CS 6968, 3D Computer Vision
Fall 2009, Prof. Guido Gerig

Assignment 2: Photometric Stereo & Epiolar Geometry

Out: Tuesday Sept-29-2009
Due: Tuesday Oct-20-2009 (theoretical and practical parts)
TA: Evrard Ohou (eohou@cs.utah.edu)
Office hours: Tue/Thu 1:45pm to 3:30pm

Required Readings: Computer Vision, Forsyth & Ponce, Chapters 4/5 and 10/11
                      In particular slides to these chapters provided on WebCT

Grading

Theoretical problems: These serve as your own study of the material using the textbook and all materials provided on WebCT. The theoretical questions will not be graded in details but just listed as completed. Detailed solutions will be provided.

Practical problem: Grading will primarily concern your solution strategy and solution of the camera calibration, and the report that describes your project, your development of the methodology, results, and critical assessments of the results.

1 I. Theoretical Problems

1.1 BDRF

a) Why are specularities bringer than diffuse reflection? (Question 4.9 Forsyth&Ponce, explain in a few sentences)

b) The eye responds to radiance. Explain why Lambertian surfaces are often referred to as having a brightness independent of viewing angle (Question 4.11, explain in a few sentences).

1.2 Reflectance Map

Some surface material cannot be simplified to have Lambertian reflectance properties. The "maria" of the moon, e.g., the dark formation as observed from the earth, is a typical example of a non-Lambertian surface. In this case, we observe that the surface emits radially equally in all directions, or in other words, brightness seems to increase with increasing angle between viewer and surface (hint: remember derivation of the equation for Lambertian surfaces where only the foreshortening of the incoming angle is considered).

a) Derive the equation for the reflectance map R(p,q) in this special case.

b) What are the isodensity curves of constant brightness in R(p,q)? (Hint: remember the discussion of circles and conic sections for the Lambertian case)

c) What are the contours of constant brightness of an image of a sphere?

d) Sketch a solution to estimate (p,q) from multiple images with different light source directions. How many would you need?

1.3   Stereo System

a) Starting from the disparity equation $d = -Bf/Z$, build the relationship between $\triangle Z$ and $\triangle d$. Hint: Build the derivative $\frac{\partial d}{\partial Z}$ and discuss changes in disparity versus changes in depth as a function of depth.

b) Estimate the accuracy of the simple stereo system with two rectified images assuming that the only source of noise is the localization of corresponding points in the two images (the disparity d). (Hint: Take the partial derivatives of Z with respect to d,B,f (disparity, baseline, focal length)). Then use the general formula for error estimation of multiple independent variables as $\partial Z = \sqrt{(\frac{\partial Z}{\delta d}\partial d)^2 + (\frac{\partial Z}{\partial B}\delta B)^2 + (\frac{\partial Z}{\partial f}\delta f)^2}$. Discuss the dependence of the error in depth estimation as a function of the baseline width and the focal length.

c) Using the previous solution, estimate the accuracy with which features should be localized in the two images in order to reconstruct depth with a relative error $|\frac{\delta Z}{Z}|$ smaller than 1%.

2   II. Practical Problems

2.1   Problem: Photometric Stereo / Shape from Shading

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 read multiple images as input along with the light source direction for each image. All data sets for this assignment are provided are provided on WebCt.
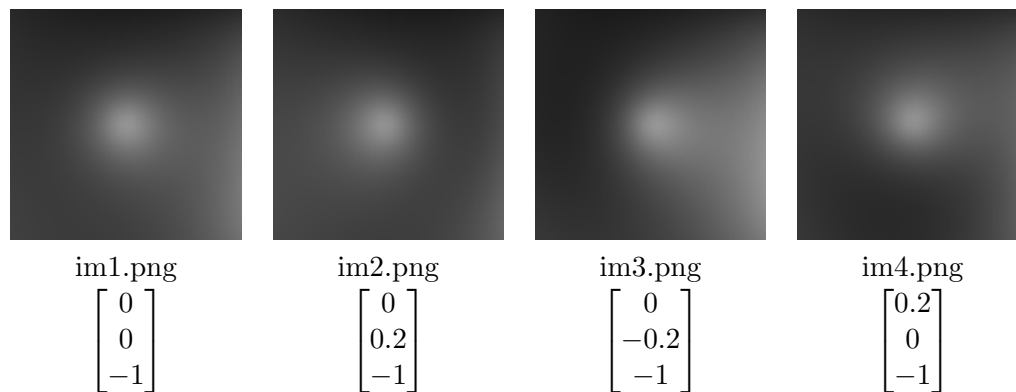

im1.png $\begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$ im2.png $\begin{bmatrix} 0 \\ 0.2 \\ -1 \end{bmatrix}$ im3.png $\begin{bmatrix} 0 \\ -0.2 \\ -1 \end{bmatrix}$ im4.png $\begin{bmatrix} 0.2 \\ 0 \\ -1 \end{bmatrix}$

Figure 1: Synthetic data: Images, file names, light source directions. You should also divide the image intensities by 255 to scale them to [0.0,1.0].

Your program should have two parts:

- Read in the images (see Fig. 1), and estimate the surface normals and albedo map.

- Reconstruct the depth map from the normals via integration.

Since we have given you four images (size 100x100), you have the possibility for an exact solution (use of 3 images only) or an overconstrained solution (all four images). The images have intensities in the range 0 to 255, so after reading the images divide them by 255 to get floating point values in the range of $[0 \cdots 1]$.

(Hint: Students reported that using images 1,2, and 4 works ok for the 3 image solution, but that the combination 1,2,3 seems problematic.)

You will have to implement linear least squares in order to use *all four images*, this is not necessary for 3 images. Hint: Follow the F&P book pages 82 and 83 which explains the calculation of the normal and the albedo at each pixel. Matlab strongly supports the solution of equation systems, see help in "Solving Linear System of Equations" in Matlab. Following the book pages 82/83, you will have to solve a system like: $(\mathcal{TV})^t\mathcal{T}i = g(x,y)$. A Matlab notion would be something like : $g = \mathcal{TV} \setminus \mathcal{T}i;$, or $g = pinv(\mathcal{TV}) * \mathcal{T}i;$.

For the second part to get a surface from normals, you should implement the naive direct integration approach to integrate the partial derivaties and construct the depth map (see algorithm 5.1 on page 85 textbook). To simplify integration, you may assume that the object that we are imaging covers the entire image.

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

- Estimated albedo map

- Estimated surface normals by either showing:

  - A needle map (e.g. 2D vectors on the 100x100 grid), or

  - three images showing the three components of surface normals (normalized normals),

- The reconstructed surface as:

  - Graylevel depth image where depth is encoded a hight z, and

  - wireframe of a depth map (you can use surf() in matlab) to display Z(X,Y) similarly to figure 5.13 on page 86, and

  - shaded image generated from a user-defined light source position (given a light source direction and surface normals, you can easily calculate shaded images for arbitrary light source positions).

- Commentary about any issues that arose, ways to improve your method, etc.

2.2 Shape from Shading from real Images

You can choose between the sphere image or the more challenging face image.

Sphere Images:

Again, divide the images by 255 to scale them back to a floating point range of $[0 \cdots 1]$. (see Fig. 2).

Face Images:

Please remember the transformation of azimuth θ and elevation ϕ to cartesian coordinates as: $x = \sin\theta\cos\phi, y = \sin\theta\sin\phi, z = \cos\theta$. Check if the resulting normals for light sources are

real1.bmp real2.bmp real3.bmp real4.bmp

$$\begin{bmatrix} -0.38359 \\ -0.236647 \\ 0.892668 \end{bmatrix} \quad \begin{bmatrix} -0.372825 \\ 0.303914 \\ 0.87672 \end{bmatrix} \quad \begin{bmatrix} 0.250814 \\ 0.34752 \\ 0.903505 \end{bmatrix} \quad \begin{bmatrix} 0.203844 \\ -0.096308 \\ 0.974255 \end{bmatrix}$$

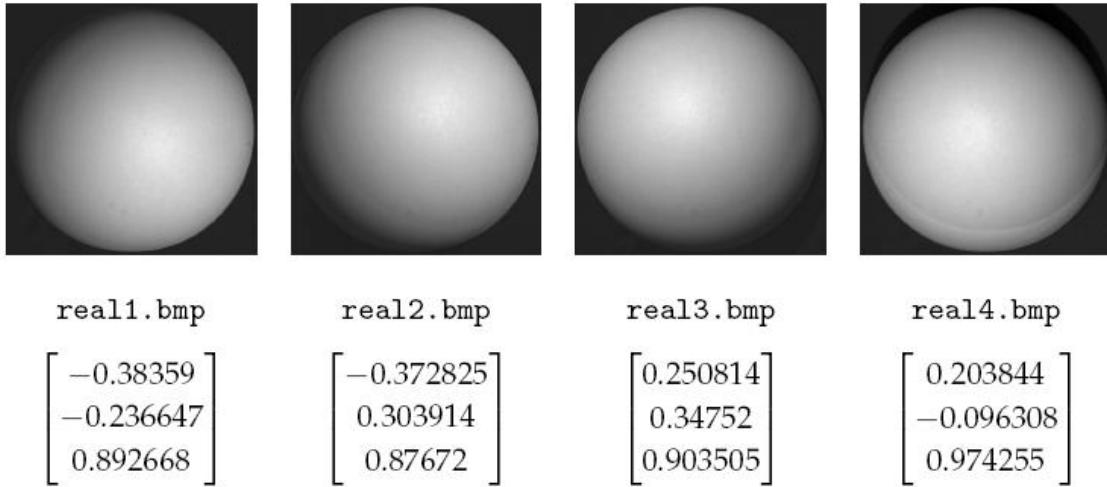Figure 2: Real data: Images, file names, light source directions. You should also divide the image intensities by 255 to scale them from [0,1].

face1.png face2.png face3.png face4.png face0.png

$$\begin{bmatrix} \theta = -5 \\ \phi = -10 \end{bmatrix} \quad \begin{bmatrix} \theta = -5 \\ \phi = 10 \end{bmatrix} \quad \begin{bmatrix} \theta = 5 \\ \phi = -10 \end{bmatrix} \quad \begin{bmatrix} a = 5 \\ e = 10 \end{bmatrix} \quad \text{ambient}$$
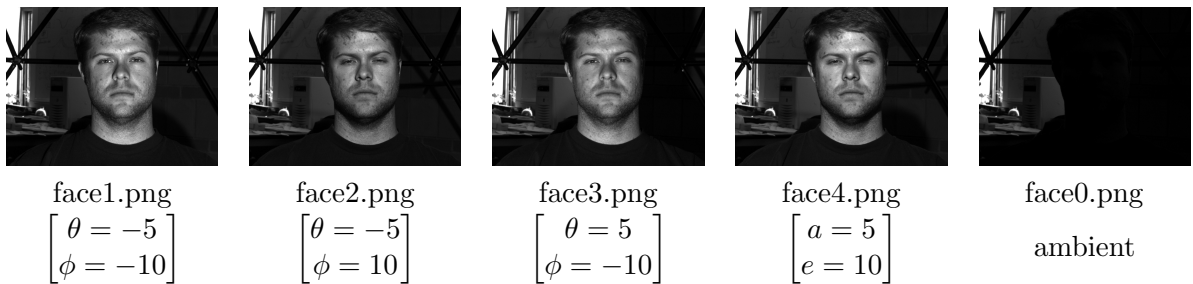
Figure 3: Real data: Images, file names, light source directions as azimuth and elevation. You should also divide the image intensities by 255 to scale them from [0,1].

feasible given the images. As a preprocessing step you should subtract the ambient image from the other images (see Fig. 3) and make sure no pixel is smaller than zero. This step makes the images closer to the Lambertian reflection model used by our reflectance map model. Again, divide the images by 255 to scale them back to a floating point range of $[0.0 \cdots 1.0]$. From the input images, crop out a rectangular region of the face, and just work with that sub-image. To read in individual images in pgm format you can use getpgmraw.m. Cropping can be done by using Matlab's *imcrop* function.

Integration:

Integration from normals to get the depth map is more difficult for noisy real images since integrability is violated and normals are only estimated approximately. You might first use a single integration step. Alternatively, you might apply integration with different start-points and then average the resulting depth map. The literature knows more sophisticated reconstructions using calculus of variation, but you are not required to do so.

Again, write a report as explained above.