Stereo + Structure from Motion

Comments	
Dates Taught	@November 3, 2020
Lecture No.	L21 L22
Links of Videos	L21-Theory, L21-Discussion
Module	SLAM: Vision
Related to All Questions (Property)	

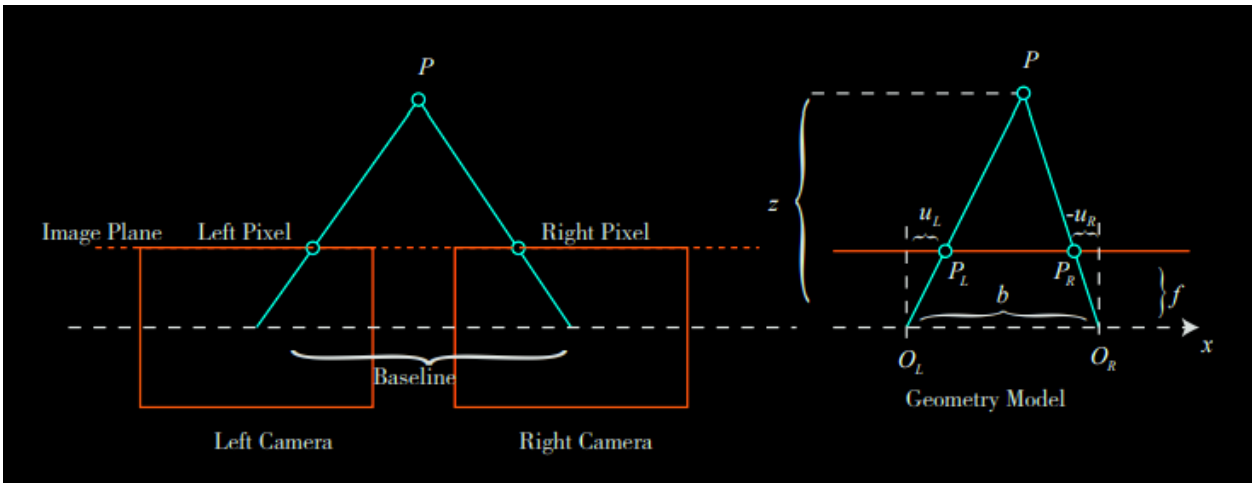
Agenda for 3/11: Stereo + Bundle Adjustment

- Understand basic stereo concept
- Understand the theory behind SfM
- Nice demo of an SfM library COLMAP

Next class: Present what you learnt so far coherently to implement Structure from Motion (SfM pipeline walkthrough)

1. Stereo Camera

> Simple experiment: Close one eye, observe one thumb at arm's length. Now switch your closed eye.



Source: Trucco, Alessandro Verri. n.d. Introductory Techniques for 3-D Computer Vision-Prentice Hall (1998). Chapter 7, Stereopsis.

💡 Just a special case of Epipolar Geometry (See Triangulation).

Finding the world point!

$$\frac{z-f}{z} = \frac{b-u_L+u_R}{b}$$
$$(Clue \rightarrow \triangle PP_LP_R \text{ and } \triangle PO_LO_R)$$

$$\implies z = \frac{fb}{d}, \quad d \triangleq u_L - u_R$$

▼ Not to get confused about u_R:

$where \;\; d \rightarrow disparity/parallax$

u_R is the "x" coordinate as measured in the O_R coordinate system. So $-u_R$ is +ve value in the above figure.

- Disparity: The "shift" that you observed when you switched your closed eye.

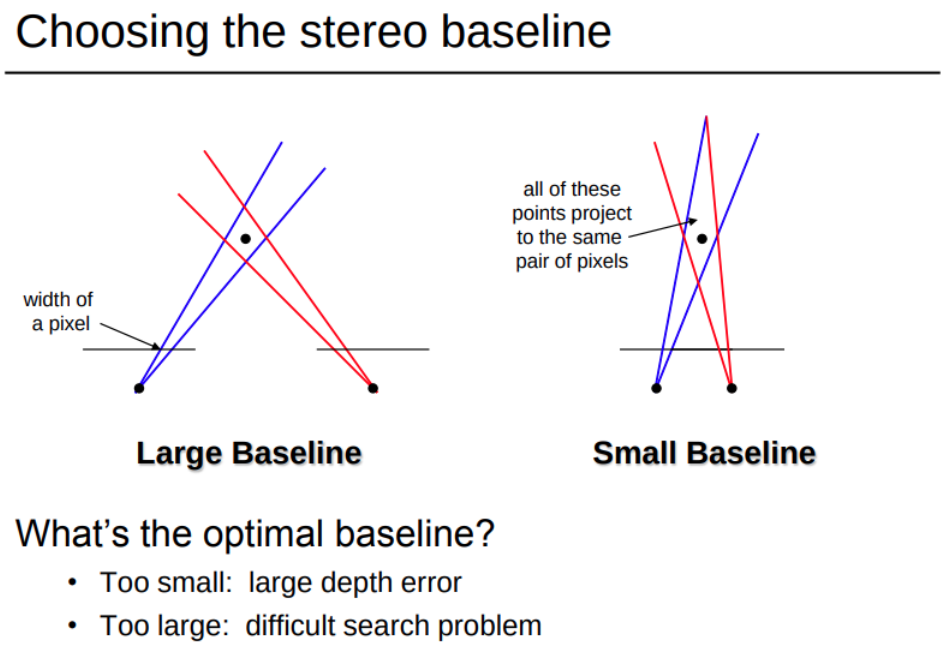
  You can think of u_L and u_R as distances relative to an absolute frame (attached to your open eye) along x-axis and thus, $u_L - u_R$ is the relative "shift" or disparity.

- When is d zero i.e. $u_L = u_R$?

  (note above to avoid confusion)

  - Both rays (normalized image coordinates) are parallel. (for example, sun at ∞.)

- Optimal Baseline



Source

- `Stereo-Homework-1:` Derive the above equation using triangulation concept
- `Stereo-Homework-2:` ***Pop Quiz***

Pop Quiz

I go see Avatar 3D with a friend.
His eyes are further apart than mine.

me

friend

Who sees objects as being closer?

Source

2. Structure from Motion

> Structure-from-motion

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".

> Bundle adjustment

An optimization algorithm used to solve it.

2.1 Brief Overview of SfM

https://www.youtube.com/watch?v=i7ierVkXYa8

💡 Bundle Adjustment problem is VERY SIMILAR to the visual SLAM problem. SLAM is some sense more general, taking different sensor modalities, different motion models into account. BA is one instance of SLAM problem.

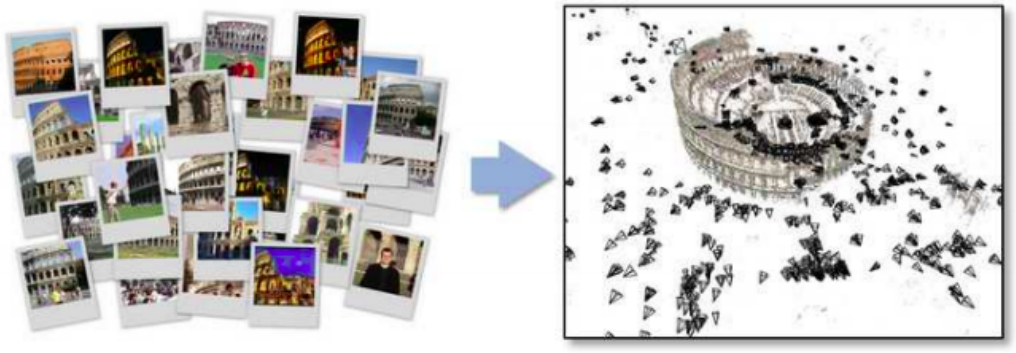
2.2 Initialization for BA

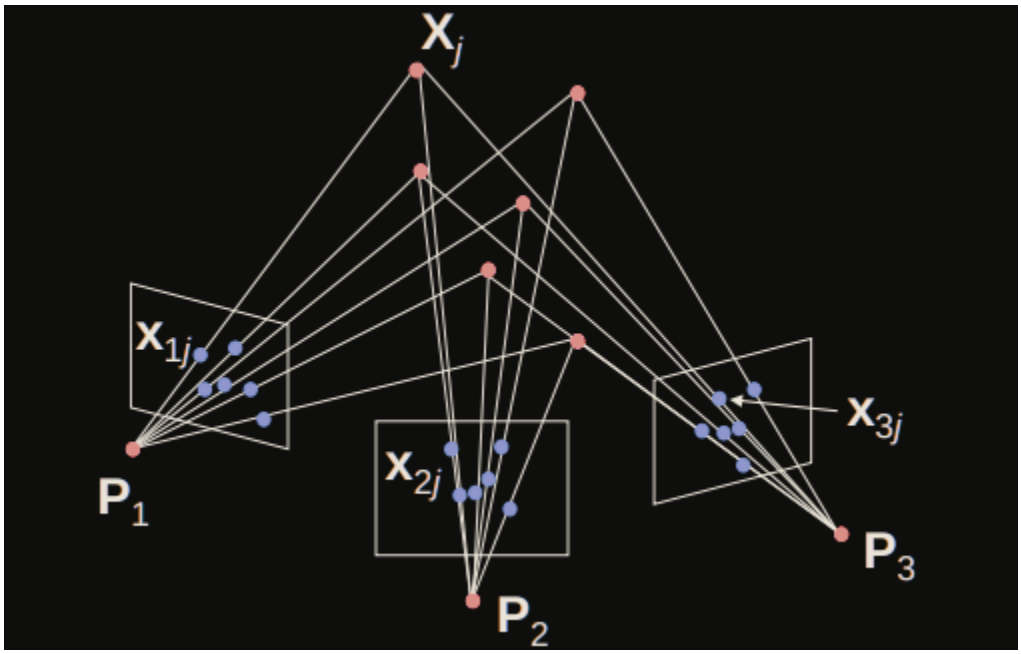
1. Real world information like IMU, Odometry → initial guess of poses; and by triangulation, obtain initial guess of 3D points as intial guess for BA
2. F estimation (8 point algorithm) first for 1st pair of images then triangulation then P3P. Or just pairwise F estimation.

(We will discuss the implementation level issues in the next class.)

2.3 Formulation

2.3.1 Reprojection Error





- m cameras and n points, with correspondences
- Unknown: m matrices P_i and n coordinates, $\vec{X}_j$
- When expressed in homogenous coordinates, camera model is given by:

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

- Known data association

Reprojection Error:

$$\| \lambda_{ij} P_i \vec{X}_j - \vec{\mathrm{x}}_{ij} \|^2$$

How closely an **estimate of a 3D point** recreates the point's **true projection**

An example:

Source

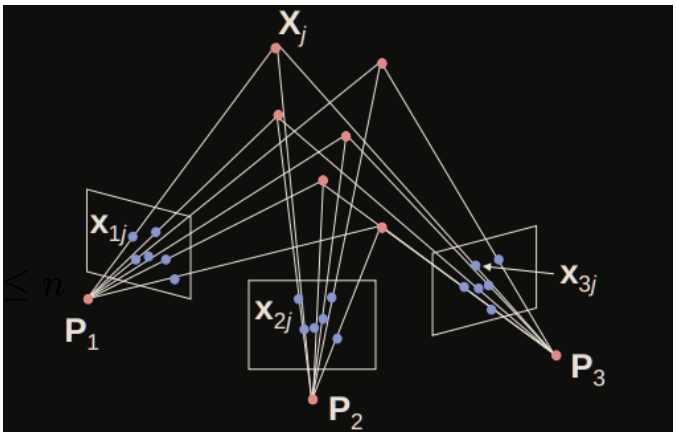
Given:

- 10k images, 1k points per image

- Each point seen 10 times on average

Question is: **How many parameters**?

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



Knowns: 20M

- Image points: 2 * 10k * 1k = 20M

Unknowns: 13M

- world pts: 1M = (10k*1k/10)
  - 1M * 3 (XYZ)
- 10k * 6: 10k orientations because 10k images
- *Scale: 10M (just like known image points but don't multiple by 2)*

Eliminating scale parameter:

- Eliminates 10M parameters and go to 3M parameters.
- Go back to Euclidean coordinate system from the homogenous coordinate system.

2.3.2 Cost Function

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

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

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

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

Reprojection Error

P_i is projection matrix of the i^{th} view, $\vec{X}_j$ is the j^{th} $3D$ point.

$\vec{x}_{ij}$ — **the observation**: the image (pixel locations) of the j^{th} pixel in the i^{th} image. *(using say SIFT matcher)*

$\hat{\vec{x}}_{ij}$ — **the predicted projection**: of initial reconstruction of the j^{th} $3D$ point to the i^{th} view.

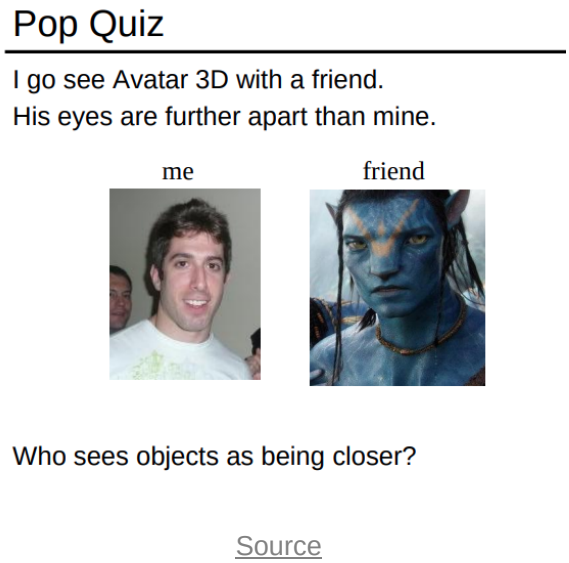
- $2MN$ equations in total (2 for each match)

This is a non-linear optimization problem wherein we minimize the sum of the squared reprojection errors of the reconstructed N $3D$ points over M images.

[3/11] Classroom Discussion

> Questions to solve/discuss

- ▼ `Stereo-Homework-1:` Derive the above equation using triangulation concept
- ▼ `Stereo-Homework-2:` ***Pop Quiz***



Source

$$\implies z = \frac{fb}{d}, \quad d \triangleq u_L - u_R$$
$$where \;\; d \to disparity/parallax$$

Demo of COLMAP

(In person: A demo of the COLMAP GUI was given in the class. See link.)

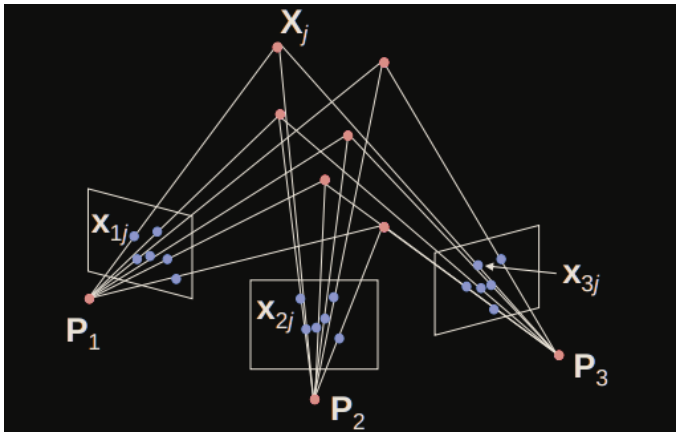
COLMAP is a state-of-the-art, free and open-source, Structure-from-Motion library.

Given a set of unordered and uncalibrated images, the program can build an accurate sparse model of the scene as well as recover the camera poses. The program can also build a dense model if a CUDA-enabled GPU is available.

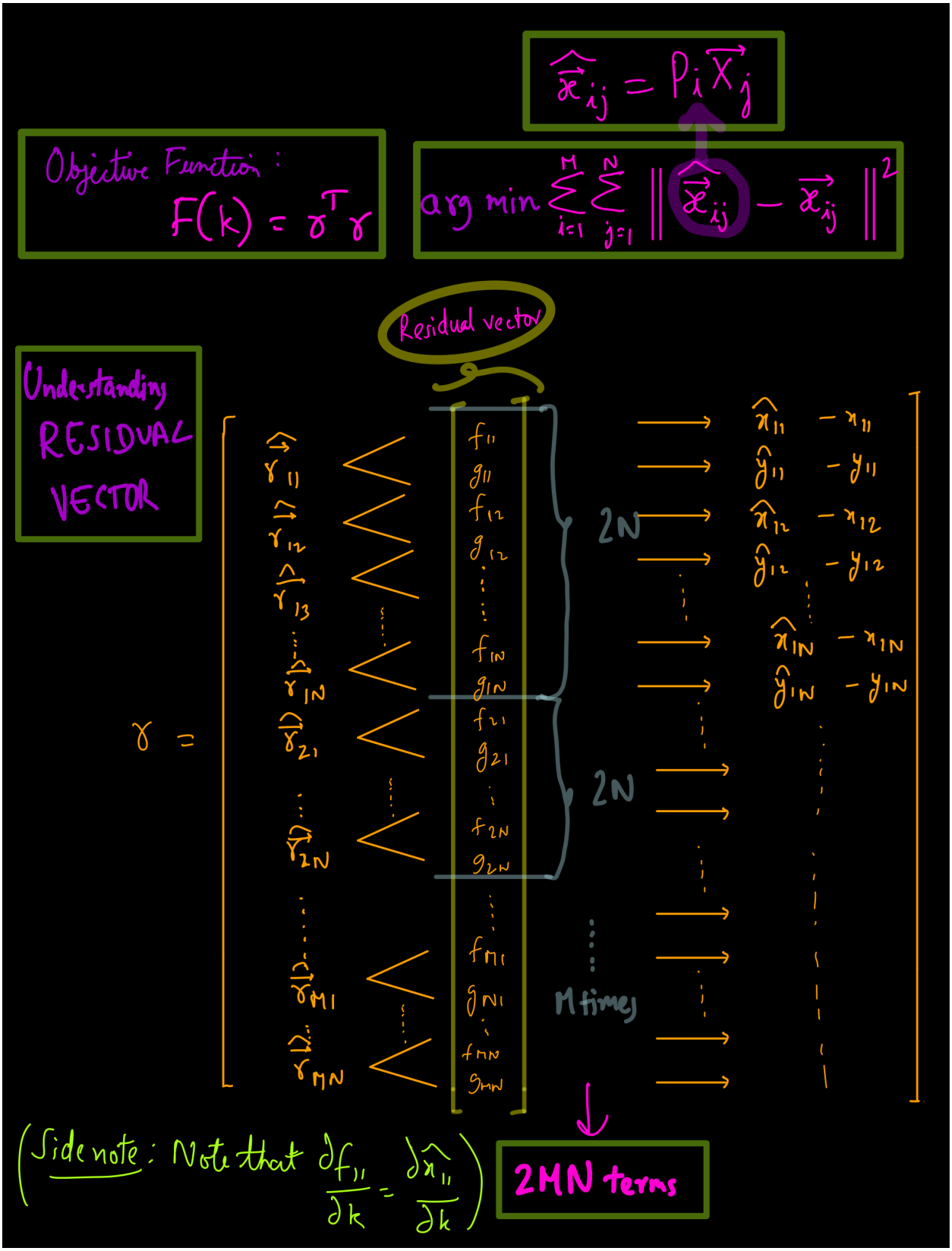
COLMAP documentation here.

2.3.3 Standard Least Squares Approach & Residual/Jacobian Structure

Recollect: LM algorithm



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


$\hat{\vec{x}}_{ij} = P_i \vec{X}_j$
Objective Function :
$F(k) = \gamma^T \gamma$
$\arg\min \sum_{i=1}^{M} \sum_{j=1}^{N} \| \hat{\vec{x}}_{ij} - \vec{x}_{ij} \|^2$
Residual vector
Understanding RESIDUAL VECTOR
$\gamma =$
2N
2N
M times
2MN terms
(Side note: Note that $\frac{\partial f_{11}}{\partial k} = \frac{\partial \hat{x}_{11}}{\partial k}$)

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

Residual vector

Objective Function: $F(k) = r^T r$

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

$$J = \frac{\partial r}{\partial k}$$

$$\left[P_1 \quad P_2 \quad \dots \quad P_M \quad \vec{X}_1 \quad \vec{X}_2 \quad \dots \quad \vec{X}_N \right]$$

$P_{11} P_{12} P_{13} \dots P_{34}$ ... $P_{M1} P_{M2} \dots P_{M34}$ $X_1 Y_1 Z_1$ ..... $X_N Y_N Z_N$

INDIVIDUAL JACOBIANS: $J_{ijm}, G_{ijm}, J_{ijs}, G_{ijs}$ ⇒

'm' otion

$$J_{ijm} = \left[\frac{\partial f_{ij}}{\partial P_i} \right] = \left[\frac{\partial f_{ij}}{\partial P_{i11}} \quad \dots\dots \quad \frac{\partial f_{ij}}{\partial P_{i34}} \right]$$

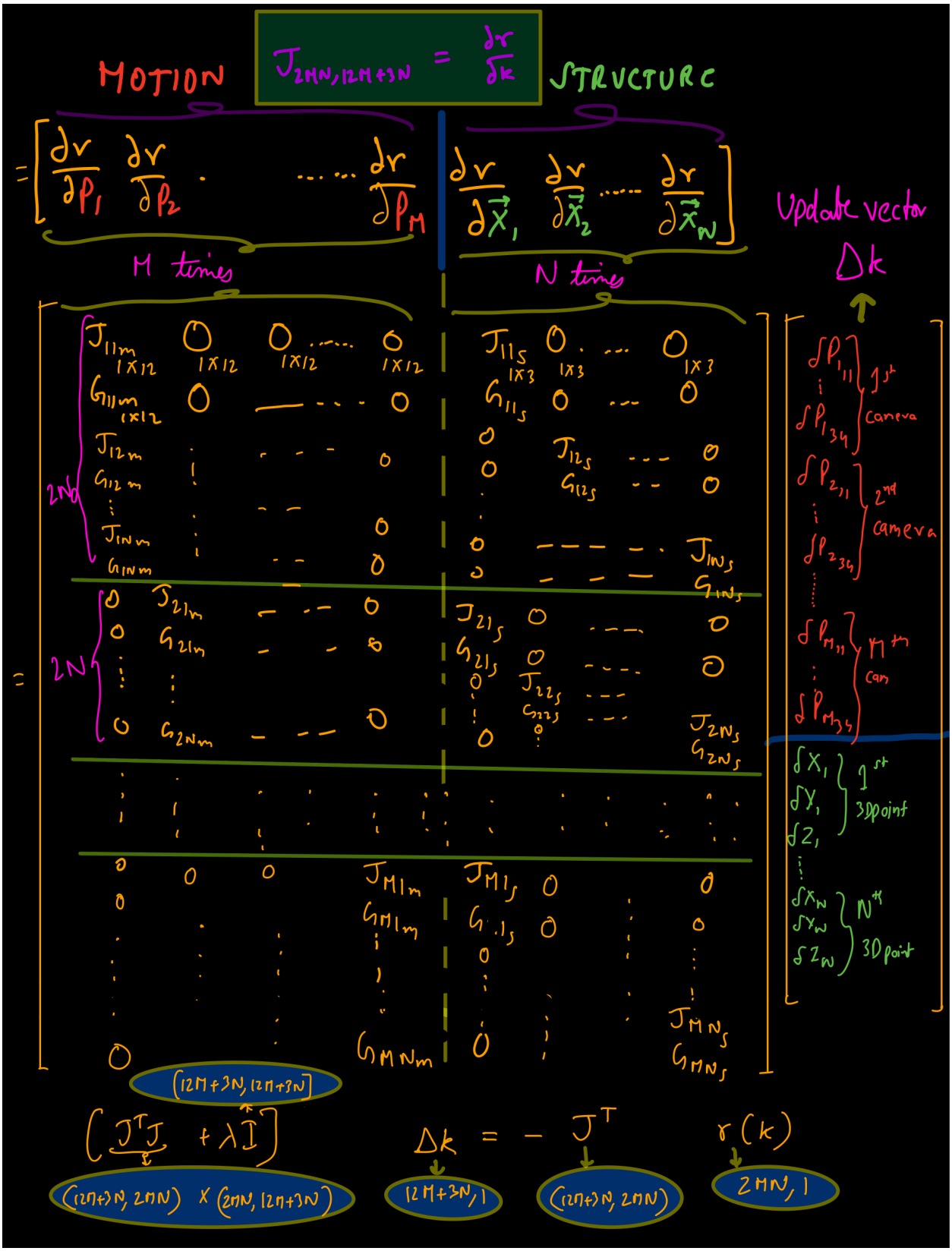
$$G_{ijm} = \left[\frac{\partial g_{ij}}{\partial P_i} \right] = \left[\frac{\partial g_{ij}}{\partial P_{i11}} \quad \dots\dots \quad \frac{\partial g_{ij}}{\partial P_{i34}} \right]$$

's' tructure

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

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

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


MOTION
STRUCTURE
Update vector
Δk
M times
N times
2M
2N
1st camera
2nd camera
Mth cam
1st 3Dpoint
Nth 3D point
[12M+3N, 12M+3N]
(JᵀJ + λI)
Δk = − Jᵀ r(k)
(12M+3N, 2MN) X (2MN, 12M+3N)
12M+3N, 1
(12M+3N, 2MN)
2MN, 1

Only the left matrix is $\mathbf{J}$. The Δ vector is written on the right. It's **NOT** multiplication: Just written together for convenience (You can compare and see the parameters corresponding to each Jacobian though — Notice how you can actually multiply them, i.e. see their matching dimensionality — $\text{No. of columns in } \mathbf{J} = dim(\Delta)$. Why is this true? Convince yourself. 👨‍🎓)

2.3.4 Coding demo + SOTA

> In person later

$$END - OF - PAGE$$

▼ Shubodh's private space, scroll up!

Original pages: