Computer Vision 1, Assignment 1

Erik Stammes Max Bos Lars Rigter Jan Zuiderveld

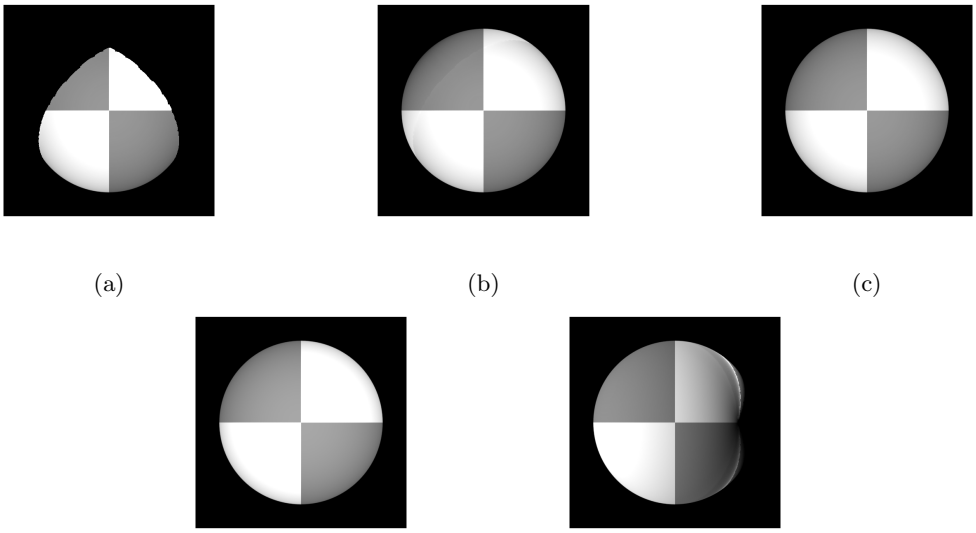
February 20, 2019

Introduction

In this assignment we experiment with different algorithms to estimate shapes by computing surface albedo and normals, do recoloring using image decomposition techniques and check for color constancy using the GrayWorld algorithm. Also, we experiment with different color spaces.

1 Photometric Stereo

1.1 Estimating Albedo and Surface Normal



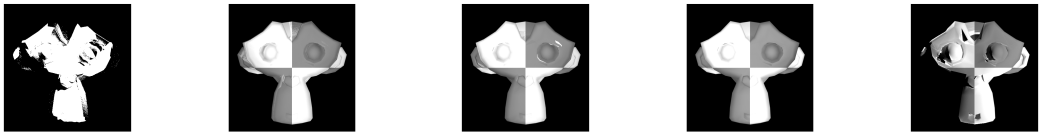
(a) (b) (c)

(d) (e)

Figure 1: Albedo images created from (a) 3, (b) 4, (c) 5 and (d) 25 gray sphere images. (e) Albedo image created from 5 gray sphere images where the light source has less variation compared to previous examples.

1. In an albedo image we expect to see the object without any lighting artifacts such as shadows. Our estimation result (Figure 1c) still shows some minor lighting artifacts, such as shadows on the outside of the sphere.

2. The minimum number of images required to estimate albedo and surface normal depends on how much variation there is in the light source direction. As shown in Figure 1 the albedo can be estimated without much error using just 5 images if there is high variation in the light source direction. With 25 images (as shown in Figure 1d) the albedo estimation is even better. Figure 1e shows the albedo



(a) (b) (c) (d) (e)

Figure 2: Albedo images created from (a) 3, (b) 4, (c) 5 and (d) 121 gray monkey images. (e) Albedo image created from 5 gray monkey images where the light source has less variation compared to previous examples.

(a) (b)

Figure 3: The absolute difference between albedo images generated with and without the shadow trick on 5 images of (a) a sphere and (b) a monkey object.

estimation using 5 images, where the light source direction does not vary much. (i.e. the used values are $X = 1$ and $Y = [-1, -0.5, 0, 0.5, 1]$). The estimation is clearly worse than the estimation in Figure 1c and therefore estimation depends on the variation in the light source direction.
When there is enough variation in the light source direction, four images seems to be the minimal number, as illustrated in Figure 1a and 1b. Figure 2 confirms these results for an image of a monkey object.

3. For simple images, such as the gray sphere and monkey images the shadow trick makes little difference. Visually, there is hardly any difference between the two, hence we calculated the absolute difference and plotted this in Figure 3. This shows some minor difference between images created using 5 images with and without the shadow trick.
The shadow trick works as follows: we construct a diagonal matrix of size $N \times N$ where N is the number of images, which has as elements the values from each image at that specific position. The image matrix is now multiplied by this matrix on both sides, which has the result of zeroing the contributions from shadowed regions, because the relevant elements of the matrix are zero at points that are in shadow [1].

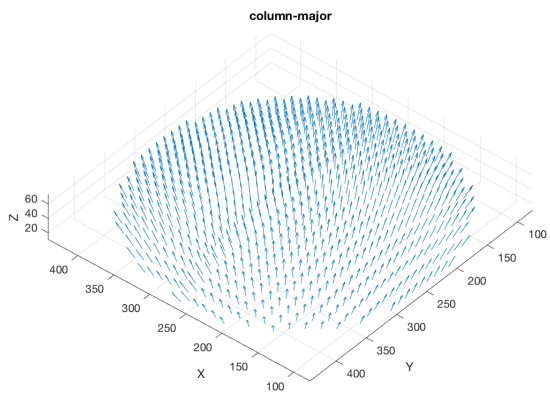
1.2 Test of Integrability

In theory, the threshold could be set at 0, but because there is some numerical instability in estimating the partial derivatives, we set the threshold to 0.005.
The number of outliers becomes less when more input images are used, as shown in Table 1. When using 3 input images, the number of outliers is more than 70.000 for both objects, while this number drops almost tenfold when 5 input images are used. The number can be further reduced, but not as much, by increasing the number of images to 25.
The reason for these errors is because the second order derivatives only hold for points on the sphere. When we compare two points where one of them lies on the sphere, and one of them not, it will result in errors when computing the second order derivatives.

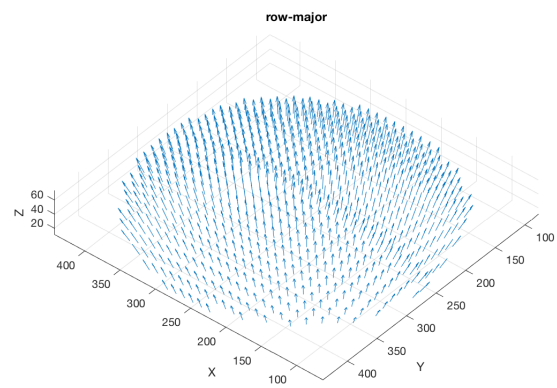
Object	# images	# outliers
Sphere	3	73339
Sphere	5	2369
Sphere	15	2025
Sphere	25	1831
Monkey	3	71998
Monkey	5	8799
Monkey	15	5407
Monkey	25	4070

Table 1: The number of outliers for the sphere and monkey objects, for different number of input images

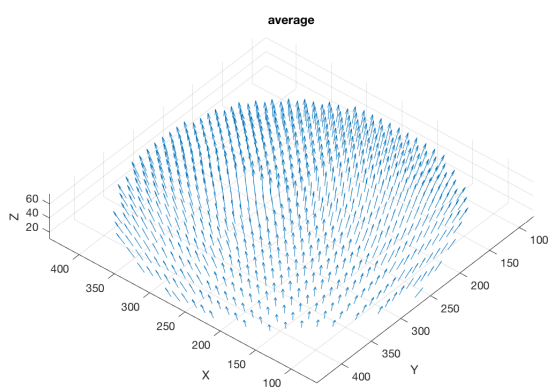
1.3 Shape by Integration



(a) Column-major



(b) Row-major



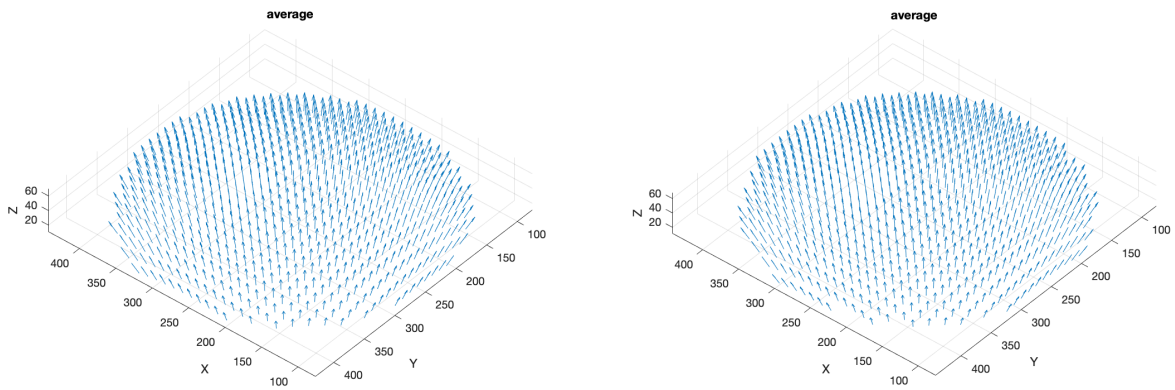
(c) Average of column- and row-major

Figure 4: Height map of the sphere when computed using (a) column-major and (b) row-major order using 5 input images. (c) is the average of the column-major and row-major height maps.

Figure 4 shows a velocity vector representation of the height maps computed using column-major order, row-major order and their average using five sphere input images. There is a subtle difference between the

shapes of the height maps in Figure 4a and Figure 4b at the borders at which the color regions change on the sphere from white to gray. The column-major integration presents the partial derivative with respect to the X-axis and the row-major integration presents the partial derivative with respect to the Y-axis, so the intensity of the height maps is higher at the edge of the white to gray regions depending on the direction of the derivative.

We see an improvement when calculating the average of the column- and row-major integration, shown in Figure 4c. The Figure shows that the average includes the color change with respect to the X-axis as well as with respect to the Y-axis, this partially cancels out the height intensity at the previously discussed regions.



(a) Average using 15 images

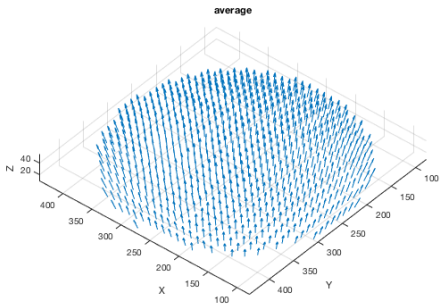
(b) Average using 25 images

Figure 5: Height map of the sphere when averaging the continuous integration of the partial derivatives in the column- and row-major order using (a) 15 sphere images and (b) 25 sphere images.

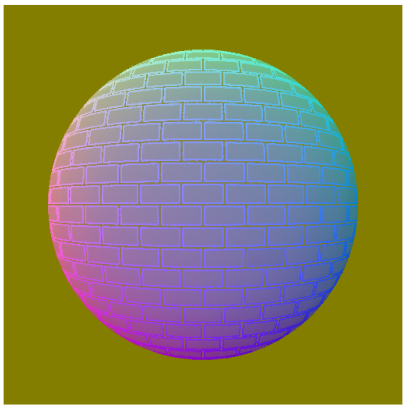
We have run experiments to test the effect of increasing the number of images under different illuminations on the shape of the surface height map. The resulting shapes for these experiments are depicted in Figure 5. The shape of the surface height map improves by adding more images under different illumination, because the shape of the surface height map is more smoothed when using 15 and 25 images compared to using just 5 images. The difference in height intensity around the previously discussed borders seem to be less present when increasing the number of images.

1.4 Experiments with different objects

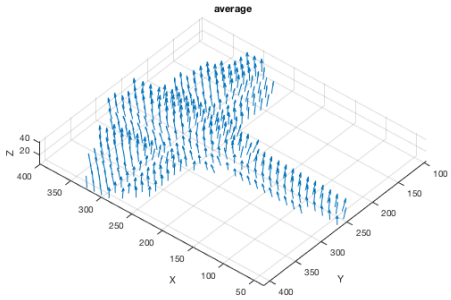
As previously seen in Figure 2, 3b and Table 1, it is more difficult to compute the surface albedo and normal map of the gray monkey object. For the same number of input images, the number of outliers is at least doubled, but contrary to the gray monkey object, there are large improvements when increasing the number of input images from 15 to 25. We expect that when the number of input images is increased further, the number of outliers will continue to decrease. The reason for the increased number of outliers is the increased complexity of the object, the monkey is a much more nuanced object that has more shadows around the eyes, ears, nose, mouth and eyebrows.



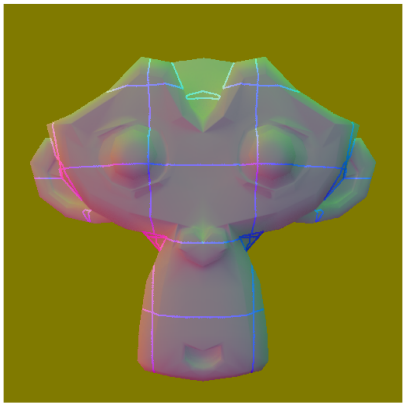
(a) Height map for SphereColor



(b) Constructed surface map for SphereColor



(c) Height map for MonkeyColor



(d) Constructed surface map for MonkeyColor

Figure 6: Constructed surface normal map and height map for the SphereColor and MonkeyColor models.

For the photometric stereo to work for 3-channel RGB inputs, the image stack for each separate channel is loaded using **load_syn_images.m** and fed into **estimate_alb_nrm.m**, yielding a 2-dimensional albedo and 3-dimensional surface normals matrix for each channel. Next, the three albedo matrices are constructed into one 3-dimensional matrix. The integrability checking is performed for each of the surface normals matrices, yielding a p and q matrix for each channel, and for each of the surface normal matrices a height map is calculated. Finally, the average of the height maps is calculated. Experiments were run using the SphereColor and MonkeyColor models. Figure 6 shows the constructed surface normal map and height map for these models. The normal height maps in Figures 6b and 6d look appropriate. On the other hand, the height maps in Figures 6a and 6c do not look correct. A zero pixel could be a problem since this would lead to a NaN type in some calculations, which leads to problems when averaging the height map matrices over the channels. This problem can be fixed by setting all NaN type values in the matrices to zero before calculating the average.

Figure 7 shows the height maps generated using 64 images from the Yale Face dataset [2][4], without using the shadow trick. These images violate the assumptions of the shape-from-shading methods because faces exhibit subsurface scattering and are not lambertian surfaces, they contain more than shadows: especially the nose, cheeks and forehead are reflective. Furthermore, not all images contain (clear) shadows, hence we created a height map using 45 images which all had sufficient shades on the face, this can be seen in Figure 7d. Figure 7a, 7b and 7c show the difference between using column-major order, row-major order and their average when computing the height map. The column-major order introduces a gradual difference in z-direction between the lower part and the upper part of the face, while the row-major order introduces

(a) (b) (c) (d)

Figure 7: Height maps generated using 64 images from the Yale Face dataset. (a) Column-major order, (b) row-major order and (c) averaged. (d) Height map generated when 19 images with minimal shadow are deleted

artifacts along the x-axis of the face. The averaged method reduces both effects and gives the best result.

(a) (b) (c) (d)

Figure 8: Height maps generated using 64 images from the Yale Face dataset. Each subfigure is a different subject.

Figure 8 shows the result of using the same method as above, but without deleting images to manually increase the effects of the shadow. Although there are some artifacts, the method is generally robust and correctly estimates the height of facial features such as the nose, lips and eyes. However, the results can likely be improved by using a method such as the one introduced by Hauagge et al. [3], which has better results for albedo estimation. This method uses a simplified image formation model and per-pixel statistics over image stacks to estimate, amongst others, the albedo of an object in an image.

2 Color Spaces

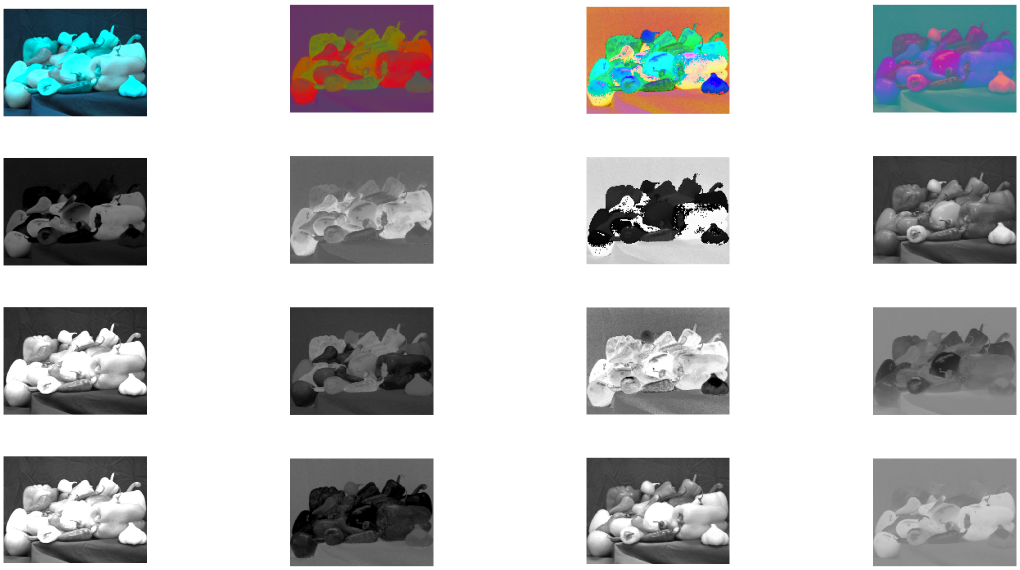
2.1 RGB Color Model

The use of RGB color model as a basis of our digital cameras and photography comes from the fact that humans are trichromats, which means that humans possess of three independent channels for conveying color information, derived from the three different types of cone cells in the eye, namely the red (R), blue (B), and green (G) channels.

A digital camera captures the total intensity of the light that strikes a surface, the sensor in the digital camera uses filtering to look at the light in its three primary colors.

2.2 Color Space Conversion

Figure 9: Peppers.png, the original image of which processed versions are shown below.



(a) (b) (c) (d)

Figure 10: Visualizations of different color space conversions. Lower 3 pictures are the separate channels forming the upper picture in RGB space (a) Opponent Color Space, (b) Normalized RGB (rgb) Color Space (c) HSV Color Space (d) YCbCr Color Space

Figure 11: Peppers.png converted to grayscale using 4 different methods, lightness (upper left), average (upper right), luminosity (lower left) and Matlab's build-in grayscale conversion method (lower right)

2.3 Color Space Properties

Opponent colour space maps colours in the image to three dimensions with opposite properties: red/green, yellow/blue, dark/light. One advantage is that it makes the image invariant to brightness. The colours will have different intensity but the ratio between the properties in the different dimensions do not change.

Normalized RGB translates the values of the different channels to a similar scale. It reduces the sensitivity to distortions. For example when there is shadow in the image it will normalize shadow points which make them less different compared to non shadow points.

HSV Color space maps RGB to HUE Saturation and Value. Hue is the deviation from the three basic colours. Saturation is the scaled distance from the diagonal e.g. the degree of the colour, 0 percent is grey and 100 percent is the full colour. Value is the mean color value or intensity, 0 percent is black and 100 percent is white. By using HSV the image can be made invariant to intensity. This can be used to make the image robust against different lightning conditions.

YCbCr maps RGB to different colour spaces. Y is the lumination or brightness. Cb is the blue component relative to the green component. Cr is the red component relative to the green component. Humans are more sensitive for brightness than to Cb and Cr. By using this mapping one could store only Y which is most relevant and disregard the other components.

Grayscale maps RGB to a gray colour space. This can be handy since in some cases colour does not add any valuable information. For example to identify an object that is independent of its colour one want to exclude colour in the feature representation since this does not add value.

2.4 More on Color Spaces

CYMK is the representation of a colour in cyan, magenta, yellow, and key. These colours are used in printers. CYMk are leftover colours when red, green and blue are subtracted from white light. When adding pigment to paper, certain colours are masked from the white backgound(paper) which yields a colour.

3 Intrinsic Image Decomposition

3.1 Other intrinsic decomposition

Another example is matting. Matting is a method to separate the foreground and the background of an image. This can be handy for example when taking a picture through a window. In this case it can be handy to separate the window reflection from the view through the window.

3.2 Synthetic images

Real world images can be too noisy. For example real world images can contain multiple light sources and complex reflections such as surface reflections. Also it is important to have the ground truth values since one could compare the constructed albedo towards the true albedo.

3.3 Image formation

In figure 12 the reconstruction can be seen.

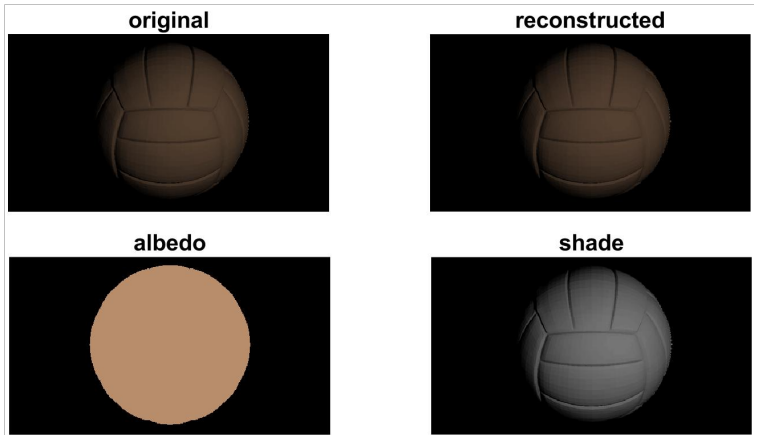


Figure 12: Image decomposition

3.4 Recolouring

The RGB colour of the albedo is [108,141,184]. The recoloured image of the ball can be seen in figure 13.

Figure 13: Recoloring of the ball to green

The reason that the colour of the reconstructed image does not seem to be uniform is due to the reconstruction with the shade. Now the colour of the ball is mixed with gray due to the reconstruction.

4 Color Constancy

4.1 Grey-World

1. The pictures in figure 14 are the result of our Grey-World script. In this script the Grey-World algorithm is applied using addition and scaling to obtain the desired average of 128 on every channel.

   because it is unclear what method is desired for the assignment, both are given (addition, scaling). If only one method is graded, please ignore the addition algorithm.

   The Grey-World Scaling algorithm seems to produce the best results in this case.

2. The Grey-World algorithm assumes a grey world on average, implying that the average of every channel is 128 (r,g,b = 128,128,128 = Grey).

   If any color is over-represented in a picture, the Grey-World algorithm will produce artefacts. Imagine a picture filled with red flowers, the means would approximate (r,g,b) = (256,0,0). If this picture was processed by the Grey-World algorithm the r channel would be heavily downscaled, and the g and b channels heavily upscaled, resulting in a distorted image containing colours it does not contain in real life.

(a) Original Image (b) GW Addition-Processed Image (c) GW Scaling-Processed Image

Figure 14: Grey-World algorithm processed images

3. The White Patch Algorithm

   Assumes a perfect white spot in the picture. What it does: Look for the maximum r,g,b values in the picture, and get a scaling factor for every channel by dividing 256 (the value white should have) by it's maximum value. The resulting factor is applied to every pixel in it's channel, mapping the maximum values to white.

5 Conclusion

In this assignment we experimented with computer vision methods such as photometric stereo, color spaces, image decomposition and color constancy. The obtained results were insightful to us however we realized that certain methods work well in particular for synthetic images. For example the reconstruction of an image taken with different light sources works fine for synthetic images but does not work well for real world images. The assignment allowed us to experiment with different methods, parameters, tricks and algorithms, giving us deeper insight into computer vision techniques.

5.1 Contribution

Everyone contributed equally. While the questions were divided amongst two teams of two members, we all came together to discuss the final result and gave feedback where needed. We all agree with the final submission.

References

[1] David A Forsyth and Jean Ponce. "Computer Vision: A modern approach". In: 17 (), pp. 80–86.

[2] Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. "From few to many: Illumination cone models for face recognition under variable lighting and pose". In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 6 (2001), pp. 643–660.

[3] Daniel Hauagge et al. "Photometric ambient occlusion". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2515–2522.

[4] Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. "Acquiring linear subspaces for face recognition under variable lighting". In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 5 (2005), pp. 684–698.