Civil 7453

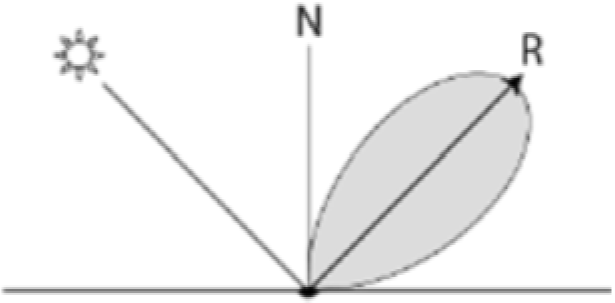
Assignment1 Report

Mao Li

Keywords: Calibrate illumination direction, Estimate surface normal direction, Lamber surface, Albedo

Background:

1. Get source light direction:
   Phong model: For shiny surface(specular reflection)



Observed pixel intensity:

$$Lr = Li(V * R)^{n}$$
$$L = 2 * (N * R) * N - R$$

For the second equation we can get the direction of the source light.

Also, for white lamber surface, we can assume the albedo as 1.

Lamber surface:

$$Lo = Li * p * (N * L)$$

Then, we can get the magnitude of the source light should correspond to the brightest point intensity on lamber surface. Lo is irradiance, Li is the enter light intensity and, N is normal direction and L is the enter light direction.

2. Get Normal for each pixel

$$\begin{pmatrix} I(1,p) \\ \dots \\ I(N,p) \end{pmatrix} = Kd \begin{pmatrix} {L_1}^{T} \\ \dots \\ {L_N}^{T} \end{pmatrix} * N(p)$$

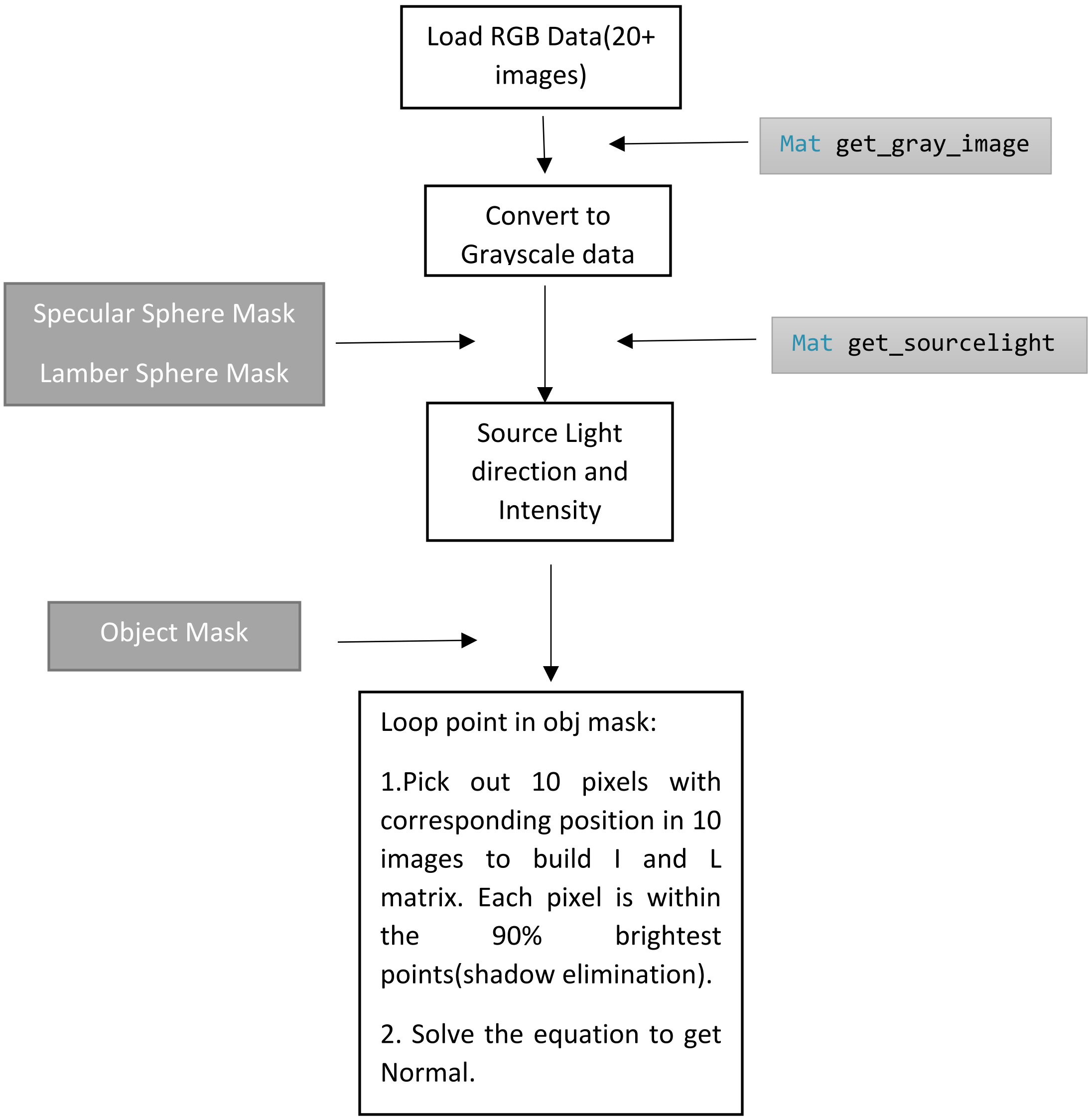
I(n,p) means the pixel in image n, with position in p. Ln means the source light for image n, and N(p) is the normal for pixel in position p.

3. Generate albedo map.
   For each point on arbitrary surface:

$$Lo = Li * p(\Theta o, \Phi o, \Theta i, \Phi i) * (N * L)$$

With Normal and source light known, we can calculate the p for each point.

Code breakdown


Load RGB Data(20+ images)
Mat get_gray_image
Convert to Grayscale data
Specular Sphere Mask
Lamber Sphere Mask
Mat get_sourcelight
Source Light direction and Intensity
Object Mask

Loop point in obj mask:

1.Pick out 10 pixels with corresponding position in 10 images to build I and L matrix. Each pixel is within the 90% brightest points(shadow elimination).

2. Solve the equation to get Normal.

Experiment result:

The first three is the albedo and re-rendered image set. The left and brighter ones are albedo map while the right ones are re-rendered images with source light set to [0 0 1]. It make sense the re-rendered map is darker because

$$Lo = Li * p * (N * L)$$

(N*L) < 1.

The last three images are the normalized normal map. I am setting my coordinate system as x axis as row, and y axis as columns, which may differ from others with different coordinate system.

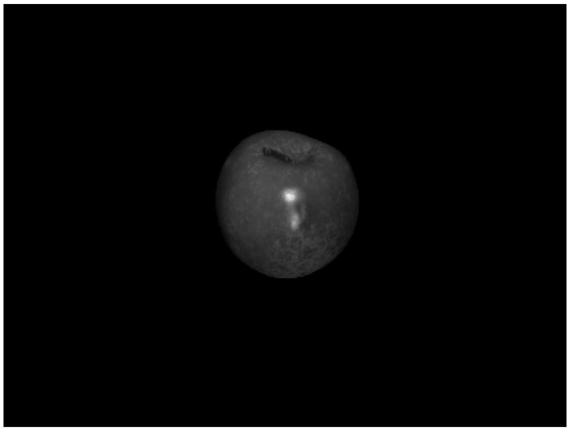


Fig.1

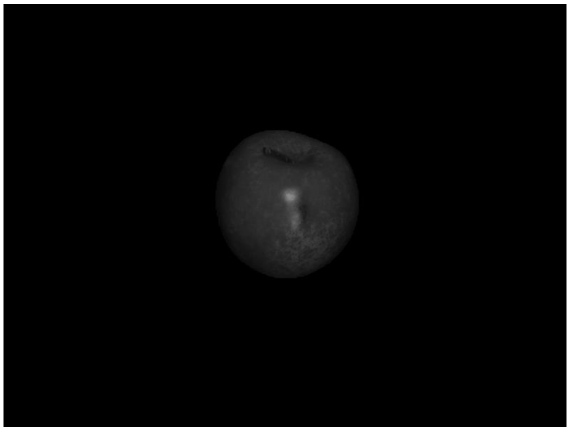


Fig.2

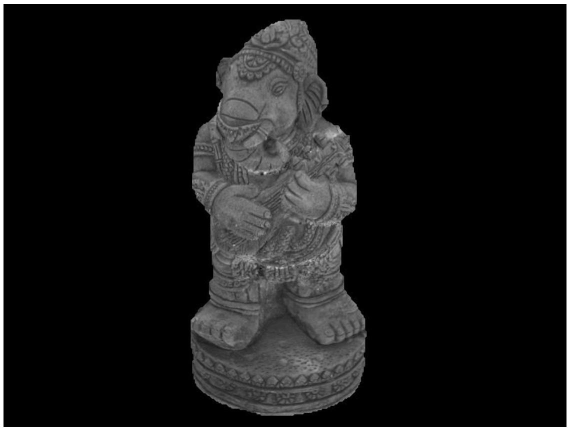


Fig.3

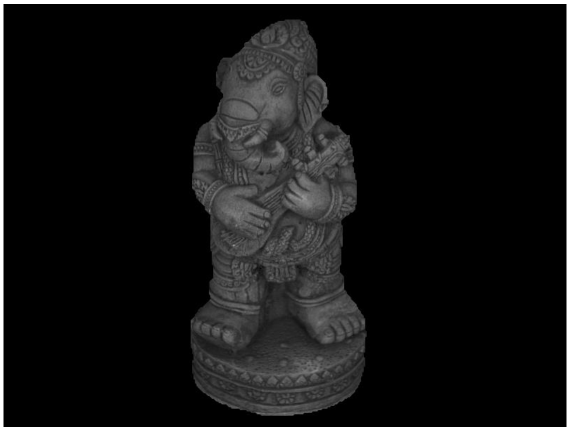


Fig.4

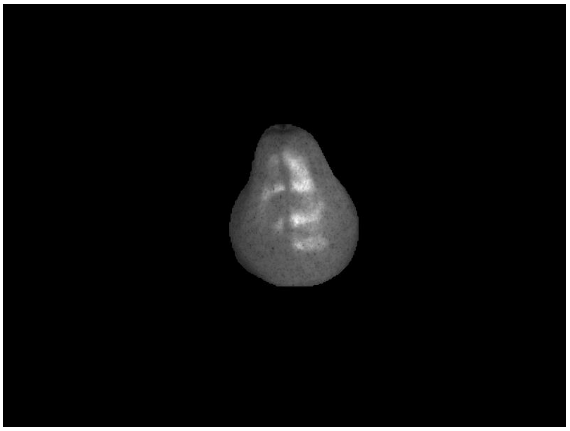


Fig.5

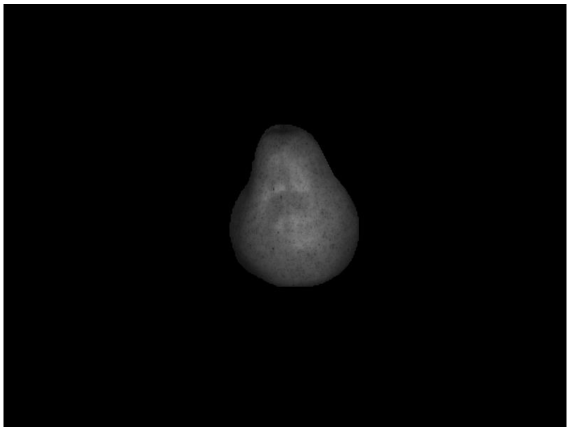


Fig.6

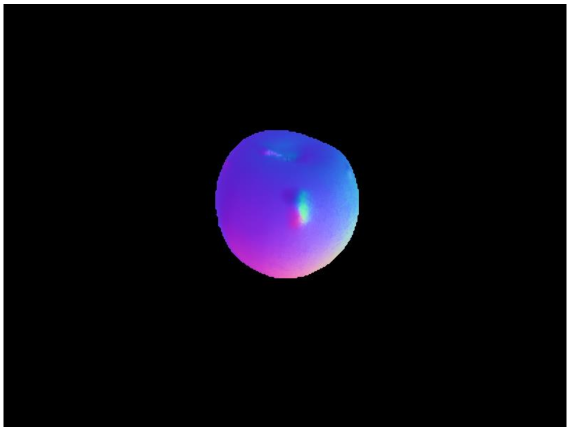


Fig.7

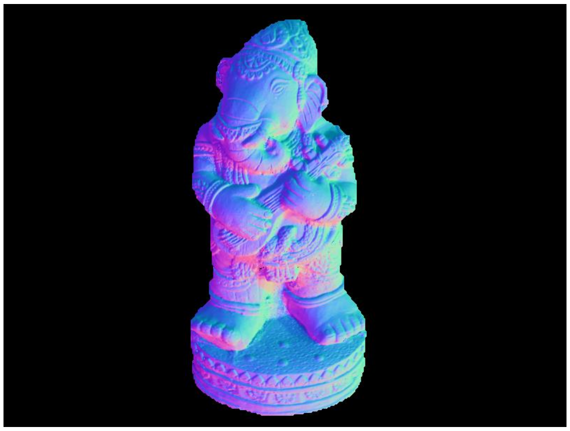


Fig.8

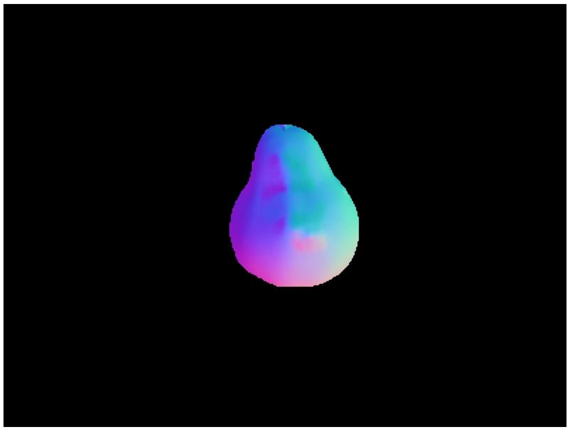


Fig.9

In the beginning, I am only choose 3 pixels to calculate the normal for each point. But the outcome is very noisy and cannot even recognized the object. I found that with more pixels to pick, the result become more reasonable. To improve my algorithm, I can pick out as many points as possible to recover the normal.