



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

1. Stereo Camera

2. Structure from Motion

2.1 Brief Overview of SfM

2.2 Initialization for BA

2.3 Formulation

2.3.1 Reprojection Error

2.3.2 Cost Function

[3/11] Classroom Discussion

2.3.3 Standard Least Squares Approach & Residual/Jacobian Structure

2.3.4 Coding demo + SOTA

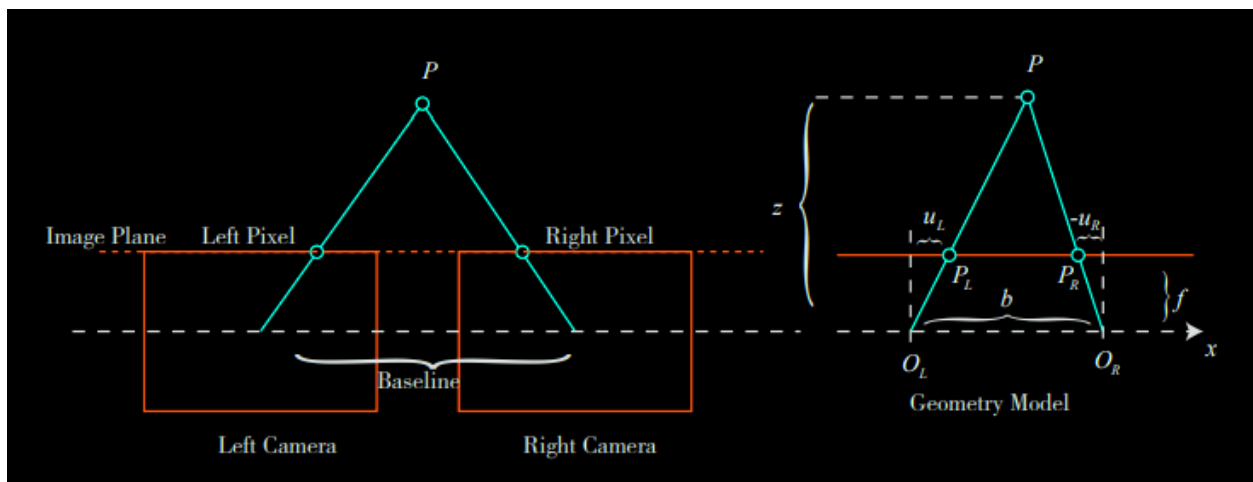
Agenda for 3/11: Stereo + Bundle Adjustment

- 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 $\rightarrow \triangle PP_L P_R$ and $\triangle PO_L O_R$)

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

▼ Not to get confused about u_R :

where $d \rightarrow \text{disparity/parallax}$

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.

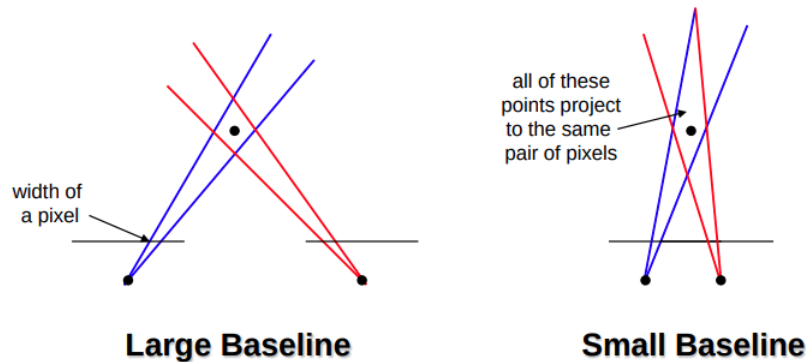
▼ When is d zero i.e. $u_L = u_R$?

([note above](#) to avoid confusion)

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

▼ Optimal Baseline

Choosing the stereo baseline



What's the optimal baseline?

- Too small: large depth error
- Too large: difficult search problem

[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.

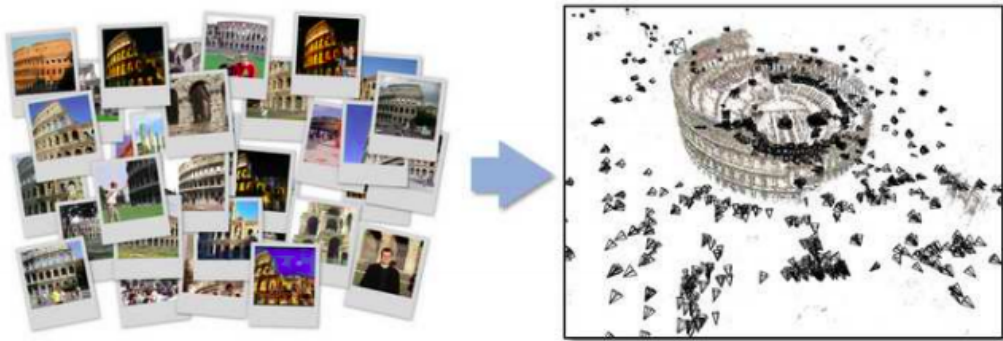
2.2 Initialization for BA

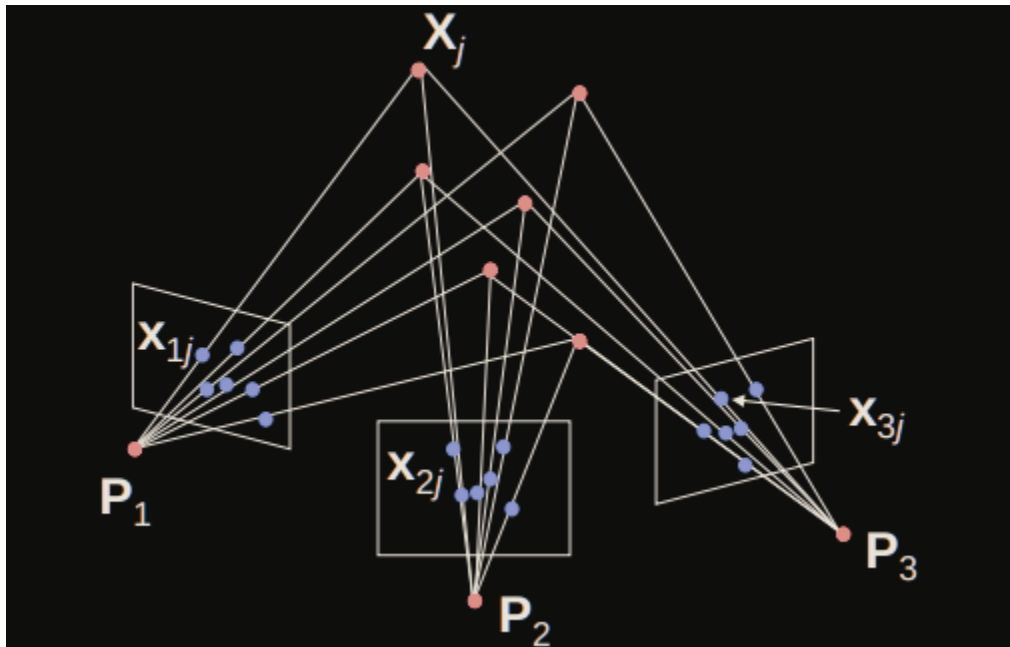
1. Real world information like IMU, Odometry → initial guess of poses; and by triangulation, obtain initial guess of 3D points as initial guess for BA
2. F estimation (8 point algorithm) first for 1st pair of images then triangulation then P3P. Or just pairwise F estimation.

(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:

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

- Known data association

Reprojection Error:

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

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

An example:

Source

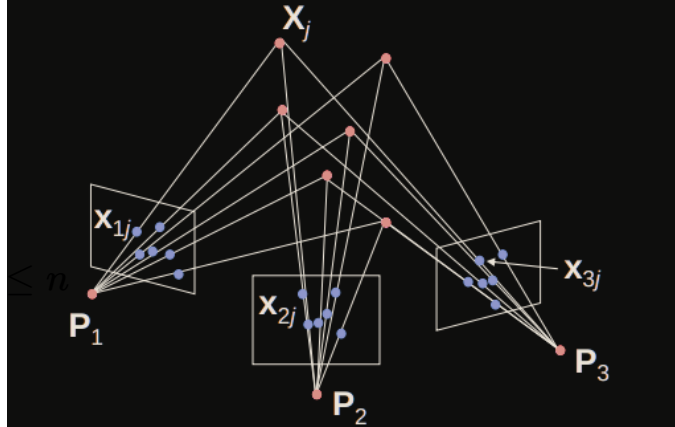
Given:

- 10k images, 1k points per image

- Each point seen 10 times on average

Question is: **How many parameters?**

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



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

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

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

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

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

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**

Pop Quiz

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



Who sees objects as being closer?

[Source](#)

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

where $d \rightarrow \text{disparity/parallax}$

Demo of COLMAP

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

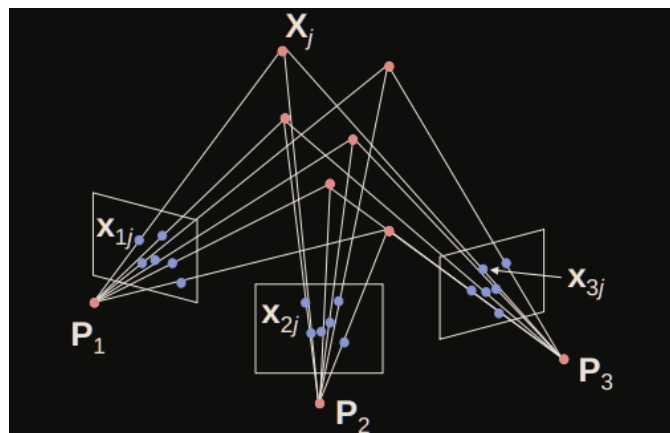
COLMAP is a state-of-the-art, free and open-source, Structure-from-Motion library.

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



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

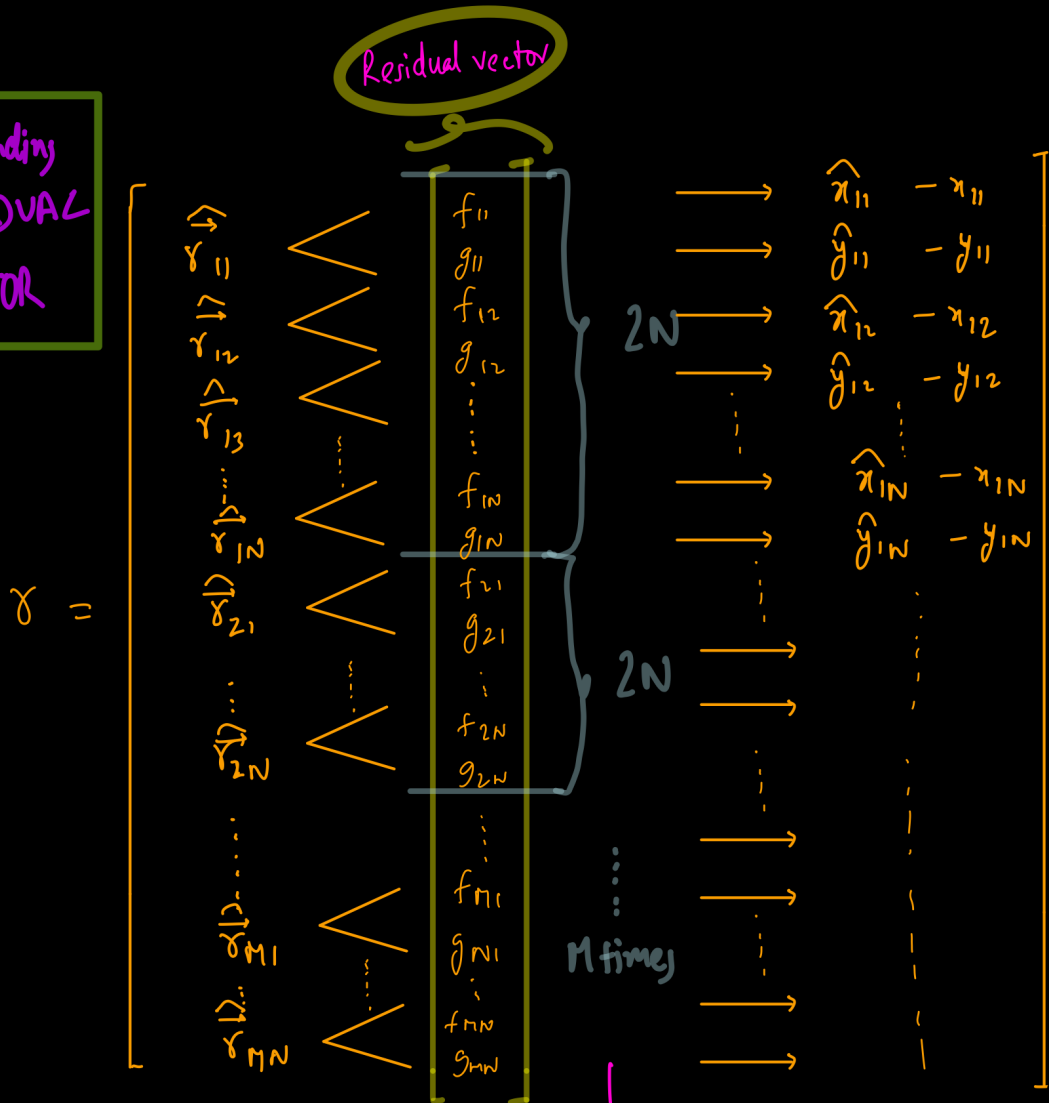
$$\hat{x}_{ij} = p_i \vec{X}_j$$

Objective Function:

$$F(k) = \delta^T \delta$$

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

Understanding
RESIDUAL
VECTOR



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

2MN terms

$\hat{x}_{ij} = p_i \vec{x}_j$

Objective Function: $F(k) = \delta^T \delta$

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

Residual vector $J = \frac{\partial \delta}{\partial k}$

$\begin{bmatrix} p_1 & p_2 & \dots & p_M & \vec{x}_1 & \vec{x}_2 & \dots & \vec{x}_N \end{bmatrix}$

$\begin{bmatrix} p_{11} p_{12} p_{13} \dots p_{134} \dots & p_{M1} p_{M2} \dots p_{M34} & x_1 y_1 z_1 & \dots & x_N y_N z_N \end{bmatrix}$

INDIVIDUAL JACOBIANS: $J_{ijm}, G_{ijm}, J_{ijs}, G_{ijs} \Rightarrow$

'motion' $\begin{bmatrix} J_{ijm} = \left[\frac{\partial f_{ij}}{\partial p_i} \right] = \left[\frac{\partial f_{ij}}{\partial p_{i1}} \dots \frac{\partial f_{ij}}{\partial p_{i34}} \right] \\ G_{ijm} = \left[\frac{\partial g_{ij}}{\partial p_i} \right] = \left[\frac{\partial g_{ij}}{\partial p_{i1}} \dots \frac{\partial g_{ij}}{\partial p_{i34}} \right] \end{bmatrix}$

'structure' $\begin{bmatrix} J_{ijs} = \left[\frac{\partial f_{ij}}{\partial \vec{x}_j} \right] = \left[\frac{\partial f_{ij}}{\partial x_j} \frac{\partial f_{ij}}{\partial y_j} \frac{\partial f_{ij}}{\partial z_j} \right] \\ G_{ijs} = \left[\frac{\partial g_{ij}}{\partial \vec{x}_j} \right] = \left[\frac{\partial g_{ij}}{\partial x_j} \frac{\partial g_{ij}}{\partial y_j} \frac{\partial g_{ij}}{\partial z_j} \right] \end{bmatrix}$

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

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

STRUCTURE

M times

2N	J_{11m} <small>1x12</small>	0	0	...	0
	G_{11m} <small>1x12</small>	0	...	0	
	J_{12m}	:	-	-	0
	G_{12m}	:	-	-	0
	J_{1Nm}	:	-	-	0
	G_{1Nm}	:	-	-	0

2N

0	J_{21m}	-	-	0
0	G_{21m}	-	-	0
:	:	-	-	:
0	G_{2Nm}	-	-	0

N times

J_{11s} <small>1x3</small>	0	...	0	<small>1x3</small>
G_{11s}	0	...	0	
0	J_{12s}	-	-	0
0	G_{12s}	-	-	0
:	:	-	-	:
0	J_{1Ns}
0	G_{1Ns}

N

J_{21s}	0	...	0
G_{21s}	0	...	0
0	J_{22s}	-	-
0	G_{22s}	-	-
:	:	-	-
0	J_{2Ns}
0	G_{2Ns}

(12M+3N, 12M+3N)

$\left[\underbrace{J^T J}_{(12M+3N, 12M+3N)} + \lambda I \right]$

x (2MN, 12M+3N)

Update vector

Δk

$\Delta k = - J^T$

\downarrow

$(12M+3N, 1)$

\downarrow

$(12M+3N, 2MN)$

\downarrow

$2MN, 1$

1st camera

2nd camera

Mth cam

Nth 3D point

$\left\{ \begin{matrix} \delta p_{11} \\ \vdots \\ \delta p_{134} \end{matrix} \right\}$

$\left\{ \begin{matrix} \delta p_{21} \\ \vdots \\ \delta p_{234} \end{matrix} \right\}$

$\left\{ \begin{matrix} \delta p_{M1} \\ \vdots \\ \delta p_{M34} \end{matrix} \right\}$

$\left\{ \begin{matrix} \delta x_1 \\ \delta y_1 \\ \delta z_1 \end{matrix} \right\}$

$\left\{ \begin{matrix} \delta x_N \\ \delta y_N \\ \delta z_N \end{matrix} \right\}$

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 — 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: