Introduction

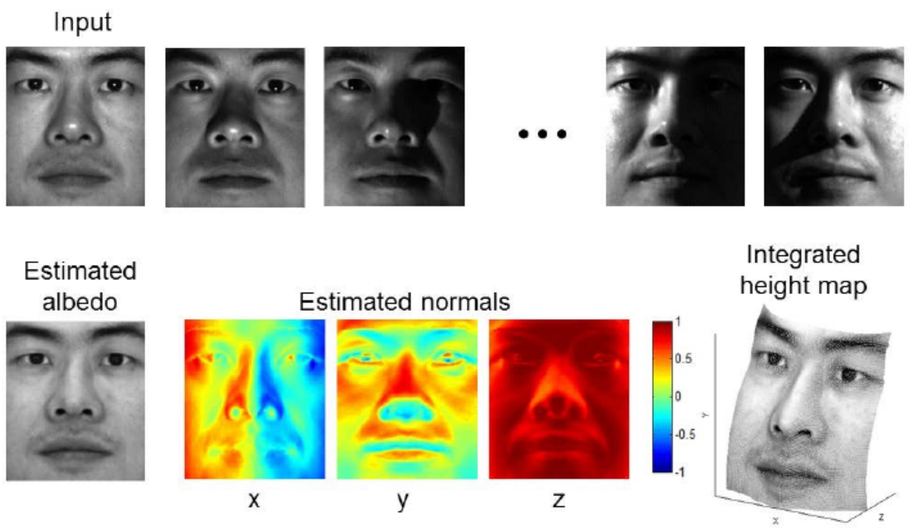
This assignment primarily involves calculating the surface normals and estimating the depth from a set of images taken under different lighting conditions. Please download the homework3.zip file from the myCourses as it contains the images and code needed for the assignment.

Requirements

You should perform this assignment in Python and feel free to use routines from OpenCV or other similar libraries. The assignment is due on **Wednesday November 14th by 11:59pm**. You are encouraged to start early and are also welcome to ask questions about and have discussions about the homework on myCourses. But please do not post your solutions or any closely related material. If there are parts of the assignment that are not clear to you, or if you come across an error or bug please don't hesitate to contact either the TAs or the Instructor.

You are allowed to collaborate with other students as far as discussing ideas and possible solutions. However you are required to code the solution yourself. Copying others' code and changing all the variable names is not permitted! You are not allowed to use solutions from similar assignments in courses from other institutions, or those found elsewhere on the web. If you access such solutions YOU MUST refer to them in your submission write-up. Your solutions should be submitted using myCourses.

Your submitted zipped file for this assignment should be named **LastnameFirstname_hw3.zip**. Failure to follow this naming convention will result in delays in grading your work. Your zipped file should contain: (i) a PDF file named LastnameFirstname_hw3.pdf with your report, showing output images from the assignment and explanatory text, where appropriate; and (ii) the source code used to generate the solutions (with code comments).



Problem 1. Estimating Shape from Shading[1]

The goal of this assignment is to implement shape from shading as described at the end of Lecture 8 (more details in Section 2.2.4 of Forsyth & Ponce 2nd edition).

Start from the `code` directory and download the data in the `croppedyale` directory, which consists of 64 images each of four subjects from the Yale Face database. The light source directions are encoded in the file names. The `code` directory also consists of several `.py` files. Your task will be to add some code to the top-level script, `run_me.py`, and to fill in the code in `load_images.py`, `photometric_stereo.py` and `get_surface.py`, as explained below.

For each subject (subdirectory in `croppedyale`), read in the images with light source info encoded in the file names:

1. Complete the function `load_images` whose signature is:
   `[ambient_image, imarray, light_dirs] = load_images(filepath, subject_name, 64)`.

   The function `load_images` should return an ambient image (i.e., image taken with all the light sources turned off), the 64 face images taken under different light conditions and the 64 light source directions.

   i. The file `[`*`subject_name`*`]_P00_Ambient.pgm` is the ambient image. You can open `.pgm` files using the helper file `read_pgm.py`.

   ii. The other `.pgm` image files should be read and stored in array `imarray`

   iii. Lastly, extract the light source direction information which is encoded in the file names:

   - Given an image file name say, `yaleB01_P00A-010E+00.pgm`, first obtain the azimuthal data as the four characters after the letter 'A', i.e. -010 which

[1]This was adapted from Svetlana Lebsnick at UIUC with permission

corresponds to -10 degrees. Similarly, the elevation is the three characters after the letter 'E', corresponding to 0 degrees. Another example: for file `yaleB07_P00A-035E+65.pgm`, the azimuthal is -35 degrees and the elevation is 65 degrees.
Extract into a 2 dimensional array `Angles` where the azimuthal is the first dimension and elevation the second.

- Convert `Angles` from spherical to Cartesian coordinates (given in file) using the function `sph2cart`, which is provided for the conversion.

2. Preprocess the data: subtract the ambient image from each image in the image source stack, set any negative values to zero, rescale the resulting intensities to between 0 and 1 (they are originally between 0 and 255).

3. Estimate the albedo and surface normals.
   For this, write in the code for `photometric_stereo`, a function which takes as input the image stack corresponding to the different light source directions and the matrix of the light source directions; it returns an albedo image and surface normal estimates.

   The latter should be stored in a three-dimensional matrix. That is, if your original image dimensions are `h × w`, the surface normal matrix should be `h × w × 3`, where the third dimension corresponds to the `x-`, `y-`, and `z-`components of the normals. To solve for the albedo and the normals, you will need to set up a linear system of equations as shown in the lecture slides. The signature for this should be:
   `[albedo_image, surface_normals] = photometric_stereo(imarray, light_dirs)`

4. Compute the surface height map by integration by summing up the discrete values of the partial derivatives over a patch. Your code implementing the integration should go in the `get_surface.py` file. To get the best results, you should compute integrals over multiple paths and average the results.

5. Display the resulting albedo, surface normals and height map images for the subject (Code for this is provided).

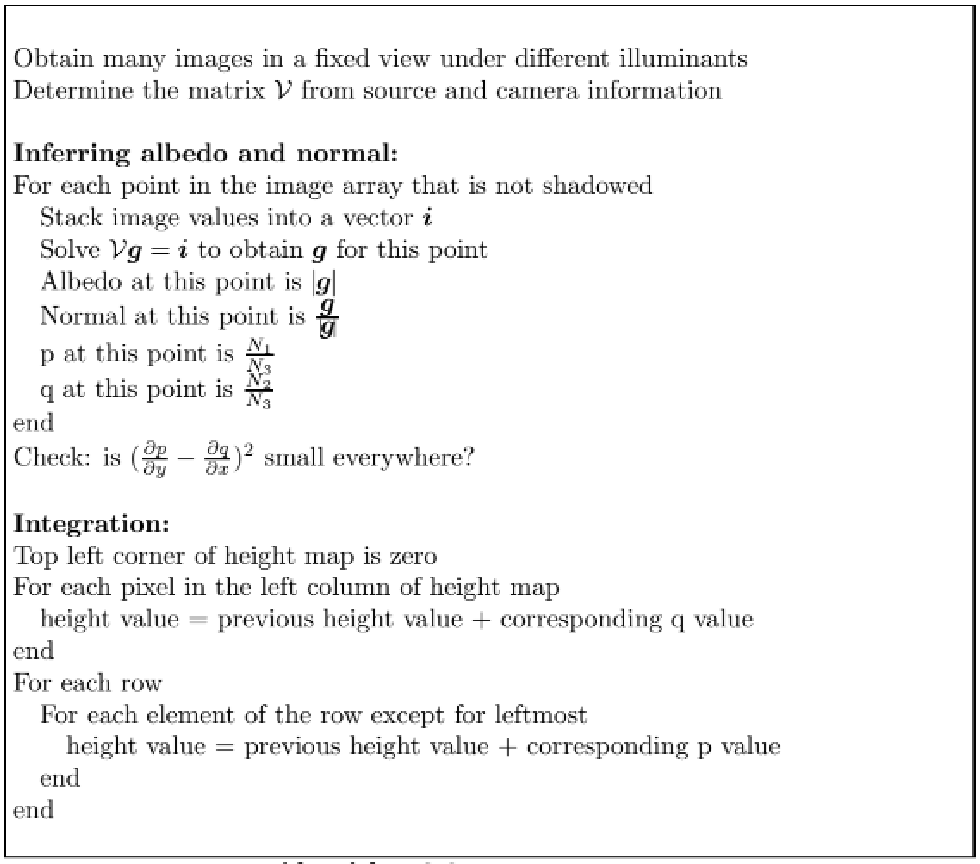
You should turn in both your code and a report discussing your solution and results. For full credit, your report should include a section for each of the following questions:

A. Briefly describe your implemented solution, focusing especially on the interesting parts of the implementation. What are some artifacts and/or limitations of your implementation, and what are possible reasons for them?

B. For every subject, display your estimated albedo maps and screenshots of height maps (feel free to include screenshots from multiple viewpoints).

C. Discuss how the Yale Face data violates some of the assumptions of the shape-from-shading method covered in the slides and discussed in the textbook. Feel free to include specific input images to illustrate your points.

Below is the pseudo-code for the assignment from page 82 of the F & P book.



```
Obtain many images in a fixed view under different illuminants
Determine the matrix $\mathcal{V}$ from source and camera information

**Inferring albedo and normal:**
For each point in the image array that is not shadowed
  Stack image values into a vector $\boldsymbol{i}$
  Solve $\mathcal{V}\boldsymbol{g} = \boldsymbol{i}$ to obtain $\boldsymbol{g}$ for this point
  Albedo at this point is $|\boldsymbol{g}|$
  Normal at this point is $\frac{\boldsymbol{g}}{|\boldsymbol{g}|}$
  p at this point is $\frac{N_1}{N_3}$
  q at this point is $\frac{N_2}{N_3}$
end
Check: is $(\frac{\partial p}{\partial y} - \frac{\partial q}{\partial x})^2$ small everywhere?

**Integration:**
Top left corner of height map is zero
For each pixel in the left column of height map
  height value = previous height value + corresponding q value
end
For each row
  For each element of the row except for leftmost
    height value = previous height value + corresponding p value
  end
end
```

Algorithm 2.2: Photometric Stereo.