Assignment 6 Solutions

CS283, Computer Vision

1. First, we introduce some notation. We assume that we have a surface patch $\mathrm{d}A$ which, from each direction $\hat{\omega}_i$, receives incoming radiance $L_i\left(p, \hat{\omega}_i\right)$. Using the notation of [1], we parameterize directions using the two angles θ_i, ϕ_i, which provide a full characterization. Also, we drop the p argument, as it can be inferred from context. Then, from the definition of radiance, we can express the amount of energy $\mathrm{d}A$ receives from the entire hemisphere during a time interval $\mathrm{d}t$ as

$$\mathrm{d}E_i = \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \,\mathrm{d}\Omega_i \,\mathrm{d}A \,\mathrm{d}t, \tag{1}$$

where Ω_i is the input hemisphere and the term $\cos\theta_i$ accounts for foreshortening. Similarly, for the energy leaving the surface patch at the same time interval we have

$$\mathrm{d}E_o = \int_{\Omega_o} L_o\left(\theta_o, \phi_o\right) \cos\theta_o \,\mathrm{d}\Omega_o \,\mathrm{d}A \,\mathrm{d}t, \tag{2}$$

where Ω_o and $\cos\theta_o$ are as before. Using the definition of the BRDF, we write (2) as

$$\mathrm{d}E_o = \int_{\Omega_o}\int_{\Omega_i} f\left(\theta_i, \phi_i, \theta_o, \phi_o\right) L_i\left(\theta_o, \phi_o\right) \cos\theta_i \,\mathrm{d}\Omega_i \cos\theta_o \,\mathrm{d}\Omega_o \,\mathrm{d}A \,\mathrm{d}t. \tag{3}$$

Due to the BRDF being constant, and from the relationship

$$f\left(\theta_i, \phi_i, \theta_o, \phi_o\right) = \frac{\rho}{\pi}, \tag{4}$$

we can write (3) as

$$\begin{aligned}
\mathrm{d}E_o &= \int_{\Omega_o}\int_{\Omega_i} \frac{\rho}{\pi} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \,\mathrm{d}\Omega_i \cos\theta_o \,\mathrm{d}\Omega_o \,\mathrm{d}A \,\mathrm{d}t \\
&= \frac{\rho}{\pi} \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \,\mathrm{d}\Omega_i \int_0^{2\pi}\int_0^{\frac{\pi}{2}} \cos\theta_o \sin\theta_o \,\mathrm{d}\theta_o \,\mathrm{d}\phi_o \,\mathrm{d}A \,\mathrm{d}t.
\end{aligned} \tag{5}$$

For the two interior integrals in (5), we have

$$\begin{aligned}
\int_0^{2\pi}\int_0^{\frac{\pi}{2}} \cos\theta_o \sin\theta_o \,\mathrm{d}\theta_o \,\mathrm{d}\phi_o &= \int_0^{2\pi} \mathrm{d}\phi_o \int_0^{\frac{\pi}{2}} \cos\theta_o \sin\theta_o \,\mathrm{d}\theta_o \\
&= 2\pi \left[\frac{\cos 2\theta}{4}\right]_0^{\frac{\pi}{2}} \\
&= \pi.
\end{aligned} \tag{6}$$

Replacing this into (5), we obtain

$$\begin{aligned}
\mathrm{d}E_o &= \frac{\rho}{\pi}\pi \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \,\mathrm{d}\Omega_i \,\mathrm{d}A \,\mathrm{d}t \\
&= \rho \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \,\mathrm{d}\Omega_i \,\mathrm{d}A \,\mathrm{d}t.
\end{aligned} \tag{7}$$

Conservation of energy implies that the total amount of energy leaving a surface during a time interval $\mathrm{d}t$ must be less than or equal to the the amount of energy arriving at the surface during that time interval, with the difference being the energy absorved by the surface. Using the notation introduced above, this gives us the inequality

$$\mathrm{d}E_i \geq \mathrm{d}E_o, \tag{8}$$

and from relations (1), (7), we have

$$\mathrm{d}E_i = \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \, \mathrm{d}\Omega_i \, \mathrm{d}A \, \mathrm{d}t \geq \rho \int_{\Omega_i} L_i\left(\theta_i, \phi_i\right) \cos\theta_i \, \mathrm{d}\Omega_i \, \mathrm{d}A \, \mathrm{d}t \Rightarrow$$
$$\rho \leq 1. \tag{9}$$

Finally, we can easily prove that $\rho \geq 0$ by using the fact that both $\mathrm{d}E_i$ and $\mathrm{d}E_o$ are nonnegative, and relations (1) and (7).

2. We list the code for all the parts at the end of this question.

   (a) In figure 1, the x-y components of the surface normals are displayed at three different resolutions, as created by quiver.

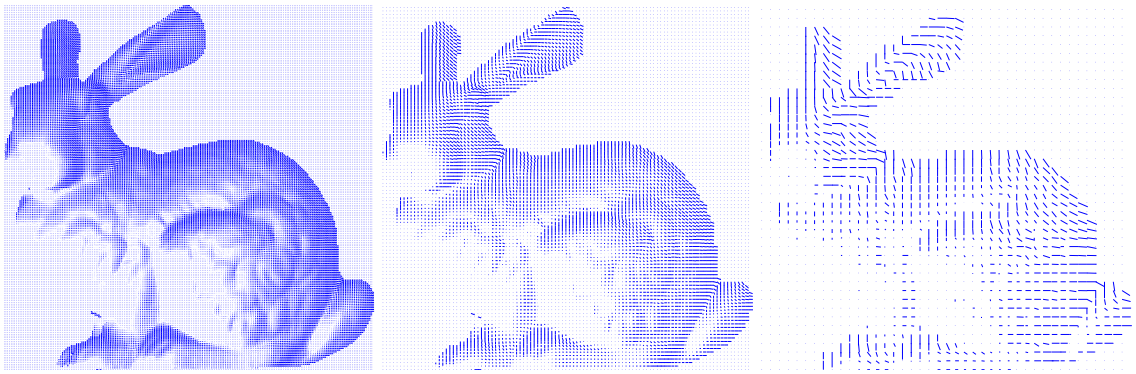

Figure 1: x-y components of the surface normals of bunny, shown in decreasing resolution from left to right.

   (b) The radiance emitted by a point source at each direction $\hat{\omega}_i$ is equal to

$$L\left(\hat{\omega}_i\right) = s\delta\left(\theta - \theta_i, \phi - \phi_i\right) \tag{10}$$

   where $\delta\left(\cdot\right)$ is the Dirac delta functional, and where we use the angles θ and ϕ to parameterize the direction sphere in the same way as in [1]. If the surface receiving this radiance from the point source is assumed to be Lambertian with constant BRDF (and thus constant albedo), then using (10) we can write the radiance emitted due to reflectance at the direction $\hat{\omega}_o$ as

$$\begin{aligned} L\left(\hat{\omega}_o\right) &= f_o \int_{\Omega} s\delta\left(\theta - \theta_i, \phi - \phi_i\right)\left(\hat{n}\right) \, \mathrm{d}\Omega \\ &= f_o s\left(\hat{n}\hat{\omega}_i\right) \\ &= f_o\left(\hat{n}\vec{s}\right) \end{aligned} \tag{11}$$

   where Ω is the hemisphere of directions, f_o is the constant albedo, $\hat{n}$ is the surface normal and $\vec{s} = s\hat{\omega}_i$. Then, from (11), we see that the reflectance and emitted radiance in this case are independent of the output direction $\hat{\omega}_o$. As the radiance term is equal, up to scale, to the intensity captured by a camera[1], by calculating the radiance we can find what a image of such a surface captured under these lighting conditions would look like.

   Due to the fact that the albedo for this surface is assumed constant and equal to 1, (11) implies that to find the surface radiance, and thus surface appearance, all we need to do is to evaluate the inner product $\hat{n}\hat{\omega}_i$. For the case of this question, $\hat{\omega}_i = (0, 0, 1)$, and therefore the inner product gives us

$$\hat{n} \cdot \hat{\omega}_i = \hat{n} \cdot (0, 0, 1) = n_z. \tag{12}$$

[1] Strictly speaking, this is only true when the camera sensor has a linear response curve and infinite dynamic range. We are not concerned with these issues in this assignment.

Using the above, by displaying the z-component of the normal vector, we can find the radiance and therefore what the surface will look like under these lighting conditions. The resulting image is shown in figure 2.

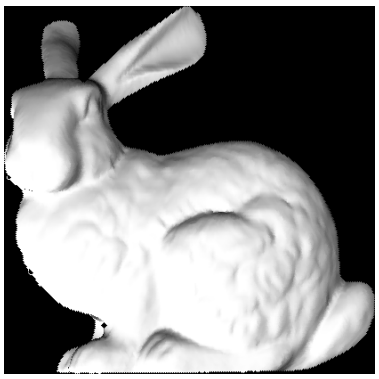

Figure 2: z component of the surface normals, or equivalently radiance emitted by the surface when illuminated by a distant point source from the frontal direction.

(c) Using the results of the previous part, all we need to do is modify $\hat{\omega}_i$ appropriately and replace it into (11).

i For the $45\,^\circ$ up case, we have that $\hat{\omega}_i = (0, \sin 45\,^\circ, \cos 45\,^\circ)$, and the inner product gives us

$$\hat{n} \cdot \hat{\omega}_i = \frac{\sqrt{2}}{2} n_y + \frac{\sqrt{2}}{2} n_z. \tag{13}$$

ii For the $45\,^\circ$ right case, we have that $\hat{\omega}_i = (\sin 45\,^\circ, 0, \cos 45\,^\circ)$, and the inner product gives us

$$\hat{n} \cdot \hat{\omega}_i = \frac{\sqrt{2}}{2} n_x + \frac{\sqrt{2}}{2} n_z. \tag{14}$$

iii For the $75\,^\circ$ right case, we have that $\hat{\omega}_i = (\sin 75\,^\circ, 0, \cos 75\,^\circ)$, and the inner product gives us

$$\hat{n} \cdot \hat{\omega}_i = \sin 75\,^\circ n_x + \cos 75\,^\circ n_z. \tag{15}$$

The results are shown in figure 3. We can detect some errors in the field of surface normals, for example at the front foot and above that. These are caused due to several simplifications we make in our image formation model, which ignore various global illumination effects. The most important effects in this case are cast shadows created due to occlusions (parts of the bunny being hidden by other parts); and interreflections (parts of the bunny receiving light that is reflected on other parts). Secondary effects have to do with the simplistic model for the reflectance properties of the bunny surface. Specifically, we model the surface as perfectly Lambertian, whereas in reality there are also weak specular reflection components and subsurface scattering.

Below we provide one possible implementation of a script for solving question 2.

```
% CS283 Fall 2015
% Student: Ioannis Gkioulekas
% Homework assignment 6, question 2
%
% Surface normals and emitted radiance

%% Part a

% load data
load ../data/bunny.mat;
```

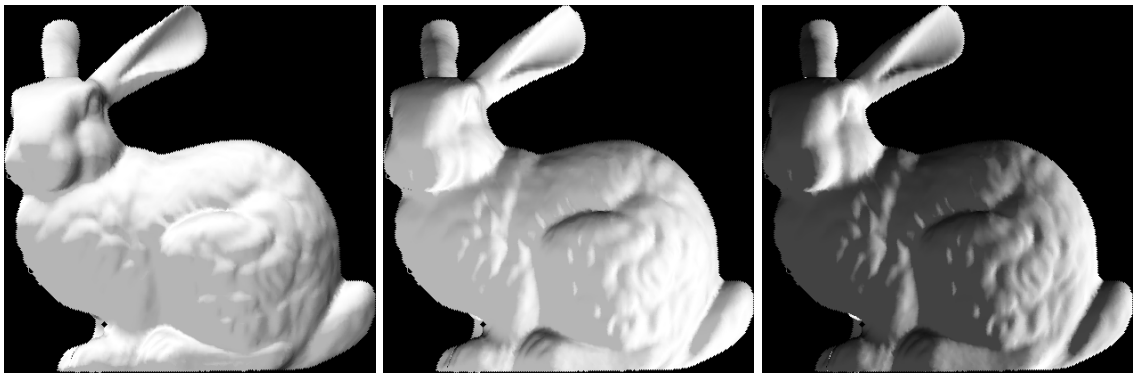

Figure 3: From left to right, radiance emitted by the surface for cases i, ii, iii.

```
11
12 % plot vector field of normal vectors using three differnt resolutions:
13 % dense
14 figure; quiver(N(:, :, 1), N(:, :, 2), '.');
15 axis image;
16 axis ij;
17 axis off;
18
19 % medium
20 figure; quiver(N(1:2:end, 1:2:end, 1), N(1:2:end, 1:2:end, 2), '.');
21 axis image;
22 axis ij;
23 axis off;
24
25 % sparse
26 figure; quiver(N(1:4:end, 1:4:end, 1), N(1:4:end, 1:4:end, 2), '.');
27 axis image;
28 axis ij;
29 axis off;
30
31 %% Part b
32
33 % plot radiance when light is at frontal direction
34 I = N(:, :, 3);
35 figure; imshow(I);
36
37 %% Part c
38
39 % calculate inner products and plot radiance under different lighting
40 % conditions:
41 % for i
42 J = max(sqrt(2) / 2 * N(:, :, 2) + sqrt(2) / 2 * N(:, :, 3), 0);
43 figure; imshow(J);
44
45 % for ii
46 K = max(sqrt(2) / 2 * N(:, :, 1) + sqrt(2) / 2 * N(:, :, 3), 0);
47 figure; imshow(K);
48
49 % for iii
50 L = max(sin(deg2rad(75)) * N(:, :, 1) + cos(deg2rad(75)) * N(:, :, 3), 0);
51 figure; imshow(L);
```

3. (a) For point X_1, the angle between the normal vector at $\mathrm{d}A$ and the line connecting P and X_1 is zero. Therefore, the solid angle subtended by $\mathrm{d}A$ at point X_1 is given by

$$\mathrm{d}\Omega_1 = \frac{\mathrm{d}A\cos 0}{(PX_1)^2} = \frac{\mathrm{d}A}{D^2}. \tag{16}$$

For point X_2, the angle between the normal vector at $\mathrm{d}A$ and the line connecting P and X_2 is equal to α. Also, the distance (PX_2) is equal to

$$(PX_2) = \frac{D}{\cos\alpha}. \tag{17}$$

Therefore, the solid angle subtended by $\mathrm{d}A$ at point X_2 is given by

$$\mathrm{d}\Omega_2 = \frac{\mathrm{d}A\cos\alpha}{(PX_2)^2} = \frac{\mathrm{d}A\cos\alpha}{\left(\frac{D}{\cos\alpha}\right)^2} = \frac{\mathrm{d}A\left(\cos\alpha\right)^3}{D^2}. \tag{18}$$

(b) As the source centered at P is Lambertian, radiance emitted by it at all directions is equal to a constant L. Using the results from part (a), the irradiance at point X_1 can be written as

$$E\left(X_1\right) = L\cos 0\,\mathrm{d}\Omega_1 = L\frac{\mathrm{d}A}{D^2}. \tag{19}$$

Similarly, the irradiance at point X_2 is equal to

$$E\left(X_2\right) = L\cos a\,\mathrm{d}\Omega_2 = L\cos\alpha\frac{\mathrm{d}A\left(\cos\alpha\right)^3}{D^2} = L\frac{\mathrm{d}A\left(\cos\alpha\right)^4}{D^2}. \tag{20}$$

Using (19) and (20), the ratio $E\left(X_1\right)/E\left(X_2\right)$ is equal to

$$\frac{E\left(X_1\right)}{E\left(X_2\right)} = \frac{L\frac{\mathrm{d}A}{D^2}}{L\cos\alpha\frac{\mathrm{d}A\left(\cos\alpha\right)^3}{D^2}} = \frac{1}{\left(\cos\alpha\right)^4}. \tag{21}$$

Equation (21) is the so-called "cosine fourth" law. In photography, it is one of the causes of the effect called *vignetting*, which is the characteristic reduction of brightness as we move away from the image center. Vignetting due to the cosine-fourth illumination falloff is called *natural vignetting*. The wikipedia article on vignetting[2] has more information, along with some nice examples.

4. We list the code for all the parts at the end of this question.

(a) In question 2, we showed that for a Lambertian surface with constant albedo and assuming a distant point light source, the radiance reflected by each surface point at each direction under the illumination of this source equals

$$I = L\left(\hat{\omega}_o\right) = p\left(\hat{n}\vec{s}\right), \tag{22}$$

where p is the albedo at that point, $\hat{n}$ is the normal vector of the surface at that point and $\vec{s}$ is a vector characterizing the light source. Note that, compared to question 2, we no longer assume that the light source is characterized by a unit vector; instead, we use the direction of the vector $\vec{s}$ to indicate the direction of the source, and its magnitude to indicate the brightness of the source. The radiance is also equal (up to scale) to the intensity measured by a camera, hence we include I in the above equation. Then, (22) can be re-written as

$$I = \left(\vec{s}\vec{b}\right), \tag{23}$$

where the albedo has been integrated into vector $\vec{b}$.
As $\vec{b}$ is three-dimensional, if we illuminate the surface from three different directions that are linearly independent, then from the linear system

$$I_1 = \vec{s_1}\vec{b} \tag{24}$$

$$I_2 = \vec{s_2}\vec{b} \tag{25}$$

$$I_2 = \vec{s_2}\vec{b} \tag{26}$$

[2] http://en.wikipedia.org/wiki/Vignetting

we can solve for the vector $\vec{b}$, and from it obtain the albedo and the normal vector. If we have $N > 3$ measurements, we can use the least-squares estimate $\vec{b}$,

$$\vec{I} = S\vec{b} \Rightarrow \vec{b} = \left(S^T S\right)^{-1} S^T \vec{I} \tag{27}$$

where S is the matrix with the $\vec{s_i}$ vectors for each light source as its rows, and $\vec{I}$ is the $N \times 1$ vector formed from the intensity values I_i of the specific pixel in the N different images. Again, we need to assume that the matrix $S^T S$ is invertible, a condition that is satisfied if at least three light source directions are linearly independent. As in previous assignments, the matrix (pseudo-)inverse in the above expression is used only for notational purposes. This heterogeneous system is most efficiently solved using a factorization technique such as QR, and this solution is implemented by the MATLAB backslash operator: `b = S \I`, Note that, even though SVD can also be used to solve this system, it is not recommended as it is considerably more expensive than QR.

Having estimated $\vec{b}$ using the above methodology, we have for the albedo and the normal vector

$$p = \left\|\vec{b}\right\|, \qquad \hat{n} = \frac{\vec{b}}{\left\|\vec{b}\right\|}. \tag{28}$$

In figure 4, we show the results of the albedo and normal estimation from the data provided, using the above methodology.

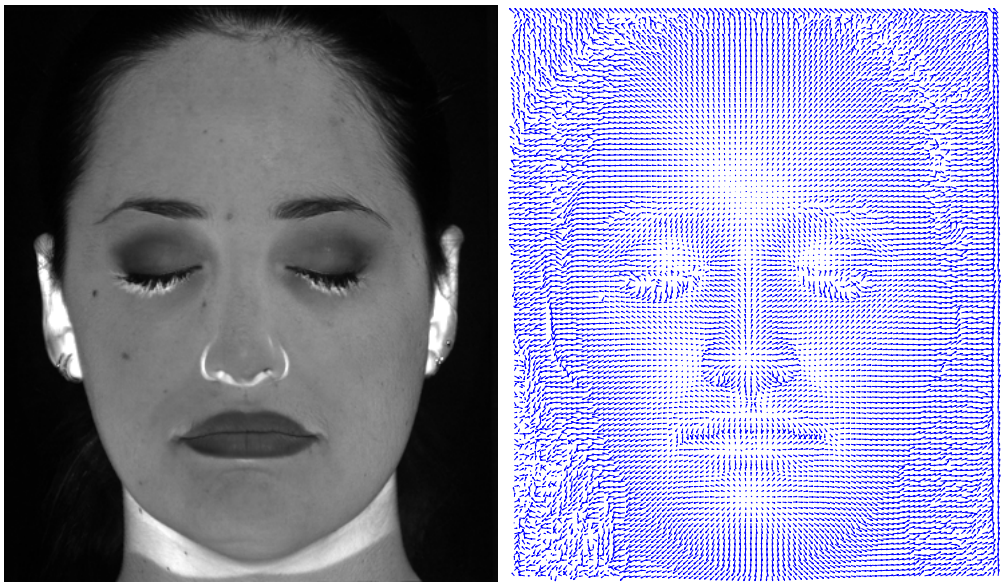

Figure 4: From left to right, albedo estimation and surface normal vector estimation.

Some students have used the method described in [3] to solve this problem. This method is analogous to the factorization algorithm for the problem of structure-from-motion (see problem 1 in assignment 4): we stack all images into a $N \times P$ matrix $\boldsymbol{I}$, where N is the number of images and P is the number of pixels. Under the assumption of far illumination (which is equivalent to saying that the illumination vector is the same in all image locations), based on the above analysis this matrix is equal to

$$\boldsymbol{I} = \boldsymbol{S}\boldsymbol{N}, \tag{29}$$

where $\boldsymbol{S}$ is a $N \times 3$ matrix where the i-th row is equal to the illumination vector for the i-th image; and $\boldsymbol{N}$ is a $3 \times P$ matrix where the j-th column is equal to the product $p\hat{n}$ at pixel j.

Therefore, one can use, for example, SVD to factorize the matrix $\boldsymbol{I}$ into $\boldsymbol{S}$ and $\boldsymbol{N}$, and then the albedo and normal vectors at each pixel are easily recovered by normalizing the columns of $\boldsymbol{N}$.

The problem with this approach is that the factorization (29) is unique only up to an invertible transformation $\boldsymbol{A}$: for a new illumination matrix $\boldsymbol{S}' = \boldsymbol{S}\boldsymbol{A}$ and a surface matrix $\boldsymbol{N}' = \boldsymbol{A}^{-1}\boldsymbol{N}$, we have

$$\boldsymbol{S}'\boldsymbol{N}' = \boldsymbol{S}\boldsymbol{A}\boldsymbol{A}^{-1}\boldsymbol{N} = \boldsymbol{S}\boldsymbol{N} = \boldsymbol{I}. \tag{30}$$

This is analogous to the ambiguity in the factorization algorithm for structure-from-motion, where we can only recover the 3D coordinates up to an affine transformation. This ambiguity arises from the fact that, in the procedure we have described above, we are not taking advantage of information available to us: we already know the illimination vectors, or in other words the real matrix $\boldsymbol{S}$. Indeed, the method in [3] is presented for the problem of photometric stereo when both the surface and illumination are unknown, which is different from our case.

In order to be able to obtain the correct solution from the factorization of $\boldsymbol{I}$, we need to do some extra computation to use the information we have about the illumination sources. Say that from the factorization we obtain two matrices $\boldsymbol{N}'$ and $\boldsymbol{S}'$. Then, given that we know the real $\boldsymbol{S}$, we can use it to find a 3×3 matrix $\boldsymbol{A}$ such that

$$\boldsymbol{S}' = \boldsymbol{S}\boldsymbol{A}. \tag{31}$$

Matrix $\boldsymbol{A}$ can be easily obtained from (31), by solving a corresponding non-homogeneous linear system with 9 unknowns (you should all by now be familiar with how to convert (31) into such a system). Once we have found $\boldsymbol{A}$, then we can obtain the true matrix $\boldsymbol{N}$ as

$$\boldsymbol{N} = \boldsymbol{A}\boldsymbol{N}', \tag{32}$$

and from it get the albedo and normal vectors through normalization of its columns.

(b) The poor estimates in these regions are due to the large shadowing and interreflection effects there, which are ignored in our treatment of the problem above. Specifically, for shadowed parts equation (22) does not hold. As a result, using measurements from these parts in the estimation problem (27) leads to the poor estimates we get for the respective albedo values.

One possible way to alleviate the above problem is to ignore measurements corresponding to images where the respective part is shadowed. This can be done by manually "masking out" shadowed regions, and the for each pixel location using only values from images where the pixel is not shadowed in the least-squares problem we formulated above.

Szeliski [2] proposes an alternative approach, weighting different rows of the photometric stero linear system (corresponding to measurements under different illuminations) by the respective intensity value. By doing so, we place very small weights on very dark pixels, which are likely to be shadowed, and therefore those have little effect in the final albedo and normal estimates.

(c) We use the same method as in exercise 2. We calculate the intensity at each point using

$$I = p\,(\hat{n} \cdot \vec{s})\,, \tag{33}$$

where p is the albedo at that point, $\hat{n}$ the normal vector and $\vec{s}$ the vector describing the illumination. It is important not to forget to include the albedo term in (33), as it varies significantly in different areas of the human face, and ignoring it produces unrealistic results. This is different from question 2, part c (see equation (11)), where we were working under the assumption that the albedo is everywhere equal to 1.

In figure 5, we show the results for the two illumination directions $\vec{s} = (0.58, 0.58, 0.58)$ and $\vec{s} = (-0.58, 0.58, 0.58)$, as calculated using the above relation and the estimates of p and $\hat{n}$ from part (a).

(d) Integrating the vector field of the surface normals estimated in part (a), using the provided function `integrate_frankot.m`, we are able to recover the surface of the face. In figure 6, we present two views of the recovered shape.

Figure 5: From left to right: illumination from $\hat{s} = (0.58, 0.58, 0.58)$; illumination from $\hat{s} = (-0.58, 0.58, 0.58)$.

Figure 6: Two views of the recovered face shape.

Below we provide one possible implementation of a script for solving question 4.

```
% CS283 Fall 2015
% Student: Ioannis Gkioulekas
% Homework assignment 6, question 4
%
% Photometric stereo.

%% Part a

% load data
load ../data/PhotometricStereo/sources.mat

% find number of images
numImgs = size(S, 1);

% load first image and make it grayscale and double
temp = im2double(rgb2gray(imread('../data/PhotometricStereo/female_01.tif')));

% find number of pixels and image dimensions
P = numel(temp);
[M N] = size(temp);

% stack images in large matrix, one image per row (so, one column of this
% matrix corresponds to the 7 measurements of one pixel)
I = zeros(numImgs, P);

% copy first image
I(1, :) = temp(:);

% copy other images, after converting to grayscale and double
for iter = 2:numImgs,
    temp = im2double(rgb2gray(imread(...
        strcat('../data/PhotometricStereo/female_0', num2str(iter), '.tif'))));
    I(iter, :) = temp(:);
end;

% calculate vector b by solving linear system in least-squares sense
b = S \ I;

% calculate albedo
p = sqrt(sum(b .^ 2));
p = reshape(p, M, N);

% calculate components of normal vector
n = reshape(transpose(b), size(temp, 1), size(temp, 2), 3) ./ ...
    repmat(p, [1 1 3]);

% plot vector field of normal vectors
figure; quiver(n(1:4:end, 1:4:end, 1), n(1:4:end, 1:4:end, 2));
axis image; axis ij; axis off;

% plot albedo
figure; imshow(p);

%% Part c

% the two different normal vectors
s1 = [0.58 -0.58 -0.58];
s2 = [-0.58 -0.58 -0.58];

% calculate inner products to find radiance under two different
% illumination conditions
Is1 = max(p .* (n(:,:,1) * s1(1) +...
        n(:,:,2) * s1(2) + n(:,:,3) * s1(3)), 0);
Is2 = max(p .* (n(:,:,1) * s2(1) +...
        n(:,:,2) * s2(2) + n(:,:,3) * s2(3)), 0);
```

```
66
67 % plot results
68 figure; imshow(Is1);
69 figure; imshow(Is2);
70
71 %% Part d
72
73 % integrate normal vectors
74 Z = integrate_frankot(n);
75
76 % plot surface
77 figure; s = surf(-Z);
78 axis image; axis off;
79 set(s, 'facecolor', [.5 .5 .5], 'edgecolor', 'none');
80 l = camlight;
81 rotate3d on
```

References

[1] Forsyth, D., and Ponce, J. *Computer Vision: A modern approach*. Prentice Hall, 2003.

[2] Szeliski, R. *Computer vision: algorithms and applications*. Springer, 2011.

[3] Yuille, A., Snow, D., Epstein, R., and Belhumeur, P. Determining generative models of objects under varying illumination: Shape and albedo from multiple images using svd and integrability. *International Journal of Computer Vision 35*, 3 (1999), 203–222.