Computer Vision Homework 1

Guocheng Qian
KAUSTID: 172525
guocheng.qian@kaust.edu.sa

1. Photometric Stereo / Shape-from-Shading [60 pts]

1.1. PhotometricStereo() function

I calculate albedo and normals using simple matrix operations based on the followed equations.

$$\rho_i = \|\boldsymbol{X}_i\|_2 \tag{1}$$

$$\boldsymbol{N}_i = \frac{\boldsymbol{X}_i}{\|\boldsymbol{X}_i\|_2} \tag{2}$$

where ρ is albedo image (the albedo for each pixel). $\boldsymbol{N}$ is the normal vectors. ρ and $\boldsymbol{N}$ can be obtained by simply element-wise matrix division using ./ in MATLAB. $\boldsymbol{X}$ can be obtained by simple matrix operations below:

$$\boldsymbol{X} = (\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{B} \tag{3}$$

where $\boldsymbol{S}$ denotes the light source directions of multiple images, which is given. Its size is $3 \times n$, n is the number of images. $\boldsymbol{B}$ denotes the pixels of multiple images, which is also given. $\boldsymbol{B}$ has a size of $3 \times M$, $M = H \times W$, and H, W is the height and width of input images respectively.

The code is available in the supplementary material (HW1_code_guochengqian.zip).

1.2. Estimate albedo and normals

Debug code. I first debug my code by comparing my results with the ground-truth using the "circle" example. I success to generate very similar results, which clearly shows the correctness of the code. One can check by setting the subjectName to circle in the code.

Estimated albedo. I show the estimated albedo for each of the five surfaces in Figure 1. The estimated quality looks very convincing. First, the circle example generates exactly the same albedo as the ground-truth (one can test it by running my code). Second, the eyebrow, the hair, the cornea (the middle part of eyes), the nostrils, and the beards are darker, which is what we expect. The reason is in these regions, the reflection ratio (albedo) is smaller. Also, look at sclera (the white part of eyes), it is whiter than the other part due to the reason that the sclera has a higher albedo (reflects the light more).

Estimated normals. I show the estimated normals for each of the five surfaces in Figure 2. The estimated normals of the circle is exactly the same as the ground-truth (one can test it by running my code). The estimated normals of real shot human face also show a great quality. The whole image looks bluish overall due to the reason the the normals are perpendicular to the paper since the humans are faced at the camera. Therefore, the normals are nearly $[0, 0, 1]^T$, which is the color of blue. At the left part of noses, the images becomes purplish because now the normal of the left side of noise is nearly $[1, 0, 1]^T$.

1.3. Generate the images of each surface for three light directions

The generated images are shown in Figure 3. The images of the top row, the middle row and the bottom row in Figure 3 are generated under three different light directions: $s_1 = [0, -1, 1]^T$, $s_2 = [0, 1, 1]^T$, and $s_3 = [1, 1, 1]^T$. Note that the circle is into the page and the four faces are towards out the page (these can be found in the height map, refer to the code). Using s_1, the light is from the bottom into the page (135 degree between y and z), so the body of the nose is dark because the nose blocks the light. For the circle, the light can lit up the upper part but the light is blocked by the shell of circle in the bottom. Remember the circle is into the page, the shape is like a bowl. Using s_2, the light is from the upper into the page (45 degree between y and z), the faces are nearly all lit up. For the circle, the light can lit up the bottom part but the light is blocked by the shell of circle in the upper part. As for the case of s_3, the light is from the top left into the page. The top left part of the faces are lit up. For the circle, the light is block by the shell of the top left part.

1.4. Estimate the 3D points of each surface

The generated images are shown in Figure 4. The top row shows the 3D points visualized using "mesh" command in MATLAB. The second row shows the 3D points visualized using "surf" command in MATLAB. The bottom row

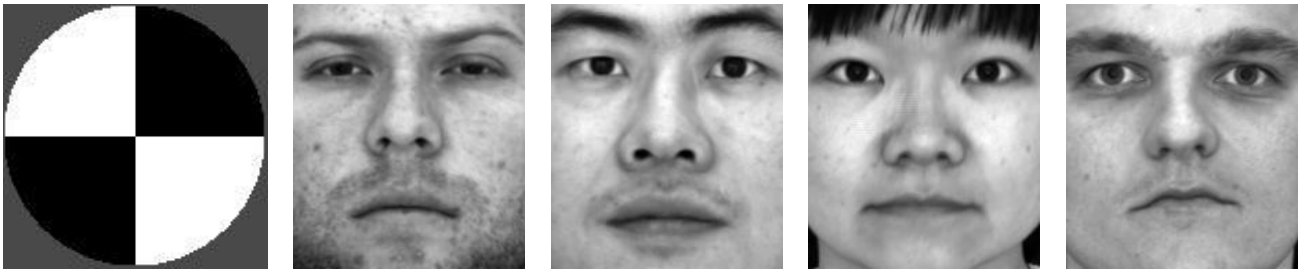


Figure 1: **Estimated albedo for each of the five surfaces.** The estimated albedo iamges are quite convincing. First, the circle example generate exactly the same albedo as the ground-truth (one can test it by running my code). Second, the eyebrow, the hair, the cornea (the middle part of eyes), the nostrils, and the beards are darker, which is what we expect. The reason is in these regions, the reflection ratio (albedo) is smaller. Also, look at sclera (the white part of eyes), it is whiter than the other part. It is because the sclera has a higher albedo.

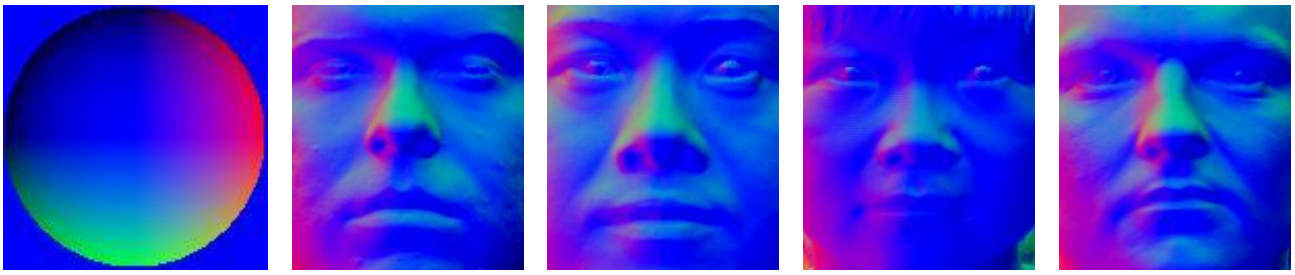


Figure 2: **Estimated normals for each of the five surfaces.** The estimated normals of the circle is exactly the same as the ground-truth (one can test it by running my code). The estimated normals of real shot human face also show a great quality. The whole image looks bluish overall due to the reason the the normals are perpendicular to the paper since the humans are faced at the camera. Therefore, the normals are nearly $[0, 0, 1]^T$, which is the color of blue. At the left part of noses, the images becomes purplish because now the normal of the left side of noise is nearly $[1, 0, 1]^T$.

shows the 3D points visualized using MeshLab.

2. Discrete Fourier Transform [10pts]

(a) matches with (g): two columns and symmetric.

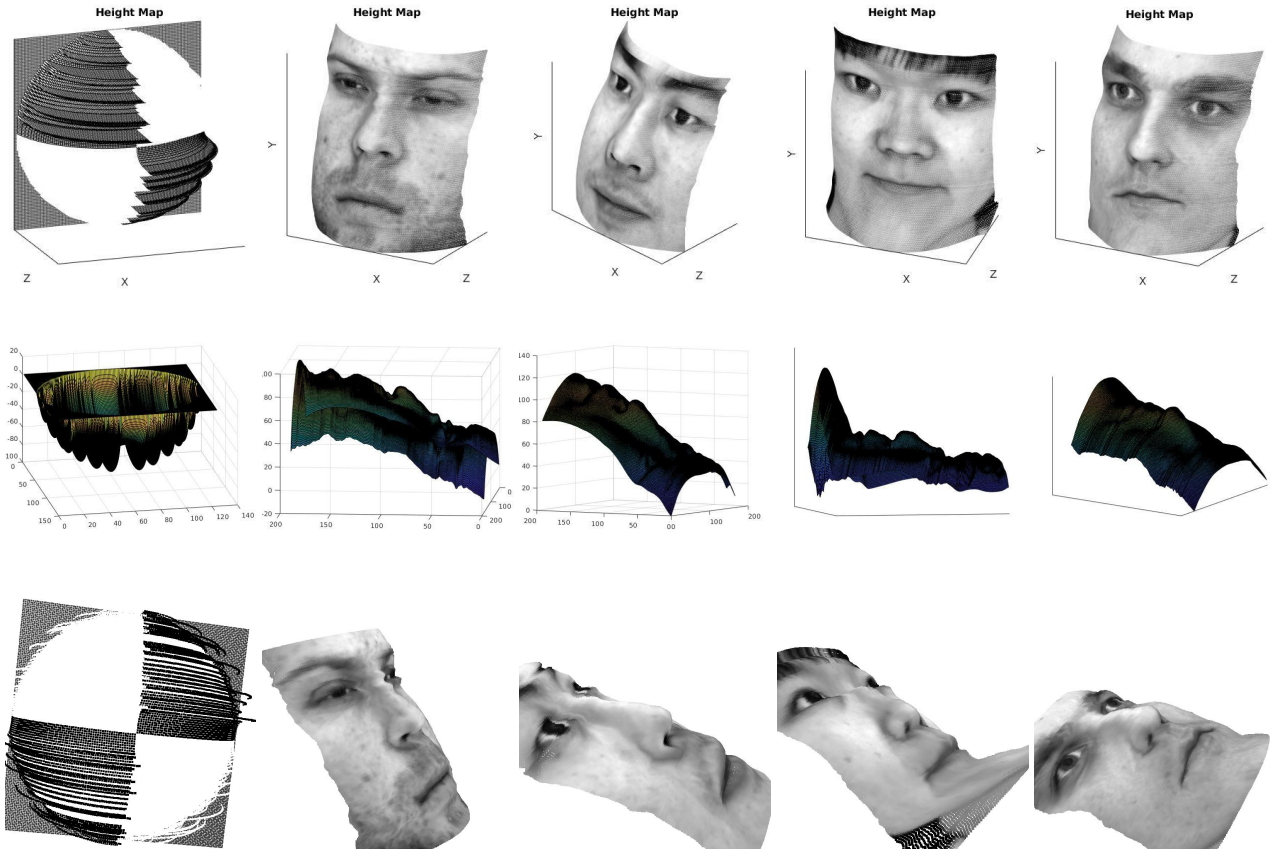
(b) matches with (i): image (b) has a slight motion blur in all direction (looks like the image is shot when there is a wind or when the camera is not stable). Therefore its Fourier image shows a even low frequency vertically and horizontally.

(c) matches with (h): sharp horizontal edges and vertical edges leads to strong vertical and horizontal frequency in the the Fourier image.

(d) matches with (f): Gaussian kernel. Low frequency pass.

(e) matches with (j): frequent horizontal waves leads to strong vertical frequency.

Figure 3: **Generated images of each surface for three light directions.** The top row, the middle row and the bottom row are generated under three different light directions: $s_1 = [0, -1, 1]^T$, $s_2 = [0, 1, 1]^T$, and $s_3 = [1, 1, 1]^T$ respectively. Note that the circle is into the page and the four faces are towards out the page. Using s_1, the light is from the bottom into the page (135 degree between y and z), so the body of the nose is dark because the nose blocks the light. For the circle, the light can lit up the upper part but the light is blocked by the shell of circle in the bottom. Remember the circle is into the page, the shape is like a bowl. Using s_2, the light is from the upper into the page (45 degree between y and z), the faces are nearly all lit up. For the circle, the light can lit up the bottom part but the light is blocked by the shell of circle in the upper part. As for the case of s_3, the light is from the top left into the page. The top left part of the faces are lit up. For the circle, the light is block by the shell of the top left part.



(a) circle (b) yaleB01 (c) yaleB02 (d) yaleB05 (e) yaleB07

Figure 4: **Estimated 3D points of each surface**. The top row shows the 3D points visualized using ”mesh” command in MATLAB. The second row shows the 3D points visualized using ”surf” command in MATLAB. The bottom row shows the 3D points visualized using MeshLab.

3. Image Reconstruction from Phase Information Only [30pts]

3.1. Reconstruction from phase only algorithm

The image reconstruction algorithm from phase only is illustrated in Algorithm 1.

Algorithm 1: Iterative algorithm for image reconstruction from phase only

Input: The phase response f with size $2H \times 2W$ of the image.

Initialize the amplitude response M with size $2H \times 2W$ (M can be unit initialized, random initialized or initialized by using the Fourier amplitude response of another reference image.

while *not terminated* **do**

1. Predict the Fourier transformation F of the image by $F = M \otimes \exp(\mathrm{j} * f)$, where $\otimes$ denotes the dot product;
2. Reconstruct the image by conducting inverse Fourier transformation and then obtain the absolute value I;
3. Zero out the padded regions of the image: $I(x, y) = 0$ except for $0 \leq x \leq H - 1, 0 \leq y \leq W - 1$;
4. Conduct Fourier transformation on the new reconstructed image I and obtain F'.
5. Update the amplitude response M by $M = \mathrm{abs}(F')$.

The termination condition can be controlled simply using the maximum iteration.

Output: the reconstructed image is $I(x, y)$ for $0 \leq x \leq H - 1, 0 \leq y \leq W - 1$;

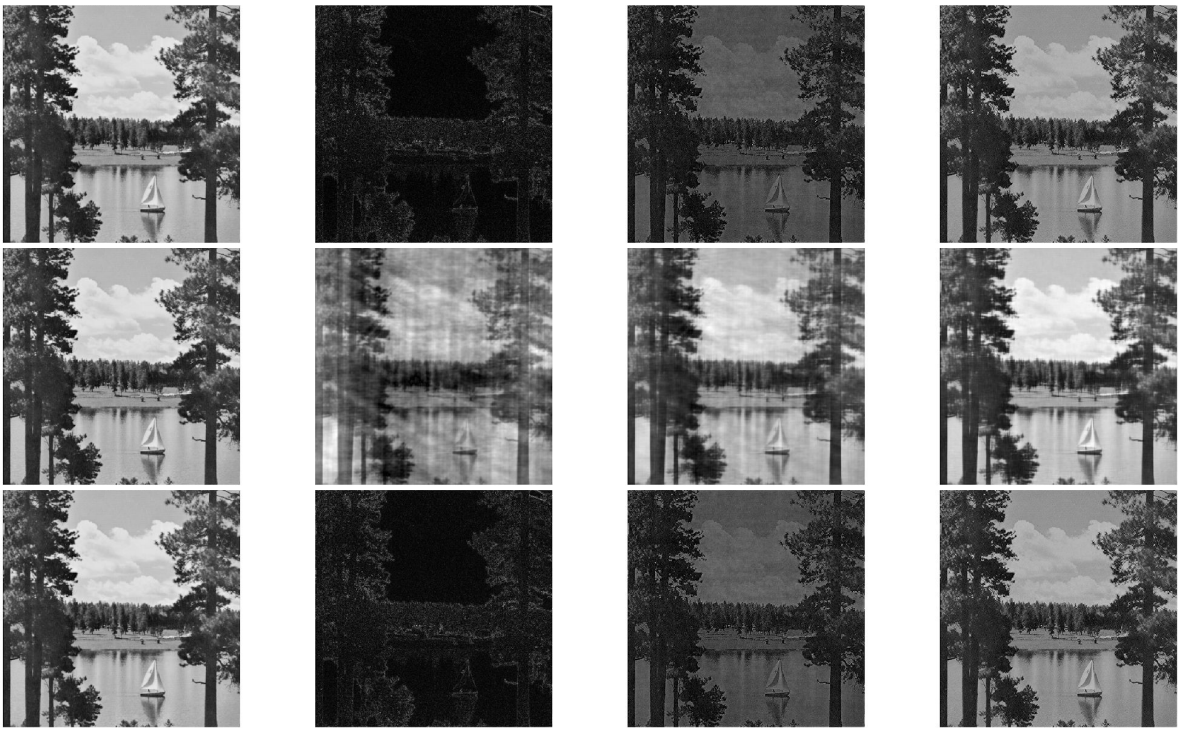
The top row in Figure 5 shows the reconstructed image using Algorithm 1 with unit amplitude response initialization. The unit initialization generates the initial guess of amplitude response equal to one for all the values. Algorithm 1 with unit amplitude response initialization successes to reconstruct images.

3.2. Apply your code using the provided lake and house images

The middle row in Figure 5 shows the reconstructed image using the house image as amplitude response initialization. The reconstructed images using a reference image shows a better quality with regards to the illumination compared to all the other methods. This is due to the reason that reconstructed image and the reference image share a similar Fourier amplitude response (this characteristic happens often among the natural images with similar illumination). Therefore the amplitude of the house image is a better prior compared to the unit initialization and random initialization. However, as one can see in the first iteration, the middle row generates some artifacts while the others do not. The reason is in the first few iterations, the strong edges in the reference image, whose pattern is not in the reconstructed image, will degrade the reconstruction quality and generate such edge artifacts.

3.3. Repeat the previous step but with a random generated positive image

The bottom row in Figure 5 shows the reconstructed image using Algorithm 1 with random amplitude response initialization. One can observe that the unit initialization and random initialization have similar performance. In the first few iterations, they suffer less from edge artifacts but they recover less content meanwhile. In the last iteration, the content of the images of all the methods look similar. However, the unit initialization and random initialization generate the images with darker illumination compared to using the house image as the amplitude response initialization.



(a) original image (b) iteration 1 (c) iteration 10 (d) iteration 100

Figure 5: **Image reconstruction results using phase information only.** The top row, middle row, and bottom row shows the phase only reconstructed images using unit initialization, the house image, and the random initialization as amplitude response initial guess respectively. All methods success to reconstruct images. The reconstructed images using a reference image (the middle row) shows a better quality with regards to the illumination. This is due to the reason that reconstructed image and the reference image share a similar illumination level. Therefore the amplitude of the house image is a better prior compared to the unit initialization and random initialization. However, as one can see in the first iteration, the middle row generates some edge artifacts while the others do not. The reason is in the first few iterations, the strong edges in the reference image, whose pattern is not in the reconstructed image, will degrade the reconstruction quality and generate such edge artifacts.