Microsoft 3D Reconstruction

CV-Beautiful

TEAM MEMBERS

梁程捷	吳奕娃	張甡源	劉知微
B11901136	B11901080	B11901123	B10705054

Table of Contents

1. Traditional Pose Estimation
2. Fast3R

Feature Detection and Matching

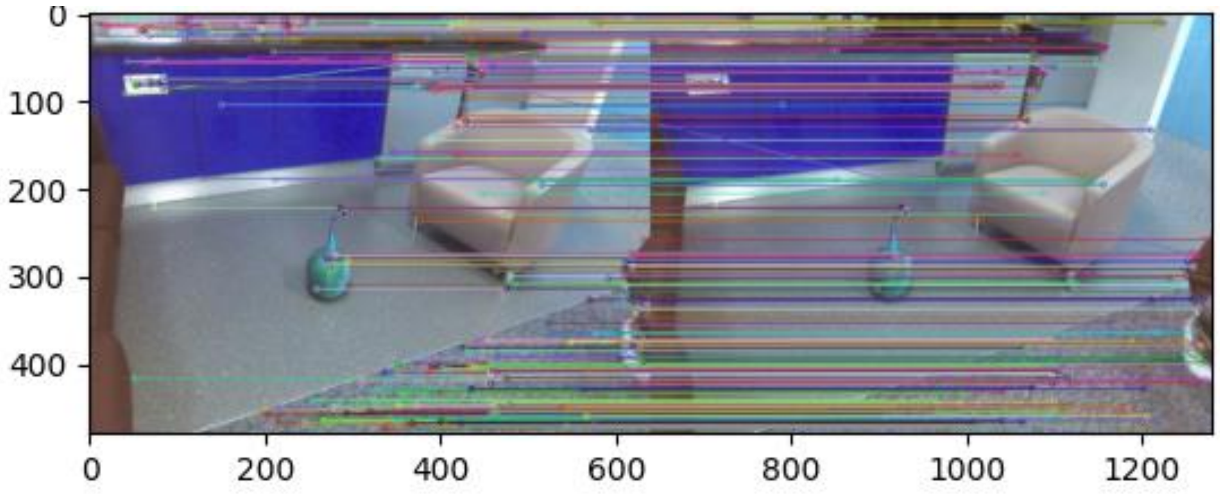
1. SIFT keypoints and descriptors are extracted from **grayscale** RGB images using OpenCV

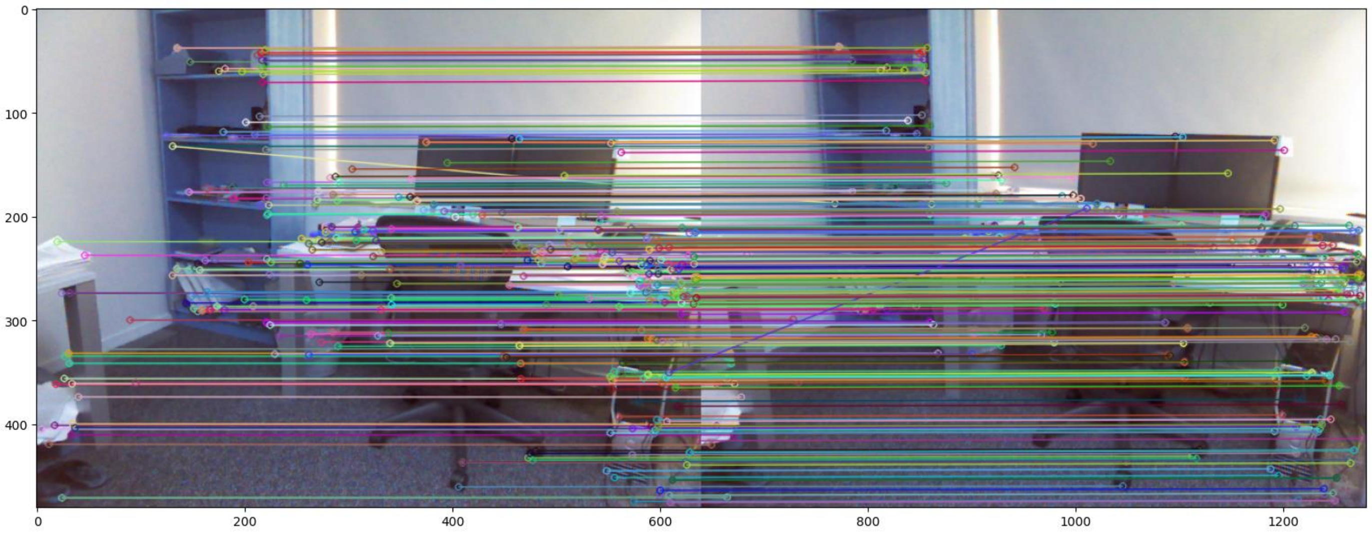
$$\boxed{\texttt{kp, des} = \texttt{sift.detectAndCompute(gray, None)}}$$

2. Feature descriptors are matched using Lowe's ratio test:

$$\boxed{m.\text{distance} < 0.75 \times n.\text{distance}}$$

Feature Detection and Matching





Rigid Alignment via Procrustes

1. Before rigid alignment, matched 2D points are projected back into 3D points using the depth map.

   This gives the 3D correspondence: $\boxed{\{\mathbf{X}_1, \mathbf{X}_2\}.}$

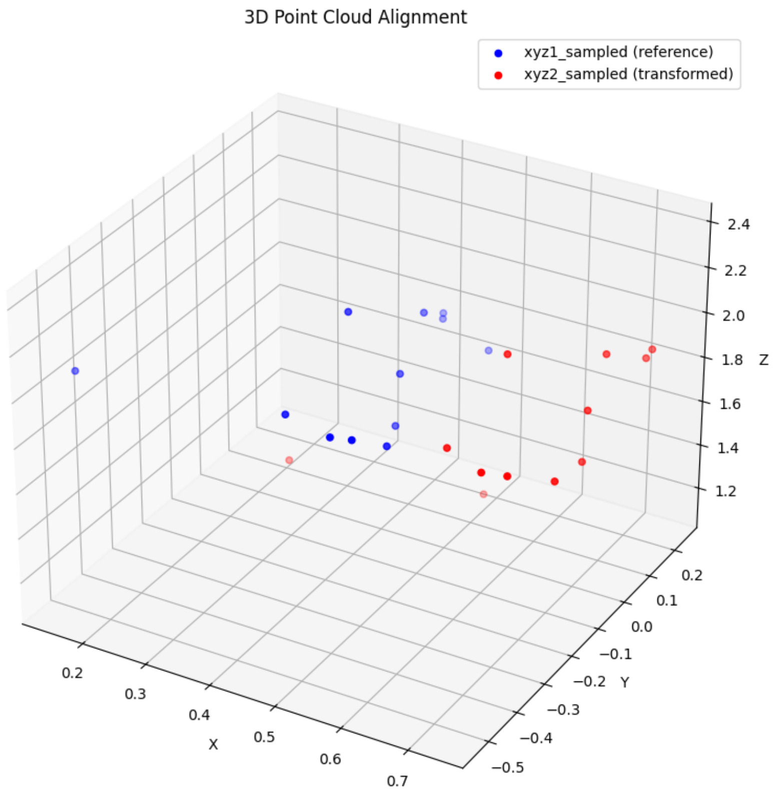
2. To estimate the transformation between two camera poses / frame:

$$\boxed{\min_{\mathbf{R},\mathbf{t}} \|\mathbf{X_2} - (\mathbf{R}\mathbf{X_1} + \mathbf{t})\|^2}$$

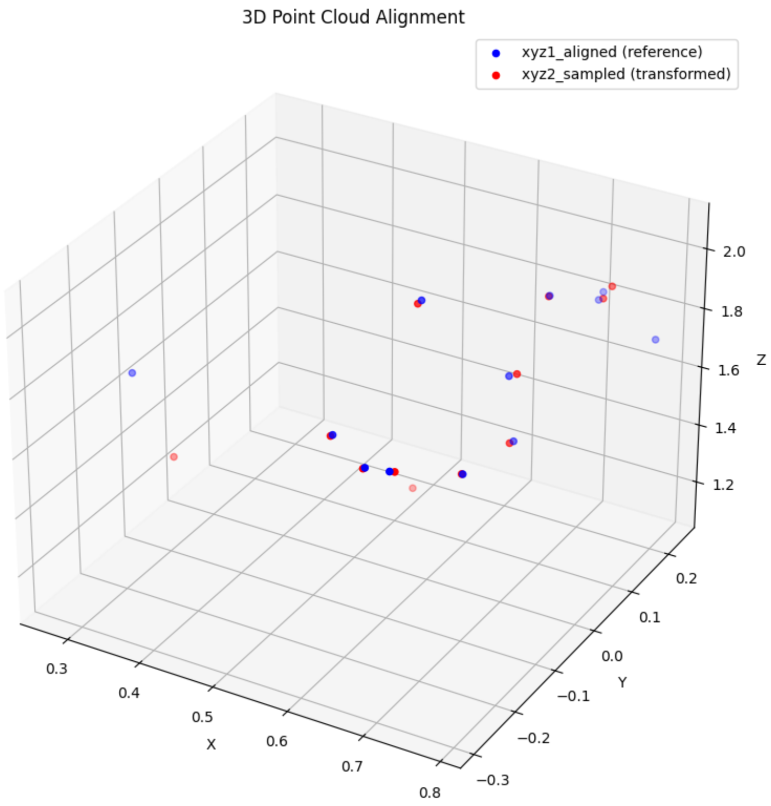
   Analytic solution via singular value decomposition (SVD):

$$\begin{aligned}
\mathbf{X_1}' &= \mathbf{X_1} - \overline{\mathbf{X_1}}, \quad \mathbf{X_2}' = \mathbf{X_2} - \overline{\mathbf{X_2}} \\
\mathbf{H} &= \mathbf{X_1}'^T \mathbf{X_2}' \\
\mathbf{H} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\
\mathbf{R} &= \mathbf{V}\mathbf{U}^T \\
\mathbf{T} &= \overline{\mathbf{X_2}} - \mathbf{R}\overline{\mathbf{X_1}}
\end{aligned}$$

Rigid Alignment via Procrustes



after alignment



Bundle Adjustment (Pose Refinement)

Minimize the reprojection error: $$\boxed{\text{error} = \sum_i \rho\left(\left\|\mathbf{x}_i^{\text{obs}} - \pi\left(\mathbf{R}, \mathbf{t}, \mathbf{X}_i\right)\right\|^2\right)}$$

where:

- ρ() is a loss function:
  - mean-squared error is used in **dense** reconstruction
  - huber-loss is used in **sparse** reconstruciton
- projection function: $$\boxed{\pi(\mathbf{R}, \mathbf{t}, \mathbf{X}) = \frac{1}{Z}\mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t})}$$

Note: Using cv2.Rodrigues, the rotation matrix is reparameterized as a rotation vector and optimized using scipy.optimize.least squares.

Output

1. Each frame’s pose is chained from the previous one:

$$\boxed{\mathbf{T}_i = \mathbf{T}_{i-1} \cdot \mathbf{H}^{-1}}$$

   where H is the transformation from frame i to $i-1$

2. The function returns a list of poses

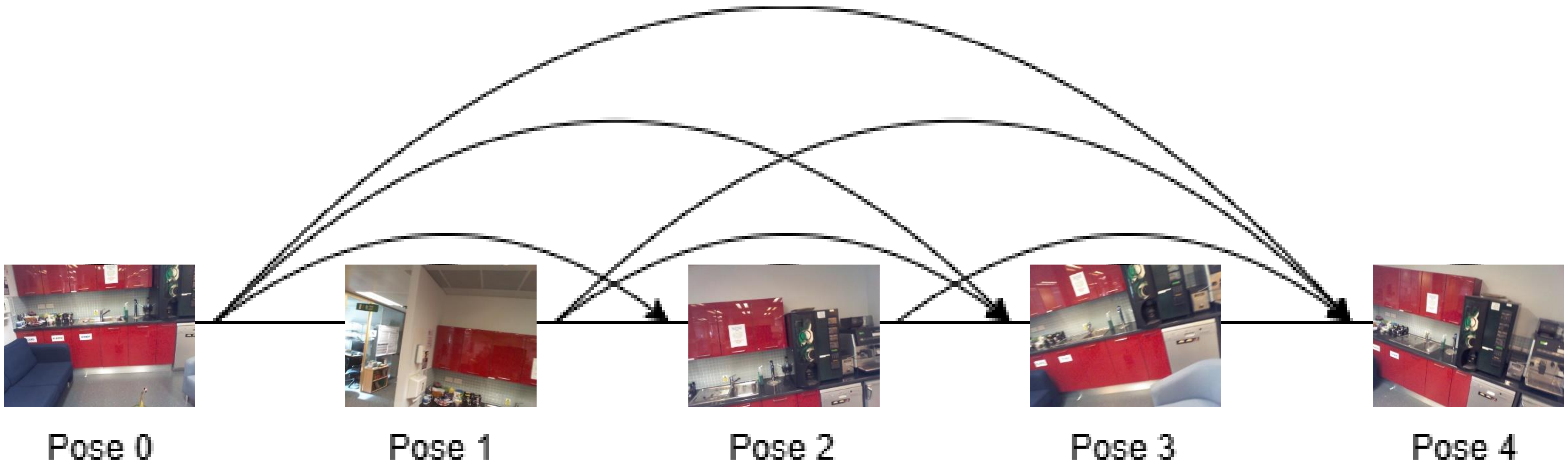
$$\boxed{\mathbf{T}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ \mathbf{0}^T & 1 \end{bmatrix}}$$

   where each pose represents the camera-to-world transformation of frame i

Bonus (Sparse Reconstruction)

Moreover, the sequence given are randomly ordered, if the frames are processed sequentially, the frames processed later know nothing about other frames.

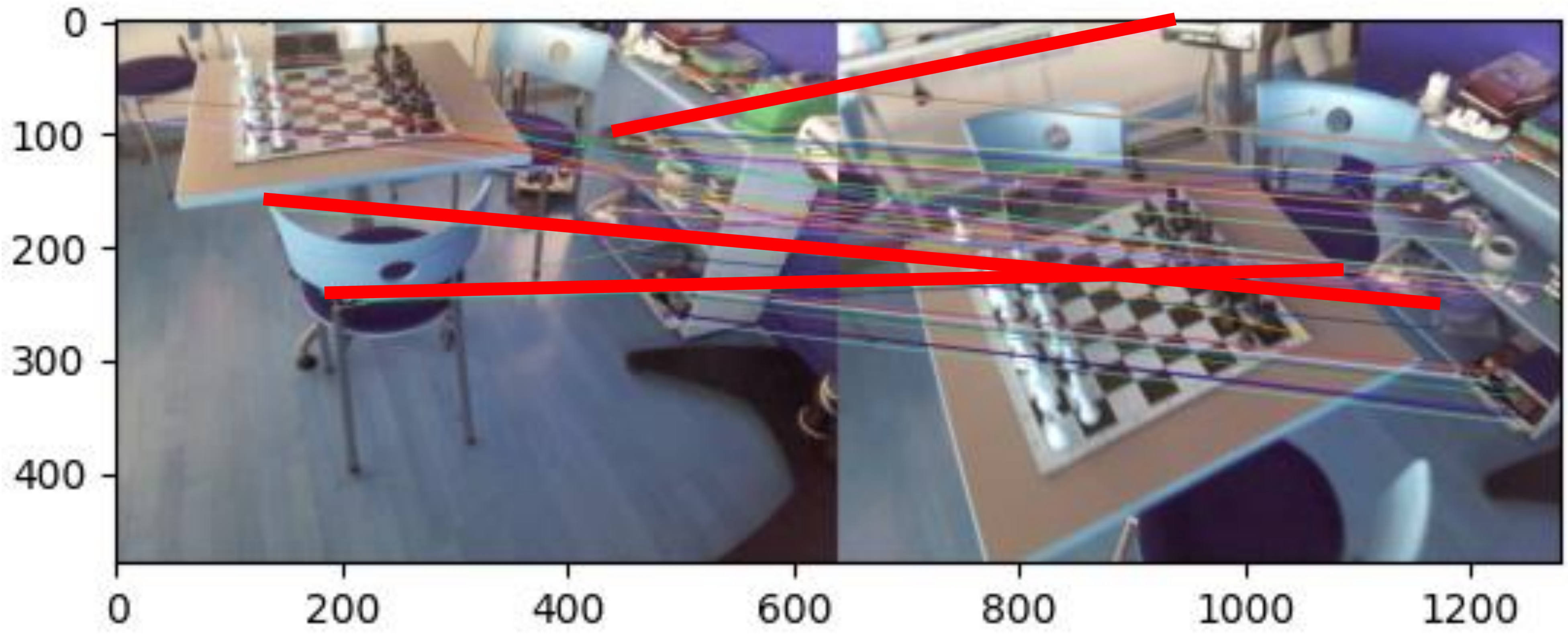
Idea: Build a Pose Graph



Bonus (Sparse Reconstruction)

Angle of view between the frames are very different from one another, mismatch of feature points occurs more often.

Noisy features !!



Noisy Feature

1. After feature matching and depth projection, we have a list of 3D feature points.
2. Trick:
   - Randomly sample 8 feature points from the list.
   - Perform rigid alignment as before, then compute the squared error between the aligned and unaligned points.
   - Repeat this process for 20 times and take the relative pose with least squared error.

Idea: Using partial feature points removes most of the outliers -> noise reduction

Experimental Results

1. Dense Reconstruction:
    - Accuracy: 0.0044
    - Completeness: 0.4937
2. Sparse Reconstruction
    - Avg accuracy : 0.0289
    - Avg completeness: 0.0575

Comparison	Metric	Mean
chess-sparse-seq-05	Accuracy	0.0422
	Completeness	0.0294
stairs-sparse-seq-04	Accuracy	0.0248
	Completeness	0.0165
pumpkin-sparse-seq-07	Accuracy	0.0176
	Completeness	0.1153
fire-sparse-seq-04	Accuracy	0.0312
	Completeness	0.0690

Fast3R

1. Efficient 3D pose estimation and reconstruction for large-scale RGB-D datasets using optimized pose graph and fast bundle adjustment.
2. Advantages:
   - **Scalable:** Handles 1000+ images.
   - **Optimized Pose Graph:** Enhances trajectory robustness and accuracy.
   - **Fast Bundle Adjustment:** Cuts computation time vs. traditional methods.
   - **Resource-Efficient:** Runs on GPUs with limited memory.
   - **Accurate & Reliable:** Robust 3D reconstructions in complex sequences.

Experimental Results

1. Accuracy: 0.0435, Completeness: 0.6101
2. Conclusion: Despite Fast3R's design advantages, it produced higher error than traditional pose estimation (**0.00257**, **0.513**), suggesting influences from dataset characteristics, parameter settings, or hardware constraints.

References

1. Garigali, D. et al. *3D Reconstruction*, GitHub repository, 2020. Available: https://github.com/dgarigali/3D_Reconstruction
2. Yang, J. et al. *Fast3R: Towards 3D Reconstruction of 1000+ Images in One Forward Pass*, CVPR 2025.