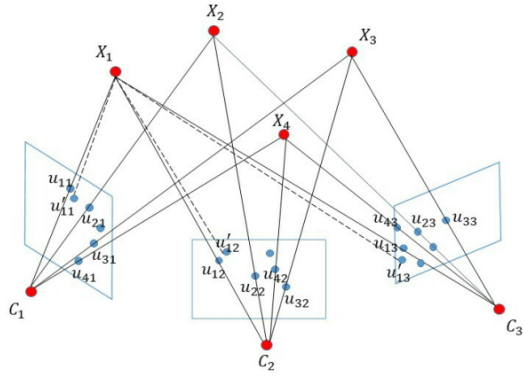
$$\begin{bmatrix} \mu \\ \nu \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X/Z \\ Y/Z \\ 1 \end{bmatrix} = \frac{1}{Z} KP$$

Where K is the intrinsic (calibration) matrix.

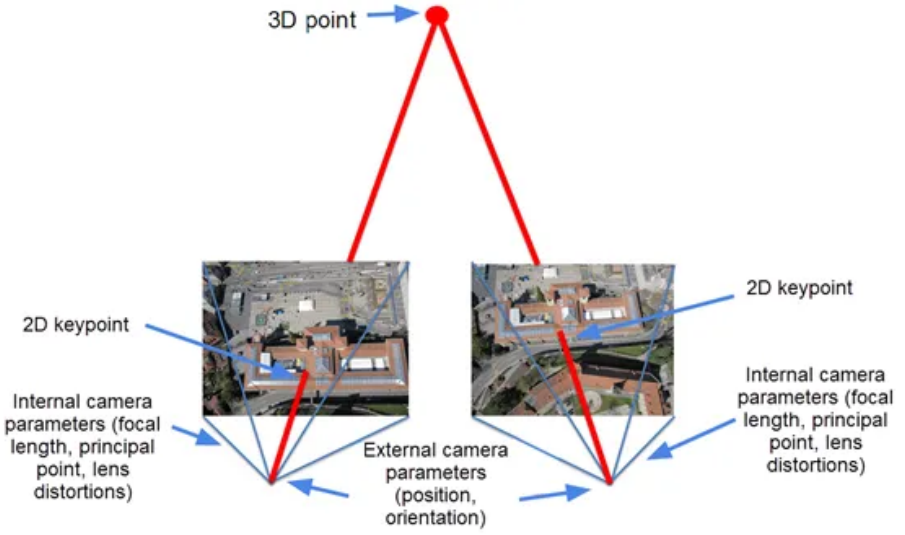
$$\begin{bmatrix} \mu \\ \nu \\ 1 \end{bmatrix} = \frac{1}{Z} \cdot K(RP_w + t)$$

Problem Statement

- To reconstruct a set of 3D points and estimate camera parameters from a set of corresponding 2D image points.



Problem Statement



Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

2. **Feature Extraction:**
   Detect keypoints and descriptors in each I_k.

Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

2. **Feature Extraction:**
   Detect keypoints and descriptors in each I_k.

3. **Feature Matching:**
   Match keypoints across views to obtain correspondences.

Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

2. **Feature Extraction:**
   Detect keypoints and descriptors in each I_k.

3. **Feature Matching:**
   Match keypoints across views to obtain correspondences.

4. **Triangulation:**
   Reconstruct 3D points $\mathbf{X}_i \in \mathbb{R}^3$

Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

2. **Feature Extraction:**
   Detect keypoints and descriptors in each I_k.

3. **Feature Matching:**
   Match keypoints across views to obtain correspondences.

4. **Triangulation:**
   Reconstruct 3D points $\mathbf{X}_i \in \mathbb{R}^3$

5. **Bundle Adjustment:**
   Minimize reprojection error over all cameras and 3D points:

Workflow

1. **Input:**
   A set of n images $\{I_1, I_2, \ldots, I_n\}$ with 2D point correspondences $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$, where $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{R}^2$ are corresponding image points.

2. **Feature Extraction:**
   Detect keypoints and descriptors in each I_k.

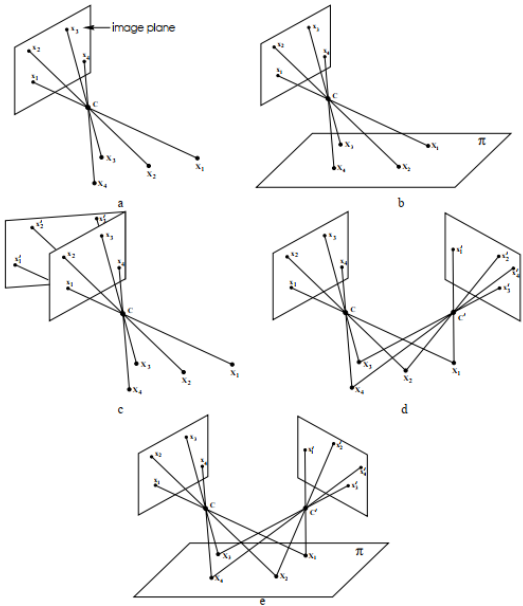
3. **Feature Matching:**
   Match keypoints across views to obtain correspondences.

4. **Triangulation:**
   Reconstruct 3D points $\mathbf{X}_i \in \mathbb{R}^3$

5. **Bundle Adjustment:**
   Minimize reprojection error over all cameras and 3D points:

6. **Output:**
   Optimized 3D structure $\{\mathbf{X}_i\}$ and camera parameters $\{P_k\}$.

The Camera Centre is the Essence!



(a) Image formation

(b) World Plane $\rightarrow$ Image Plane

(c) Images with same camera centre

(d) Multiple camera centres Problem at hand!

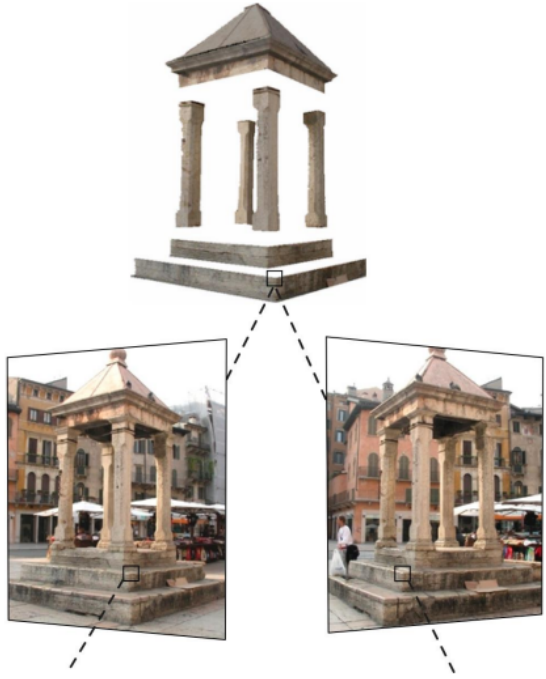
(e) (d) + world is planar

Two-View Geometry

Goal!!!

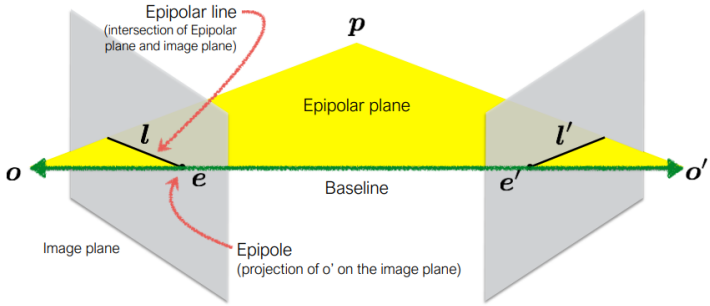
Reconstruct 3D scene from 2D point correspondences across two views.

- **Input**: Point correspondences $x_i \leftrightarrow x_i'$ in two images.
- **Unknowns**:
  - 3D points X_i
  - Camera matrices P and P'
- **Assumption**: $x_i = PX_i$ and $x_i' = P'X_i$
- **Ambiguity**:
  - Reconstruction possible only up to a projective transformation H
  - $(P_j H^{-1})(HX_i)$ yields same projections



Epipolar Geometry

- Describes the **geometric relationship** between two views of a 3D scene.
- A 3D point P is projected onto two images as points x and x'.
- Together with camera centers o and o', point P defines the **epipolar plane**.



Epipolar Geometry

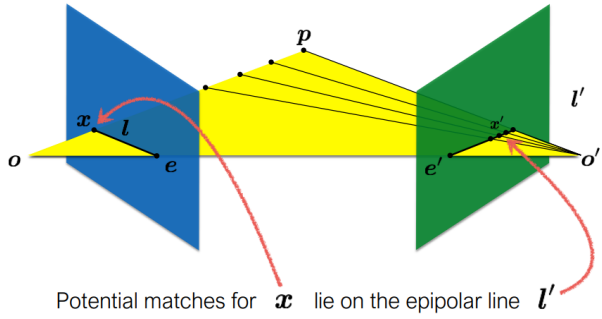


Figure: Epipolar Constraint

Why is it important?

- Reduces 2D correspondence search to a **1D search along epipolar lines**.
- Crucial for estimating the **fundamental matrix (F)**.
- Basis for **triangulation** and **3D reconstruction**.

Fundamental Matrix

- Mapping of point in one image to epipolar line of image $\mathbf{x}$
- **Assumption** : Uncalibrated Cameras

Mathematically the line is expressed as,

$$\mathbf{l}' = F\mathbf{x} = K'^{-1}EK^{-1}\mathbf{x}$$

- Since, $\mathbf{x}'$ is on $\mathbf{l}'$, by the point on line definition we know,

$$\mathbf{x}'^{T}\mathbf{l}' = 0$$

We can thus relate corresponding pair $(\mathbf{x}, \mathbf{x}')$ to each other,

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

This equation says that the point $\mathbf{x}'$ in the second image must lie on the epipolar line corresponding to $\mathbf{x}$ in the first image.

cmi

Algorithm

Require: A set of n images $\{I_1, I_2, \ldots, I_n\}$ with corresponding 2D points $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_{i=1}^{N}$

1: **for** each image I_k **do**
2: Detect keypoints and extract descriptors
3: **end for**
4: Match keypoints across image pairs to get correspondences
5: Estimate relative camera poses and epipolar geometry (e.g., Fundamental or Essential matrix)
6: Triangulate 3D points $\mathbf{X}_i$ from matched points and estimated camera matrices
7: Initialize camera parameters $\{P_k\}$ and 3D points $\{\mathbf{X}_i\}$
8: Perform Bundle Adjustment:
9: **return** $\{\mathbf{X}_i\}$ and $\{P_k\}$

Feature Extraction

Difficulty :

- Automatically find the correspondence of each pixel in images of different points of view

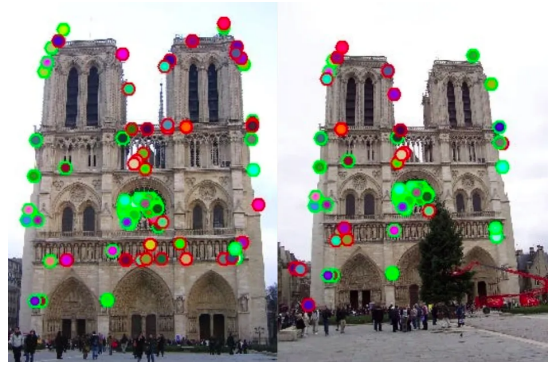


Figure: Extracting Key Feature Points

Feature Extraction

- Detect good features (key points)
  - Corners
  - SIFT points

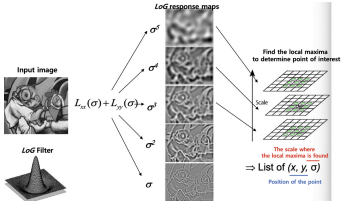


Figure: SIFT

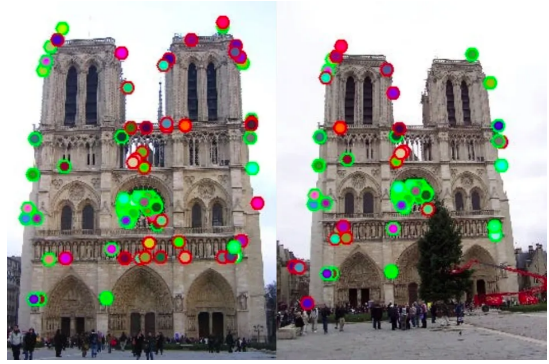


Figure: Extracting Key Feature Points

Feature Matching

- Find correspondences between frames.
- Match each key point to its equivalent in each point of view.
  - Template Matching
  - Optical Flow

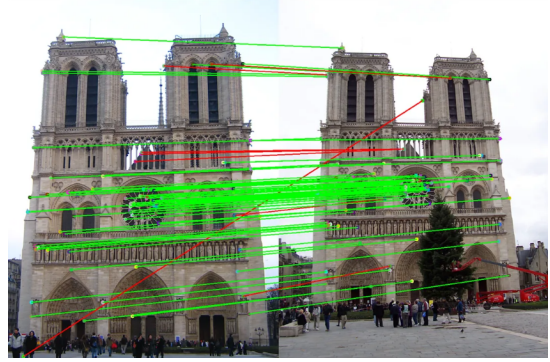
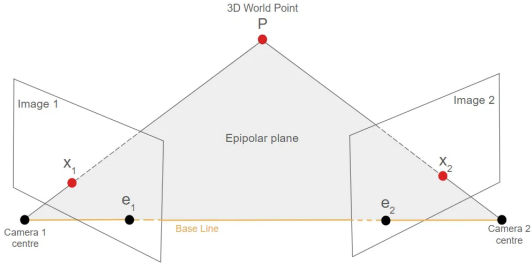


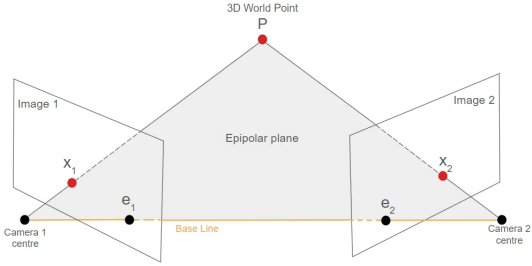
Figure: Feature Matching

Triangulation



- Compute *epipolar geometry* of the points using *fundamental matrix*
- With known projection matrices, *triangulation* computes the 3D location of a point from its 2D correspondences across views.
- Ideally, the 3D point lies at the intersection of the back-projected rays. However, due to noise, projected rays do not intersect.

Triangulation



- Compute *epipolar geometry* of the points using *fundamental matrix*
- With known projection matrices, *triangulation* computes the 3D location of a point from its 2D correspondences across views.
- Ideally, the 3D point lies at the intersection of the back-projected rays. However, due to noise, projected rays do not intersect.

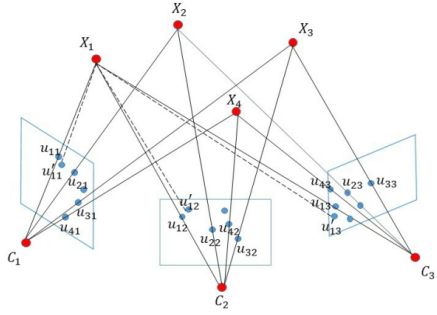
Need to Minimise an appropriate Error Metric !!!

Bundle Adjustment

Non - linear least sqaures optimisation problem

Objective !!!

To minimise the sum of errors between 2D observations and the predicted 2D points, where the predicted points are re-projected from 3D structures by camera parameters.



Bundle Adjustment

Non - linear least sqaures optimisation problem

3D $\rightarrow$ 2D mapping

- a function on intrinsics K, extrinsics R and t
- measurement affected by noise

Mathematically,

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} \left\| \mathbf{u}_{ij} - \pi(C_j, \mathbf{X}_i) \right\|^2$$

where,

- $\mathbf{u}_{ij}$: observed 2D keypoint of the i^{th} 3D point in j^{th} image.
- $\pi(C_j, \mathbf{X}_i)$: Projection of i^{th} 3D point $\mathbf{X}_i$ onto the j^{th} image
  - $\pi(C_j, \mathbf{X}_i) = K_j \cdot [R_j \mid t_j] \cdot \mathbf{X}_i$

Bundle Adjustment

Non - linear least sqaures optimisation problem

The reprojection error for a single point is,

$$r_{ij} = \mathbf{u}_{ij} - \pi(C_j, \mathbf{X}_i)$$

and the total reprojection error to be minimised is simplified as,

$$E = \sum_{i=1}^{n} \sum_{j=1}^{m} \|r_{ij}\|^2$$

The optimization problem is solved using the **Levenberg–Marquardt algorithm**

- Smoothly interpolates between the Gauss–Newton method and gradient descent
- Exploits the sparsity of Jacobian (of residuals)

Applications

Structure from Motion

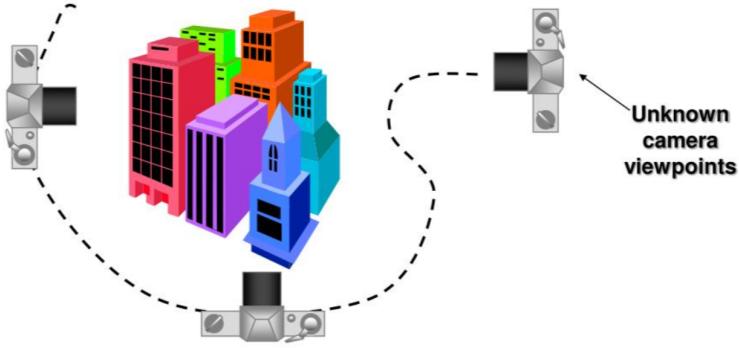


Figure: Structure from Motion

Reconstruct scene geometry and camera motion from multiple images of different views.

SLAM

- Simultaneous Localization and Mapping
- Computational problem of constructing or updating a map of an unknown environment
- Simultaneously keeps track of an agent's location within it

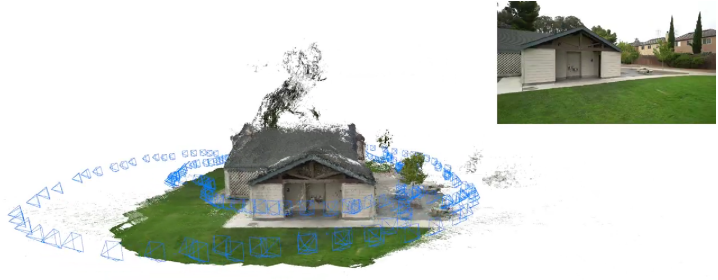


Figure: SLAM

Code Implementation

Future Scope

1. **Real-Time Reconstruction**
   - Integrate Bundle Adjustment with real-time SLAM systems for robotics and autonomous vehicles.
2. **Scalability to Large-Scale Scenes**
   - Develop distributed and parallel optimization methods to handle city-scale or drone-based mapping.
3. **Robustness to Noise and Occlusion**
   - Leverage deep learning models to improve feature matching and outlier rejection in challenging environments.

References

Bundle Adjustment Revisited

Yu Chen1, Yisong Chen1, Guoping Wang1
1 Peking University, Department of Computer Science and Technology,
Graphics and Interactive Lab, Beijing, China

***Abstract*—3D reconstruction has been developing all these two decades, from moderate to medium size and to large scale. It's well known that bundle adjustment plays an important role in 3D reconstruction, mainly in Structure from Motion(SfM) and Simultaneously Localization and Mapping(SLAM). While bundle adjustment optimizes camera parameters and 3D points as a non-negligible final step, it suffers from memory and efficiency requirements in very large scale reconstruction. In this paper, we study the development of bundle adjustment elaborately in both conventional and distributed approaches. The detailed derivation and pseudo codes are also given in this paper.**

I. Introduction

Bundle adjustment plays an important role in geodesy and 3D reconstruction(both SLAM and SfM), but it was deemed as solved until recently. Bundle adjustment constitutes a core component in most state-of-the-art multi-view geometry systems and is typically invoked as a final refinement stage to approximate initial scene estimates as well as a means for removing drift in incremental reconstructions. The Levenberg-Marquardt algorithm has proven to be the most successful method for solving this formulation, as it is simple to implement, robust to initialization, and its framework makes it very amenable to taking advantage of the forms of sparsity that typically arise in multi-view geometry problem. Each step of this algorithm produces an estimate of the parameters that improves upon the previous and the resulting series of iterates can be shown to converge to a local minimum of the objective function at hand.

As a result, a number of approaches to bundle adjustment have been proposed in the last decade. These approaches divide into two groups: the first branch focuses on making the bundle adjustment algorithm as efficient as possible, while the second focuses on reducing the size or frequency of invocation of individual bundle adjustment.

Bundle adjustment is the key to refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter estimates. Optimal means that the parameter estimates are found by minimizing some cost function that quantifies the model fitting error, and jointly that the solution is simultaneously optimal with respect to both structure and camera variations. The name refers to the 'bundle' of light rays leaving each 3D feature and converging on each camera center, which are 'adjusted' optimally with respect to both feature and camera positions[23].

Since the basic mathematics of the bundle adjustment problem are well understood[23] and several implementation-aimed works are given[15], [26], we can use some open-source software packages easily[15], [27], [3], [26], [2]. However, bundle adjustment is still a bottleneck in large-scale Structure from Motion because of matrix storage and frequent matrix manipulation[12], [25], [17], [20], [22], [24], [9]. The naive LM algorithm requires $O((m+n)^3$ operations for each iteration, and memory on the order of $O(mn(m+n))$. However, exploiting matrix structure and using the Schur complement approach, the number of arithmetic operations can reduced to $O(m^3+mn)$, and memory use to $O(mn)$. Further reduction can be achieved by exploiting secondary sparse structure[14]. The conjugate gradient approaches in [3], [6] can reduce the time complexity to $O(m)$ per iteration, making it essentially linear in the number of cameras. [15] uses dense Cholesky decomposition to solve the normal equation directly, and it needs much memory to store Jacobi and the reduced camera system, thus it's not suitable for large scale reconstruction work. [3] uses preconditioning on either hessian or schur complement and avoid the explicit storage of Jacobi, and the preconditioned conjugate gradient approaches[6], [5], [13] makes the procedure of solving the reduced camera system more stable because it has lower condition number than solving the Gauss-Newton[18] or Levenberg-Marquardt[16] problem directly.

Robust approaches[4], [7] are typically used to protect world point and camera parameter estimates from effect of outliers, which for BA are incorrect point correspondences that have gone undetected.

The development of hardware, especially GPU(Graphics Processing Unit), makes some works concentrate on the implementation of parallel bundle adjustment[26] and to accelerate the reconstruction.

However, when the data scale gets larger, e.g, city-scale reconstruction[28], [29], [30], the approaches discussed above cannot meet the memory and efficiency requirements. Thus, some works[28], [19], [11] concentrate on performing bundle adjustment in distributed manner. The limitation of memory can be avoided once we have enough computers. Besides, a large improvement in both accuracy and efficiency would happen. While[11] uses Douglas-Rachford splitting methods to split the conventional bundle adjustment problem into distributed manner, [28], [19] implements *ADMM*(Alternating Direction Method of Multipliers) to transform the original problem into a distributed one.

The main purposes of this paper are two folds:

- To give a detailed investigation and derivation of bundle adjustment problem, in both theoretical and practical levels.
- To show the development of bundle adjustment in parallel

Figure: Reference 1

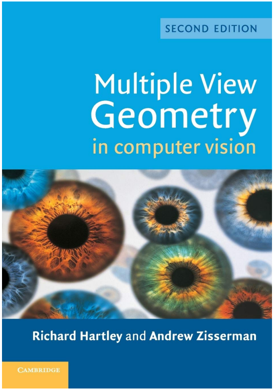


Figure: Reference 2

Thank You

soumyajoy.mds2023@cmi.ac.in
chandranath.mds2023@cmi.ac.in

cmi