# Computer Vision 1: Photometric Stereo

Arsen Sheverdin, Mátyás Schubert, Ard Snijders

September 14, 2020

## Introduction

Cameras are able to capture the 3 dimensional world and transform it into 2 dimensional pictures. In order to make artificial agents understand the objects they see through cameras, sometimes it is necessary to recover their original, 3D shape. Capturing the object from several angles might be the obvious solution, but sometimes it's easier to leave the object and the camera and instead capture the object under various illuminations. From the information of different lighting and the differences between the pictures, we can reconstruct the albedo and surface normal for a model. Using this information we can estimate the shape of various objects. Furthermore, we explore a variety of colour spaces, discussing their properties, differences, and practical use cases. We also discuss the topic of intrinsic image decomposition, and perform a simple experiment on synthethic data. Finally, we explore colour constancy by covering the grey-world algorithm.

## Photometric Stereo

If we want to understand the shape of an object, we first have to understand its surface. Albedo and normal values can be computed from the illumination property [1] (or properties for more pictures) and the matrix of image brightnesses for each pixel the image (or images). With this data we can estimate the albedo values of a surface for each pixel by solving a linear system.

The results for running the algorithm on the SphereGray5 folder can be seen on figure 1. Compared to the result in the provided material [1], we can see a difference in albedo and normal on the edges of the sphere compared to the middle. Possibly, from this we could assume that the sphere is not completely flat, but a bit curved.

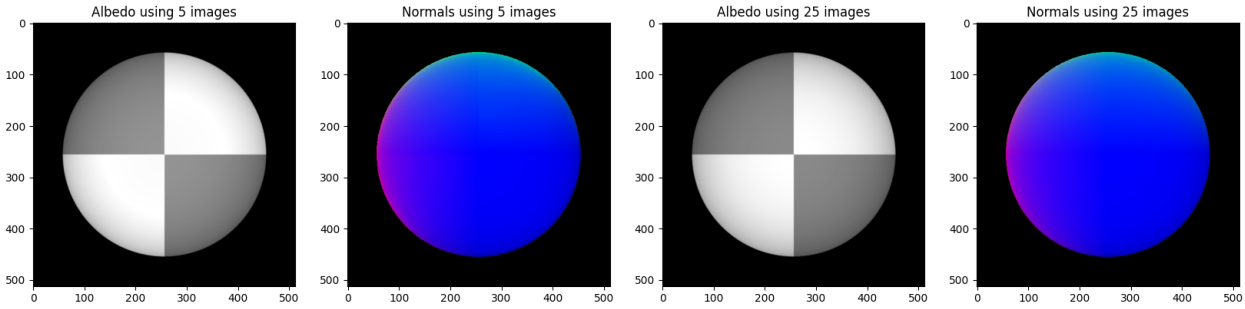


Figure 1: Albedo and normals estimation for different number of images

The albedo and surface normals for an image can be estimated from a single picture, however it is better to use more than three for a correct least-squares solution. Naturally, it is best to use as much pictures as possible with different angles of illumination. On figure 1 we can also see the estimations using 25 images.

Although the difference is subtle compared to 5 images, we can see that the edges of the object have a smoother colour gradient now, meaning a more accurate estimation. It is easy to spot that the vertical line is gone from the top part of the sphere. Achieving such an observable difference is only possible when comparing the results from 5 images with the results of five times more images at once.

Different positions of light might cause different parts of the image to remain in shadow. This means pixels which are in shadow have a 0 value for the corresponding picture. These 0 values can cause albedo estimation to give inaccurate results. In order to prevent these values to affect the computation, we create a diagonal matrix out of the intensity values for each pixel and multiply the left side of both sides of the equation. With this, these zero values are no longer impacting the estimated albedo value of the pixel. This trick brings a significant improvement when used on a small amount of images. The difference between normals can be clearly seen on figure 2. Without this trick for a small amount of images, artefacts appear in the estimation which are gone when using it. There is less difference when working with higher amount

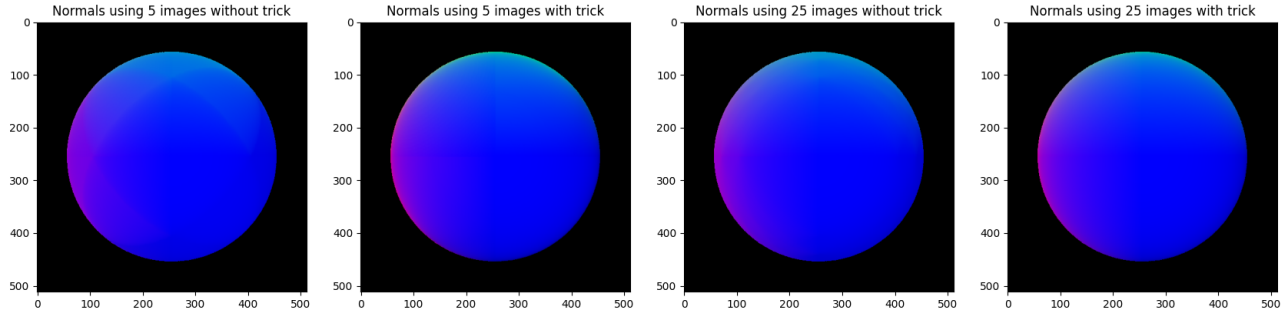


Figure 2: Normals estimated from 5 pictures when using the trick (left) and without (right)

of pictures. However there are still subtle artifacts when not using the trick. On the figure we can see that the estimation for more pictures without the trick still contains (barely) visible artifacts. These results show that eliminating the impact of shadowed points can improve albedo estimation even if we are working with a higher amount of input pictures.

With normals for a given object, we can calculate its surface height map by determining the slope of each axis for every point. Differences between second derivatives of these two slopes should be zero for each point. However because of errors in the estimations of numerical methods, we have to allow a little deviation from this. We should still check if this value is under a given threshold for most points. To choose such a threshold, we have to make an educated guess by analyzing these differences for 5 pictures. On figure 3 we can see how much of these differences are larger than different thresholds. The average is very close to 0 and only around 1.75% of the values are larger this. We can see a spike in percentages under 0.2. Zooming in on this plot tells us that this spike starts around 0.02, which means most of our differences are below this threshold. When using 5 images, 0.95% of differences go over this threshold of 0.02. Using 25 images this drops to 0.82%. On figure 4 we can see how using different number of images changes the percentages for different thresholds. It is clearly visible that using more pictures results in lower differences. Using these slope values, we can reconstruct the shape of the object by computing the height of each point one by one based on their neighbours starting from the top left pixel which we assign zero height. Both column-major and row-major order height map construction provide us with a map which has visible lines of steepness spikes all over it. This happens because we computer each column (or row) separately which results in considerable differences between points which are next to each other in a row but in different columns (or vice versa). These lines can be observed on figure 5. These errors are not completely gone when using averages but they impact the map less since neighbouring values both in a row and a column should have similar values now.

The number of pictures used to construct the height map greatly impacts the quality of the map. Figure 6 shows how much more errors and value spikes are in the height map created with only a handful of images compared to the one with several times more.

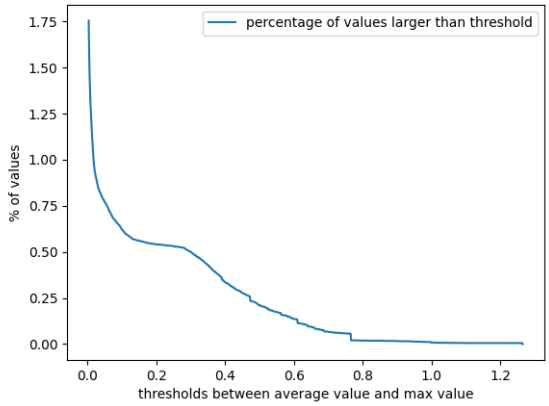


Figure 3: Percentage of differences larger than different thresholds

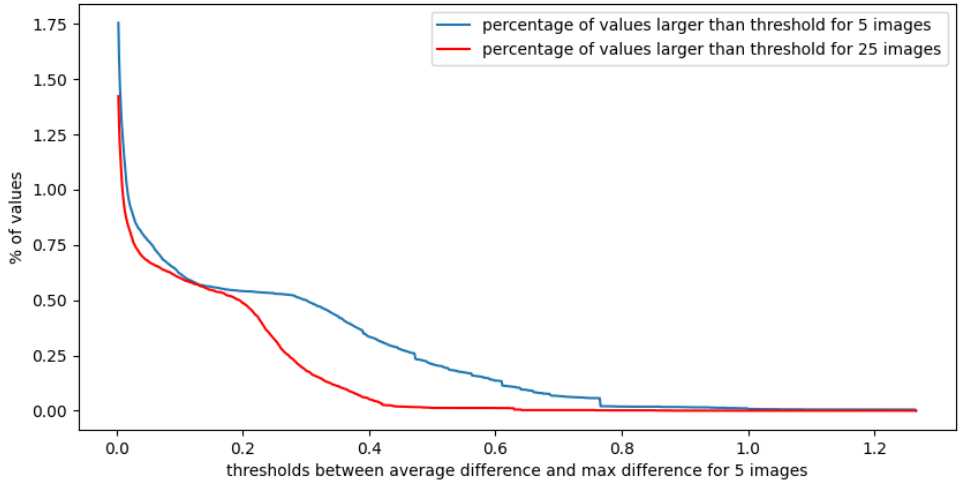


Figure 4: Comparison for percentages of differences larger than different thresholds between using different number of pictures

We can see how the normal directions and slope intensity changes between the edges of the object and the center on figure 7.

## Experiments with different objects

To get a better sense of the algorithm's generalisation we perform experiments on a variety of synthetic and real data. First, we find that obtaining albedo and surface normals seem more difficult for complex objects, as can be seen in figure 8; we observe only subtle improvements for increasing the amount of input images, and even at 121 images there is still a considerable degree of shading. Possibly, this is due to the geometric complexity of the monkey, making it more difficult to eliminate shading.

We also explore the degree to which the algorithm performs when incorporating colour. We can expand the algorithm to work in full-colour by 1) computing separate albedo's for each colour channel (and then

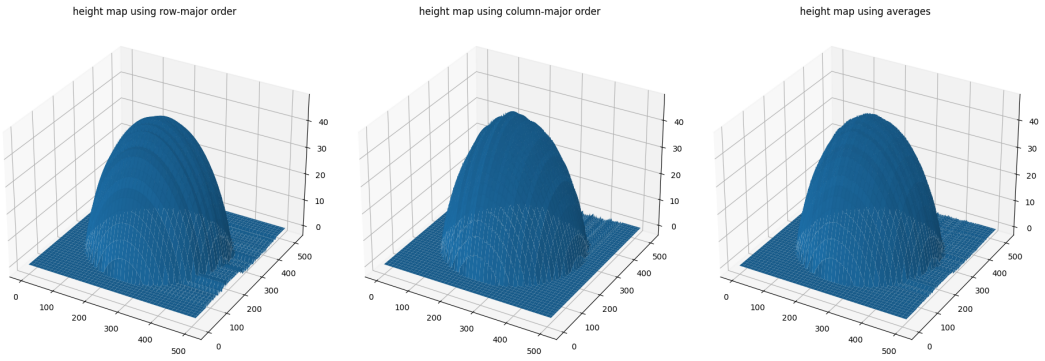


Figure 5: Height maps constructed using different techniques

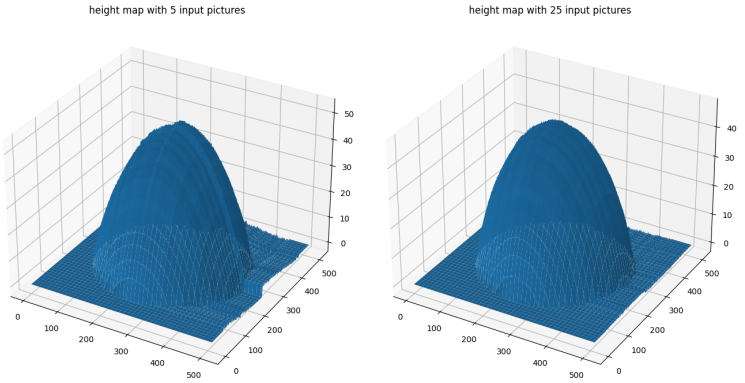


Figure 6: Height maps constructed using different number of images

stacking these across the 3rd axis into a single albedo) and 2) computing separate surface normals and taking the average across channels, for $x$, $y$ and $z$. Results can be seen in figure 9. While the albedos can be recovered accurately, computation of the surface normals seems to be sensitive to stark colour differences - for the coloured monkey, the checkered colour difference gives rise to inaccurate, 'striped' normals (figure 9).

Finally, we assess how the algorithm performs on natural images of human faces. Results for row-wise, path-wise and average integration can be seen in figure 11. These images violate several assumptions of the shape-from-shading model: faces do not exhibit Lambertian reflectance, and do not have a smooth surface; possibly, these images were also captured with some global illumination present [2]. Row-wise integration produces the most stable height map; with column-wise (and subsequently, averaging) integration, noticeable artefacts appear particularly around the eyes. Possibly, this is due to bright, white reflections in the eyes, which are present on most images; this would explain the sudden 'perceived' change in height, producing the observed distortions. Note: we chose to discard many of the original images as they were poorly lit and/or visibly glitched, which resulted in considerably poorer height maps.

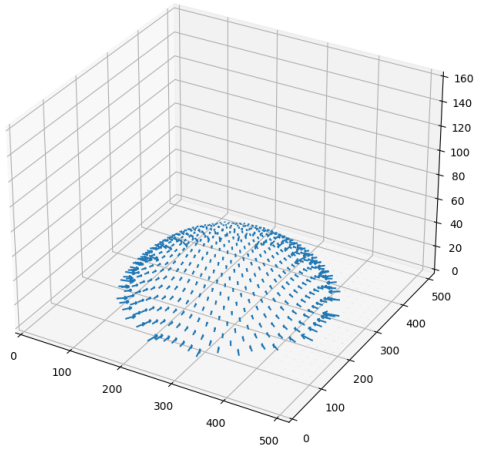

Figure 7: Quiver plot showing the directions and intensities of normals

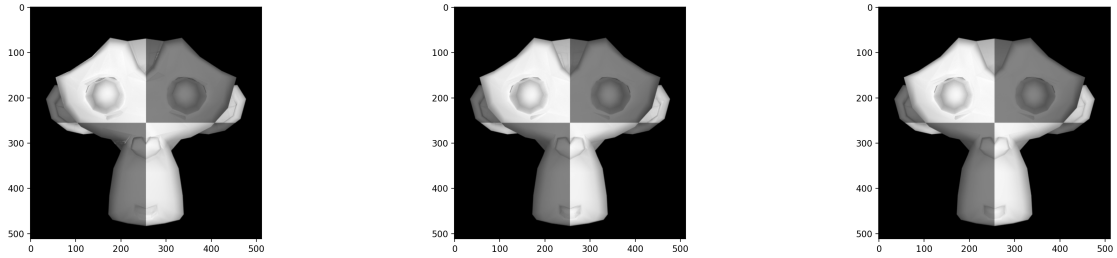

Figure 8: Albedo's generated for GreyMonkey with 4, 41, and 121 input images


Figure 9: Albedo and normals for coloured sphere and monkey.

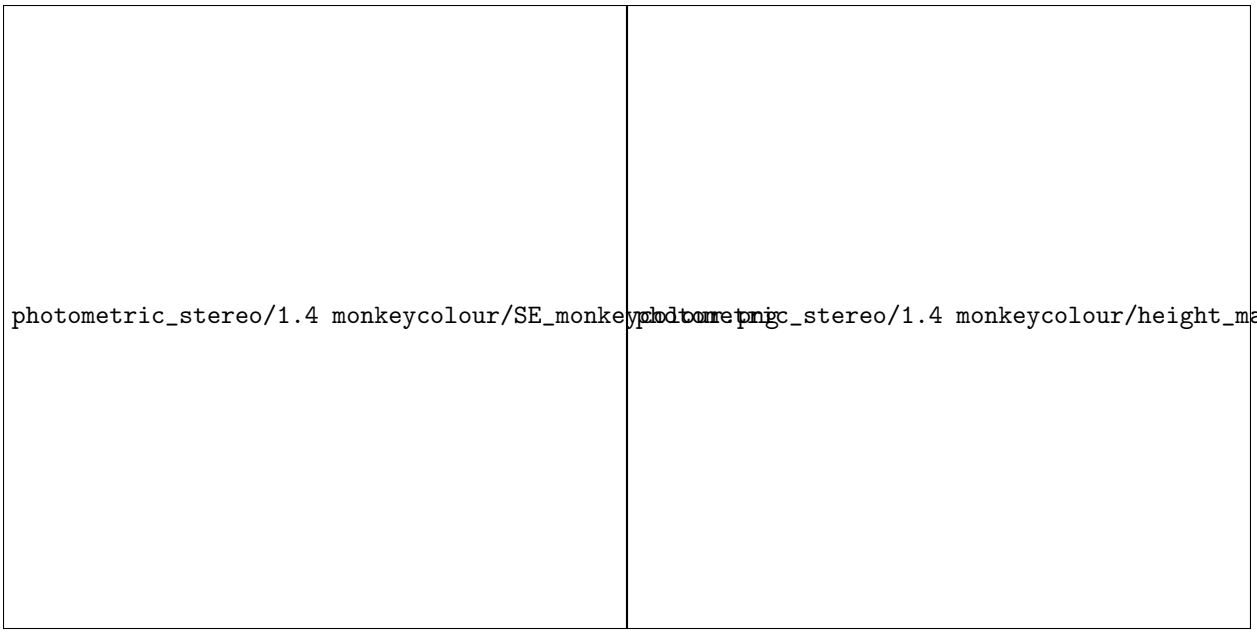


Figure 10: SE and height maps for coloured monkey.

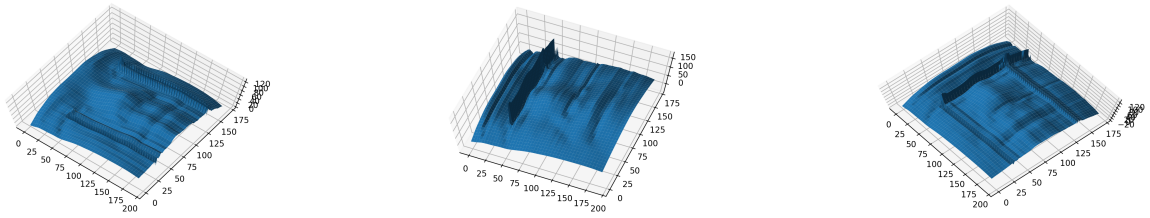

Figure 11: Yale face height maps. From left to right: row-wise, column-wise and average path integration.

# Colour Spaces

We also explore various colour spaces. The most conventional colour model is arguably RGB. This colour model, which is predominant in both digital camera technology and photography, stems from the way humans have evolved to perceive colour. There are two main theories that model human colour perception: trichromatic theory and the opponent process theory. Trichromatic theory, also known as Young–Helmholtz theory, states that three different types of cones in human's retina are responsible for different colour perception; particularly: red, green and blue. A standard digital camera captures RGB colour images in a similar fashion, by capturing the different intensities of red, green and blue light by use of separably filtered image sensors, after which the image processor can re-assemble the various intensities to approximate the 'true' colour per pixel.

## Opponent

The opponent colour model has three components: $O_1$, a luminance component; $O_2$, a red-green channel, and $O_3$, a blue-yellow channel. Opponent theory argues that human visual system interprets colours in an adversarial way, based on differences between colours: red vs. green, blue vs. yellow, black vs. white. This

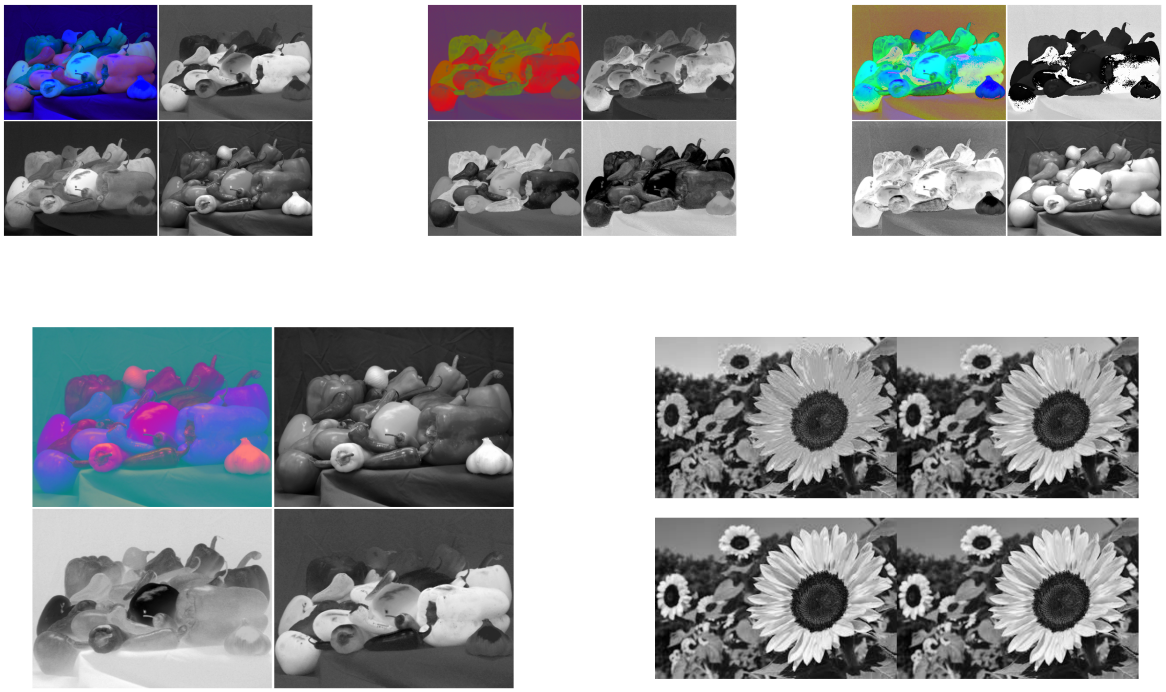

Figure 12: Various Colour Spaces. In clockwise order, from top-left: opponent, normalized RGB, HSV, $YC_bC_r$ and grayscale. The grayscale colour spaces are (in clockwise order from top-left): Lightness, Average, CV2, Luminosity.

theory builds on the hypothesis that human colour perception hinges on two chromatic axes; red-green and yellow-blue. As can be seen in figure 12, conversion with the opponent space yields images with a high degree of distinguishability between differently coloured objects.

## Normalized RGB

The way colours are captured in an image depends both on the colours of objects and the illumination. The normalized RGB space builds on the notion of normalizing colours to compensate for lighting conditions; this can be useful in a variety of computer vision settings, such as object detection. Normalized RGB means that for each pixel, one divides RGB value (of each channel) by the sum of values across all 3 channels. As can be seen in figure 12, converting to normalized RGB produces starker images with a greater degree of segmentation between objects.

## HSV

HSV (hue, saturation, value) model represents colours of each hue in a radial slice of a cylinder, where they change with height of a cylinder, which ranges from black at the bottom to white at the top. Saturation dimension of HSV model is responsible for the control of how much colour is used. Meaning 100% would mean purest colour possible, whereas 0% results in greyscale image. Value is a parameter which controls how bright the colour is. If the values are set to 0%, then the image will be black, and if 100% is set, there will be no black colours in the image. A particular usecase of HSV is for colour selection in web applications; HSV is also a dominant colour space in digital image editting software. HSV is better suited for colour-based

image detection and segmentation tasks, compared to RGB. The reason is that it effectively separates colour or hue from the saturation, thus, helping to distinguish one colour set from another, excluding the effect of saturation and false illumination.

### YCbCr

YCbCr represents colour not as a combination of red, green and blue but in terms of Luma (intensity), Chroma Blue (blue-difference) and Chroma Red (red-difference). This colour space builds on the idea that humans are more sensitive to light intensity, and less to colour. This has practical uses in for instance storage and transmission of large amounts of image data, as we can afford to degrade the less important chroma components to lower the total information content, whilst retaining a relatively high degree of perceptibility via the Luma channel. This can be observed in 12, where the Y component alone captures the scene in high detail.

### Grayscale

For the grayscale colour spaces, we employ 4 different methods: Lightness, Average, Luminosity and the built-in CV2 method. The lightness method averages the most prominent and least prominent colours, reducing contrast, while the average method just averages values. The luminosity method takes into account human perception, employing a weighted function with an emphasis on greens - The CV2 method is somewhat similar, albeit with different weights ($0.299R + 0.587G + 0.114B$). The luminosity method generally achieves the most transparent conversion, but there are cases where other methods are more effective. This colour scheme provides an advantage over RGB in some tasks of edge detection. If the colour does not affect it significantly, reducing to grayscale helps to solve the same problem with less amount of data. For example in Figure 12, all edges are visible even after the image was converted to gray scale.

### CIELAB

Another colour space which was found in literature concerns the CIELAB colour space [4]. This colour space was designed to be perceptually uniform, meaning that equal changes should produce equally noticable differences. It defines colour using $L^*$ for brightness, $a^*$ denoting green to red, and $b^*$ from blue to yellow. In practice, this colour space is used to convert imagery from RGB to CMYK (a common colour space for printing) prior to printing, as it includes the complete subset of colours of both colour spaces.

## Intrinsic Image Decomposition

The study of image intrinsics concerns the components an image can be decomposed into. In principle, there are no other components in which an image can be decomposed other than its albedo (the colour reflecting of an object as a property of its surface) and its shading (the illumination of the scene). However, arguably one could extend the notion of an intrinsic by separating shading itself into the shape of an object, its texture, and the illumination of the scene. Extracting intrinsics from real-life scenes is an inherently underdetermined problem, making the collection of natural image intrinsics a costly process. With synthetic imagery researchers can control for a variety of conditions, thereby allowing for straight-forward extraction of ground-truth intrinsics. we explore the process of image decomposition in such a synthetic setting. The intensity at each pixel can be approximated as the element-wise product of the image albedo $R(\overrightarrow{x})$ and shading $S(\overrightarrow{x})$, such that

$$I(\overrightarrow{x}) = R(\overrightarrow{x}) \times S(\overrightarrow{x})$$

With this knowledge, we can reconstruct a synthetic image of a ball by combining its albedo and shading intrinsics accordingly, as can be seen in figure 13. We can also recolour the ball from its original beige $[184, 141, 108)$ to a pure green, by multiplying the shading intrinsic with a vector $(0, 255, 0)$. It can be observed in figure 13 that the recoloured ball's green hue is not as saturated as the initial albedo; this is due to

the geometry of the object (causing the bottom of the ball to be more shadowy) and the overall illumination of the scene (which in this case suggests a darkly casted image).

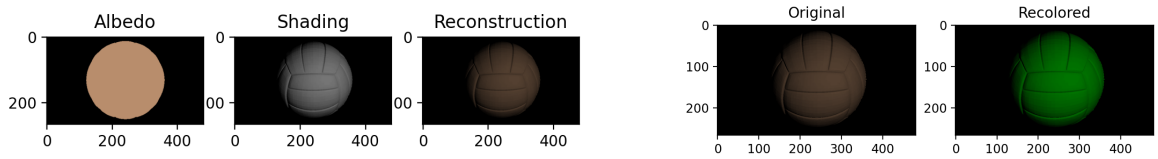


Figure 13: Reconstruction and recolouring via image intrinsics.

## Colour Constancy

Apart from estimating the true shape, we can also estimate the true colour of objects. While we might be able to observe an apple as red under most lighting conditions, a camera might fail to do so. For this reason several colour normalization algorithms have been created which try to correct the image so that objects are represented with their true colour. The Grey-World algorithm's purpose is to make the average colour of the picture grey, by correcting each pixel according to the average of each colour channel. The algorithm works very well for colourful pictures where each channel is represented but they have unrealistic values, such as too much of a colour on the picture or not enough compared to the target average. We can see an example of the algorithm working expected on figure 14. The original picture has too high values in the red channel which makes the scene look unrealistic. The corrected image has these red colours suppressed which result a more natural looking scene.

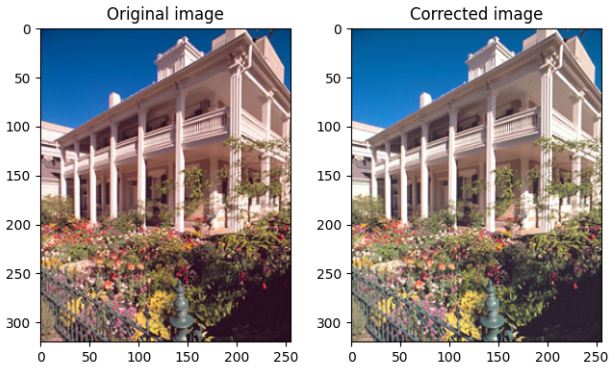


Figure 14: Example of Grey-World algorithm working well for image correction

Unfortunately the assumption for Grey-World algorithm does not always work. Some scenes have a colour balance which does not result in a grey average. An example for such case can be seen on figure 15. The problem is that red is not only underrepresented on the input image but on the expected output image too. The algorithm corrects the lack of red resulting in a darker picture with a red shade all over it which looks just as unrealistic as the input image.

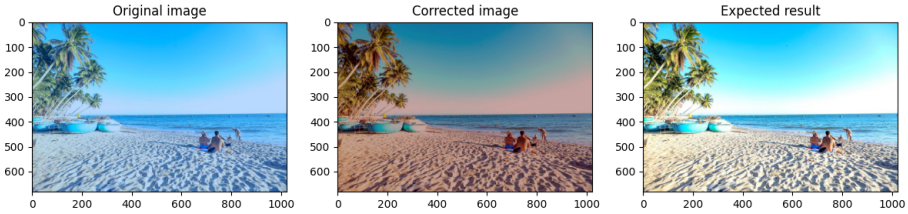


Figure 15: Example of Grey-World algorithm resulting in unrealistic colours

The Grey-World algorithm might not always work but fortunately many more colour normalisation techniques exist. One of them is the von Kries chromatic adaptation method. Chromatic adaptation [3] is an ability of the eye to perceive the same colour for objects under different illuminations. Our eyes achieve this with three types cone cells (L, M and S) sensitive to different wavelengths of light. The Von Kries model corrects the image with the help of a chosen reference white. First, it transforms each pixel by assigning the values of how much it affects these specific wavelengths. Then it scales these values by the reference white's corresponding value. At the end the picture gets transformed back to its original space.

# Conclusion

We successfully estimated the shapes of different objects by analyzing them under different illumination properties. From the difference between pixel intensities for different lighting angles, we could recover normals for a given object. With the method to eliminate noise from shadowed points, we could improve our results significantly. Treating the normal as a function which assigns a value for each $(x, y)$ point, allowed us to compute its partial derivative for the two axes. The height map for the object could be constructed by applying the computed slopes for each pixel compared to its neighbours. We saw that using more input images and lighting conditions can greatly improve the results these algorithms deliver. We then explored a variety of colour spaces, highlighting what set them apart and expanding on some of their use cases in various domains. Next, we demonstrated how in a synthetic setting, it is possible to reconstruct any image as the product of its albedo and shading (assuming these intrinsics are present), and we showed how intrinsics can be manipulated to recolour images. Finally, with the Grey-world algorithm we could easily recover true colours of objects even if the picture had unrealistic colour values. While a camera might be vulnerable to different lighting conditions, colour normalisation algorithms allow us to simulate the ability of our eyes to observe objects in their true colours. This is especially important if we want agents to recognize objects easier.

# References

[1] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach.*

[2] T. Haines. A rough guide to shape from shading, 2012.

[3] Hamed P. Viewing conditions and chromatic adaptation.

[4] J. Schwiegerling. *Field guide to visual and ophthalmic optics.* 2004.