The goal of this assignment is to implement an algorithm that reconstructs a surface using the concept of photometric stereo. You can assume a Lambertian reflectance function, but the albedo is unknown and nonconstant in the images. Your program will take in multiple images as input along with the light source direction for each image. All data sets for this assignment is provided in http://www.cs.ucsd.edu/classes/sp05/cse152/hw3-data.zip.

- **Test Case 1: Synthetic Data**
  The following four test images will serve as input. The first row shows the image, the second row is the file name, and the third row is the light source direction used to generate the image. You should also divide the image intensities by 255 to scale them from [0,1].

  ![Image 1](im1.png) ![Image 2](im2.png) ![Image 3](im3.png) ![Image 4](im4.png)

  \[
  \begin{bmatrix}
  0 \\
  0 \\
  -1
  \end{bmatrix},
  \begin{bmatrix}
  0 \\
  0.2 \\
  -1
  \end{bmatrix},
  \begin{bmatrix}
  0 \\
  -0.2 \\
  -1
  \end{bmatrix},
  \begin{bmatrix}
  0.2 \\
  0 \\
  -1
  \end{bmatrix}
  \]

- Your program should have two parts:
  - Read in the images, and estimate the surface normals and albedo map.
  - Reconstruct the depth map from the normals.

For the second part, you should implement two methods to integrate the partial derivatives:
  - The naive direct integration approach (the first approach in the lecture notes)
  - The iterative integration approach (the method derived from calculus of variations).

You have also been provided with some detailed notes on the iterative integration technique (http://www.cs.ucsd.edu/classes/sp05/cse152/iterative.pdf). To simplify integration, you may assume that the object that we are imaging covers the entire image; in
which case $m_{i,j} = 1$ for all $i,j$. If you would like to obtain more accurate results, you can create a mask (a binary image) and implement the masked iterative integration technique. Since we have given you four images, try your algorithm with these three sets of images:

- im1.png, im2.png, im4.png (Easiest)
- im1.png, im2.png, im3.png (problematic)
- All four images (most accurate)

You will have to implement linear least squares in order to use all four images.

**What to include in your report**

You should write up a report containing your BEST result of the:

- Estimated albedo map
- Estimated surface normals by either showing
  * A needle map
  * Three images showing three components of surface normal
- Wireframe of a depth map (you can use surf() in matlab).

**TEST CASE 2: REAL DATA**

For this part we will use a real data set, given by the following table:

<table>
<thead>
<tr>
<th>real1.bmp</th>
<th>real2.bmp</th>
<th>real3.bmp</th>
<th>real4.bmp</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[-0.38359]$</td>
<td>$[-0.372825]$</td>
<td>$[0.250814]$</td>
<td>$[0.203844]$</td>
</tr>
<tr>
<td>$-0.236647$</td>
<td>$0.303914$</td>
<td>$0.34752$</td>
<td>$-0.096308$</td>
</tr>
<tr>
<td>$0.892668$</td>
<td>$0.87672$</td>
<td>$0.903505$</td>
<td>$0.974255$</td>
</tr>
</tbody>
</table>

**What to include in your report**

Run your algorithm on this dataset and report your BEST result with the:

- Estimated albedo map
- Estimated surface normals by either showing
  * A needle map
  * Three images showing three components of surface normal
– Wireframe of a depth map (you can use surf() in matlab).

**Test Case 3: Face Data (Extra Credit)**

For this part we will use data the Yale Face Database B, which you can access through ftp://plucky.cs.yale.edu/CVC/pub/images/yalefacesB/. You should use yaleB01_P00.tar.gz, which is under the TarSets directory, but you can also use other datasets if you wish. Each dataset contains 64 images per person, but you’ll only want to use a modest number of images (at least three and perhaps six images). From the input images, crop out a rectangular region of the face, and just work with that sub-image. To read in individual images you can use getpgmraw.m, and to read in all images you can use LoadFaceImages.m. Alternatively you can use the given data set face1.bmp, face2.bmp, face3.bmp, face4.bmp with light source directions in the following table (given as azimuth \((a)\) and elevation \((e)\)):

<table>
<thead>
<tr>
<th></th>
<th>face1.bmp</th>
<th>face2.bmp</th>
<th>face3.bmp</th>
<th>face4.bmp</th>
</tr>
</thead>
<tbody>
<tr>
<td>([a = -5, e = -10])</td>
<td>([a = -5, e = 10])</td>
<td>([a = 5, e = -10])</td>
<td>([a = 5, e = 10])</td>
<td></td>
</tr>
</tbody>
</table>

As a preprocessing step you should subtract the ambient image from the other images and make sure no pixel is smaller than zero. In addition, you may reduce noise by convolving the image with a small Gaussian kernel (blurring), but this might not be necessary.

**What to Include in Your Report.** Run your algorithm on a face dataset and report your BEST result with the:

– The cropped input images for your algorithm
– The albedo map
– Normal field
– Wireframe of the depth map
– Commentary about any issues that arose, ways to improve your method, etc.

**What to Hand In:** You should hand in

– A report with the items described above, any other figures, and comments/issues that you find relevant.
– Hardcopy of your code.
- An email entitled CSE 152 Assignment 3 to wychang@cs.ucsd.edu with tar or zip file containing the code and report.

- For reference, here is a map of dome flashes used to generate the face datasets.