NYU CS-GY 6643, Computer Vision
Spring 2016, Prof. Guido Gerig

Assignment 3: Photometric Stereo

Out:	Mon Mar-12-2016
Due:	Wed April-06-2016, midnight (theoretical and practical parts)
TA:	Padmashree Teeka
TA:	Rupanta Rwiteej Dutta
Office hours	TA: Tue 2-5pm MetroTech Center, 10th floor, cubicle space 10.098E
	Instructor Mo 2pm-5pm, 2 MetroTech Center, 10.094

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

Grading

Theoretical problems: These serve as your own study of the material using the textbook and all materials provided on WebCT. 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 (total 16pts)

1.1 Reflectance map (10pts)

1.1.1 Special case

(Book problem 2.1). We see a diffuse sphere centered at the origin, with radius one and albedo ρ, in an orthographic camea, looking down the z-axis. The sphere is illuminated by a distant point light source whose source direction is (0,0,1). There is no other illumination. Show that the shading field (reflectance map) in the camera is:

$$\rho\sqrt{1-x^2-y^2}.$$

1.1.2 General light source direction

Starting from the reflectance map equation for a general light source direction (see slides and book), show that reflectance map isophotes (curves of constant brighness in R(p,q)) represent conic sections, i.e. curves with squared and linear terms in p and q. You may choose an example for a light source direction and plot the result in (p,q)-space for several values of constant brightness.

1.1.3 Number of images for reconstruction of normals?

Following the discussions in the book chapter 2 and slides, explain:

Why is one image with on light source direction not enoug?

Getting images with more than one light source directions, justify how many images we would need as a minimum. You can use drawings or equations if you like.

1.2 Stereo System (6 pts)

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.

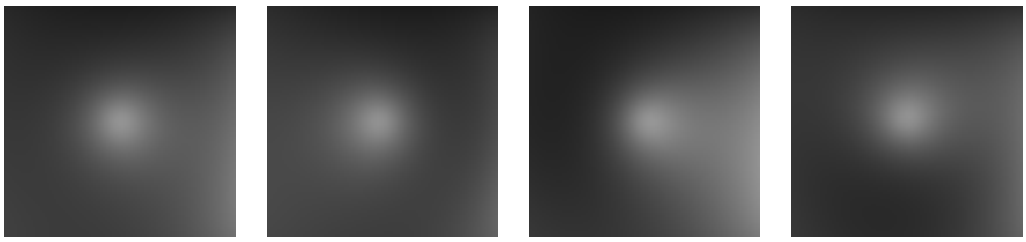
b) Estimate the accuracy of the simple stereo system with two rectified images assuming that the only source of noise is the localization of corresponding points in the two images (the disparity d). (Hint: Take the partial derivatives of Z with respect to d,B,f (disparity, baseline, focal length)). Then 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 (total 36 pts)

2.1 Problem: Photometric Stereo: Synthetic Images (22pts)

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 canvas and the class website.
Write a report on your solutions for the theoretical problems. This report can be handed in on paper during the class lecture on Monday February 13 or can be added to the electronic pdf/Word report of the theoretical part (submitted to the CADE handin system by the specified deadline).



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

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:

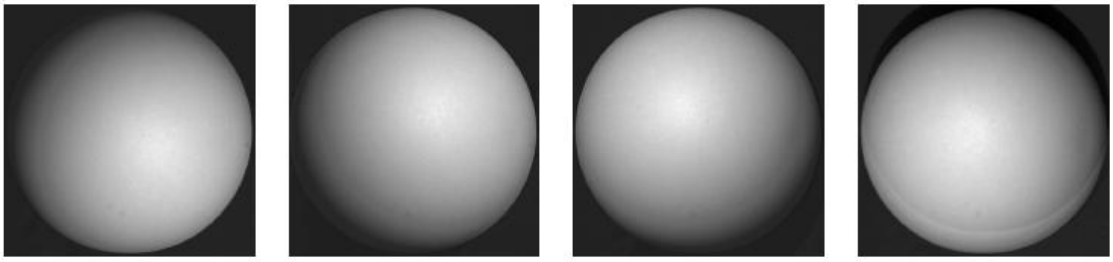
- Read in the images (see Fig. 1), and estimate the surface normals and albedo map.
- Reconstruct the depth map from the normals via integration.

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. For integration, you may consider starting in the middle of the image rather than a corner since background noise may created distortions of the reconstructed surface.

2.2 Shape from Shading from real Images (12pts)

You can choose between the sphere images or the dog images (courtesy to Christopher Bireley, class F2009) or do both once you have a running code.

Sphere Images:

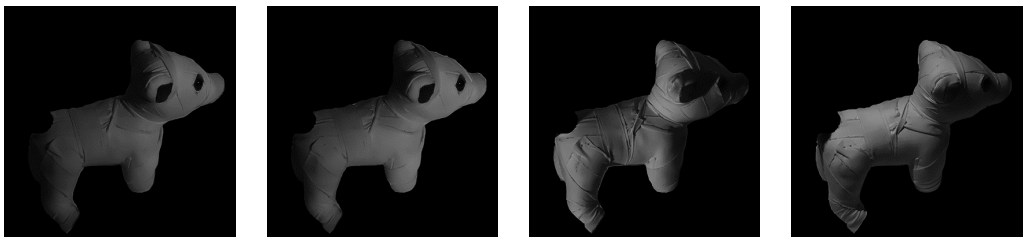


real1.bmp	real2.bmp	real3.bmp	real4.bmp
$\begin{bmatrix} -0.38359 \\ -0.236647 \\ 0.892668 \end{bmatrix}$	$\begin{bmatrix} -0.372825 \\ 0.303914 \\ 0.87672 \end{bmatrix}$	$\begin{bmatrix} 0.250814 \\ 0.34752 \\ 0.903505 \end{bmatrix}$	$\begin{bmatrix} 0.203844 \\ -0.096308 \\ 0.974255 \end{bmatrix}$

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].

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

Dog Images:



dog1.tif	dog2.tif	dog3.tif	dog4.tif
$\begin{bmatrix}16\\19\\30\end{bmatrix}$	$\begin{bmatrix}13\\16\\30\end{bmatrix}$	$\begin{bmatrix}-17\\10.5\\26.5\end{bmatrix}$	$\begin{bmatrix}-9\\25\\4\end{bmatrix}$

Figure 3: Dog image data: Images, file names, light source directions as non-normalized 3D vectors. You should also divide the image intensities by 255 to scale them from [0,1].

Images see Fig. 3. The light source vectors first need to be normalized before using them in your code. Again, divide the images by 255 to scale them back to a floating point range of $[0.0 \cdots 1.0]$. We have got very good results with images 2, 3 and 4, but you can use all 4 images by using SVD for solving for the overconstrained problem.

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 as hown in the slides and in our slides and Forsyth and Ponce book. Alternatively, you might apply integration with different start-points and then average the resulting depth map.

Bonus: Integration with the Chellappa method (bonus recorded separately)

The literature knows more sophisticated reconstructions using calculus of variation, and if you have additional capacity you may try the following:
Matlab code for computer vision, (`http://www.csse.uwa.edu.au/~pk/research/matlabfns/`), under "Surface Normals to Surfaces", surface integration can be done by "frankotchellappa.m" (`http://www.csse.uwa.edu.au/~pk/research/matlabfns/Shapelet/frankotchellappa.m`.

2.3 Report

You should write up a report including your approach and results:

- Estimated albedo map
- Estimated surface normals by either showing:
  - A needle (vector) map (e.g. 2D vectors on the 100x100 grid), or

  - Three images showing the three components of surface normal vector (normalized normals), or
  - Creating a new shaded image by choosing a new light source position vector and calculating the dot product of surface normals and light source vector (both normalized) at each pixel, resulting in an image showing a light source not used for reconstruction.

- The reconstructed surface (after integration) as:
  - Graylevel depth image where depth z is encoded as intensity, and
  - Wireframe of a depth map (you can use surf() or meshgrid() in matlab, see bottom) 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, critical assessment of results, etc.

2.4 Matlab mesh viewing

```
1. Use 'surf' for the surface rendering

[ X, Y ] = meshgrid( 1:C, 1:R );
figure;
surf(  X, Y, DepthMap, 'EdgeColor', 'none' );
camlight left;
lighting phong

- You may want to use or change the lighting options for the better visualization

2. Use 'mesh' for the mesh surface rendering

[ X, Y ] = meshgrid( 1:C, 1:R );
figure;
mesh( X, Y, DepthMap );

- R, C are the dimension of the image
```