Programming Assignment #1

Robust Photometric Stereo

Goal

Robust Photometric Stereo

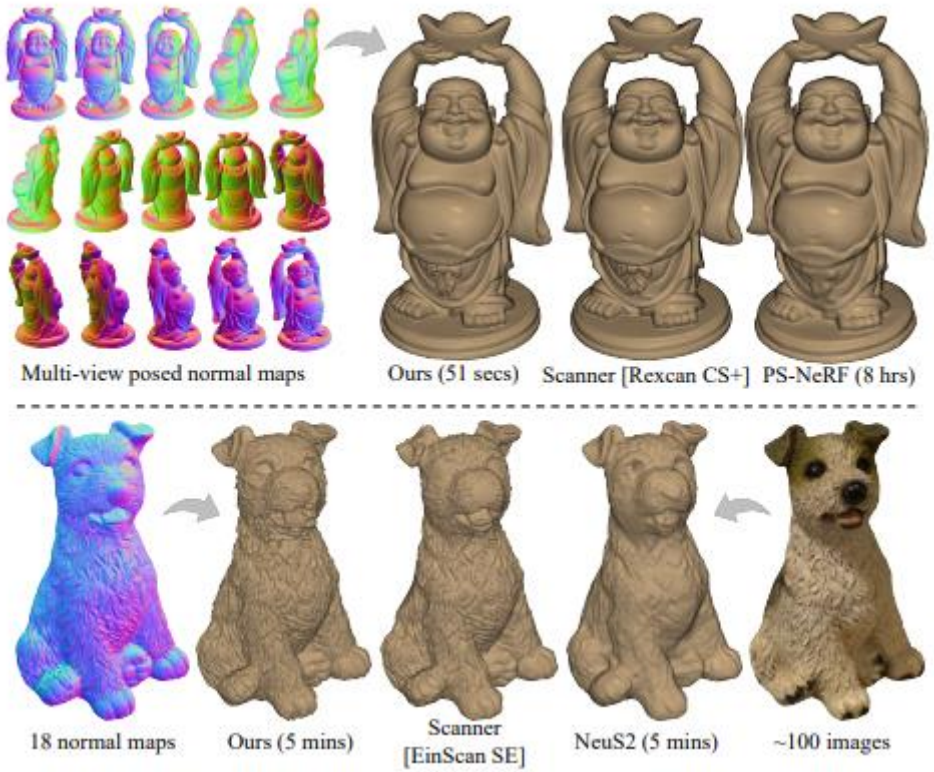
Programming Assignment #1 will cover:

- Photometric Stereo: How to estimate normal/albedo from a set of images?
- Robust PCA: How to recover low-rank matrix from corrupted observations?

Preliminaries

Photometric Stereo - Applications

What can we do with surface normals and albedo?



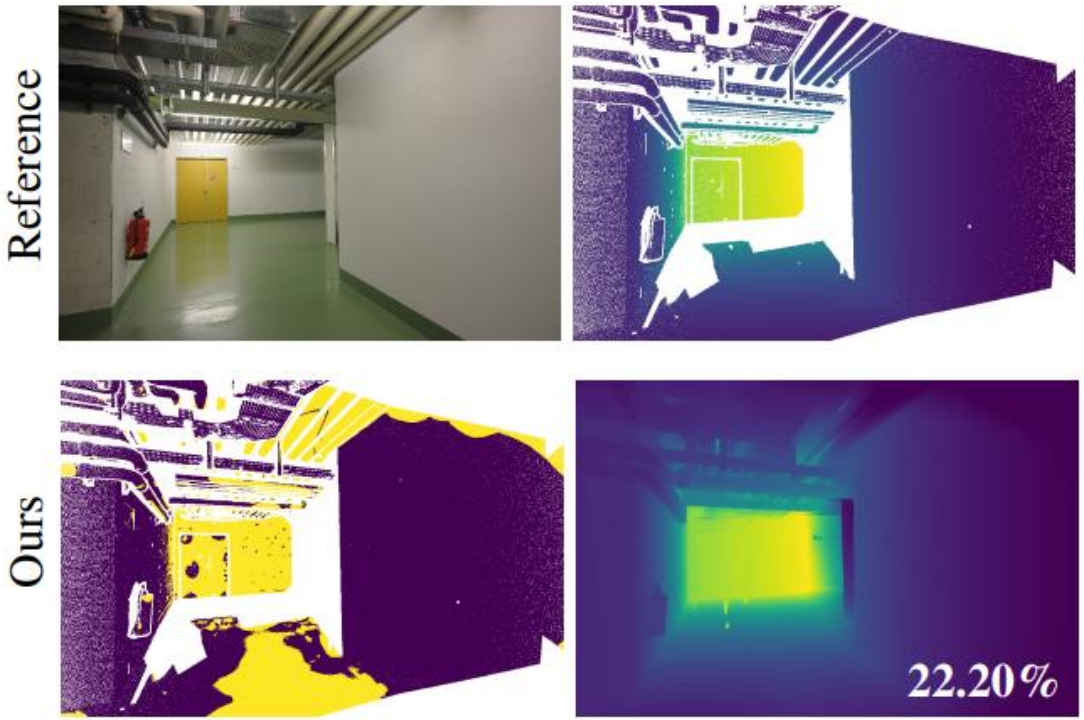
3D Shape Recovery

Neural Surface Reconstruction via Multi-View Normal Integration. CVPR 2024

Preliminaries

Photometric Stereo - Applications

What can we do with surface normals and albedo?



Depth Refinement

Joint Graph-Based Depth Refinement and Normal Estimation. CVPR 2020

Preliminaries

Photometric Stereo - Applications

What can we do with surface normals and albedo?

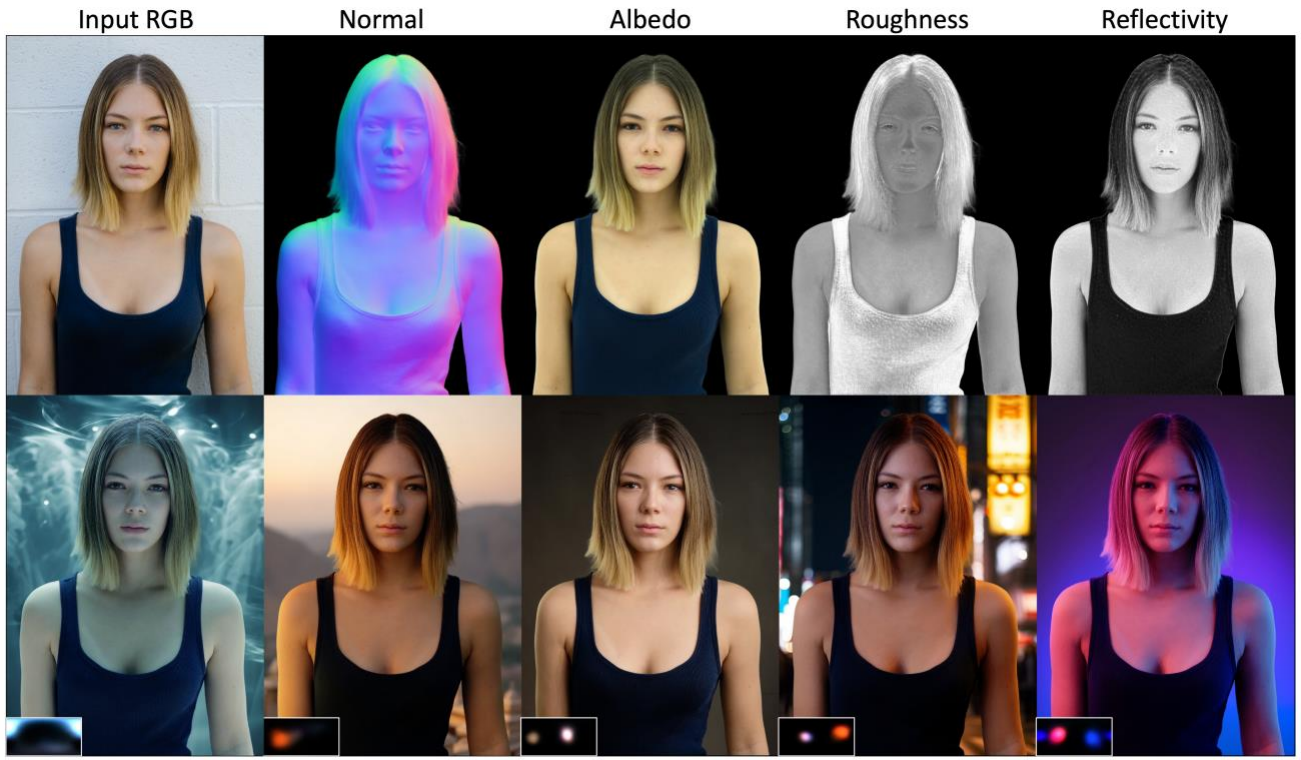


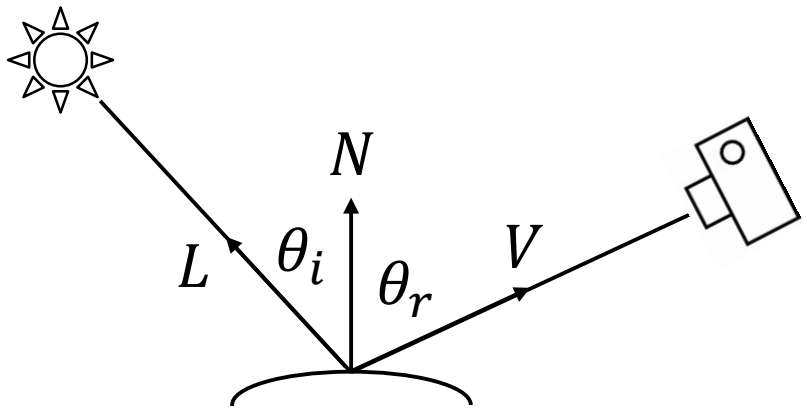
Image Relighting

SwitchLight: Co-design of Physics-driven Architecture and Pre-training Framework for Human Portrait Relighting. CVPR 2024

Preliminaries

Light and Surface

How does light interact with a surface?



N: Surface Normal
L: Light Direction
V: Viewing direction

Outgoing radiance along V $\leftarrow L_r = \rho(\theta_i, \theta_r) L_i \cos\theta_i \rightarrow$ Incoming radiance along L

Bi-directional reflectance function (BRDF)

Preliminaries

Light and Surface

How does light interact with a surface?

Special Case 1: Perfect Mirror

$\rho(\theta_i, \theta_r) = 0, \quad \theta_i \neq \theta_r$

Special Case 2: Matte Surface

$\rho(\theta_i, \theta_r) = \rho$

called albedo

Preliminaries

Lambertian Surfaces

For a Lambertian Surface:

$$L_r = \rho L_i \cos\theta_i = \rho L_i L \cdot N$$

Assume for all points on the surface;

- The camera viewing angle is same. => the camera is infinitely far away
- All rays are parallel and have same intensity ($L_i = 1$) => the light sources are infinitely far away

Now, the image intensity for Lambertian surfaces can be formulated as:

$$I(x, y) = \rho N(x, y) \cdot L$$

Lambertian Object

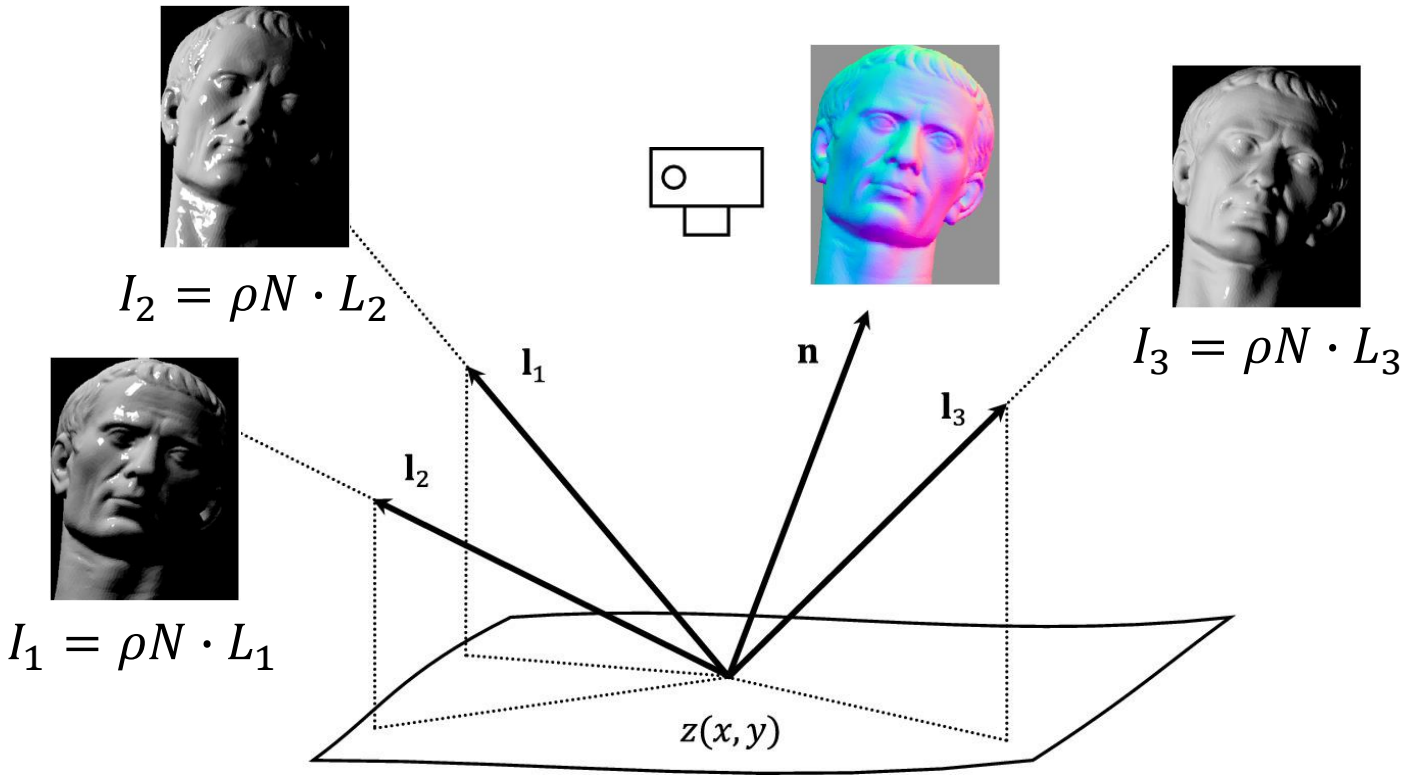
Non-Lambertian Object

I : pixel intensity
ρ : albedo (Lambertian; constant)
N : surface normal (3d vectors)
L : light directions (3d vectors)

Preliminaries

Photometric Stereo

Photometric stereo : A way to determine the surface normal from the measured pixel brightness obtained under different lighting conditions



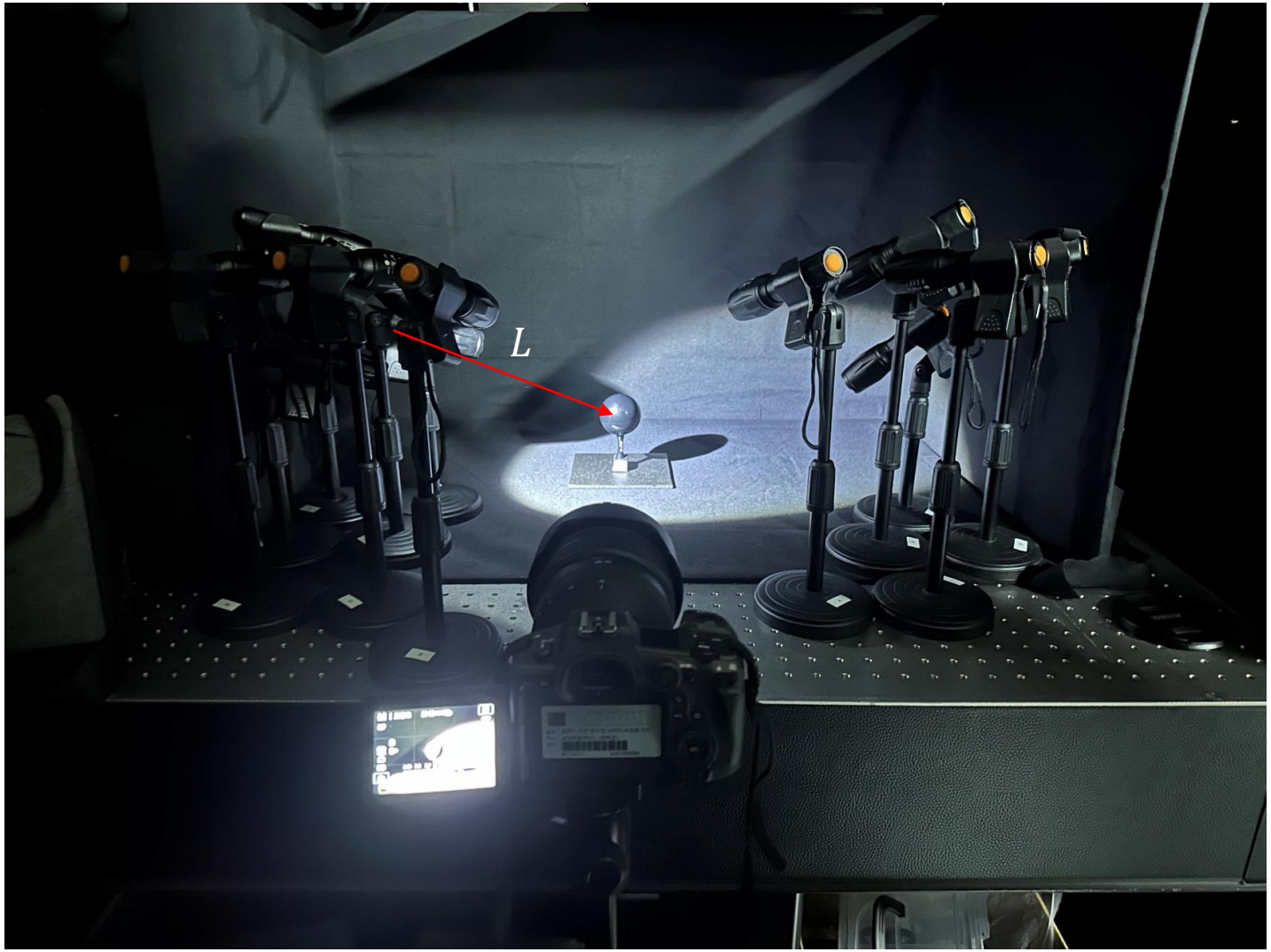
As a matrix equation:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \rho \begin{bmatrix} L_1^T \\ L_2^T \\ L_3^T \end{bmatrix} N$$

Multiple images and lights give constraints on ρ and N

Preliminaries

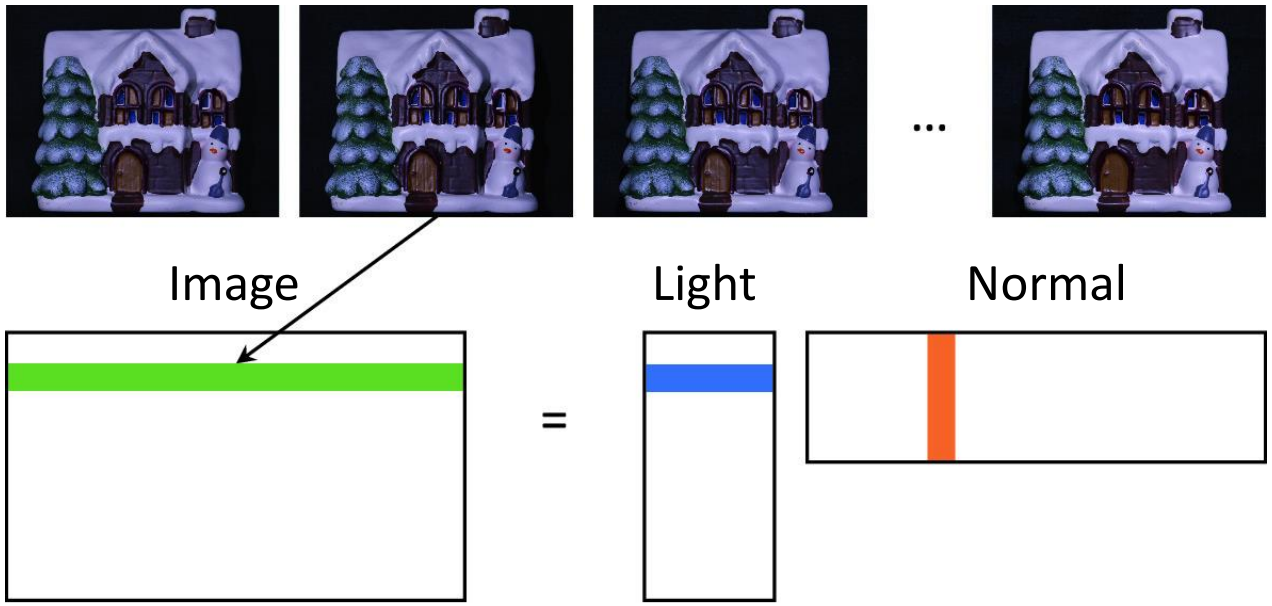
Photometric Stereo



Preliminaries

Problem Setup : Photometric Stereo

Assume you have n images with m pixels each, captured under n different light sources.



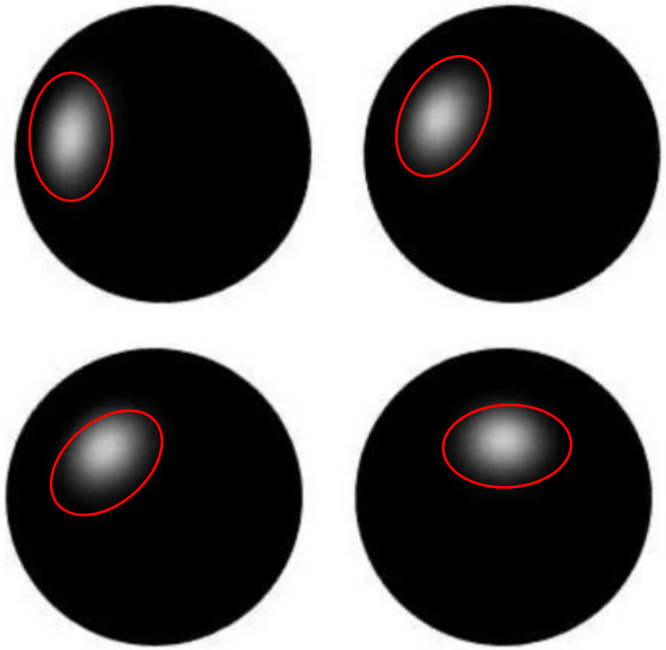
$I \in \mathbb{R}^{n \times m}$ ✅

$L \in \mathbb{R}^{n \times 3}$ 🤔

$N \in \mathbb{R}^{3 \times m}$ ❗

Goal: Recover Light Direction (Step 1)

Simple Trick : Place a chrome sphere in the scene

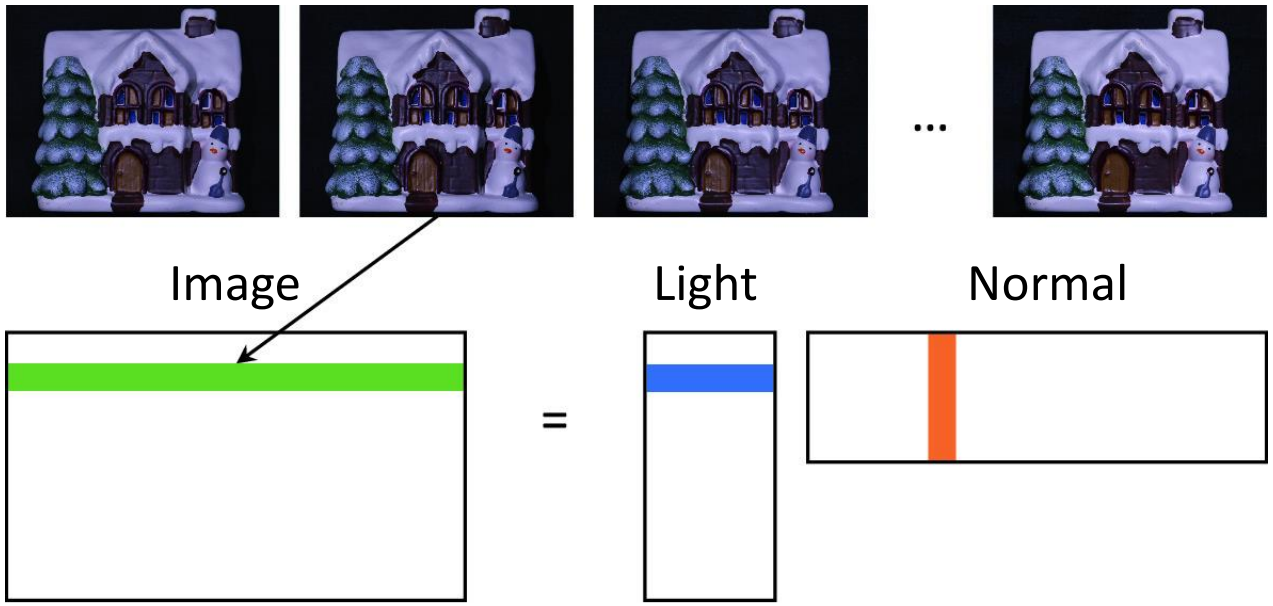


The location of the highlight(brightest point) tells you where the light source is!

Preliminaries

Problem Setup : Photometric Stereo

Assume you have n images with m pixels each, captured under n different light sources.



$I \in \mathbb{R}^{n \times m}$ ✅

$L \in \mathbb{R}^{n \times 3}$ ✅

$N \in \mathbb{R}^{3 \times m}$ ❗

Goal: Recover Surface Normals (Step 2, 3)

Implementation Details

Lambertian Least Squares Photometric Stereo – Mathematical Formulation

The image can be formulated as: $I = L^T \rho N$

Let $G = \rho N$, then

$$G = L^{-T} I$$

To solve for G, we use the least squares solution:

$$\min_G \|I - L^T G\|^2$$

$$\min_G I^T I + G^T L L^T G - 2 G^T L I$$

Take the derivative with respect to G and set to 0:

$$2 L L^T G - 2 L I = 0$$

$$G = (L L^T)^{-1} L I$$

Recall that $G = \rho N$:

$$\|G\| = \rho, \ \frac{G}{\|G\|} = N$$

Implementation Details

[Summary] Lambertian Least Squares Photometric Stereo

Photometric Stereo estimates surface normals and albedo using multiple images captured under different lighting conditions.

The brightness at each pixel can be modeled as:

$$I = \rho(N \cdot L)$$

By capturing images under $N \geq 3$ different lighting directions, we can estimate surface normals.
This results in a linear system of equations:

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} L_1^x & L_1^y & L_1^z \\ L_2^x & L_2^y & L_2^z \\ \vdots & \vdots & \vdots \\ L_N^x & L_N^y & L_N^z \end{bmatrix} \begin{bmatrix} \rho N_x \\ \rho N_y \\ \rho N_z \end{bmatrix}$$

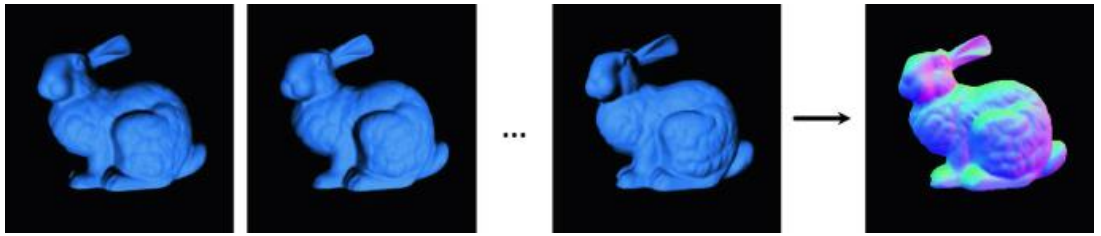
I : pixel intensity
ρ : albedo (Lambertian; constant)
N : surface normal (3d vectors)
L : light directions (3d vectors)

To solve for ρN, we use the least squares solution:

$$I = L \cdot \rho N$$
$$L^{-1} I = \rho N$$

If L is not square (more than 3 lights), use Moore-Penrose pseudoinverse instead.

$$L^T I = L^T L \cdot \rho N$$
$$(L^T L)^{-1} L^T I = \rho N$$

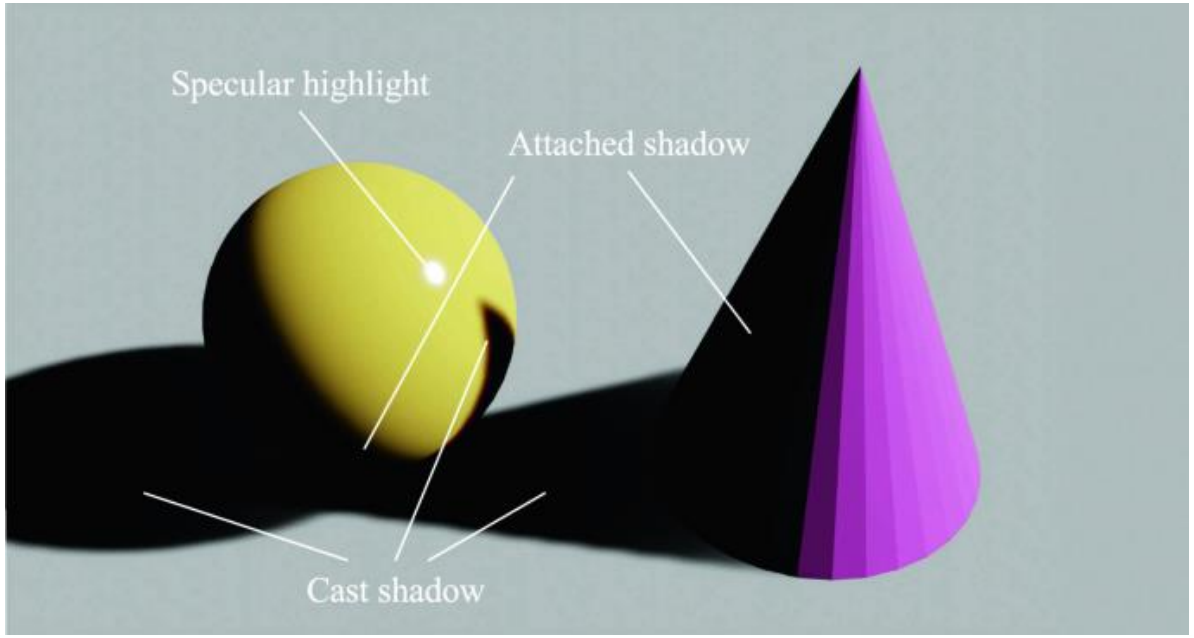


Lambertian Surface: An ideal diffuse surface that appears equally bright from all viewing angles

Preliminaries

RPCA Photometric Stereo - Shadows & Highlights

Does the Lambertian surface assumption hold in real-world scenarios?



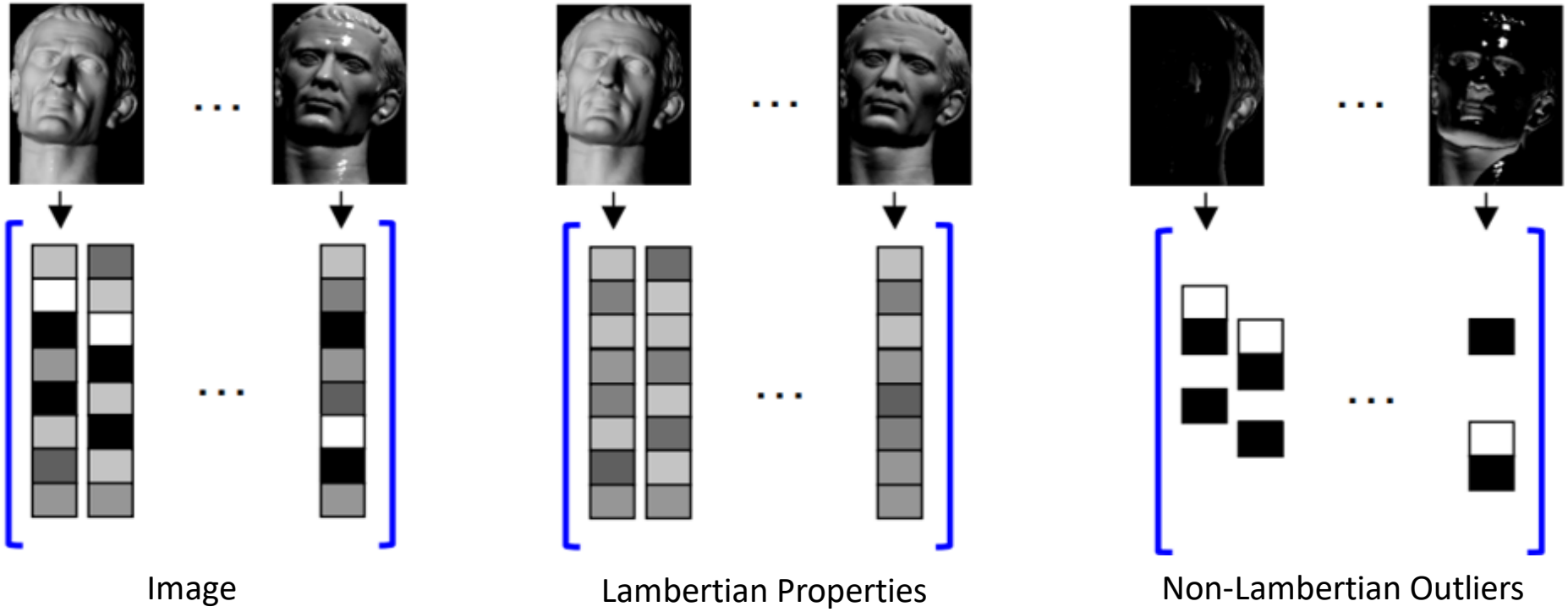
Non-Lambertian Outliers

How can we avoid it? Robust Photometric Stereo!

Preliminaries

RPCA Photometric Stereo - Non-Lambertian Outliers

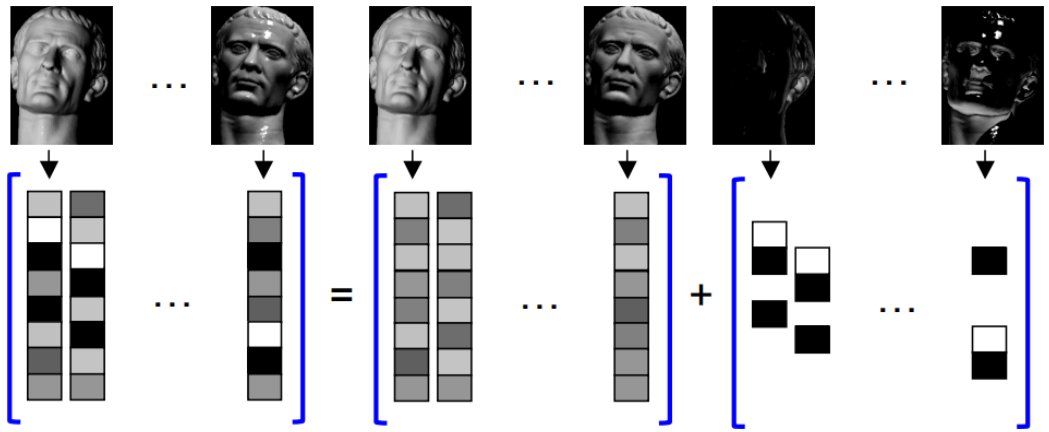
Given Non-Lambertian outliers, an image can be divided into:

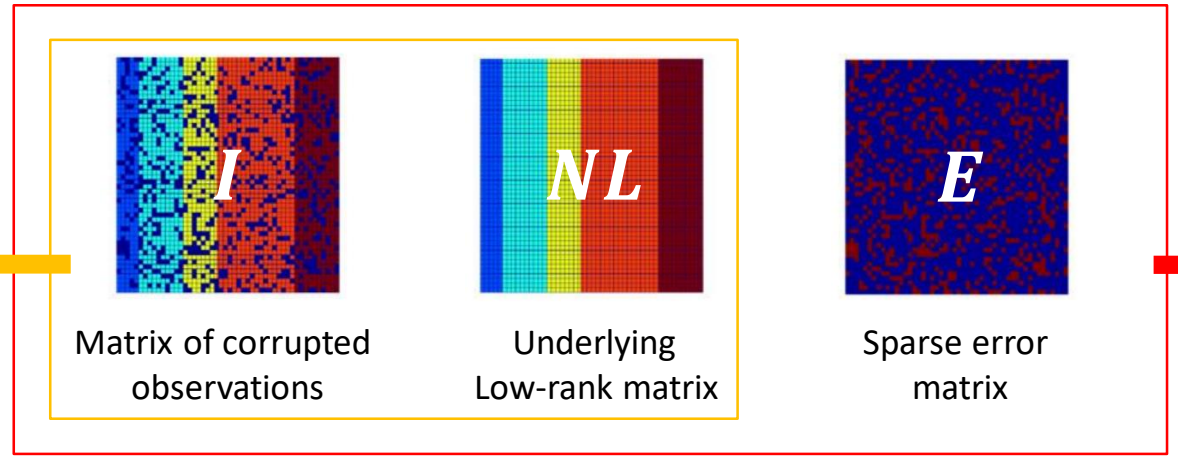


Preliminaries

RPCA Photometric Stereo – Problem Setup

Can we recover NL and E from I?





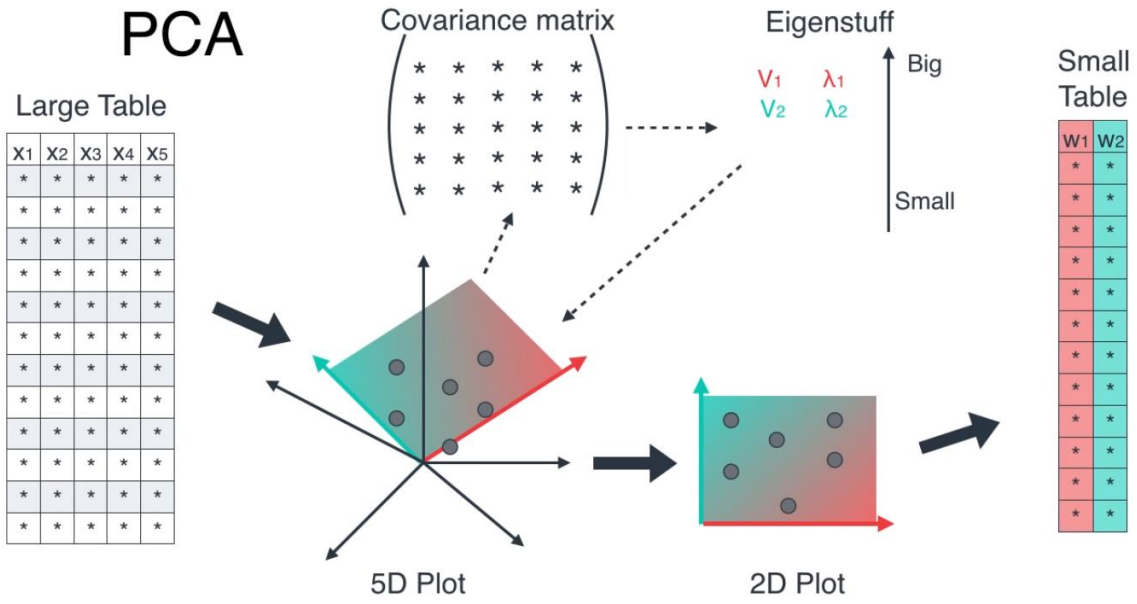
Holds only under Lambertian assumptions

Accounts for non-Lambertian deviations

Preliminaries

RPCA Photometric Stereo – What is PCA?

"Can we reduce the dimensionality of data while preserving its most important information?"



Principal Components Analysis?

1. Compute the Covariance Matrix
2. Find Eigenvalues & Eigenvectors
3. Select the top K Components
4. Transform the Data

PCA relies on variance; can be sensitive to outliers.

"PCA works well, but what if the data contains outliers?"

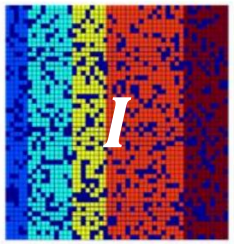
Preliminaries

RPCA Photometric Stereo – What is RPCA?

Robust Principal Component Analysis? (RPCA)

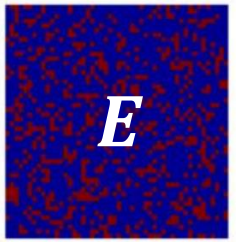
Given $D = A + E$, where A and E are unknown, but A is known to be low rank and E is known to be sparse, recover A from D

$$\min_{A,E} rank(A) + \gamma \|E\|_0 \quad s.t. \quad D = A + E$$



Matrix of corrupted observations

Underlying Low-rank matrix



Sparse error matrix

NL is *low rank*; both N and L are at most rank 3

Assume E is *sparse*; Lambertian assumptions obey

→ RPCA problem

"RPCA improves PCA by decomposing a matrix into two components: Low-Rank and Sparse!"

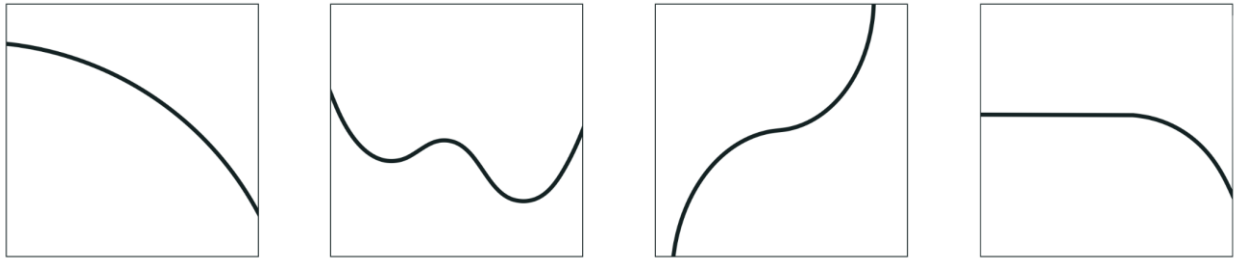
Preliminaries

RPCA Photometric Stereo – Mathematical Formulation

The given problem is:

$$A^*, E^* = \underset{NL,E}{\operatorname{argmin}}\{rank(NL) + \gamma\|E\|_0\} \quad s.t. \quad I = NL + E$$

- Minimizing $rank(A)$ reduces the number of principal components
- $\|E\|_0$ counts the number of non-zero elements (outliers)
- Both rank and L0-norm are **non-convex**, making the problem hard to solve



Preliminaries

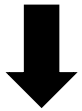
RPCA Photometric Stereo – Mathematical Formulation

To make this problem convex, replace:

- $rank(A) \rightarrow \|A\|_*$: Sum of singular values
- $\|E\|_0 \rightarrow \|E\|_1$: Sum of absolute values

Then, this problem can be formulated as:

$$A^*, E^* = \underset{NL,E}{\operatorname{argmin}}\{\|NL\|_* + \gamma\|E\|_1\} \quad s.t. \quad I = NL + E$$



Now, how to find the minimum value?

Preliminaries

RPCA Photometric Stereo – Convex Optimization Prob.

We are given the following convex optimization problem:

$$\min_{x_1,x_2} f(x_1,x_2) = x_1^2 + 2x_2^2 - 2x_1x_2 - 0.5x_1 - 0.5x_2$$

Subject to this constraint:

$$g(x_1,x_2) = x_1,x_2 - 1 = 0$$

That is, we need to minimize $f(x_1,x_2)$ while ensuring that $x_1 + x_2 = 1$

Preliminaries

RPCA Photometric Stereo – Lagrange Multiplier (LM)

For such constrained optimization problems, use the **Lagrange Multiplier Method**:

$$L(x_1, x_2, \lambda) = \boxed{x_1^2 + 2x_2^2 - 2x_1x_2 - 0.5x_1 - 0.5x_2} + \boxed{\lambda(x_1 + x_2 - 1)}$$

f is kept as it is

g is multiplied by the Lagrange multiplier λ

Now take partial derivatives and solve the equation:

$$\frac{\partial L}{\partial x_1} = 2x_1 - 2x_2 - 0.5 + \lambda = 0$$

$$\frac{\partial L}{\partial x_x} = 4x_1 - 2x_2 - 0.5 + \lambda = 0$$

$$\frac{\partial L}{\partial \lambda} = x_1 + x_2 - 1 = 0$$

Then, the optimal solution is:

$$x_1^* = \frac{3}{5}, x_2^* = \frac{2}{5}, \lambda^* = 0.1$$

Preliminaries

RPCA Photometric Stereo – Augmented Lagrange Multiplier (ALM)

The Lagrange method may converge slowly if λ is not updated properly and the constraint is not strictly enforced.

To address this, use the **Augmented Lagrange Method**, which introduces a penalty term:

$$L(x_1, x_2, \lambda, \mu) = x_1^2 + 2x_2^2 - 2x_1x_2 - 0.5x_1 - 0.5x_2 + \lambda(x_1 + x_2 - 1) + \frac{\mu}{2}(x_1 + x_2 - 1)^2$$

Preliminaries

RPCA Photometric Stereo – Augmented Lagrange Multiplier (ALM)

Now, back to RPCA!

$$A^*, E^* = \underset{NL,E}{\operatorname{argmin}}\{\|A\|_* + \lambda\|E\|_1\} \quad s.t. \quad I = A + E$$

Use the Augmented Lagrange Multiplier (ALM) method:

$$L_\mu(A, E, Y) = \|A\|_* + \lambda\|E\|_1 + \langle Y, D - A - E\rangle + \frac{\mu}{2}\|D - A - E\|_F^2$$

- Y is the Lagrange multiplier enforcing $D = A + E$
- μ is a penalty parameter that strengthens the constraint.
- $\|D - A - E\|_F^2$ is a quadratic penalty term to ensure constraint satisfaction

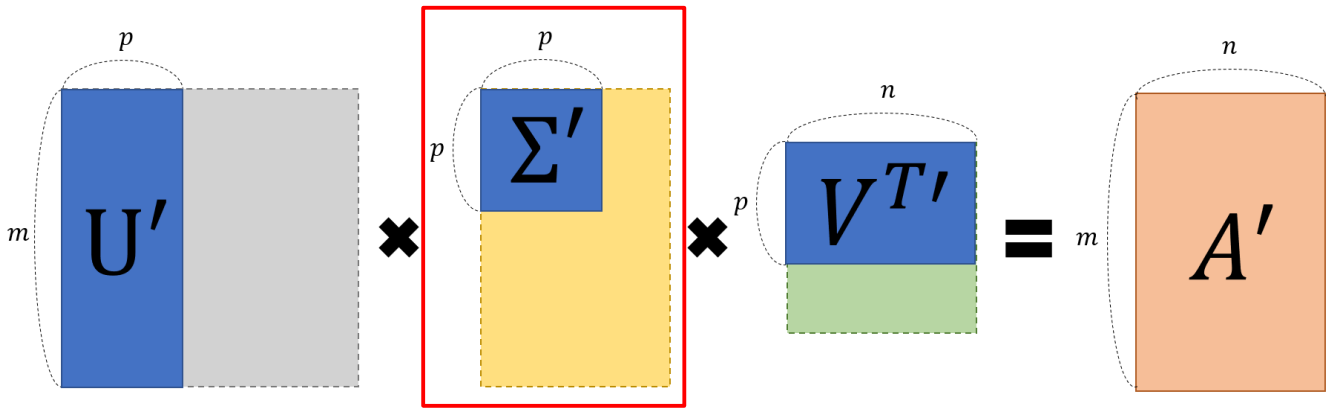
Preliminaries

RPCA Photometric Stereo – What is Singular Value Decomposition?

Singular Value Decomposition? (SVD)

For a given matrix A, it can be represented as:

$$A = U\Sigma V^T$$



Original Image : A

Low-Rank Image : A'
(with 39.6% of A)

Important information is concentrated in singular values;
Removal of small singular values can reduce noise, leaving only the core structure.

Implementation Details

RPCA Photometric Stereo – ALM Optimization Steps

Step 1. Low-Rank Matrix Extraction (Updating A)

Compute the Singular Value Decomposition (SVD):

$$A = U\Sigma V^T$$

Then, apply singular value thresholding:

$$A = US_{1/\mu}(\Sigma)V^T$$

Where $S_{1/\mu}(\cdot)$ is the soft thresholding function:

$$S_\tau(\sigma_i) = \max(\sigma_i - \tau, 0)$$

This step shrinks small singular values to zero, effectively removing noise and preserving only essential information.

Implementation Details

RPCA Photometric Stereo – ALM Optimization Steps

Step 2. Sparse Outlier Detection (Updating E)

Apply soft-thresholding to enforce sparsity:

$$E_{\lambda/\mu} = S(D - A + Y/\mu)$$

This removes small noise elements while keeping significant outliers.

Implementation Details

RPCA Photometric Stereo – ALM Optimization Steps

Step 3. Lagrange Multiplier Update

Update Y to better enforce the constraint:

$$Y = Y + \mu(D - A - E)$$

This step adjusts the constraint penalty, ensuring convergence.

Implementation Details

RPCA Photometric Stereo – ALM Optimization Steps

Step 4. Iterative Optimization

Repeat Steps 1–3 until convergence:

1. Compute SVD and shrink singular values.
2. Apply soft-thresholding to promote sparsity.
3. Update multipliers to satisfy the constraint.

Implementation Details

RPCA Photometric Stereo – Overall Algorithm

Algorithm 1 (Matrix Completion and Recovery via ALM).

INPUT: $D \in \mathbb{R}^{m \times n}$, $\Omega \subset \{1, \ldots, m\} \times \{1, \ldots, n\}$, $\lambda > 0$.
Initialize $A_1 \leftarrow 0$, $E_1 \leftarrow 0$, $Y_1 \leftarrow 0$.
while not converged $(k = 1, 2, \ldots)$ **do**
  $A_{k,1} = A_k$, $E_{k,1} = E_k$;
  while not converged $(j = 1, 2, \ldots)$ **do**
    $E_{k,j+1} = \text{shrink}\left(\pi_{\Omega^c}(D) + \frac{1}{\mu_k} Y_k - \pi_{\Omega^c}(A_{k,j}), \frac{\lambda}{\mu_k}\right)$;
    $t_1 = 1$; $Z_1 = A_{k,j}$; $A_{k,j,1} = A_{k,j}$;
    while not converged $(i = 1, 2, \ldots)$ **do**
      $(U_i, \Sigma_i, V_i) = \text{svd}\left(\frac{1}{\mu_k} Y_k + \pi_{\Omega^c}(D) - E_{k,j+1} + \pi_{\Omega}(Z_i)\right)$;
      $A_{k,j,i+1} = U_i \, \text{shrink}\left(\Sigma_i, \frac{1}{\mu_k}\right) V_i^T$, $t_{i+1} = 0.5\left(1 + \sqrt{1 + 4t_i^2}\right)$;
      $Z_{i+1} = A_{k,j,i+1} + \frac{t_i - 1}{t_{i+1}}(A_{k,j,i+1} - A_{k,j,i})$, $A_{k,j+1} = A_{k,j,i+1}$;
    end while
    $A_{k+1} = A_{k,j+1}$; $E_{k+1} = E_{k,j+1}$;
  end while
  $Y_{k+1} = Y_k + \mu_k \, \pi_{\Omega^c}(D - A_{k+1} - E_{k+1})$, $\mu_{k+1} = \rho \cdot \mu_k$;
end while
OUTPUT: $(\hat{A}, \hat{E}) = (A_k, E_k)$.

Implementation Details

[Summary] RPCA Photometric Stereo

In reality, Lambertian surfaces rarely exist due to the illumination effects stated in previous slide.

These Non-Lambertian observations can be treated as outliers that do not fit the Lambertian model.

We can therefore consider an extended Lambertian model that accounts for such outliers:

$$D = NL + E$$

We know that the matrix NL is low rank because both L and N are at most rank 3.
Therefore, we can regard NL as a low-rank matrix A.

Now the problem can be formulated as finding a sparse matrix E such that the matrix $A = D - E$ has the lowest possible rank:

$$\min_{A,E} rank(A) + \gamma\|E\|_0 \quad s.t. \quad D = A + E$$

But this optimization is NP-hard, so we can use another efficient solution:

$$\min_{A,E} \|A\|_* + \gamma\|E\|_1 \quad s.t. \quad D = A + E$$

This problem can be solved by Augmented Lagrange Multiplier Method:

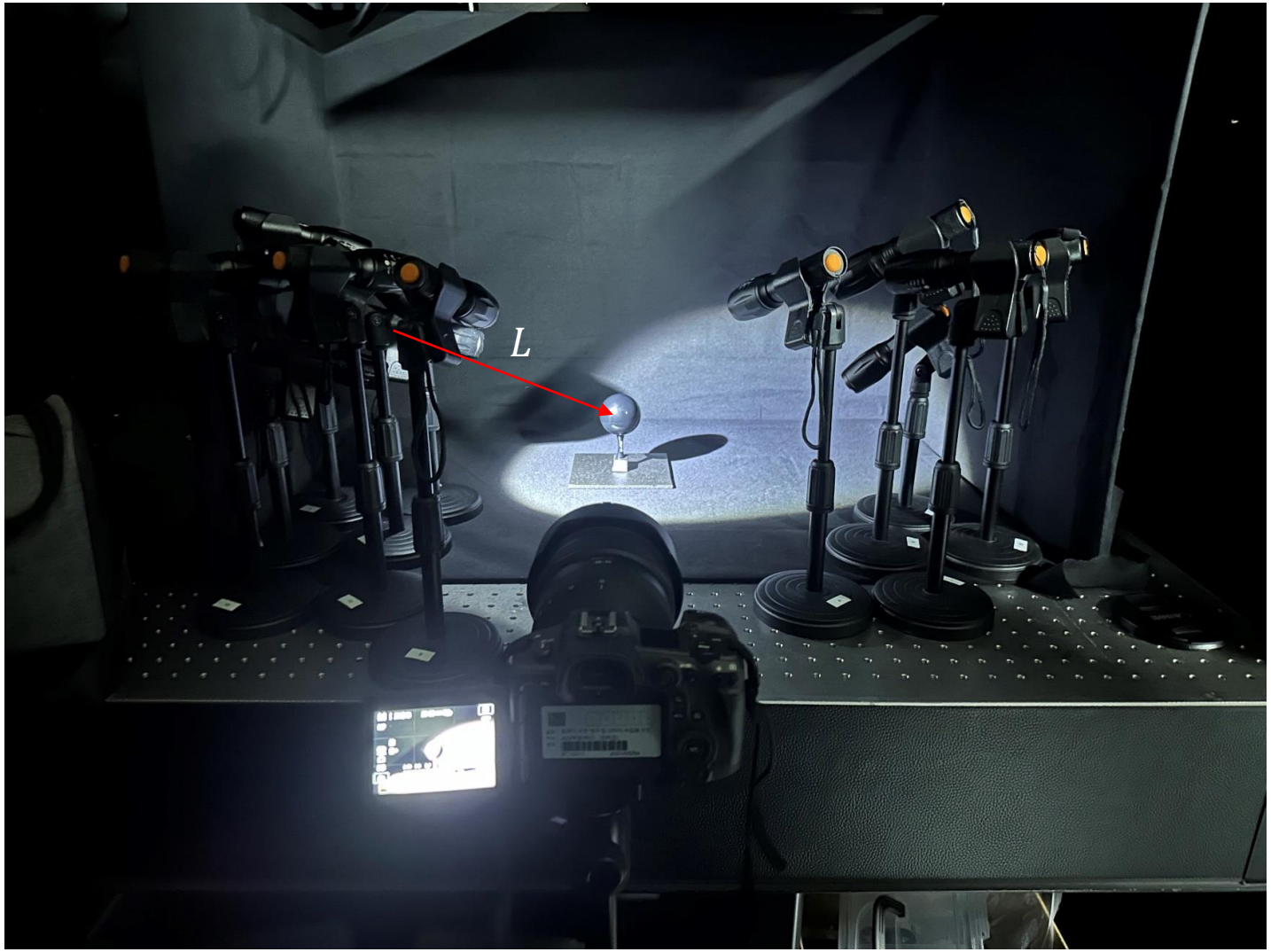
$$L_\mu(A, E, Y) = \|A\|_* + \lambda\|E\|_1 + \langle Y, D - A - E\rangle + \frac{\mu}{2}\|D - A - E\|_F^2$$

Let $(\hat{A}, \hat{E})$ be the optimal solution, then we can easily recover the matrix N from $\hat{A}$ as:

$$N = \hat{A}L^\dagger$$

Opening

Recall



Opening

Problem Setup

You will be given a collection of images:

For each of four objects you will receive 12 images.

Toothless

Nike

Moai

Choonsik

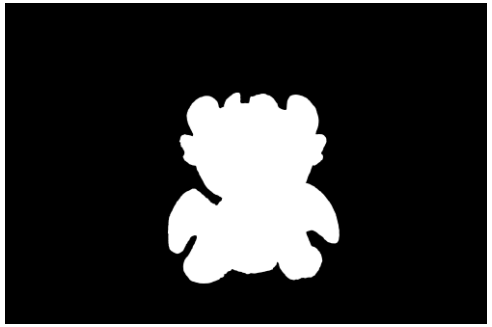
For a single object, all the images are taken from the same camera position and viewing direction.

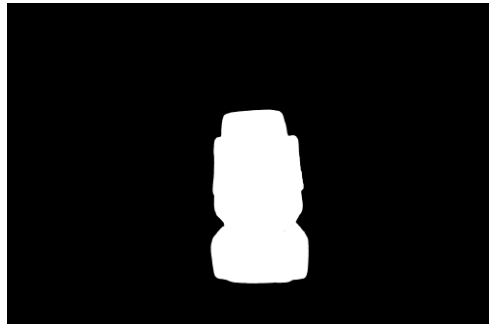
They are, however, taken under different light sources; this is the key.

Opening

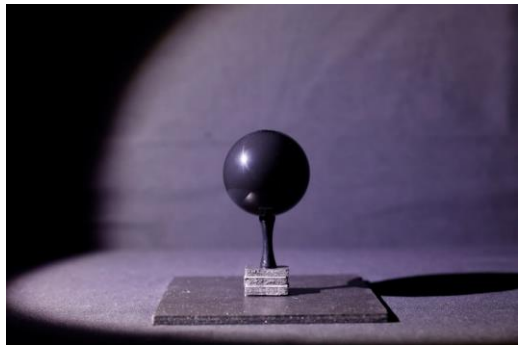
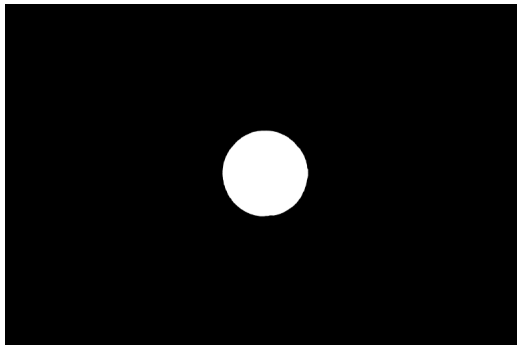
Problem Setup

And there is a mask image for each object that specifies those pixels to which the object projects in the images.






Finally, in addition to the objects there are images taken of a chromeball and a corresponding mask.

In this assignment, you will solve a simple photometric stereo problem, in which lighting directions are given.

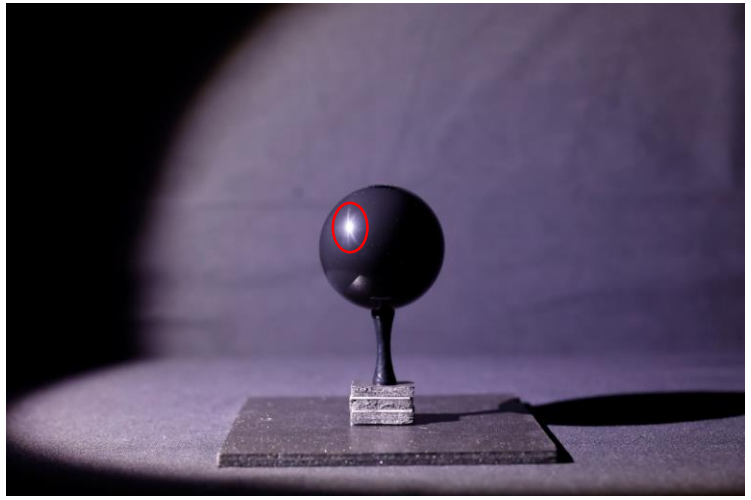
Code Implementation

Step 1. Recovery for Light Direction

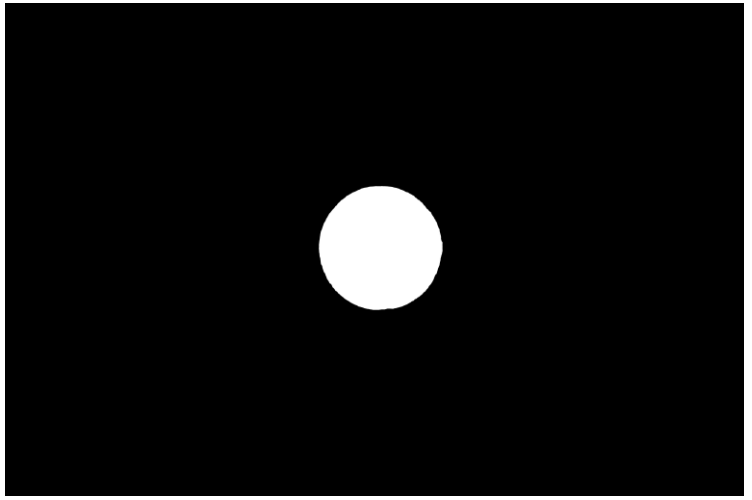
Before we estimate surface normals and albedo, it's essential to calibrate lighting setup.

One way to do this is by using a chromeball to estimate the light source directions.

Your Task) Complete the function in **recover_lightdir.py** so that it estimates the light source directions.



chromeball



mask

Code Implementation

Step 2. Lambertian Least Squares Photometric Stereo

For a single color channel, we can recover surface normals with more than 3 image samples under different lightings.

This can be done by solving a linear least squares problem for:

$$I = L \cdot \rho N$$

$$L^{-1} I = \rho N$$

Your Task) Complete the function in **least_squares.py** to obtain normal map and albedo.

Code Implementation

Step 3. Robust PCA Photometric Stereo

In real-world situations, if the reflectance deviates from the Lambertian assumption, errors can exist in surface normal.

Step3 is motivated by the reasonable observation that

Non-Lambertian observations emerge primarily in limited areas of each image.

With an assumption of sparse corruption E, the solution for the low-rank component A represents the diffuse observations that are free from sparse outliers:

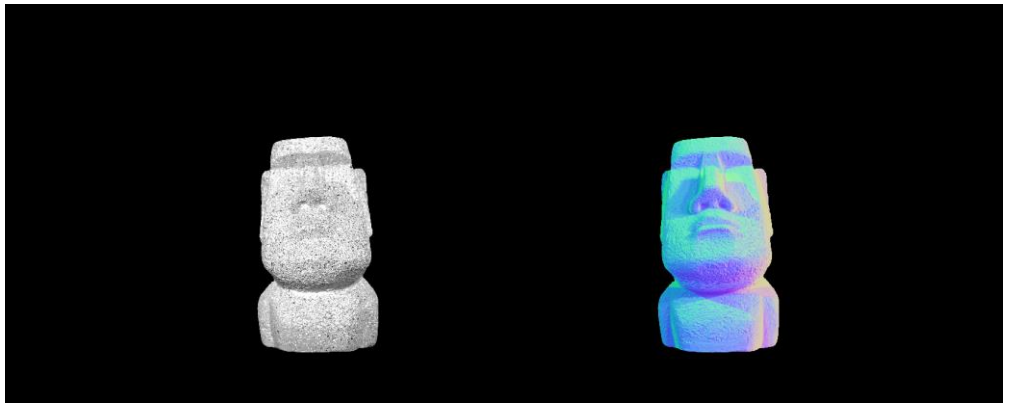
$$\min_{A,E} rank(A) + \gamma \|E\|_0 \quad s.t. \quad D = A + E$$

Your Task) Complete the function in **rpca.py** to obtain normal map and albedo.

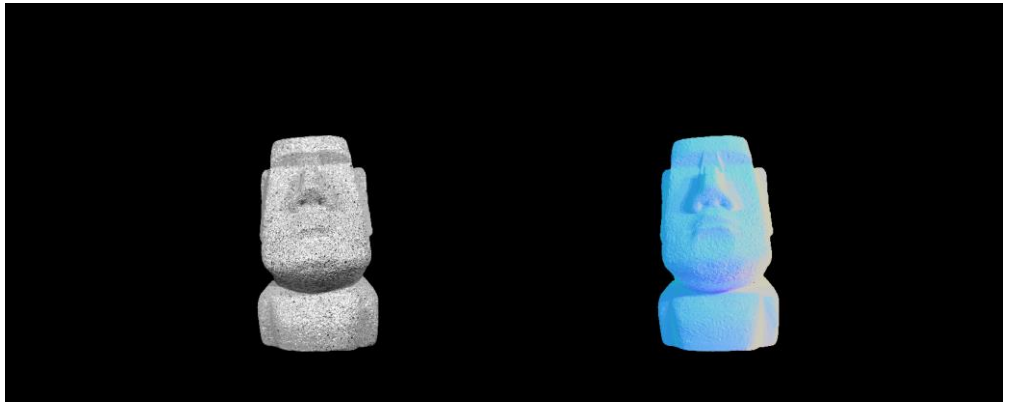
hint: solving Algorithm 1 in [3]

Code Implementation

Example Results for Step 2, 3



Lambertian Least Squares



Robust PCA

Code Implementation

Step 4. Relighting with an Estimate of Unknown Light

Since we have an estimate of the surface normals, we can now apply a practical application: relighting!

Using a given light direction, relight the object and evaluate the results by computing the Mean Squared Error (MSE) between the re-illuminated output and the ground truth.

Your Task) Complete the function in **relight.py** to relight an unknown object.

Code Implementation

Example Results for Step 4

Ground Truth

Relit Result

Guidance

Complete your code

Following the steps below to complete your code based on a given skeleton.

Step 1. Recover Light Directions using chromeball hints [recover_lightdir.py]

Step 2. Implement Lambertian Least Squares Photometric Stereo [least_squares.py]

Step 3. Implement Robust PCA Photometric Stereo [rpca.py]

Step 4. Measure Performance with Relighting Results

Make sure to carefully review the ***README file*** and ***main.py*** before starting.

You should fill in the ***#todo*** blank on a given skeleton.

Your result will be evaluated by running ***main.py***.

Guidance

Write your report

After completing your code, you need to write a report on your implementation.

Your report should include:

Understanding the Steps	*Explain the algorithms and key concepts used.*
Visualizing the Results	*Present surface normals, albedo, and relighting results for each object.*
Analyzing the Results	*Discuss any issues and possible solutions.*

At least three pages are allowed.

Instructions

You should implement:

- Step 1. Recover Light Direction (3 points)
- Step 2. Lambertian Least Squares (5 points)
- Step 3. Robust PCA (7 points)
- Step 4. Apply Relighting and Evaluate (2 points)

You should write:

- A report (3 points)

Additional credit with your own idea (up to 2 pts)

Remember!

0. No Plagiarism
0. No delay
0. No use of any open libraries/functions

TA session : 2025.4.1 & 2025.4.3

Due Date: 2025.4.5

Any Questions: jws5271a@gm.gist.ac.kr (TA)

Good Luck!

References

For your understanding of the concepts and implementation of the codes;

You should read [3], [5] to implement your codes well.

[1] Photometric method for determining surface orientation from multiple images

[2] Robust Principal Component Analysis?: Recovering low-rank matrices from sparse error

[3] Convex Optimization Based Low-Rank Matrix Completion and Recovery for Photometric Stereo and Factor Classification

[4] The Augmented Lagrange Multiplier Method for Exact Recovery of Corrupted Low-Rank Matrices

[5] Robust Photometric Stereo via Low-Rank Matrix Completion and Recovery