Bundle Adjustment

Structure from Motion (SfM)

- Recovering the 3D geometry or “structure” of the scene and the camera motion from a set of 2D images when a camera is subject to “motion”
  - The structure refers to the 3D world coordinates of the captured points
  - The motion refers to the 3D world coordinates from which various images were captured from the camera(s)

Bundle Adjustment (BA)

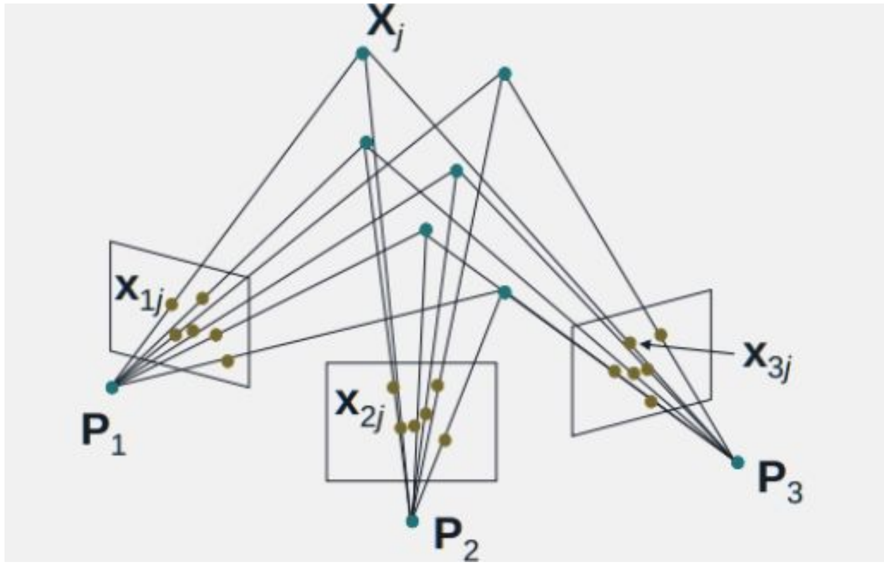
- This is an optimisation algorithm to solve the problem of SfM
- The non-linear least squares constraint problem which we discussed previously is used to model the cost function in the Bundle Adjustment setting

Initialisation for BA

- The following approaches help provide initialisation to BA: (brief overview)
  - Real world information like IMU, Odometry from which we obtain our initial guess for poses. Then by triangulation, obtain initial guess of 3D points as intial guess for BA
  - Fundamental Matrix estimation using the 8 point algorithm on the first pair of images to get the relative poses. Again, by triangulation we can obtain guesses for the 3D points from P3P
  - The relative poses and 3D point estimates can also be obtained by just pairwise fundamental matrix estimation

Reprojection Error

- What we have:
  - m cameras
  - n points
  - Known correspondences across all these cameras and points
- Unknowns:
  - m matrices P_i
  - n coordinates X_i

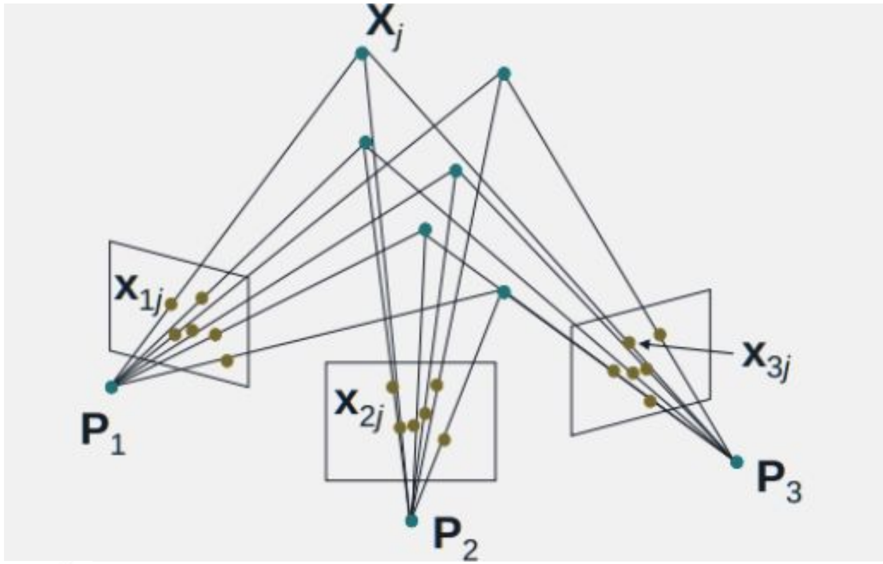


- Known:
  - Data association, i.e., correspondences across images which relate the pixels that correspond to the same real world point taken from various cameras and hence, viewpoints

Reprojection Error

- What we have:
  - m cameras
  - n points
  - Known correspondences across all these cameras and points
- The camera model expressed in homogeneous coordinates is given by the following set of equations

$$\vec{\mathbf{x}}_{ij} = \lambda_{ij} P_i \vec{X}_j, \quad 1 \le i \le m, 1 \le j \le n$$



- And finally, the reprojection error is given by $\|\lambda_{ij} P_i \vec{X}_j - \vec{\mathbf{x}}_{ij}\|^2$, which denotes how closely the estimated 3D point resembles its actual projection
  - The lambdas are scaling factors to ensure that the third coordinate in the image coordinates in homogeneous system is 1

An Example

- Given that we have 10k images with 1k points each. Each point is seen 10 times on avg
- How many known and unknown parameters do we have?
- Known parameters: 20M
  - These are the image coordinates: 2 * 10k * 1k = 20M known params
  - Why are we multiplying by 2? Pixel coordinates
- Unknown parameters: ~13M
  - No of world points = (10k * 1k) / 10 = 1M. Division by 10 as the same point is repeated in 10 images on average. Hence, we have 1M * 3 = 3M world point params
    - Why are we multiplying by 3? World coordinates
  - For each of the 10k images, we have 10k corresponding orientations of the camera that captured the image. Hence, 10k * 6 = 60k orientation params
  - For each of the image points, there is a single corresponding scaling factor. Hence, the scaling factors contribute to 10k * 1k = 10M scaling params

Can we do better?

- Get rid of the scale parameters
- How? Convert from homogeneous coordinate system to Euclidean coordinate system

Cost Function

$$\arg\min_{\vec{X}j,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left(\left[\frac{P_{i11}X_j + P_{i12}Y_j + P_{i13}Z_j + P_{i14}}{P_{i31}X_j + P_{i32}Y_j + P_{i33}Z_j + P_{i34}} - x_{ij}\right]^2 + \left[\frac{P_{i21}X_j + P_{i22}Y_j + P_{i23}Z_j + P_{i24}}{P_{i31}X_j + P_{i32}Y_j + P_{i33}Z_j + P_{i34}} - y_{ij}\right]^2 \right)$$

$$\arg\min_{\vec{X}j,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left\| \frac{P_{(1:2)i}\vec{X}_j}{P_{3i}\vec{X}_j} - \vec{x}_{ij} \right\|^2$$

Cost Function

$$\arg\min_{\vec{X}j,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left(\left[\frac{P_{i11}X_j + P_{i12}Y_j + P_{i13}Z_j + P_{i14}}{P_{i31}X_j + P_{i32}Y_j + P_{i33}Z_j + P_{i34}} - x_{ij} \right]^2 + \left[\frac{P_{i21}X_j + P_{i22}Y_j + P_{i23}Z_j + P_{i24}}{P_{i31}X_j + P_{i32}Y_j + P_{i33}Z_j + P_{i34}} - y_{ij} \right]^2 \right)$$

$$\arg\min_{\vec{X}j,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left\| P_i \vec{X}_j - \vec{x}_{ij} \right\|^2$$

$$\arg\min_{\vec{X}j,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left\| \frac{P_{(1:2)i}\vec{X}_j}{P_{3i}\vec{X}_j} - \vec{x}_{ij} \right\|^2$$

$$\arg\min_{Xj,P_i} \sum_{i=1}^{M}\sum_{j=1}^{N} \left\| \widehat{\vec{x}}_{ij} - \vec{x}_{ij} \right\|^2$$

Reprojection Error

- P_i is the projection matrix of the ith view and X□ is the world coordinate of the jth world point
- x_i□ is the known observation while $\hat{x}_{ij}$ is the predicted projection of the jth world point in the ith view

What do we finally have?

- We have 2mn equations in total (2 for each match)
- This is a non-linear optimisation problem wherein we minimise the sum of the squared reprojection errors of the reconstructed n 3D points over m images
- How do we solve this familiar looking formulation of a non-linear least squares cost function? **Levenberg-Marquardt Algorithm**

$$\left(\mathbf{J}^\top \mathbf{J} + \lambda \mathbf{I}\right) \Delta \mathbf{k} = -\mathbf{J}^\top \mathbf{r}(\mathbf{k})$$

- Here r is the residual vector, which is the difference vector between the observed image coordinate vector and the predicted projection of the corresponding world point through the view

$$\boxed{\hat{\vec{x}}_{ij} = P_i \vec{X}_j}$$

Objective Function:

$$\boxed{F(k) = r^T r}$$

$$\boxed{\arg\min \sum_{i=1}^{M} \sum_{j=1}^{N} \left\| \hat{\vec{x}}_{ij} - \vec{x}_{ij} \right\|^2}$$

Residual vector

Understanding RESIDUAL VECTOR

$$
r = \begin{bmatrix}
\vec{r}_{11} & \left< \right. & f_{11} & \longrightarrow & \hat{x}_{11} - x_{11} \\
 & & g_{11} & \longrightarrow & \hat{y}_{11} - y_{11} \\
\vec{r}_{12} & \left< \right. & f_{12} & \longrightarrow & \hat{x}_{12} - x_{12} \\
 & & g_{12} & \longrightarrow & \hat{y}_{12} - y_{12} \\
\vec{r}_{13} & \left< \right. & \vdots & \vdots & \vdots \\
\vdots & & \vdots & & \\
\vec{r}_{1N} & \left< \right. & f_{1N} & \longrightarrow & \hat{x}_{1N} - x_{1N} \\
 & & g_{1N} & \longrightarrow & \hat{y}_{1N} - y_{1N} \\
\vec{r}_{21} & \left< \right. & f_{21} & \vdots & \vdots \\
 & & g_{21} & & \\
\vdots & & \vdots & \longrightarrow & \\
\vec{r}_{2N} & \left< \right. & f_{2N} & \longrightarrow & \\
 & & g_{2N} & & \\
\vdots & & \vdots & \vdots & \vdots \\
\vec{r}_{M1} & \left< \right. & f_{M1} & \longrightarrow & \\
 & & g_{M1} & \longrightarrow & \\
\vdots & & \vdots & \vdots & \\
\vec{r}_{MN} & \left< \right. & f_{MN} & \longrightarrow & \\
 & & g_{MN} & \longrightarrow & \\
\end{bmatrix}
$$

2N

2N

⋮ M times

$\downarrow$

$$\boxed{\text{2MN terms}}$$

(Side note: Note that $\dfrac{\partial f_{11}}{\partial k} = \dfrac{\partial \hat{x}_{11}}{\partial k}$)

$$\boxed{\hat{\vec{x}}_{ij} = P_i \vec{X}_j}$$

Residual vector

$$\boxed{\text{Objective Function: } F(k) = r^T r}$$

$$\boxed{\arg\min \sum_{i=1}^{M} \sum_{j=1}^{N} \left\| \hat{\vec{x}}_{ij} - \vec{x}_{ij} \right\|^2}$$

$$J = \frac{\partial r}{\partial k} \qquad \Big[\underbrace{P_1}_{P_{1_{11}} P_{1_{12}} P_{1_{13}} \cdots P_{1_{34}}} \quad P_2 \quad \cdots \quad \underbrace{P_M}_{P_{M_{11}} P_{M_{12}} \cdots P_{M_{34}}} \quad \underbrace{\vec{X}_1}_{X_1 Y_1 Z_1} \quad \vec{X}_2 \quad \cdots\cdots \quad \underbrace{\vec{X}_N}_{X_N Y_N Z_N} \Big]$$

$$\boxed{\text{INDIVIDUAL JACOBIANS: } J_{ij_m},\ G_{ij_m},\ J_{ij_s},\ G_{ij_s}} \Rightarrow$$

'm'otion

$$J_{ij_m} = \left[\frac{\partial f_{ij}}{\partial P_i}\right] = \left[\frac{\partial f_{ij}}{\partial P_{i_{11}}} \quad \cdots\cdots \quad \frac{\partial f_{ij}}{\partial P_{i_{34}}} \right]$$

$$G_{ij_m} = \left[\frac{\partial g_{ij}}{\partial P_i}\right] = \left[\frac{\partial g_{ij}}{\partial P_{i_{11}}} \quad \cdots\cdots \quad \frac{\partial g_{ij}}{\partial P_{i_{34}}} \right]$$

's'tructure

$$J_{ij_s} = \left[\frac{\partial f_{ij}}{\partial \vec{X}_j}\right] = \left[\frac{\partial f_{ij}}{\partial X_j} \quad \frac{\partial f_{ij}}{\partial Y_j} \quad \frac{\partial f_{ij}}{\partial Z_j} \right]$$

$$G_{ij_s} = \left[\frac{\partial g_{ij}}{\partial \vec{X}_j}\right] = \left[\frac{\partial g_{ij}}{\partial X_j} \quad \frac{\partial g_{ij}}{\partial Y_j} \quad \frac{\partial g_{ij}}{\partial Z_j} \right]$$

So, from above, notice when we write J_{12_m} simply (Notice 'm') means derivative w.r.t P_1. And J_{12_s} would mean derivative w.r.t $\vec{X}_2$.

$$\boxed{J_{2MN,\,12M+3N} = \frac{\partial r}{\partial k}}$$

MOTION STRUCTURE

$$= \Big[\underbrace{\frac{\partial r}{\partial P_1} \;\; \frac{\partial r}{\partial P_2} \;\cdots\cdots\; \frac{\partial r}{\partial P_M}}_{M \text{ times}} \;\Big|\; \underbrace{\frac{\partial r}{\partial \vec{X}_1} \;\; \frac{\partial r}{\partial \vec{X}_2} \cdots\cdots \frac{\partial r}{\partial \vec{X}_N}}_{N \text{ times}}\Big]$$

$$= \left[\begin{array}{c|cccc|cccc}
 & & & & & & & & \\
\multirow{8}{*}{$2N$} & \underset{1\times12}{J_{11_m}} & \underset{1\times12}{0} & \underset{1\times12}{0} \cdots & \underset{1\times12}{0} & \underset{1\times3}{J_{11_s}} & \underset{1\times3}{0} \cdots & & \underset{1\times3}{0} \\
 & \underset{1\times12}{G_{11_m}} & 0 & \cdots & 0 & G_{11_s} & 0 & \cdots & 0 \\
 & J_{12_m} & \vdots & \cdots & 0 & 0 & J_{12_s} & \cdots & 0 \\
 & G_{12_m} & \vdots & \cdots & 0 & 0 & G_{12_s} & \cdots & 0 \\
 & \vdots & \vdots & & & \vdots & & & \\
 & J_{1N_m} & \vdots & \cdots & 0 & 0 & \cdots & \cdots & J_{1N_s} \\
 & G_{1N_m} & & \cdots & 0 & 0 & \cdots & \cdots & G_{1N_s} \\
\hline
\multirow{4}{*}{$2N$} & 0 & J_{21_m} & \cdots & 0 & J_{21_s} & 0 & \cdots & 0 \\
 & 0 & G_{21_m} & \cdots & 0 & G_{21_s} & 0 & \cdots & 0 \\
 & \vdots & \vdots & & & 0 & J_{22_s} & \cdots & \\
 & & & & & \vdots & G_{22_s} & \cdots & \\
 & 0 & G_{2N_m} & \cdots & 0 & 0 & \vdots & & J_{2N_s} \\
 & & & & & & & & G_{2N_s} \\
\hline
 & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\hline
 & 0 & 0 & 0 & J_{M1_m} & J_{M1_s} & 0 & \vdots & 0 \\
 & 0 & \vdots & \vdots & G_{M1_m} & G_{M1_s} & 0 & \vdots & 0 \\
 & \vdots & \vdots & \vdots & \vdots & 0 & & & \vdots \\
 & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & J_{MN_s} \\
 & 0 & & & G_{MN_m} & 0 & \vdots & & G_{MN_s}
\end{array}\right]$$

Update vector

$$\Delta k \uparrow \quad \left[\begin{array}{l}
\left.\begin{array}{l} \delta P_{1_{11}} \\ \vdots \\ \delta P_{1_{34}} \end{array}\right\} 1^{st} \text{ camera} \\
\left.\begin{array}{l} \delta P_{2_{11}} \\ \vdots \\ \delta P_{2_{34}} \end{array}\right\} 2^{nd} \text{ camera} \\
\vdots \\
\left.\begin{array}{l} \delta P_{M_{11}} \\ \vdots \\ \delta P_{M_{34}} \end{array}\right\} M^{th} \text{ cam} \\
\hline
\left.\begin{array}{l} \delta X_1 \\ \delta Y_1 \\ \delta Z_1 \end{array}\right\} 1^{st} \text{ 3D point} \\
\vdots \\
\left.\begin{array}{l} \delta X_N \\ \delta Y_N \\ \delta Z_N \end{array}\right\} N^{th} \text{ 3D point}
\end{array}\right]$$

$$\overset{(12M+3N,\,12M+3N)}{\Big[\underbrace{J^T J}_{(12M+3N,\,2MN)\,\times\,(2MN,\,12M+3N)} + \lambda I\Big]} \; \underbrace{\Delta k}_{12M+3N,\,1} = - \underbrace{J^T}_{(12M+3N,\,2MN)} \; \underbrace{r(k)}_{2MN,\,1}$$

A Short Detour - Lie Algebra

- For Transformation Matrices, it's better to work in the Lie Space than in the R12 or R3 space due to the specific properties of Rotation Matrices, more details in the corresponding chapters in the pdfs provided in the folder. (Note: This is not compulsory, you can do it if you are interested - the derivatives given are mostly for ICP-SLAM and not SFM, might need to change it a little bit to adapt it for this task)

A few resources:

- ICP-SLAM-ShubhodhFin.pdf : More ICP SLAM than SFM, but a good read nonetheless to understand Lie Algebra
- https://karnikram.info/blog/lie/
- slambook-en.pdf : Chapter 3
- OptimizationOverManifolds.pdf: Page 39 onwards (Go in this particular order, would make things easier to understand)
- https://github.com/princeton-vl/lietorch

Demos

1. COLMAP
   https://github.com/colmap/colmap

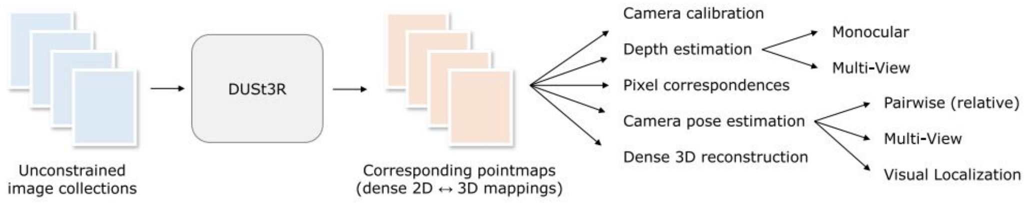
2. HLoc
   https://github.com/cvg/Hierarchical-Localization

What if we trained a Network for this task?

1. Dust3r
   https://europe.naverlabs.com/research/publications/dust3r-geometric-3d-vision-made-easy/

2. Mast3r
   https://europe.naverlabs.com/research/publications/grounding-image-matching-in-3d-with-mast3r/

Dust3r



Trained a transformer network to predict a joint pointmap given a pair of images.

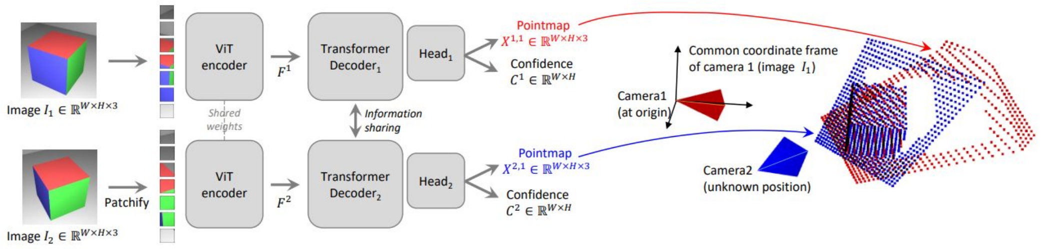


Image $I_1 \in \mathbb{R}^{W\times H\times 3}$
Image $I_2 \in \mathbb{R}^{W\times H\times 3}$
Patchify
ViT encoder
Shared weights
ViT encoder
F^1
F^2
Transformer Decoder$_1$
Transformer Decoder$_2$
Information sharing
Head$_1$
Head$_2$
Pointmap $X^{1,1} \in \mathbb{R}^{W\times H\times 3}$
Confidence $C^1 \in \mathbb{R}^{W\times H}$
Pointmap $X^{2,1} \in \mathbb{R}^{W\times H\times 3}$
Confidence $C^2 \in \mathbb{R}^{W\times H}$
Common coordinate frame of camera 1 (image I_1)
Camera1 (at origin)
Camera2 (unknown position)

The outputs are the 3D points in the first image's frame for each pixel, along with the confidence of the 3D point for each pixel

Mast3r

1. Trains a new decoder Head to predict pixel correspondences.
2. Dust3r's decoder was originally scale invariant, it trains it with metric data now to predict metric depth, so that it can be used for pose estimation as well.

Grounding Image Matching in 3D with MASt3R

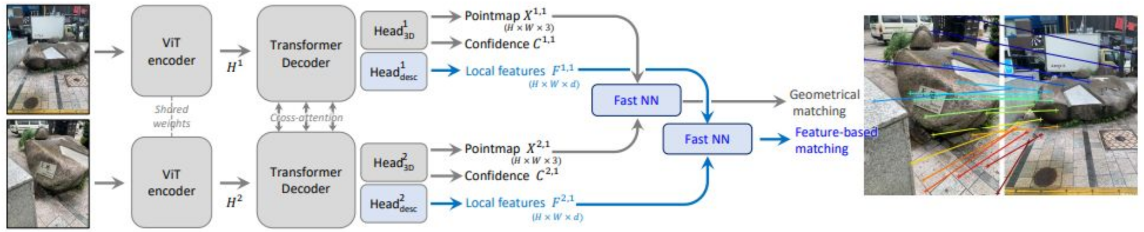


Figure 2: Overview of the proposed approach. Given two input images to match, our network regresses for each image and each input pixel a 3D point, a confidence value and a local feature. Plugging either 3D points or local features into our fast reciprocal NN matcher (3.3) yields robust correspondences. Compared to the DUSt3R framework which we build upon, our contributions are highlighted in blue.

Questions?

Acknowledgements

• Borrowed most of the slides from the SfM notion doc by Sai Shubodh

References

• Muti view geometry by Hartley and Zisserman

• RRC Summer School 2023

• Bundle Adjustment – Part I Introduction & Application - Cyril Stachniss

• Bundle Adjustment – Part II Numerics of BA - Cyril Stachniss

• Lecture: Photogrammetry I & II (2021, Uni Bonn, Cyrill Stachniss) - Lectures 53, 54, 55

Further Readings

• Incremental SfM, Failure cases of SfM, Multi View Stereo, and much more

  ○ Structure from motion, L. Lazebnik, N. Snavely, M. Herbert

  ○ Structure from Motion, Derek Hoiem

Thank You!