Computer Vision 1
Assignment 1

Ivan Bardarov, Puck de Haan, Tim van Loenhout, Lino Miltenburg

October 2, 2019

Introduction

In this report we will take a look at photometric stereo and color in computer vision. First, we will implement the photometric stereo algorithm as described in Forsyth and Ponce (2002) and discuss the different techniques that were used, together with the final results. Secondly, we will run several experiments on color spaces and compare the RGB color model to other color models, such as the HSV color space. Subsequently, intrinsic image decomposition into its formation components and the recoloring of images will be discussed and illustrated using several experiments. To conclude, we will implement the *Grey-World Algorithm*, take a more detailed look at this algorithm and give a concise review of different color constancy algorithms.

1 Photometric Stereo

1.1 Estimating Albedo and Surface Normal

1. We expected to see an equal brightness across all locations on the sphere, since we assumed the object metrial to be Lambertian and the albedo is the material color of a object without any illumination effect. However, our result displays slightly lower albedo values at the edges opposed to the high values in the center, due to which some shading in still perceived in the image (Figure 1).

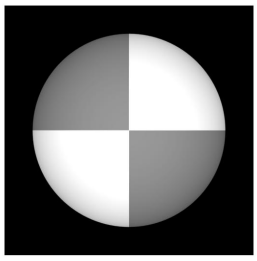
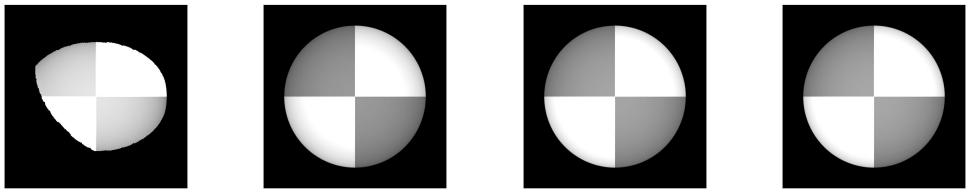


Figure 1: Albedo image, generated from 5 pictures.

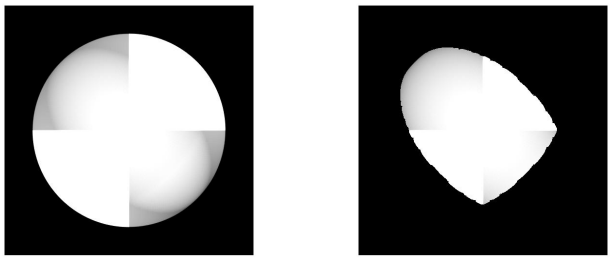
2. The minimum number of images that is necessary to estimate albedo and surface normal is three, such that the resulting linear system with 3 variables (x, y, z) can be solved. We increased the number of pictures from 3 to 5, 15 and 25. The more images that were used, the more uniform the albedo (Figure 2). By using more pictures, the shading effect almost completely disappeared, which matched our previously discussed expectation of an equal albedo across the whole object.



(a) 3 images (b) 5 images (c) 15 images (d) 25 images

Figure 2: Albedo image, incremental increase from 3 to 25 pictures.

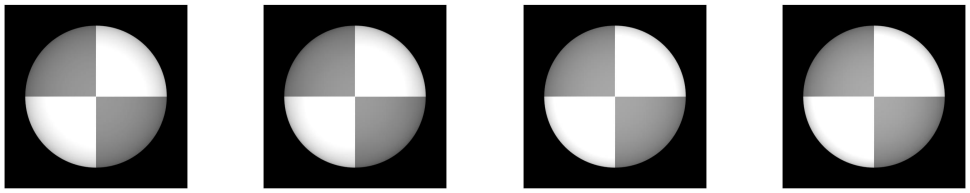
3. Shadows are the parts of an object were no light reaches, and thus no light is reflected. However, as displayed in Figure 3a, when computing the albedo on these parts, a value is returned where it should be zero. In order to circumvent this, a shadow trick can be used. This trick applies a diagonal matrix transformation on the V matrix (containing the properties of illumination and the camera), setting its equations for shadowed parts to zero. The result is an albedo image where the parts that receive no light return a reflection value of zero, as can be seen in Figure 3b.



(a) Shadow trick off (b) Shadow trick on

Figure 3: Reconstructed albedo, generated from 3 pictures.

The impact of the shadow trick decreases when generating an albedo image from a set of pictures in which every part of the object receives at least some light. For instance, as seen in Figure 4, the effect becomes insignificant when applying the shadow trick on the albedo image generated from 25 instead of 5 images.



(a) Shadow trick off (b) Shadow trick on (c) Shadow trick off (d) Shadow trick on

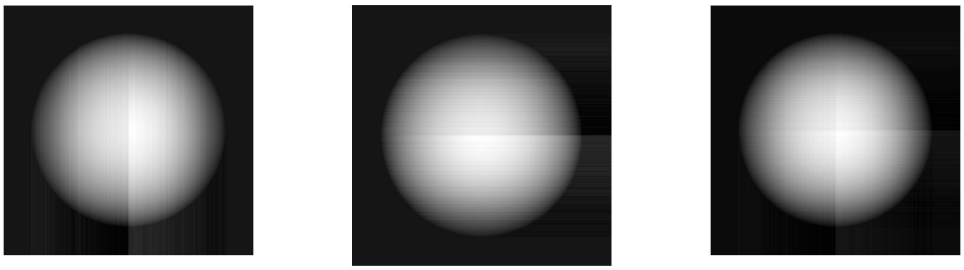
Figure 4: Reconstructed albedo, generated from 5 (left) and 25 (right) images.

1.2 Test of Integrability

Since we are estimating the partial derivatives numerically, it could be that some small errors appear, because of the residual error that could result from the least squares solution used to solve the linear system per point in the image from which our normals originate. The numerical precision of Matlab could also cause these slight errors. When using a threshold of 0.005, there are 50411 outliers when a set of 5 images is used for the reconstruction process and 45634 outliers when a set of 25 images is used. This difference indicates that a more precise solution can be found when using a larger set of images, resulting in less outliers.

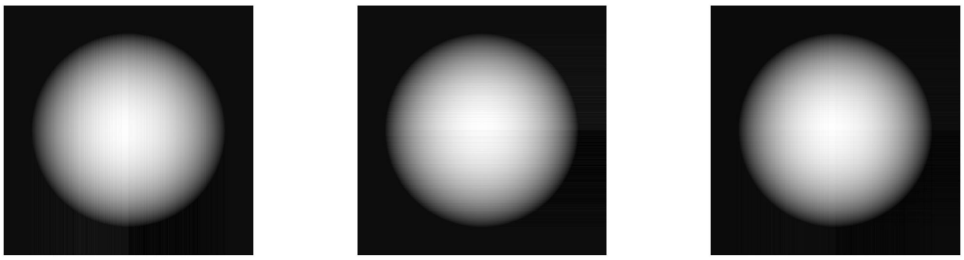
1.3 Shape by Integration

We constructed the surface height map of the sphere using both the column-major order and the row-major order. A clear difference is visible between the two. The map that was constructed using the row-major path shows vertical stripes throughout the image of the map, whereas we can see horizontal stripes on the image of the map that was constructed using the column-major path (Figure 5a, 5b). When taking the average of both results, we get a height map with both horizontal and vertical stripes when using 5 images, but a very uniform height map when using 25 images (Figure 5c, 6c). Overall, the number of streaks decreases when using a larger set of images. (Figure 5, 6).



(a) Row-major path (b) Column-major path (c) Average

Figure 5: Surface height map, generated from 5 pictures.



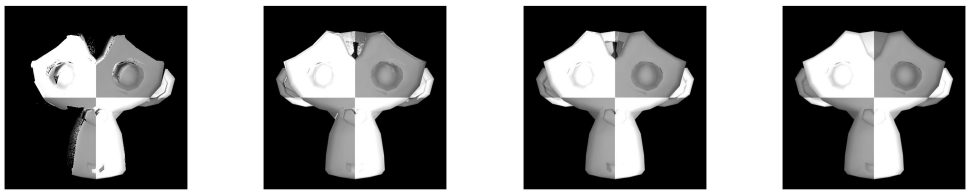
(a) Row-major path (b) Column-major path (c) Average

Figure 6: Surface height map, generated from 25 pictures.

1.4 Experiments with different objects

4. When running the photometric stereo algorithm on the MonkeyGray model, more albedo errors are observed than on the GreySphere model. This is most likely the result of the substantial

amount of height differences across the object. Sections of the object which are located next to steep ascends are in shadow for a multitude of light directions. As a consequence, little information can be subtracted from such a region on the magnitude of its reflection
However, when using more images, with different light directions, the amount of light these regions receive increases, resulting in more information on its albedo. For instance, the area between the eyes of the monkey in Figure 7 is located between two substantially higher surfaces. When generating the albedo images based on 3 pictures, no light reaches this area, resulting in no albedo information. When increasing the number of pictures, the albedo on this area becomes more accurate, resulting in a lower error.

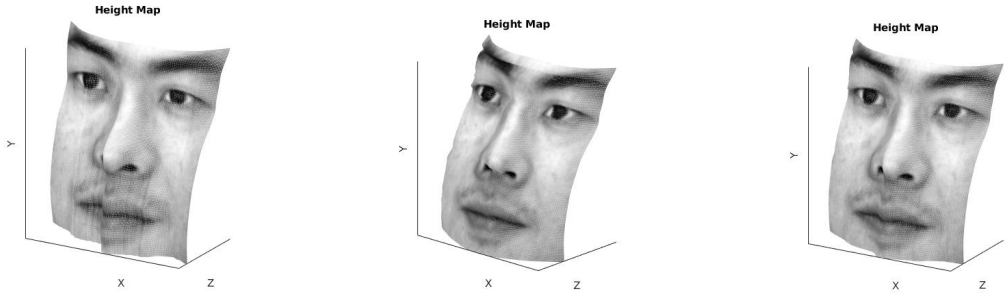


(a) 3 pictures (b) 5 pictures (c) 10 pictures (d) 121 pictures

Figure 7: Albedo image, incremental increase from 3 to 121 pictures.

5.

6. First of all, we reconstructed the image albedo and height map using all 64 given YaleB02 images and used them to create a three-dimensional image (Figure 8). The row-major path result is the least natural looking of all three results; there seems to be a vertical stripe in the middle of the face that is unnaturally high, compared to the relief of the rest of the face. The model that was constructed with the column-major path looks better than the row-major one, but also a little bit flattened out. The average of both model produces the best-looking 3d-model, but still has the unnatural vertical strip in the middle of the face.



(a) Row-major path (b) Column-major path (c) Average

Figure 8: 3d models created with the Yale face images, using different integration paths.

Compared to the GraySphere and MonkeyGrey model, the Yale face database model has a relatively high albedo error. This discrepancy could be caused by the albedo properties of the YaleB02 mode. That is, the sphere and monkey are constructed from a Lambertian material, whereas the YaleB02 model consists of multiple materials, of which some have a more directional diffuse. On top of that, the varying light intensities across the different pictures in the YaleB02 database are unknown, making photometric stereo unfeasible. See for instance (Figure 9) where

the light from the right is stronger than the light from the left. However, such a discrepancy between light intensity from the left and from the right exists across many pictures, so taking all these out of the database goes at the expense of a lot of information. Therefore, not much improvement is visible between the height maps based on the full and filtered database.

(a) Angle +050E+0 (b) Angle -050E+0

Figure 9: Light source discrepancy between otherwise similar pictures

2 Color Spaces

In this exercise we explored different color spaces and experimented with an image that was converted from an RGB model to various other color systems. The original image can be seen in Figure 10.

Figure 10: RGB representation of peppers

2.1 RGB Color Model

Photographic images are usually based on 3 primary colors - Red, Green and Blue, hence the abbreviation. They are called additive primary colors since combining them can produce new colors and when they are combined together in equal amounts they form white. Similarly, when there is no color selected, the image appears black. The existence of these three colors is a result of the tri-stimulus nature of the human eye. It contains 3 different kinds of cones and each of them is sensitive to a different portion of the color spectrum. The color space can be illustrated as a cube by mapping the colors to the x, y and z axis.
A camera integrates light according to the spectral response function of its red, green, and blue sensors. Most of today's digital still and video cameras use a color filter array (CFA), where alternating sensors are covered by different colored filters. The most commonly used pattern in color cameras today is the Bayer pattern (Bayer 1976), which places green filters over half of the sensors (in a checkerboard pattern), and red and blue filters over the remaining ones. Usually, there are as many green filters as blue and red combined, because the luminance signal is determined mainly by the green values.The different signals are then interpolated to create the final image.

2.2 Other Color Space Properties and Conversions

Opponent RGB

Opponent RGB comes from the opponent process theory which specifies how information from the cones in the eyes is processed. The primaries are based on the 3 psychological opponent axes (white-black, red-green, and yellow-blue). These three channels correspond to the L*a*b color space where L (lightness) is the black-white channel and a and b correspond to red/green and blue/yellow. Opponent is useful for graphics applications as it increases speed.

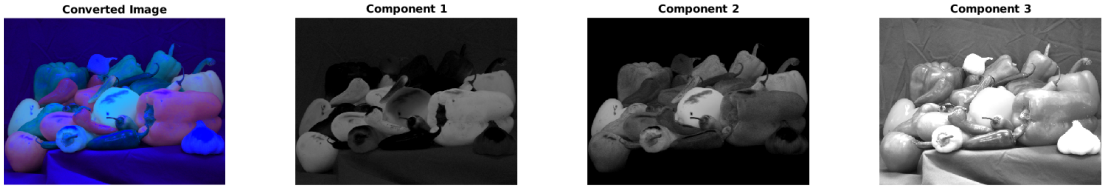


Figure 11: RGB image converted to opponent RGB

Normalized RGB

The normalized RGB (rgb) color system aims to deal with the distortions which are caused by shadows. It is computed by dividing each channel by the sum of all the channels. The r, g, b points only depend on the ratio of the R, G and B coordinates, so they are insensitive to surface orientation to the light source. This color conversion is often used for segmentation tasks as it gets rid of the shadows and clearly displays the borders of the items. Normalizing the values is also useful for deep learning applications as it reduces the scale of the pixel values. As can be seen in Figure 12 the shadows have disappeared and the shape of the items is much easier to see.

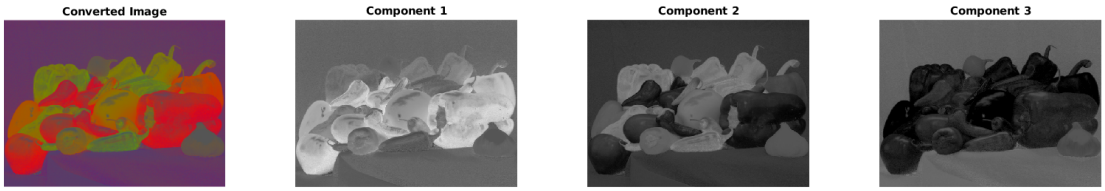


Figure 12: RGB image converted to rgb (normalized RGB)

HSV

HSV is another color system which is even easier for humans to understand. It remaps RGB values to H(ue), S(aturation) and V(alue). The dimensions of this system are independent of each other, so for example if the Value is 0% the image is going to be black regardless of the H and S. Similarly, if the Saturation is set to 0%, the Hue does not matter. Because the Hue is cicular, the color system is illustrated as a cylinder. HSV (hue, saturation, value), also known as HSB (hue, saturation, brightness), is often used by artists because it is often more natural to think about a color in terms of hue and saturation than in terms of additive or subtractive color components.

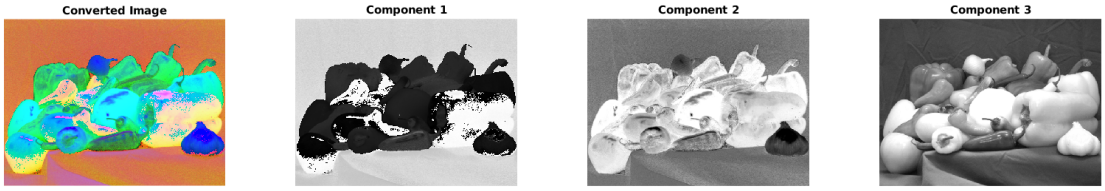


Figure 13: RGB image converted to HSV

YCbCr

YCbCr Color Space is a color space which represents color using luminance and chrominance. The luminance is contained by the Y channel while the chrominance information is stored as the other two components Cb - the difference between the blue component and a reference value and Cr - the difference between the red component and a reference value. Additionally, this color system is very similar to how the rods and the cones in the human eyes work and is widely used for digital video. YCbCr is used widely in video and image compression schemes such as MPEG and JPEG.

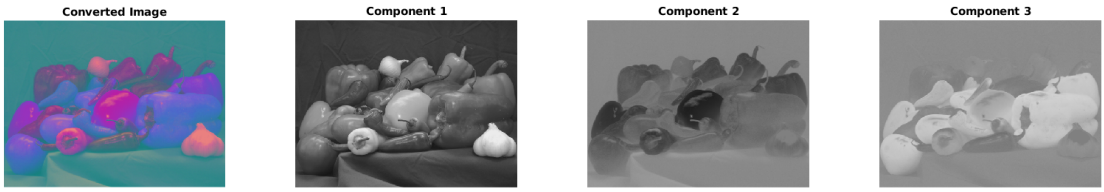


Figure 14: RGB image converted to YCbCr

Grayscale

Grayscale only contains information about luminance and has no information about color. Each pixel is only represented by one value which shows how white/black it is. There are different algorithms for converting an RGB image to gray scale and some of them are displayed on Figure. 15.

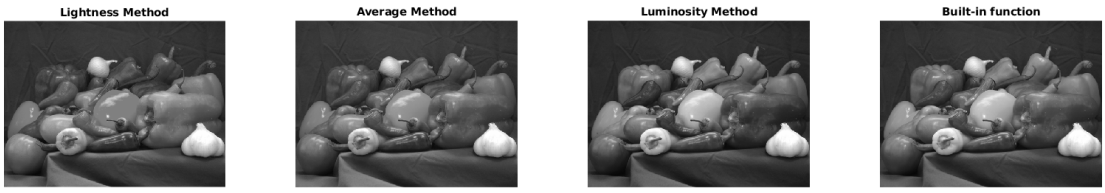


Figure 15: RGB image converted to grayscale using 4 different methods

Mathlab has a built-in function to create a grayscale image from an RGB image. It forms a weighted sum of the RGB components according to the formula:

$$0.2989 * R + 0.5870 * G + 0.1140 * B$$

From Figure 15 it can be seen that for this picture they perform almost the same. The most noticeable difference is for the luminosity method that produced a darker image. For comparison, the luminosity method and the built-in MATLAB function have similar approaches (weighing each channel) so their performance seems to be most similar. The first two methods rely on averaging the values of the channels so these approaches also have similar performance.

CMYK

CMYK stands for cyan, magenta, yellow and black. To avoid confusion with RGB and its blue, the letter that represents black is K. Unlike RGB, CMYK is a subtractive color model. That means that the more colors are blended, the darker the result gets. It is used in printing, where the background (paper for example) is white and colors are mixed on top of it. To produce the real dark tones, black has been introduced as a separate color, which also results in the decrease in the usage of ink in the printers.

3 Intrinsic Image Decomposition

In intrinsic image decomposition the image can be separated in its components. In this section we focus on the decomposition of the formation elements albedo and shading. Furthermore, later on in this section recoloring of an image will be applied with the intrinsic image components.

3.1 Other Intrinsic Components

Different from the albedo and shading, other intrinsic components are distance, reflection, orientation and illumination (Barrow and Tenenbaum, 1978). These components can be composed in individual images from the traditional image. The distance image would describe the distance from the viewpoint to the objects in the image, in which brighter points have smaller distance to the camera. The orientation provides information of the surface at each point in the image by visualizing the surface normal. In fact, this is the gradient of the distance image (Barrow and Tenenbaum, 1978). To provide information of the direction of the light source, the illumination component is used. This intrinsic provides information of surfaces in the image to be shadow. Altogether, these intrinsic images can provide enough image information to combine the individual components for a reconstruction of a simple image.

For example, take an image of a ball. The distance image would provide information of shape of the ball as the contours are visualized by this intrinsic and also the background can be distinguished from the object. Alongside, the albedo or reflectance shows the different colour reflections per point in the image. The orientation or vector gives the orientation of the normal of the surface, indicating the direction of reflection on each surface. At last, the illumination (direction) provides useful information of the darker areas in the images which are due to shadow. For example, if the light source was slightly more right than the camera position, the image would contain a shadow left from the ball which is visualized by the illumination.

All in all, this example shows that this image decomposition approach works for simple circumstances.

3.2 Synthetic Images

Almost all image decomposition datasets are synthetic. To be able to test the performance of the image decomposition algorithms, comparison between the estimation and its ground-truth is desired. The ground-truth of a synthetic image is more straight-forward, since the formation components are known from the synthesis of the image.

3.3 Image Formation

The result of the reconstruction of the image can be seen in Figure 16. In script **idd_image_formation.m** it can be seen how an element-wise product is applied between the albedo and the shading of the original image. By combining the reflectance of the albedo and illumination of the shading, a reconstruction can be formed. In this approach of decomposing an image, it is assumed that the object surface is diffuse, there is linear sensor response and has narrow band filters.

Figure 16: The reconstruction of the original image by its intrinsic components (albedo and shading). Element-wise multiplication of the albedo and shading result in the reconstructed image on the far right.

3.4 Recoloring

The albedo, or color reflectance, of an image determines the color of a spatial coordinate (x, y) within an image. For color adjustment, the albedo can be changed to the desired RGB space. To find out the RGB space of the ball in the *ball.png* image, which is uniform in this case, we extract the maximum values of each color matrix in the third dimension of the albedo matrix. This results in the following rgb-coordinates; $\mathbf{r} = \mathbf{184}$, $\mathbf{g} = \mathbf{141}$, $\mathbf{b} = \mathbf{108}$.

In the **recoloring.m** script, the color of the ball is adjusted to pure green (0,255,0). The values (184,141,108) of the matrices in the third dimension of the albedo-matrix are substituted by the rgb-values of pure green (0,255,0). Subsequently, the element-wise product of the albedo and shading is applied to compose the new image. The result of the recoloring is presented in figure 17.

Figure 17: The result of the recoloring to pure green is shown with on the left the original image and on the right the recolored outcome.

Although the ball has been recolored, the pure green color is not displayed and therefore the color distribution does not seem to be uniform. This is because the composition of the image is not only based on the albedo but also on the shading (illumination). The shading image shows nonuniform values for the ball, which results in a nonuniform color of the ball after the element-wise product between the albedo and shading.

4 Color Constancy

Color constancy algorithms aim to provide a light source correction of the original image so that the image appears to be taken with a white light source. The Gray-World Algorithm is such a color

constancy algorithm. It is based on the calculation of the average color for each channel, which is then compared to gray to normalize for the difference.

The condition for this approach is that the colours in the image are balanced (Agarwal, Abidi, Koschan, and Abidi, 2006). The algorithm does not work well in case the color distribution is imbalanced. In this case the algorithm would overcompensate for the color imbalance to achieve an average grey image. For example, in the case of an image of the ocean. In this case, the algorithm is overcompensating for the imbalanced presence of blue which leads to an incorrect estimation of the illuminant.

Figure 18: Gray-World algorithm results

Another color constancy algorithm is the Scale-by-max algorithm (Agarwal et al., 2006) In this approach the estimation of the light source is executed by observing the RGB-values of the brightest pixels. The brightest pixels of the image are assumed to be representing a white object, and can be used to derive the illuminant light source. This method only works under the condition of an achromatic patch being present in the image.

Conclusion

In this report we first implemented photometric stereo on several models, from which we conclude the algorithm works best on Lambertian objects opposed to objects with a more directional diffuse. On top of that, we found there to be a positive relationship between the number of pictures used and the accuracy of the generated albedo image. This added value from using more pictures is highest for objects with a complex shape.

As a part of the color spaces exercise we explored how to convert an RGB image to different color spaces and we observed how it changes when these conversions are made. This allowed us to dive into the properties of each system and understand better how they describe images.

Also, we have seen that an image can be decomposed in intrinsic image components. With a synthetic data set of simple images, reconstruction and recoloring of images have been performed. In the last part of this report, two color constancy algorithms have been compared, where we came to the conclusion that the Grey-World algorithm fails for imbalanced color distributions and the Scale-by-max algorithm for an image without a white patch.

References

Agarwal, V., Abidi, B., Koschan, A., & Abidi, M. (2006). An overview of color constancy algorithms. *Journal of Pattern Recognition Research*, *1*, 42–54. doi:10.13176/11.9

Barrow, H. & Tenenbaum, J. (1978). Recovering intrinsic scene characteristics from images. *Computer Vision Systems*, 3–26.

Forsyth, D. A. & Ponce, J. (2002). *Computer vision: A modern approach*. Prentice Hall Professional Technical Reference.