Project 5: Photometric Stereo

Siddharth Patel

I. P5.1 Lighting a Scene

In this section, we are computing the scene lighting of a Lambertian object from its normals and its albedo considering that we know where the light is coming from. Basically, for every point on the surface x, we will be implementing the brightness formula.

$$ObservedBrightness(x) = Albedo(x)cos(\theta_x) = Albedo(x)[l*n(x)] \quad (1)$$

The first step in implementing light_scene function is to create an array the size of albedos and set each element in the array to 0. We will name this variable lit. The second step is to loop through each elements of the albedos parameter and finally, save the above calculated equation's output to each index of lit variable and return it.

Below, I have attached the 4 different lightings of the bunny scene generated by mainpulating the x and y values using the interactive widget slider.

As you can see, the light position changes in the four different images and the difference can be observed based on how much the bunny is lit. When the lighting is [0, 0], the light is directly hitting the bunny, making it completely visible.

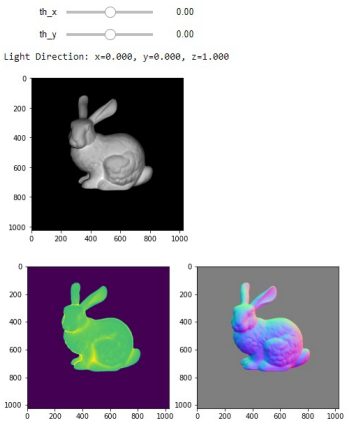

Fig. 1. Caption

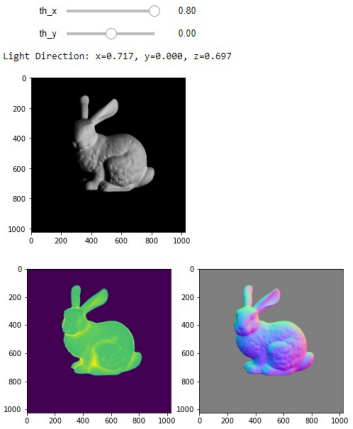

Fig. 2. Caption

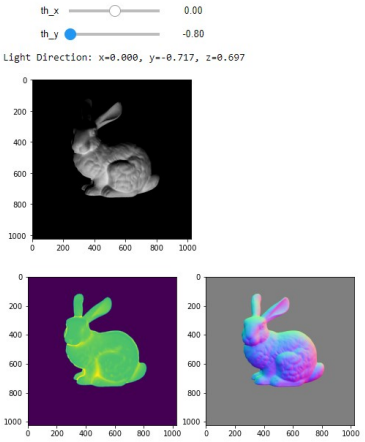

Fig. 3. Caption

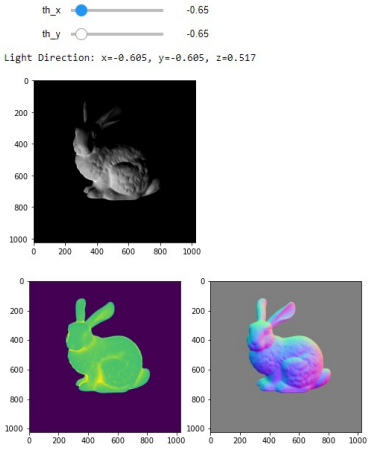

Fig. 4. Caption

II. P5.2.1 Using Provided Data

Now that we know how to light up a scene, we will generate images using 3-element vectors which will represent the coordinates of the direction of light source from the object. We will use these coordinates to generate a fully lit image.

As desrcibed in the L18 slides, I followed these steps to compute the photometric stereo:

1) Estimate light source directions
2) Compute surface normals
3) Compute albedo values
4) estimate depth from surface normals
5) Relight the object (with original texture and uniform albedo)

Below, you can visualize the plots of computed albedos followed by normals. Finally, I have integrated the reconstruction depth image difference via integrateFrankot function provided for each buddah and scholar datasets.

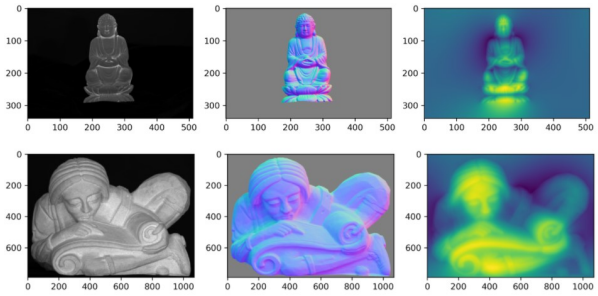

Fig. 5. Caption

III. P5.2.2 Imperfections in Photometric Stereo

While we are doing the photometric computation, it is important to keep in mind that we are performing some significant assumptions about the structure of the scene and the material properties of the object. Again, we will be reconstructing the image and the main goal in this section will be to comment on the differences between the reconstructed vs. original image.

The steps that I have taken to compute code for this section are as follows:

1) I am fetching the scholar data set that I filtered and stored in the previous part of this assignment.
2) Using the light_scene function from 5.1, I am relighting the scholar object to approximately recreate the original image, image 11.png. For this purpose, I have used light_position[10] from the scholar data as instructed.
3) Finally, I am plotting the original image, my reconstructed image, and the depth difference between the two images.

A. Question: Explain (in 2–3 sentences) why there is a significant difference between the two images to the lower-left of the scholar's face.

The main reason for this is because the lower-left corner of the scholar's face is extremely dark which makes it hard to detect. As I mentioned above, we are performing significant assumptions about the structure of the scene and its material properties one of which is to detect well-lit images. Clearly, the lower-left side is barely visible to me in the original picture. Therefore, it would be extremely difficult for the re-lighting algorithm to detect such lighting. Notice that we are also using one particular image and one image position from the scholar data. Another reason for this is also because we have limited amount of data. We do not possess enough data to determine whether the lower-left image is part of the scholar or not.

B. Question: Provide an explanation (in 2–3 sentences) why the body of the scholar is (on average) darker in the original image than in the reconstruction. (There are a few potential explanations to this question.)

As I said before, we are performing significant assumptions about the structure of the scene and its material properties and one of those assumptions is the strength and position of our light source. We may think that the light is falling from the "sky" like sun, but the original image's light might be coming from a whole different angle. As a result, our image is slightly brighter than the original image.

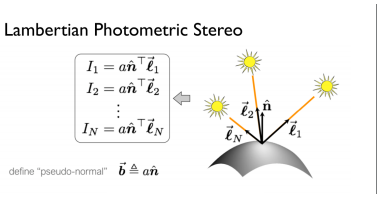

Fig. 6. Caption

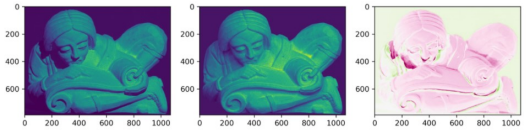

Fig. 7. Caption

IV. P5.2.3 Lambertian Objects in Blender

As requested, I have provided the 4 images generated from blender below. The steps to compute these lambertian objects that I generated in blender are the same as section 5.2.1. You gather the data from these images, load the light positions from the light_positions.txt file, normalize those positions, and compute for albedos and normals by calling the photometric_stereo function which takes the image data, light positions, and a threshold value which in this case is none or 0. After our computation, we plot the albedos, normals, and depth difference as you can see in the below image.

V. P5.2.4 Non-Lambertian Objects in Blender

The process for computing non-lambertian objects through the images (see below) generated from blender is no different from the above computation.

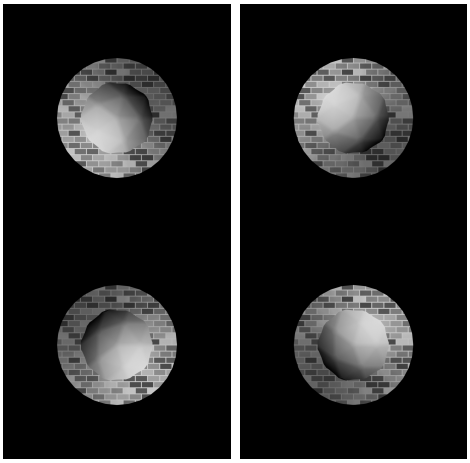

Fig. 8. Caption

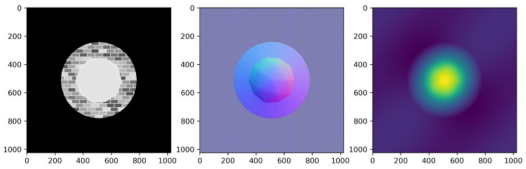

Fig. 9. Caption

A. Question: Explain (2-3 sentences) why the albedo and normals change. Is there a linear solution to finding the correct normals?

The albedos and normals change because the object is shinny. You can think of albedos as paint. If the surface is white colored, your albedos will be high, but if your surface is more towards black or dark color, your albedos will have low. Therefore, albedos and normals varies from point to point. If you have a surface with same texture and color throughout then we can have constant normals across the field. Therefore, there is a possibility of linear solution.

B. Question: Find a research paper (I recommend using Google Scholar) that attempts "non-lambertian Photometric Stereo": trying to recover material properties for a non-lamberitan object, like the one shown above. Include a citation of the paper and (in 3–5 sentences) explain how it works and how the approach differs from the lambertian photometric stereo we studied in class.

This paper addresses the problem of photometric stereo, in both calibrated and uncalibrated scenarios, for non-Lambertian surfaces based on deep learning. We first introduce a fully convolutional deep network for calibrated photometric stereo, which we call PS-FCN. Unlike traditional approaches that adopt simplified reflectance models to make the problem tractable, our method directly learns the mapping from reflectance observations to surface normal, and is able to handle surfaces with general and unknown isotropic reflectance. At test time, PS-FCN takes an arbitrary number of images and their associated light directions as input and predicts a surface normal map of the scene in a fast feed-forward pass. To deal with the uncalibrated scenario where light directions are unknown, we introduce a new convolutional network, named LCNet, to estimate light directions from input images. The estimated light directions and the input images are then fed to PS-FCN to determine the surface normals. Our method does not require a pre-defined set of light directions and can handle multiple images in an order-agnostic manner. Thorough evaluation of our approach on both synthetic and real datasets shows that it outperforms state-of-the-art methods in both calibrated and uncalibrated scenarios.

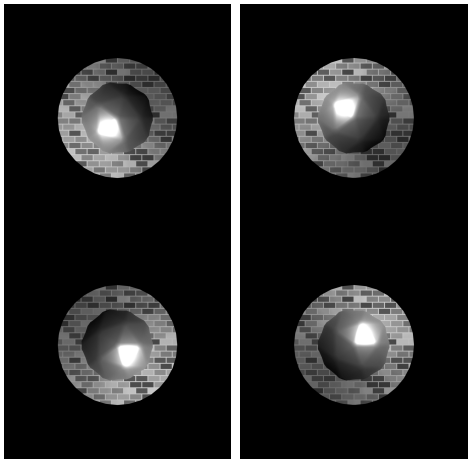

Fig. 10. Caption

The approach taken in the above explanation takes the deep learning route. The advantage with this is as we do more tests, our algorithm will improve to be able to recover material properties in non-lambertian objects. But the algorithm we are using is constant and will not be able to improve over certain amount of trials essentially never able to recover material properties.

Citation: G. Chen, K. Han, B. Shi, Y. Matsushita and K. -Y. K. Wong, "Deep Photometric Stereo for Non-Lambertian Surfaces," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2020.3005397.

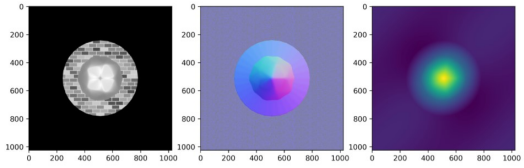

Fig. 11. Caption

VI. P5.3.1 Structure from Motion Implementation

Given the affine summary image to implement Affine SFM, we will first find feature matches using OpenCV which is already implemented for us. Given m images and n features, for each image i, we will center the feature coordinates. The final goal of the implementation of our algorithm is to elimiate affine ambiguity.

```
M (before correction):
 [[ 31.76908744   9.81257784   0.57296022]
 [ 10.93594577 -27.72025187   1.5225108 ]
 [ 34.572499     6.11387381 -10.36066116]
 [  7.31406288 -28.48146533   3.01590273]
 [ 31.83508604   9.75726727   0.52710383]
 [ 12.69445236 -27.96025204  -7.97658129]
 [ 28.25157989  12.88835753  10.64002626]
 [ 12.5407986  -26.42941398   6.79090004]]
A (after correction):
 [[ 0.94260326  0.33219952  0.00790791]
 [ 0.33250862 -0.94320166  0.02101346]
 [ 0.96427876  0.22318072 -0.14299623]
 [ 0.23377894 -0.97129645  0.04162502]
 [ 0.94429913  0.33039144  0.007275  ]
 [ 0.33125749 -0.93699084 -0.11009153]
 [ 0.8949612   0.4213843   0.14685198]
 [ 0.40948901 -0.90734985  0.09372694]]
```

Fig. 12. Caption

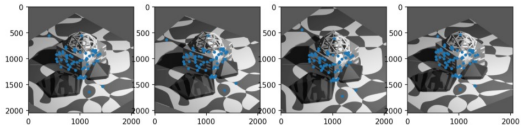

Fig. 13. Caption

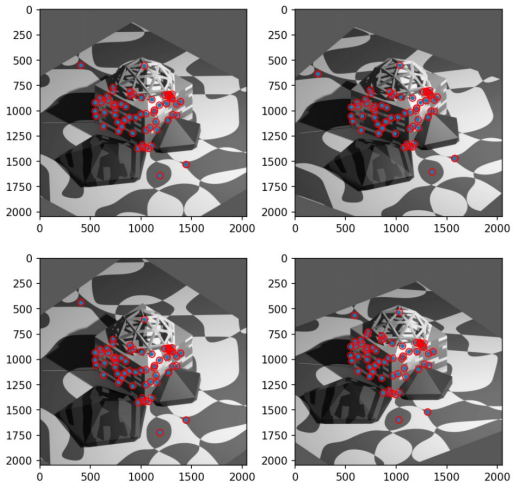

Fig. 14. Caption

VII. P5.3.2 Followup Conceptual Questions

A. Question: What is the relationship between the Affine matrix you have computed and the 3x4 camera projection matrix? Pick one of the input images above. Using the affine transformation matrix you computed for that image, what is the 3x4 camera projection matrix for this image? Include it in your writeup.

B. Question: I have provided you with the function correct_affine_ambiguity so that the Affine projective matrices produced by our algorithm satisfy certain properties. What are these properties? You may refer to the code itself or the course notes in your answer. It may help you to print the matrices before and after the correction is applied.

We are basically implementing the Cholesky decomposition which given lower triangular matrix with real and positive diagonal entries and its conjugate transpose, which is useful for efficient numerical solutions. When it is applicable, the Cholesky decomposition is roughly twice as efficient as the other decomposition algorithms for solving systems of linear equations.

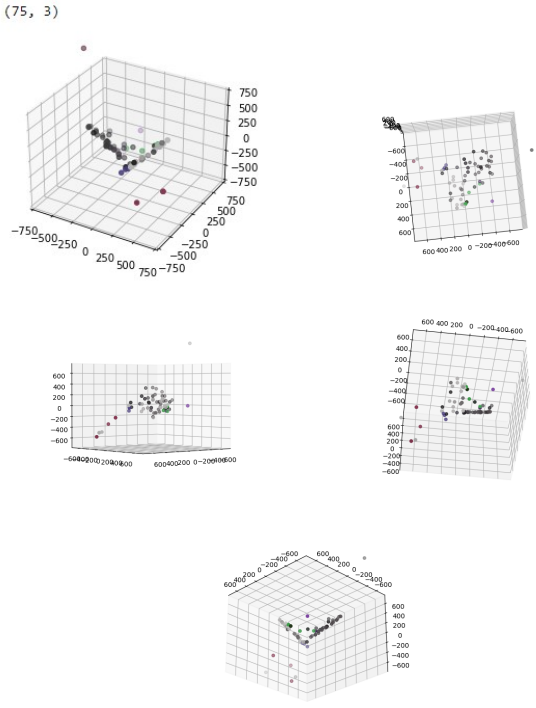

Fig. 15. Caption