Homework 2
M1522.001000 Computer Vision (2019 Spring)

Due: Wednesday April 11 02:00PM.

1 Submitting Your Assignment

Your submission for this assignment will comprise of answers to a few theory questions, the code for your **MATLAB** implementation and a short writeup describing any thresholds and parameters used, interesting observations you made or things you did differently while implementing the assignment. The answers to the theory questions in Section 2 should be in a plaintext file or a pdf named `theory.txt or theory.pdf`. Each of the MATLAB functions you write as described in Section 3 along with any extra helper functions you wrote should be in a folder named `matlab`, upload only `.m` files to this folder and make sure all the files needed for your code to run (except data) are included. The writeup describing your experiments should be in a plaintext file or a pdf named `experiments.txt` or `experiments.pdf`. Your writeup should be **typed** with **English**.

Put your **code and writeup** into a directory called "`(studentid)-(yourname)-HW2`" and pack it into a `tar.gz` or `zip` named "`(studentid)-(yourname)-HW2`". For example, `201921234-gildonghong-HW2.tar.gz` or `201921234-gildonghong-HW2.zip`. and send it to TA's email(`ta.cv@vision.snu.ac.kr`) with title as "`(studentid)-(yourname) -HW2`".

Your submitted zip file should include the files arranged in this layout:

- `theory.txt` (or `.pdf`): answers to the theory questions in Section 2
- `experiments.txt` (or `.pdf`)
- Folder <u>`matlab`</u>
  - `q2Script.m` (*already given, upload your modified copy*)
  - `q1Script.m`, `findCircle.m`, `findLight.m`, `computeNormals.m`, `separateGlobalDirect.m`, `q3Script.m`
- Folder <u>`data`</u>: (*already given*)

Refer to Section 3 for the details of the above files.

Your zip file should be sent before the due. Later than that, you will use one late day.

2 Theory Questions [30 pts]

Question 1: Combining Light Sources (5 points)

A Lambertian surface is illuminated simultaneously by two distant point sources in the directions $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ with intensities I_1 and I_2. Show that for all normals on the surface that are visible to both sources, illumination can be viewed as coming from a single "effective" direction $\boldsymbol{s}_3$. How is $\boldsymbol{s}_3$ related to $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$? And what is the intensity of the "effective" source?

Question 2: Solid Angle (5 points)

Calculate the solid angle subtended at the center of an icosahedron by one of its faces.

Question 3: Lambertian albedo (5 points)

For Lambertian surfaces, the BRDF is a constant function of the input and output directions. For such a material, we often describe the reflectance in terms of its *albedo*, which is given the symbol ρ. For a Lambertian surface, the BRDF and albedo are related by $f_r(\widehat{v}_i, \widehat{v}_o) = \rho/\phi$. Using conservation of energy, prove that $0 \leq \rho \leq 1$.

Question 4: Computing Normals of a Sphere (10 points)

Consider an image of a sphere formed under orthographic projection. Let the center of the sphere be the point (a, b) in image coordinates and let the radius of the sphere's image be R pixels. Derive a formula for the normal direction to the sphere's at any point (u, v) on the sphere's surface (specified in image coordinates). The formula should give the normal vector in a 3D coordinate system with origin at the sphere's center and with x and y axes oriented with the image axes.

Question 5: Computing Scene Normals (5 points)

Assume that we have 3 images of a Lambertian object taken using different light sources with same intensity(brightness), and we already know the directions of the light sources, $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3$, and the intensities of a image pixel, I_1, I_2, I_3. Describe the method to compute the normal direction and the albedo(unnormalized) of the pixel. What if the number of light source is larger than 3?

3 Programming [45 pts]

Part 1 Surface normal (10 points)

Obtain the file `bunny.mat` from the data folder of this assignment and load it into Matlab. There is a single variable in this file; the variable N is an $h \times w \times 3$ array of surface normals. N(i,j,1), N(i,j,2), and N(i,j,3) are the x, y, and z components of the surface normal at the ij^{th} surface point, as observed by an orthographic camera with view direction (0,0,1). (assume radient intensity is π)

- Use `quiver` to display the x-y components of the normal field and print the result.

- Compute and display the radiance emitted from each point assuming Lambertian reflectance and constant albedo (equal to one), with a distant point light source in direction $\widehat{s} = (0, 0, 1)$.

Part 2 Photometric Stereo

In this part of the assignment you will develop a vision system that recovers the orientation and reflectance of an object's surface. For this purpose you will use photometric stereo.

You will be given 3 images of an object taken using three different light sources. Your task is to compute the surface normals and albedo for this object. To do this, you will first need to find the directions and intensities of the 3 light sources. You will compute the light source directions and intensities from 4 images of an object of known geometry (a sphere) and use this information about the lighting to compute the shape and albedo of a new object of unknown geometry. The file `q2Script.m` reads in the data, makes calls all the functions you will write in this section and visualizes the output. You can modify the script as needed (adding parameters, adding extra arguments to function calls) but make sure it generates the same set of output images.

The data folder `data/q2` contains 7 greyscale images, `sphere0`, ..., `sphere3` and `object1`, ..., `object3`. Your program will be tested also with additional test images.

You can make the following assumptions about the images

- The surface of all objects (including the sphere) is Lambertian. This means there is only a diffuse peak in the reflectance map (no specular component).
- All the images are orthographic projections
- Image files with the same indices are taken using the same light source. For example, sphere1 and object1 are taken using light source number 1 only. The image sphere0 is taken using ambient illumination.
- The objects maintain the same position, orientation and scale through the different images – the only difference is the light source. For example, the sphere in sphere0 ... sphere3 has the same coordinates and the same radius.
- Each light sources maintains constant direction and intensity over all images.
- The light sources are not in singular configuration, i.e. the S-matrix that you will compute should not be singular.
- You may NOT assume that the light sources are of equal intensities. This means that you need to recover not only the directions of the light sources but also their intensities.

Finding Circles (5 points)

Write a function to find the centroid and radius of spheres in the included sphere images

```
function [cx cy r] = findCircle(img, threshold)
```

The function will input a greyscale image `img` and a scalar threshold `threshold` for binarizing the image. The outputs are the image coordinates of the center of the sphere `cx` and `cy` and the radius of the sphere in pixels `r`.

Under orthographic projection, the sphere projects into a circle on the image plane. You need to threshold the greyscale image to obtain a binary one. Make sure you choose a good threshold, so that the circle in the resulting image looks clean. Find the centroid and radius of the circle in the resulting binary image.

Finding Light Sources (10 points)

Write a function to find the direction and intensity of the light source in an image of a sphere given the sphere's parameters under the assumption that there is a single point light source in the scene.

```
function [lv] = findLight(img, cx, cy, r)
```

The function will input a greyscale image`img` along with the center coordinates of the sphere in that image (`cx` and `cy`) along with the sphere's radius `r`. The function will output `lv` a vector of length 3 pointing in the direction of the light source.

Find the normal direction corresponding to the brightest pixel in the image. Assume that the direction of this normal is the same as the direction of the light source in the image. The intensity of the light source is proportional to the magnitude (brightness) of the brightest pixel on the sphere, scale the direction vector `lv` so that its length equals this value.

Computing Normals and Albedos (10 points)

Write a function that inputs three images of an object along with the directions of the light sources in the three images and outputs the surface normal directions at a regularly sampled grid across the image.

```
function [normals albedo] = computeNormals(img1, img2, img3, lv1,
lv2, lv3, threshold)
```

The function will input 3 greyscale images `img1`, `img2` and `img3` and the light sources for each of those images `lv1`, `lv2` and `lv3` respectively. `threshold` decides which pixels to ignore when calculating normals. If a pixel isn't illuminated by all three light sources, then the normal can not be computed. If the minimum value of a pixel is below the threshold then assume it falls into the shadows in one of the images and do not compute a normal. `normals` is a $3\times$`N` matrix which stores the normal directions at each pixel in the image. `albedo` is an image of the scene where each pixel represents the (unnormalized) albedo of a scene point, not the apparent brightness.

Refer to the class notes on how to compute normals and albedos.

Part 3 Separating Global and Direct Illumination (10 points)

In this part of the assignment you will be separating direct and global components of a scene using high frequency illumination patterns. The technique you will use was first described in [1]. The task is to take a series of images of a scene illuminated by high frequency patterns and generate two output images, one that shows the light received by the camera that bounced off the scene and returned straight to the camera (the direct component) and another image containing the rest of the light that underwent multiple reflections or was scattered in the scene before returning to the camera (the global component).

Data for this task is in the `data/q3` folder. There are two scenes `scene1` and `scene2` and 25 images per scene. Each image shows an image of the scene illuminated with a high frequency checkerboard pattern. The checkerboard pattern shifts along the x and y axes.

Write a function

```
[globalImg directImg] = separateGlobalDirect(dirname)
```

That input `dirname` is a path to the set of images to run your separation algorithm over (you may assume the images will always be `.png` files). The outputs `globalImg` and `directImg` should be images of the same size as the input images containing the global and direct components of the illumination respectively.

Let `maxImg` be the image whose pixel value at `(i,j)` is the maximum value of the `(i,j)` pixel in all the input images. Similarly, the `minImg` is the image formed by taking the pixelwise minimum over all the input images. The global illumination image is just two times `minImg` and the direct illumination image is `maxImg - minImg`.

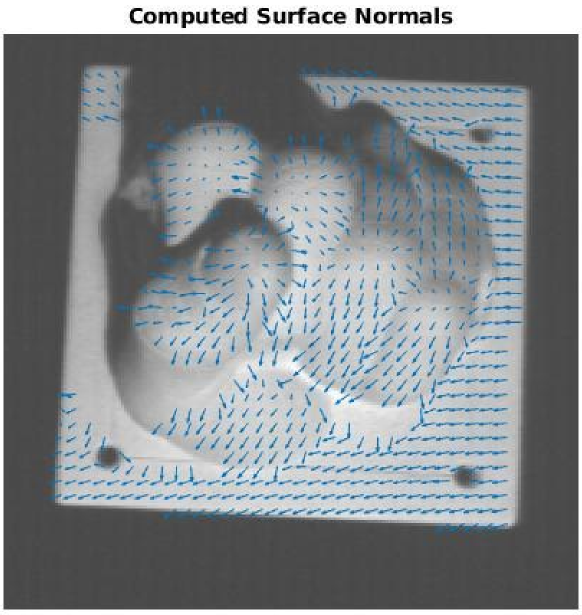
Write a script named `q3Script.m` to run your function on the two scenes provided in the `data/q3` folder. Describe the results and any interesting things you noticed in your experiments writeup.

References

[1] S.K. Nayar, G. Krishnan, M. D. Grossberg, and R. Raskar. Fast Separation of Direct and Global Components of a Scene using High Frequency Illumination. *ACM Trans. on Graphics (also Proc. of ACM SIGGRAPH)*, Jul 2006.

Revision History

Revision	**Date**	**Author(s)**	**Description**
1.0	2019/03/28	TA	HW2 out, Add a reference result image of computed normal map. Note that the results may vary depending on the threshold value.
1.0	2019/03/28	TA	unnormalized albedo $\rho' = \rho kc/\pi$


Computed Surface Normals