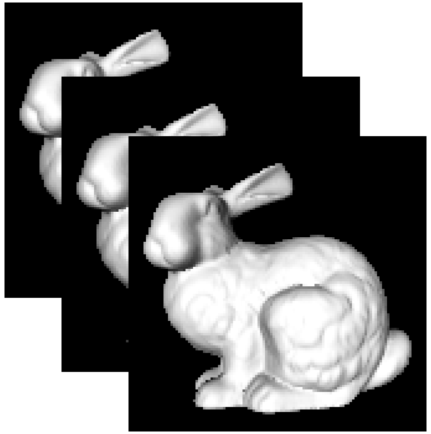
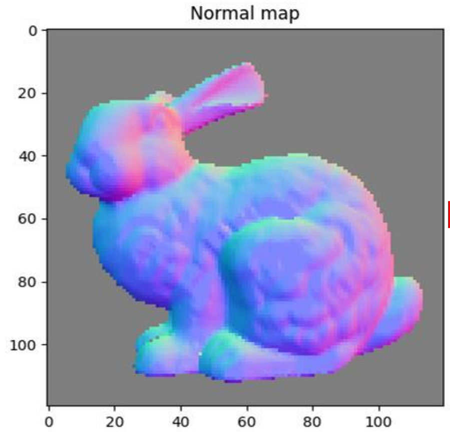
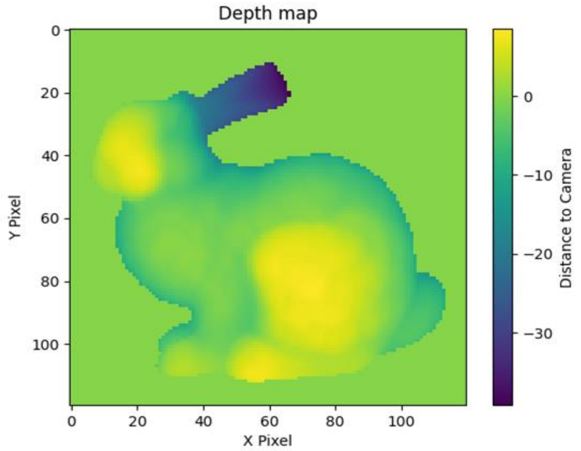
Homework 1

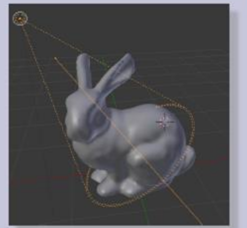
Computer Vision 2024 Spring

Photometric Stereo









Lambertian Reflection

$$i = \mathrm{k_d} \mathrm{l}(\mathbf{s}^\intercal \mathbf{n})$$

$$\longrightarrow$$

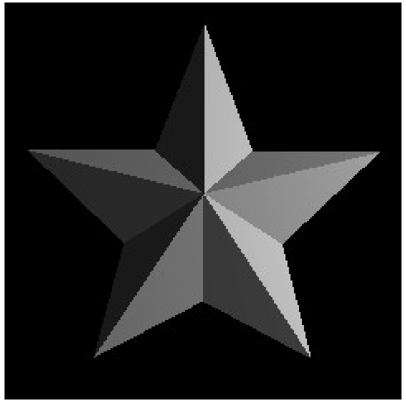
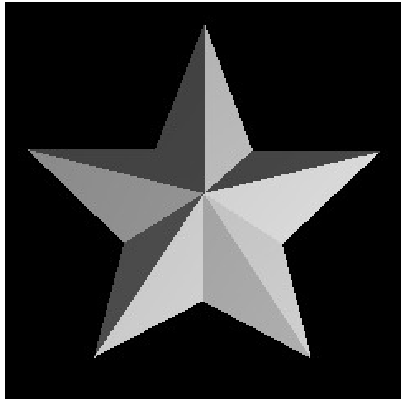
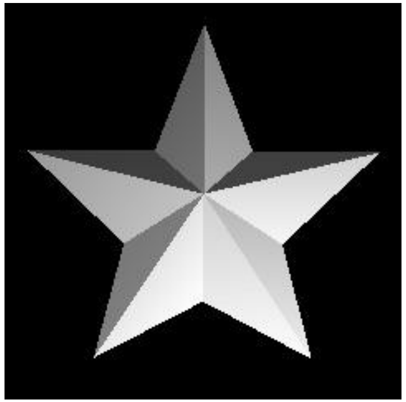
$$\longleftarrow$$

$$\mathrm{i} = \mathrm{k_d} \mathrm{l}(\mathbf{s}^\intercal \boldsymbol{n})$$

Normal Estimation

According to the reflection model, we suppose that the unknown normal vector n and the intensity i in the mth image at pixel (x,y) will satisfy the equation.
Kd is the coefficient of material , l is the "unit vector" of light vector

$$i_{x,y}^{(m)} = l_m K_d n$$

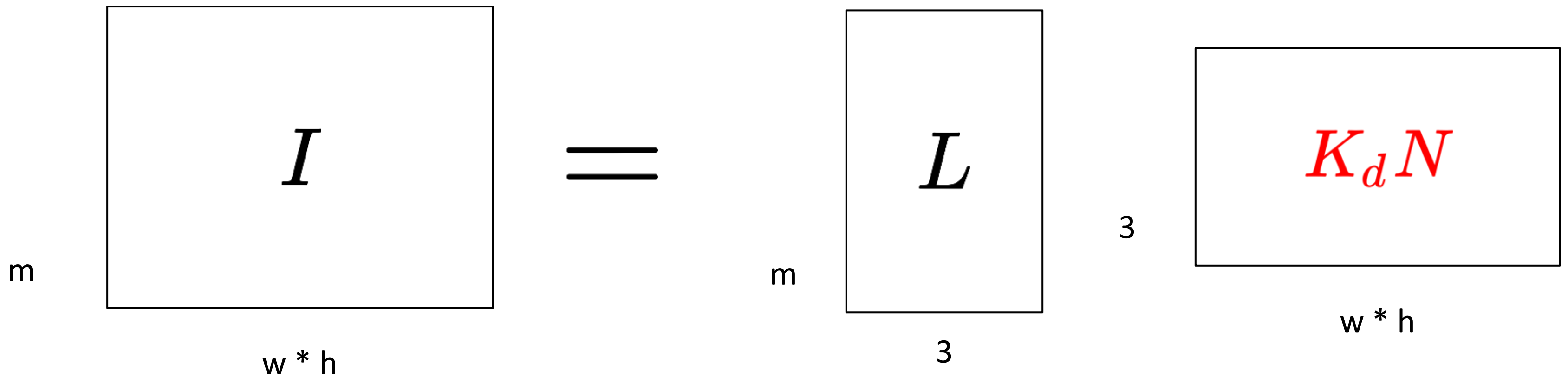




Normal Estimation

Assume we have m images and image width is w and height is h

We can construct an over-determined linear system and solve it to get the normal vector of every pixel

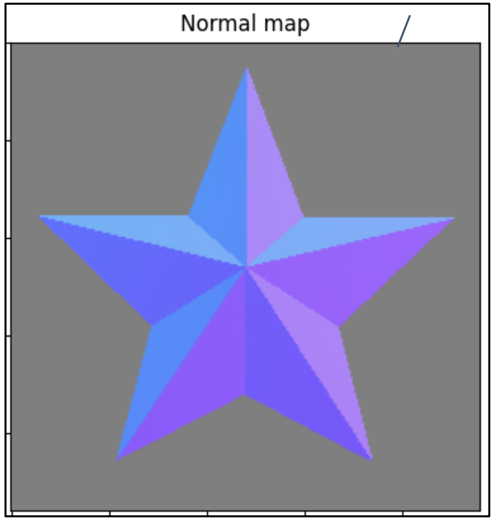
$$i_{x,y}^{(m)} = l_m K_d n \longrightarrow I = L K_d N$$



Normal Estimation

Solving this equation through pseudo-inverse (or QR , SVD decomposition)

$$I = LK_d N \Rightarrow L^T I = L^T L K_d N \Rightarrow K_d N = (L^T L)^{-1} L^T I$$



The result is KdN but we need "unit normal vector". Fortunately for n(x,y) , Kd(x,y) is a constant so we can directly apply vector normalization on KdN(x,y)

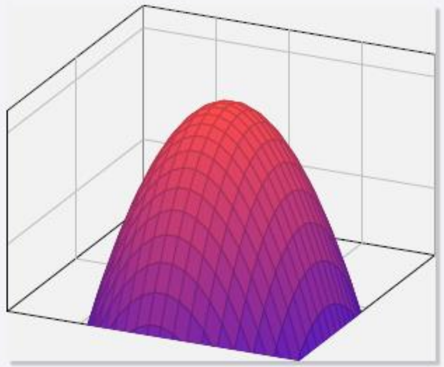
$$N = \frac{K_d N}{||K_d N||}$$

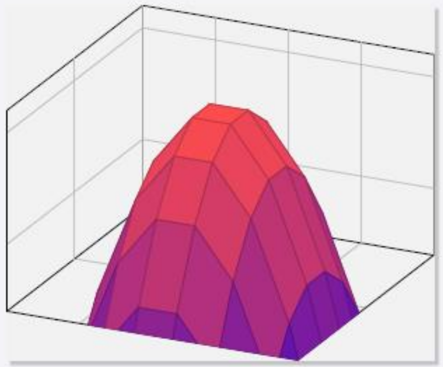
Surface Reconstruction 1.

The surface $z(x, y)$ near pixel $(\mathrm{x}^*, \mathrm{y}^*)$ can be approximated by the tangent plane:

$$\mathrm{n}_1(x - \mathrm{x}^*) + \mathrm{n}_2(y - \mathrm{y}^*) + \mathrm{n}_3(z - z(\mathrm{x}^*, \mathrm{y}^*)) = 0 \qquad (1)$$

where $(\mathrm{n}_1, \mathrm{n}_2, \mathrm{n}_3)^\mathsf{T}$ is the normal vector at $(\mathrm{x}^*, \mathrm{y}^*)$.





Surface Reconstruction 1.

The equation 1 can be rewritten as

$$z_{\mathsf{approx}}(x, y) = \left(-\frac{\mathrm{n}_1}{\mathrm{n}_3}\right) x + \left(-\frac{\mathrm{n}_2}{\mathrm{n}_3}\right) y + \mathsf{constant}$$

We can reconstruct the surface $\tilde{z}(x, y)$ as we know the gradient of z_{approx} at each pixel, for example, by

$$\tilde{z}(x, y) = \sum_{i=0}^{x-1} \left.\frac{\partial z_{\mathsf{approx}}}{\partial x}\right|_{(i,0)} + \sum_{j=0}^{y-1} \left.\frac{\partial z_{\mathsf{approx}}}{\partial y}\right|_{(x,j)}$$

Surface Reconstruction 1.

- Other Tips
  - You may need to use some method such like integral from different direction or begining at different initial point and average those result to let surface more smooth

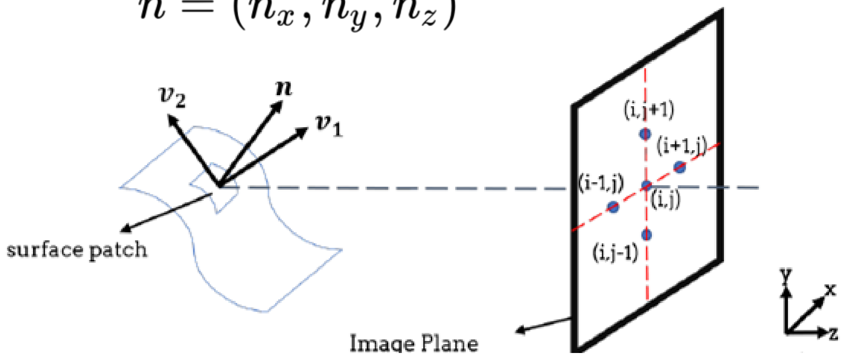
  - Sanity Check $\dfrac{\partial^2 z}{\partial x \partial y} = \dfrac{\partial^2 z}{\partial y \partial x}$

Surface Reconstruction 2.

- The Normal n must be orthogonal to the vector v1 & v2

$$V_1 = (x+1, y, z_{x+1,y} - (x, y, z_{x,y})$$
$$V_2 = (x, y+1, z_{x,y+1} - (x, y, z_{x,y})$$
$$n = (n_x, n_y, n_z)$$

$$\boxed{\begin{array}{l} v_1 \cdot n = 0 \\ v_2 \cdot n = 0 \end{array}}$$
$$\big\downarrow$$
$$\boxed{\begin{array}{l} n_x + n_z(z_{x+1,y} - z_{x,y}) = 0 \\ n_y + n_z(z_{x,y+1} - z_{x,y}) = 0 \end{array}}$$
$$\big\downarrow$$
$$\boxed{\begin{array}{l} z_{x+1,y} - z_{x,y} = -\dfrac{n_x}{n_z} \\ z_{x,y+1} - z_{x,y} = -\dfrac{n_y}{n_z} \end{array}}$$



Reference :
https://pages.cs.wisc.edu/~csverma/CS766_09/Stereo/stereo.html

Surface Reconstruction 2.

- We can use this two equations of every pixels to construct a linear system $Mz = V$
  M may be a big sparse matrix
  (Let image size S = image width * image height)
- We can use pseudo inverse again to solve it

$$z_{x+1,y} - z_{x,y} = -\frac{n_x}{n_z}$$

$$z_{x,y+1} - z_{x,y} = -\frac{n_y}{n_z}$$

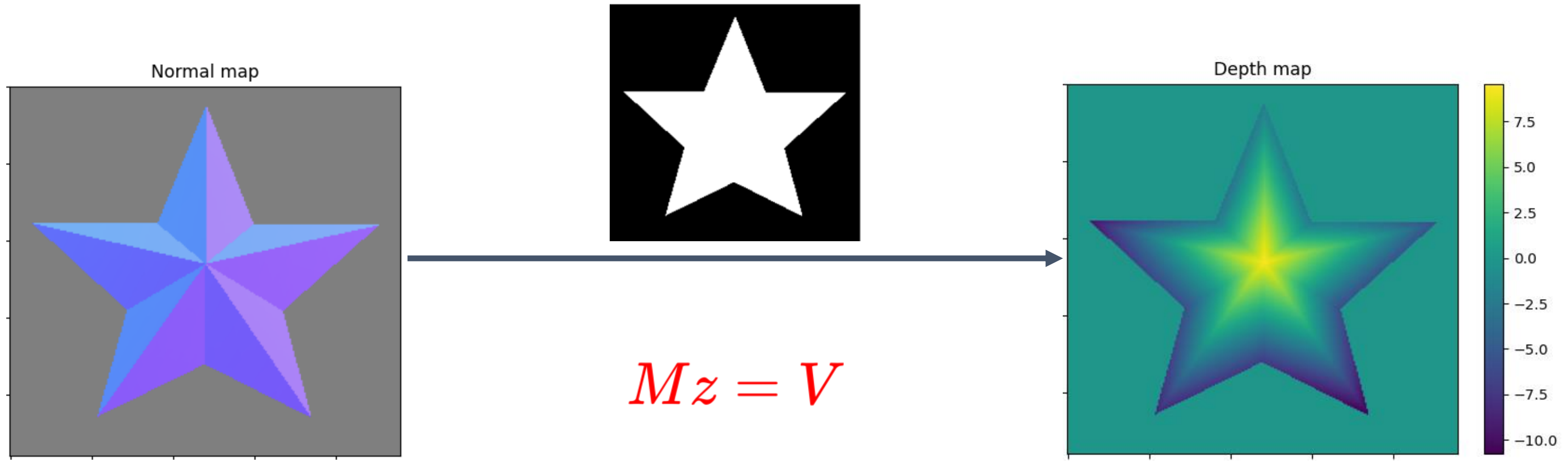
$$M^T M z = M^T V$$

$$z = (M^T M)^{-1} M^T V$$

$$\underset{2S \times S}{\overset{M}{\begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & -1 & 1 & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & -1 & \cdots & \cdots & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}}} \underset{S \times 1}{\overset{z}{\begin{bmatrix} \vdots \\ z_{50,50} \\ z_{51,50} \\ \vdots \\ z_{50,51} \\ \vdots \end{bmatrix}}} = \underset{2S \times 1}{\overset{V}{\begin{bmatrix} \vdots \\ -\frac{n_x^{50,50}}{n_z^{50,50}} \\ \vdots \\ \vdots \\ -\frac{n_y^{50,50}}{n_z^{50,50}} \\ \vdots \end{bmatrix}}}$$

Surface Reconstruction 2.

- Other Tips
  - You may need to eliminate some pixel already known its depth value is 0 to reduce the size & complexity of $Mz = V$
  - You can create a "mask" to focus on those unknow depth



$$Mz = V$$

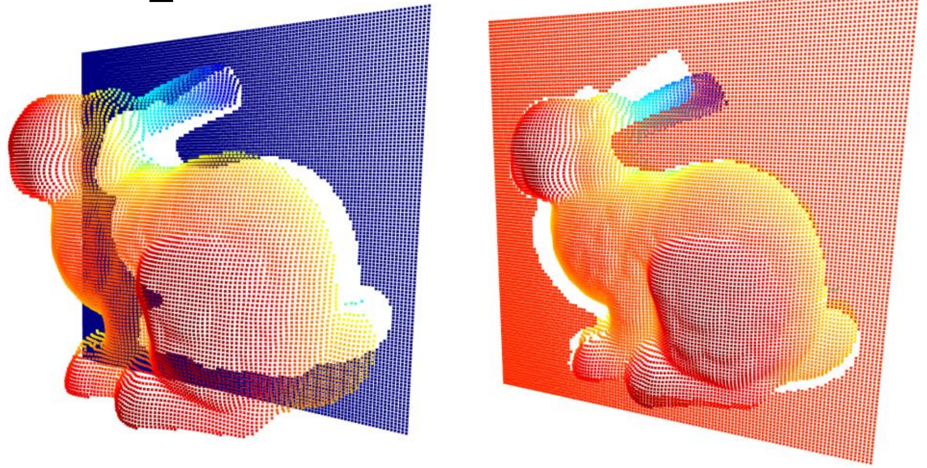
Install

- Python
- OpenCV : http://opencv.org
    - pip install opencv-python
- open3D: http://www.open3d.org/
    - pip install open3d
- matplotlib: https://matplotlib.org/
    - pip install matplotlib

- You can install other library which you are family with for solving linear equation
(But don’t use the library directly complete the photometric stereo or you may not get score)

Input & Output

- Input:
  - 4 case (bunny , star , venus, noisy_venus)
  - 6 .bmp image
  - LightSource.txt



- Output:

  .ply file(Polygon File Format)

  (named as bunny.ply , star.ply , venus.ply, noisy_venus.ply respectively)
  - using the open3D to output ply file
    - You can use the function "save_ply()" we provide
      (this function may set 0 depth vaule to minimum depth value let whole model float on a plane)
    - If you don't like the way to display whole model floating on a base plane, you can change the way to save ply file. The only limitation is that the ply file need to be able to open and show the result by "show_ply()" function.

Grading

50% Reconstruct surfaces of “bunny” & “star” data (25% / per data)

(Please reduce unnecessary distortion and rugged surfaces. The score you get will depend on the quality of your reconstructed surface)

20% Show the normal map and depth map of “bunny” & “star” in report. (5% each)

10% Simply explain your implementation and what kind of “method” you use to enhance the result and compare the result (in report)

(Don’t just paste the code with comment)

10% Reconstruct surfaces of “venus” (This case you may need to find some ways to deal with some extrem normal result, or you wont’s get full score.) and explain in report.
(we may treat “venus” more strictly than “bunny” & “star”)

10% Reconstruct surfaces of “noisy venus” where Gaussian noise has been applied to the input images. (If you incorporate specific approaches to tackle noisy venus data, please explain them in the report.)

Deadline

- Deadline : 2024/03/31 (Sun.) 11:59 pm
- Please zip the all files and name it as {studentID}_HW1.zip :
  ex. 311553000_HW1.zip (wrong file format may get -10% panelty)
  - Zip file format:
    - 1. {studentID}_report.pdf
    - 2. bunny.ply, star.ply, venus.ply, noisy_venus.ply
    - 3. your code
- Penalty of 3% of the value of the assignment per late day
  - late one day : your_score * 0.97
  - late two days : your_score * 0.94 …
  - If late, you need to notify all TAs through e-mail or you won’t get scored.
- Feel free to ask questions on E3 forum, or send e-mails through E3 platform to all TAs for personal questions.