Assignment 1

Jenni Simon
09-116-005

Photometric Stereo

1. Calibration

Algorithm Let r be the radius and $c = (c_x, c_y)$ be the image-coordinates of the sphere. Further let $s_i = (s_{ix}, s_{iy})$ be the image-coordinates of the highlights on the sphere in image i for $i \in \{1, \ldots, N\}$. Let the unit vector $d = (0, 0, -1)$ denote the direction of the camera from any point on the sphere. Both r and c can easily be inferred from the provided object-masks and the spot-highlights s_i can also easily be located by looking for maximal intensity values in the images.

The 3D-coordinates of the surface-normal at a position $p = (p_x, p_y)$ on the sphere can then be computed as

$$\begin{aligned} n_x(p) &= \frac{p_x - c_x}{r} \\ n_y(p) &= \frac{p_y - c_y}{r} \\ n_z(p) &= -\sqrt{1 - n_x^2 - n_y^2} \end{aligned} \tag{1}$$

Note that the normal $n(p) = (n_x, n_y, n_z)$ then already has unit length. The unnormalised directions of the light-sources L_i for $i \in \{1, \ldots, N\}$ are then given by

$$L_i = 2 \cdot (n(s_i))^T d \cdot n(s_i) - d \tag{2}$$

Resulting Light-Directions Normalising these light-directions I obtained the following results (row i corresponds to L_i^T):

$$\mathbf{L^T} = \begin{bmatrix} 0.496 & -0.466 & -0.733 \\ 0.245 & -0.129 & -0.961 \\ -0.038 & -0.180 & -0.983 \\ -0.103 & -0.438 & -0.893 \\ -0.323 & -0.506 & -0.799 \\ -0.117 & -0.558 & -0.822 \\ 0.287 & -0.418 & -0.862 \\ 0.094 & -0.438 & -0.894 \\ 0.209 & -0.342 & -0.916 \\ 0.095 & -0.327 & -0.940 \\ 0.129 & -0.046 & -0.990 \\ -0.136 & -0.359 & -0.923 \end{bmatrix}$$

2. Computing Surface Normals and Grey Albedo

Algorithm Let I_i be a gray-level image and let $I_i(p)$ denote the intensity at pixel p for $i \in \{1, \ldots, N\}$. The pixel intensity can then be modeled as $I_i(p) = \alpha(p)(n(p)^T L_i)$ (assuming diffuse objects and distant point light sources). We can further combine the I_i's into a $M \times N$ matrix of image-intensities by vectorising the relevant image pixels of image I_i. M therefore stands for the number of image pixels corresponding to the object and N for the number of images (and light-sources). Let n denote the $3 \times M$ matrix of the unknown surface normals. The problem can then be written as $I = (\alpha n^T)L$, or equivalently as $I^T = L^T(\alpha^T n)$ and we can solve for the the unknown matrix $A = \alpha^T n$ via

$$A = (LL^T)^{-1} L I^T \tag{3}$$

The matrix $(LL^T)^{-1}L$ is also called the pseudo-inverse of L. From the definition $A = \alpha^T n$ we see that the albedo α is given by the magnitude of the columns of A and the normals n are given by the normalized columns of A.

Having computed the normals, we can now solve for the individual red-, green- and blue-albedo of the objects. To this end, let I_r denote the $M \times N$ matrix of vectorized red-channel images. Our model again is $I_r = \alpha_r(n^T L)$, where α_r is the unknown red-albedo. Let $G := (n^T L)$. We solve for α_r by minimizing the error

$$\alpha_r = \underset{\alpha}{\operatorname{argmin}} \|I_r - \alpha_r G\|^2 \tag{4}$$

Taking derivatives and setting them to zero, we obtain the follow formula for

the albedo at pixel p:

$$\alpha_r(p) = \frac{\sum_{i=1}^{N} G(p,i) \cdot I_r(p)}{\sum_{i=1}^{N} G(p,i)^2} \tag{5}$$

Results Applying the above procedure to the dataset we obtain the gray albedos in figure 1 and the normal-maps shown in figure 2. The normals in figure 2 are visualized by mapping the x-, y- and z-coordinates of the normals to RGB-values in the range $1 - 255$. Finally, figure 3 shows the recovered color albedos.

Fig. 1: Recovered gray albedo

3. Surface Fitting (35 points)

Algorithm Let $S(x,y) = (x,y,z(x,y))$ be the surface we would like to recover. The tangent vectors are then given by

$$\begin{aligned} t_x &= \frac{\partial S}{\partial x} = (1,0,z_x)^T \\ t_y &= \frac{\partial S}{\partial y} = (0,1,z_y)^T \end{aligned} \tag{6}$$

where $z_x = \frac{\partial z}{\partial x}$. The estimated normal $n(x,y)$ should then be consistent with

$$n(x,y) = \frac{t_x \times t_y}{\|t_x \times t_y\|} = \frac{(z_x, z_y, -1)^T}{\|(z_x, z_y, -1)\|} \tag{7}$$

Fig. 2: Normal maps

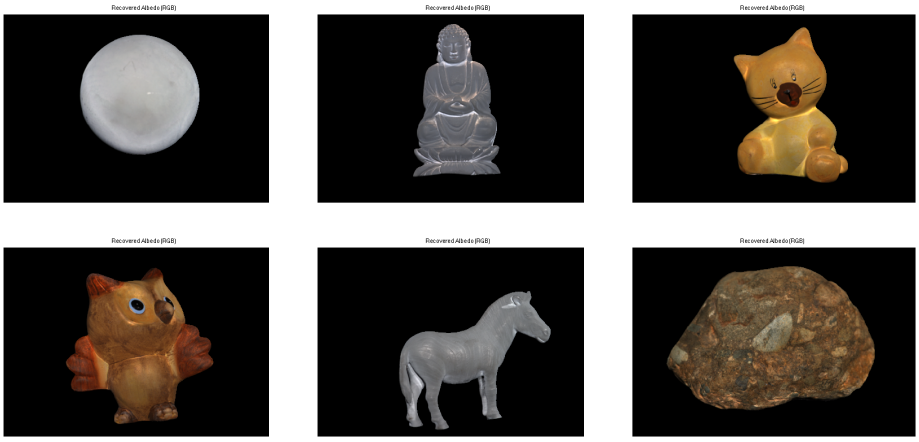

Fig. 3: Recovered RGB albedo

Using forward differences to approximate the depth-derivatives, we get

$$\begin{aligned} z_x &\approx z(x+1,y) - z(x,y) \\ z_y &\approx z(x,y+1) - z(x,y) \end{aligned} \tag{8}$$

Substituting equation (8) in equation (6) we get an approximation of the tangent vectors as

$$\begin{aligned} t_x(x,y) &\approx (1,0,z(x+1,y) - z(x,y))^T \\ t_y(x,y) &\approx (0,1,z(x,y+1) - z(x,y))^T \end{aligned} \tag{9}$$

We know that these tangent vectors should be perpendicular to the normal-vector and can therefore introduce the constraints

$$\begin{aligned}(1, 0, z(x+1, y) - z(x, y)) \cdot n(x, y) = 0 \\ (0, 1, z(x, y+1) - z(x, y)) \cdot n(x, y) = 0\end{aligned} \tag{10}$$

Which is equal to:

$$\begin{aligned}(z(x+1, y) - z(x, y)) \cdot n_z(x, y) = -n_x(x, y) \\ (z(x, y+1) - z(x, y)) \cdot n_z(x, y) = -n_y(x, y)\end{aligned} \tag{11}$$

Let $\tilde{z}$ be the vectorization of the $m \times n$ matrix z such that $z(x, y) = \tilde{z}((x-1)n+y)$ and let $\tilde{n}(p) = (\tilde{n}(p)_x, \tilde{n}(p)_y, \tilde{n}(p)_z)$ be defined similarly. Equation (11) can then be rewritten as

$$A\tilde{z} = b, \tag{12}$$

where A is an $2mn \times mn$ matrix and b is a vector of length mn. A and b encode equation (11) as follows:

Let $i = (x-1)n + y$ for any position (x, y) on the object. The entries of A are then given by

$$A(2i-1, j) = \begin{cases} -\tilde{n}_z(i) & \text{if } j = i \\ \tilde{n}_z(i) & \text{if } j = i + n \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

and

$$A(2i, j) = \begin{cases} -\tilde{n}_z(i) & \text{if } j = i \\ \tilde{n}_z(i) & \text{if } j = i + 1 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The entries of b can in turn be computed as:

$$b(k) = \begin{cases} -\tilde{n}_x(i) & \text{if } k = 2i - 1 \\ -\tilde{n}_y(i) & \text{if } k = 2i \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Note that A is a sparse matrix and the system (12) can therefore be solved efficiently. The so recovered depth z is then fixed such that minimum depth is equal to zero.

Omitted detail: In case points $(x+1, y)$ or $(x, y+1)$ would not belong to the object, backward-differences were used in the derivation of A (adjusting equation (9) accordingly)

Results Figure 4 shows the recovered depth-maps for the six datasets. The image intensities correspond to the relative depth of the pixels, where brighter pixels are closer than darker ones.

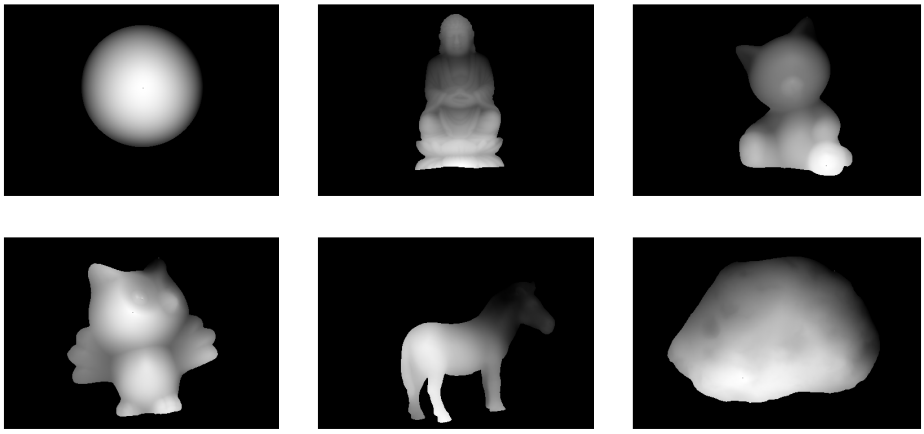

Fig. 4: Recovered depth-maps

Discussion It can be observed that sometimes the method can fail in dark regions like the owl's eye, where we can observe a spike in the depth-map. The reason for this is most likely that shading can hardly be observed in dark regions as they barely reflect light.

Another source of errors, namely shadows, can much better be observed in the recovered albedos and normals. Because the method, as it is presented here, doesn't handle shadowed areas separately, the lower brightness in these areas is "interpreted" as shading by the method, which then leads to wrong results. These effects can be best observed in the neck-region of the buddha and cat, as well as the ear and leg of the zebra.

Specular reflections, another likely cause of failures, are not clearly observable in the given data and the assumption of Lambertian surfaces seems justified.

In bright regions with diffuse reflections and without shadows the results of the method are rather convincing.