Computer Vision 1

Assignment 1: Photometric Stereo & Colour

Group: 12

Victor Wernet
Sameer Ambekar
Apostolos Panagiotopoulos

Professor:
dr. Shaodi You

September 19, 2021

1 Introduction

In this assignment we are exploring and experimenting with different properties of images, like illumination, shadows and color spaces. The first part regards recovering the 3D shape from multiple 2D images taken with fixed camera and object positions but varying angle of illumination sources. We exploit the difference in illuminations and shadows created in every picture to estimate the normal vector of each point in the original 3D space and then reconstruct an approximate 3D model. In the following parts, we experiment with transforming images to different color spaces that represent the image in different manner, decomposing images into their intrinsic components (which can leads to easier image processing) and improving color accuracy. We also observe the color constancy methods and take a look at current and improvised methods that aim to solve this task similar to how we humans correct the colors.

2 Photometric Stereo

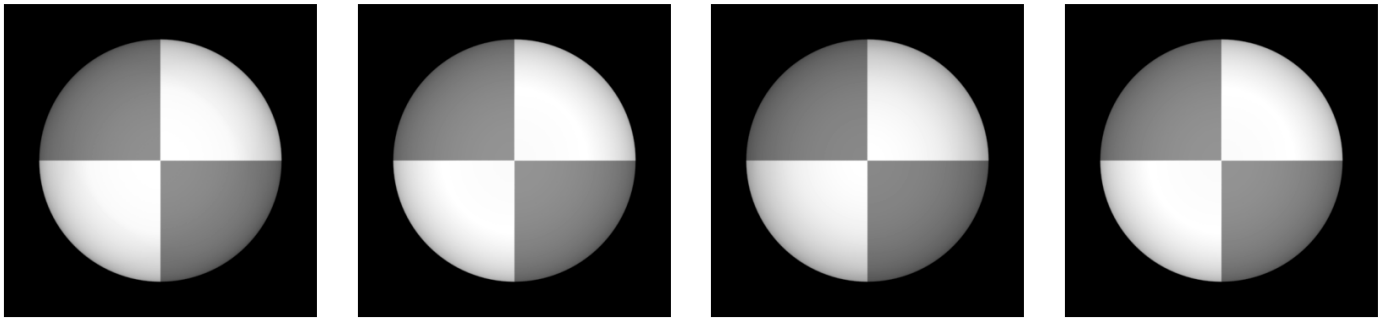
2.1 Estimating Albedo and Surface Normal

2.1.1 Questions

1. The true underlying color of the given pixel at location $i(x, y)$ for a given $M * N$ image without any shading, shadow or lighting. For the example of `SphereGray5` we would expect uniform colour in the 4 quadrants, but instead we still observe some shading.

2. In practice, the minimum number of images needed is $n = 3$. Each image typically involves different directions of illumination, specifically, the slant angles should be chosen between 30° and 60° [1]. These numbers are chosen by dividing $180°/n$. Arguably, to get a robust result, redundancy is needed. Therefore, it's very common to have more than 3 light sources with different directions. The downside of this is that, as n increases, so does the processing time [11, 3].

3. When a region is covered by shadows, the number of available directions of illumination reduces, which can lead to ambiguities for that region. Shadows are essentially dealt with by multiplying both side of this equation $i = V * g(x, y)$ with I, where I is an $N * N$ matrix containing variations in brightness under different sources, which in turn code the shape of the surface on its diagonal. This essentially has the effect of zeroing the contributions from shadowed regions, because the relevant elements of the matrix are zero at points that are in shadow. The more images you have, the more precise the image becomes when it comes to shadows, so in turn you would want to have shadow trick disabled when you have many images (it won't be that useful), since the increase of number of images correlates with robustness [12].



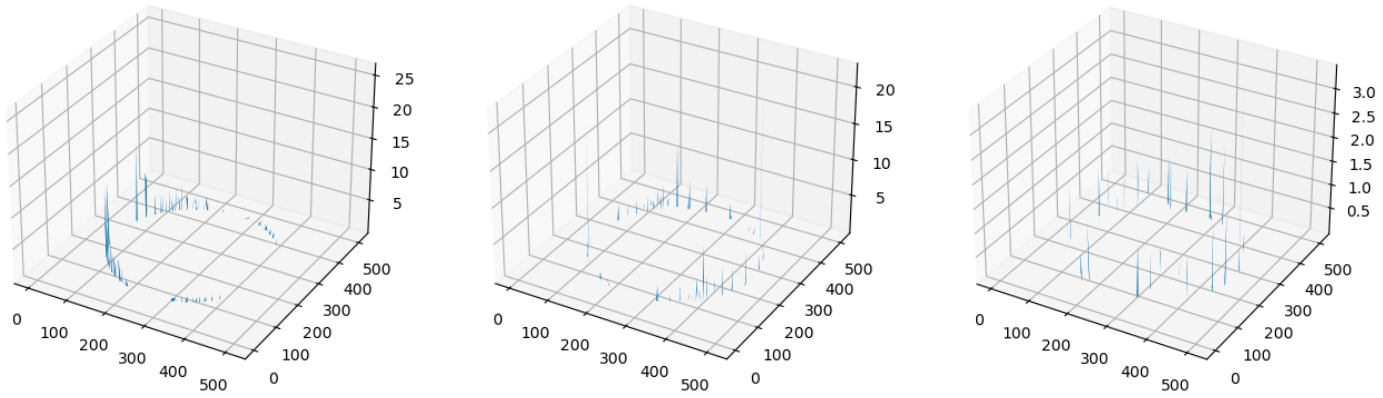
(a) SphereGray5 with shadow trick disabled (b) SphereGray5 with shadow trick enabled (c) SphereGray25 with shadow trick disabled (d) SphereGray25 with shadow trick enabled

Figure 1: Shadow trick. We can see that using the shadowing trick leads to more illuminated (brighter) images.

2.2 Test of Integrability

These errors occur due to ambiguity, and most likely caused by the perspective projection. The test of integrability essentially exploits the relations between the normal map and the derivatives of the depth map. So any time the partial derivatives do not end up being 0, this can be considered an error and hence we can use a threshold to alleviate the problem a bit that essentially filters these out. Below we can see a few experiments with different n images.

[1] `https://www.mvtec.com/doc/halcon/13/en/photometric_stereo.html`

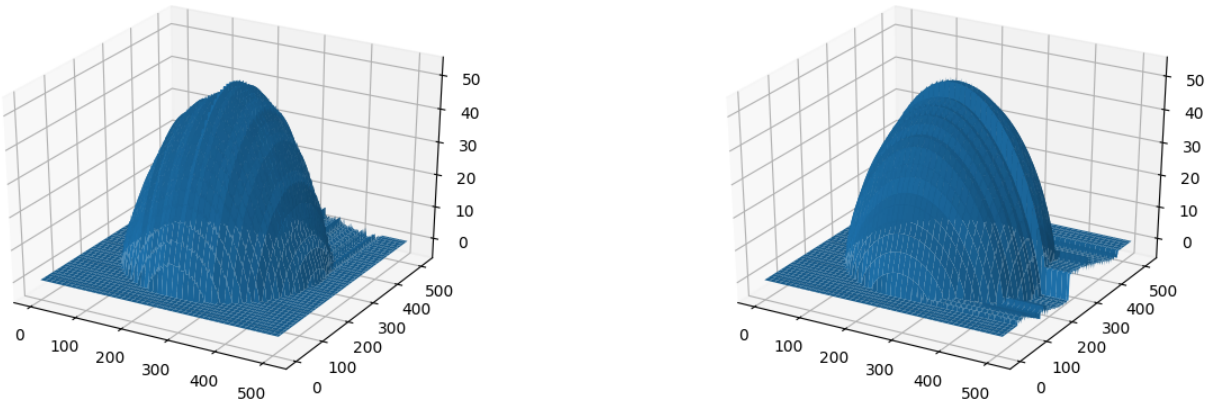


(a) SphereGray with 2 images (b) SphereGray with 5 images (c) SphereGray with 25 images

Figure 2: The Test of Integrability with shadow trick enabled

2.3 Shape by Integration

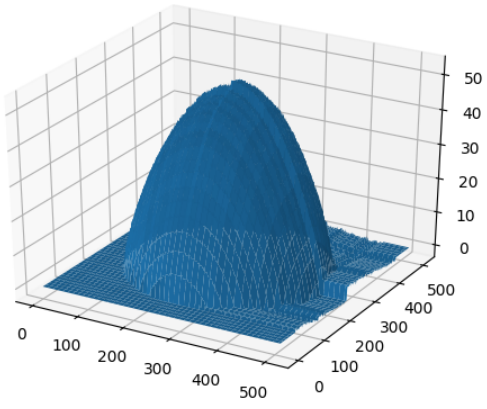
1. Below in Image 3 we can see visually the difference between using path type column and row respectively. Essentially the difference between the two types boils down to either 1) summing the first column and then summing each individual row, or 2) summing over the first row, and then summing each individual column respectively. The main reason for doing this is due to the fact that we are dealing with discrete values, hence we can simply sum the values at of the previous neighboring pixel $i(x, y)$ in order to move the summation along as we updated each pixel accordingly based on the path type we must walk. This then reconstructs (or approximates) the shape of the object in question, by walking along the path and calculating the height, and saving it.



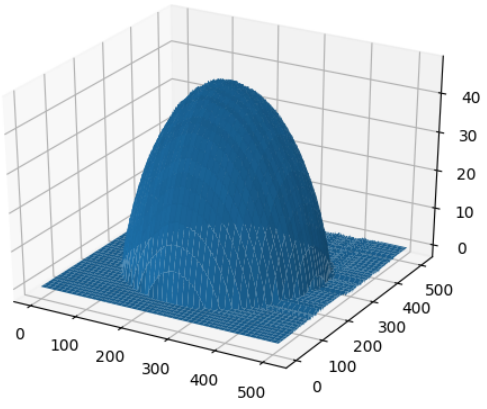
(a) Path type "column" (b) Path type "row"

Figure 3: Reconstructing the surface column-, and row-wise respectively

2. By averaging, we end up summing in both directions, which leads to a more uniform look, or inherently, a smoother visual as number of n images increases. As can be seen in Image 4.



(a) Path type "average" using 5 images

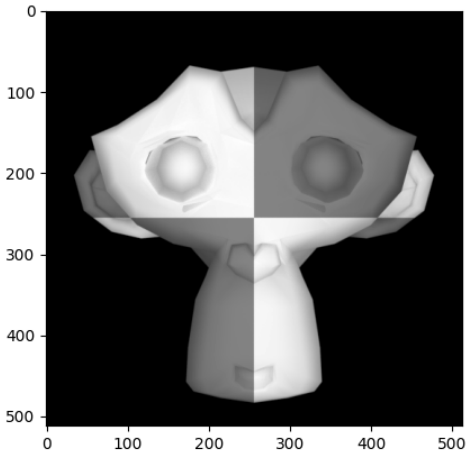


(b) Path type "average" using 25 images

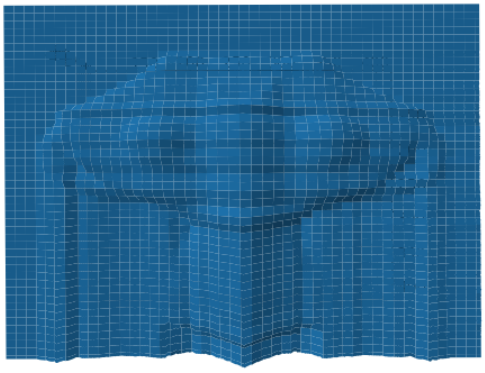
Figure 4: Reconstructing the surface by averaging the sum of columns, and rows

2.4 Experiments with different objects

1. The result for the albedo, and height map result can be seen in Image 12.



(a) Albedo map for MonkeyGray



(b) Height Map for MonkeyGray

Figure 5: MonkeyGray using 121 images

2. .

3. It's clear, from the different path types, that the "row" gives the best result, as seen in Image 7. This is most likely due to the fact that when summing over the rows, the dataset provides more information when it comes to the shadows (or less ambiguity), while the other 2 results suffer from this ambiguity. Also, various of the data samples seem to provide no relevant information regarding the shape of the image, due to the angle of the light causing most regions to be completely dark, which in turn creates many ambiguities and provides no relevant information about the shape. A few of these examples can be seen by images 9, 10, and 11

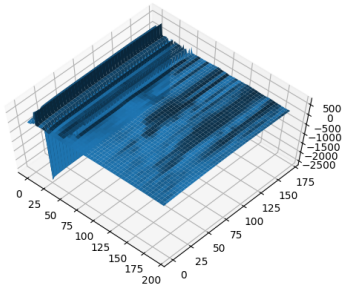


Figure 6: column

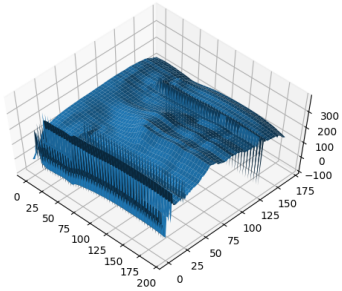


Figure 7: row

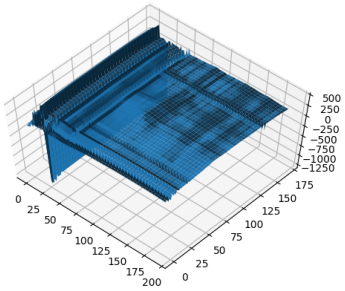


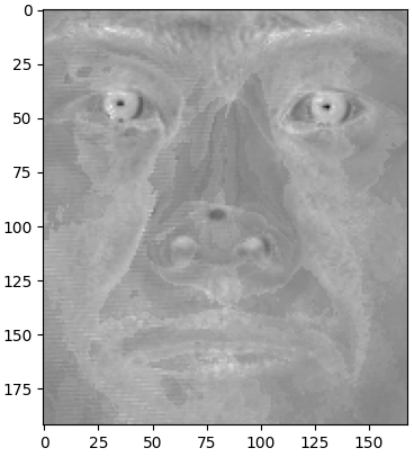
Figure 8: average

Figure 9: P00_Ambient

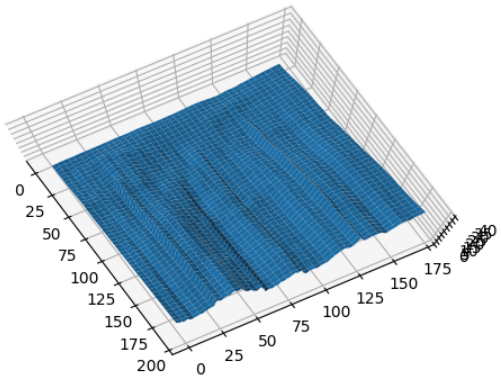
Figure 10: P00A-120E+00

Figure 11: P00A-130E+20

12.



(a) Albedo map for Yale Faces with filtered data samples



(b)

Figure 12: Height map for Yale Face with filtered data samples. Using only 36 images.

3 Color Spaces

We make use of color spaces to represent the colors in different ways. A color space usually consists of a color model and a mapping function [7]. Color spacers help us quantify the colors as mathematical models in the form of numbers. Each color space has its own way of representing the colors. There are many color spaces that are widely used for various image processing/computer vision purposes. Some of them are HSV, LAB, YUV, YCbCr, YPbPr, YIQ, XYZ, HED, LCH and

CMYK [4]. Amongst mentioned, RGB is widely used in many computer vision applications for capturing, saving, processing the images. RGB (Red, Green, Blue) consists of 3 channels 8 bit each and the value of each channel ranges from 0-255.

RGB is widely used for most of the consumer applications such as Cameras, computer screens and is often referred as Industrial standard because it spans a wider range of color tones and allow accurate representation of colors. Additionally, since human eye can usually sense RGB colors in the day-to-day objects, devices like camera and monitor use RGB color space so as to recreate the same color experience/representation our Human eyes see. RGB color space model is used more often where the intention is to generate colors like electronic screens such as monitors, mobile phones, etc. Whereas color spaces like CMYK is wwidely used for printing due to its subtrative based color modelling. Since monitors, mobile, cameras, phone screens are connected to ecosystem of arts, computer graphics, computer vision areas; in order to simplify the things and ensure consistency, and keep the workflow together - RGB color space is used widely.

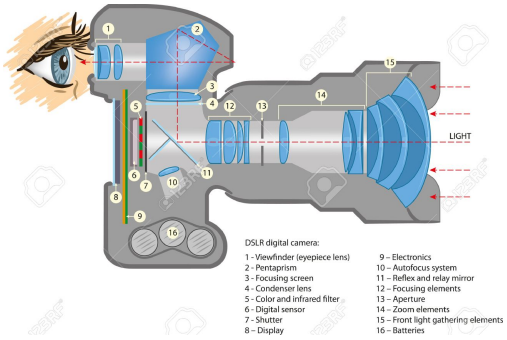


Figure 13: Camera's components

Camera consists of the components as shown in 13. Initially, when the shutter opens itself on clicking the camera shutter button to capture an image, the shutter opens itself to allow light to pass in it for a fixed amount of time (shutter speed) and also opens as wide as it is specified (camera exposure). The object that is placed in front of the camera reflects the light that is falling on it and thus the light passes inside the shutter of the camera which in turn hits the mirror placed in the camera. The light that passes through the aperture could be from an artificial source of light or natural source of light based on the environment. Usually the shutter opens and closes faster than the blink of the human eye. The amount of focus can be controlled by adjusting the lens accordingly through rotating the lens. After the light passes through the mirror placed in the camera, based on if the camera is a traditional camera or a digital camera it may have either a film to record the light that is falling onto it or a CCD sensor that captures light and stores in a form a digital signal. The CCD or the CMOS sensor convert the light to electrons in the form of a 2d image. whenever light strikes the electric sensor it is held as small electric charge in the photo sensor. the charges in line of pixels that are closest to the amplifiers that amplify the output are amplified. this is repeated until all the lines of pixels are covered and amplified ADC converter (Analog to Digital Converter) converts the analog signal in the form of zeros and ones.

Color spaces and their importance:

i) HSV - HSV stands for Hue Saturation Value. The central section represents the Intensity. Hue is defined in the range of (0,2Pi) relative to Red axis and red at angle 0. Green at 2pi/3 and red at 2pi [13]. HSV like other color spaces serves as pre processing step for segmenting of carious real world objects like skin, road, faces as in [6], [10]. HSV was designed in such a manner that it mimics how humans perceive color.

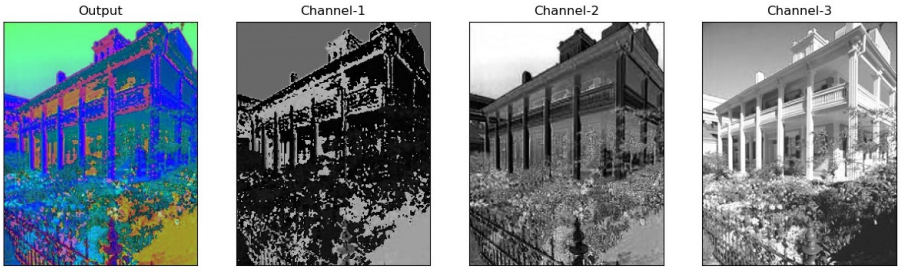


Figure 14: HSV color space and its channel wise representation.

From fig 14 we can observe that we can use it for image processing applications such as thresholding areas that may not be possible in RGB image. In Channel-2 we observe that the building is separated from the background behind it like the sky, thus converting to HSV can serve as a pre-processing for further analysis of the image. HSV can be used for detecting skin from images, [8] performs skin detection effectively using this and image thresholding and prove results empirically.

ii) YCbCr : where Y stands for luma component and Cb and Cr and blue, red difference of chroma components. This color space is widely used in European television studios and for the task of compressing. Since it allows to get ride of unwanted components information it is widely used for compressing widely known video formats such as MP4. The human eye is known to observe intesity changes quickly and comparitively less senesitive towards changes of hue. Therefore, this color space lays more emphasis on light intensity changes than hue changes so as to provide better representation of the images [9].

Figure 15: YCbCr color space and its channel wise representation.

We observe in 15 that the channel wise outputs do a better job in differentiating the objects and scene when compared to RGB. Similar to HSV, we can use image thresholding techniques to segment the objects for the image analysis.

iii) Opponent RGB (oRGB) Since the perception of color is not best represented in RGB color space, we make use of Opponent RGB space (oRGB). It has three components - O1 corresponds to luminance component, O2 corresponds to red-Green Channel and O3 corresponds to blue yellow channel. In the paper [1] they introduce oRGB that can be used for computer graphics extensively. It can also be used for color adjustment, color transfer, other color based applications. However it has limitations such as traditional colorimetric applications. Also, it is not known to be bijection of RGB color space [1].

Figure 16: oRGB color space and its channel wise representation.

iv) Normalized RGB color space It can be used where we want to get rid of distortions visible an image and to reduce the effects of light. Specifically, it can be used in computer vision applications where we to localize/focus on a particular object for subsequent processing.

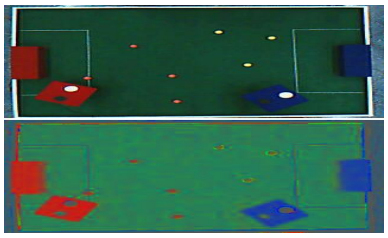


Figure 17: Left: Shows simple RGB image. Right: shows: Image obtained after Norm-RGB.

Shows how Normalized RGB can be useful for scenes where we want to primarily focus on several objects only and get rid of unwanted things. We also observe the color difference between the objects is much more visible now so as to segment the objects using image processing techniques. For instance, if we want to know one of the player's position we can directly threshold just the blue color values and we would obtain the position of that player quickly, without any additional operation. We observe that Norm-RGB helps us perceive more information from the image and depicts exact boundary lines for the objects shown in the Fig 18. Particularly, in computer vision it could be a very useful feature such that it allows us to track an object if its in motion.

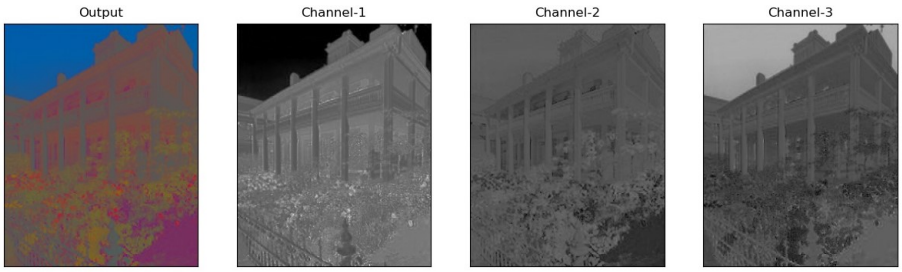


Figure 18: oRGB color space and its channel wise representation.

v) Grayscale color space A greyscale image typically needs only the intensity information. Typically values range from 0 -255 where 0 being the lowest and 255 being the highest (If we assume the scale of 0-255). Since it makes of very less information, accordingly it only uses 1 byte or 8bits to store the information. In popular libraries like OpenCV, grayscale makes use of a 2-dimensional array that is spread across width*height. Any image can be easily converted into grayscale due to its low-information design. Grayscale can be extremely useful for dimension reduction: because of its low-information design when compared to other image spaces like RGB, YCbCr that make use of more bytes of memory for storage. It helps reduce the model complexity: Instead of training the neural network on RGB or any other images, if the task doesnt need the color based information, we make use of Grayscale images for the training, that helps us in the model faster (due to less information and less noise) and even the input layer will accept the image in 1-channel only instead of 3 channels. Edge detection: For the task of edge detectors like sobel, canny they make use of grayscale images primarily to segment the objects base don the pixel values and their own transforms.

vi) CMYK color model - CMYK is widely used for the printing in color. It consists of Cyan,Magenta, Yellow, Key (Black). The color model while printing doesn't mix the color's to obtain the color black instead use a separate black color to represent it. It works by masking the colors when on a white background. It belongs to the class of subtractive color method. It is used to print colors on various objects such as clothes.

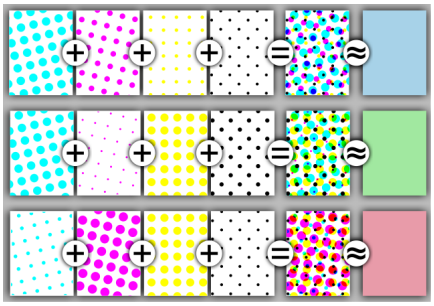


Figure 19: Representation of how colors can be obtained in CMYK color model

From fig 19 we see example of three separate grids, of how human eye would observe the resultant of the individual colors that have been used by taking CMYK color model into consideration. Although while image processing using libraries like OpenCV and softwares like Photoshop we make use of RGB color space primarily to operate, we however use CMYK color space to print them.

4 Intrinsic Image Decomposition

Intrinsic Image Decomposition is the process of separating the image into an illumination-invariant component, the reflectance (albedo), and a illumination-variant component, the shading. Although, lately, research focuses mostly on intrinsic image decomposition, one can decompose an image I into other components, like *structure* (u) and *texture* (v), such that $I = u + v$ [2]. The structure component preserves the geometry of the original object, and hence is suitable for image segmentation and object recognition, while the texture component is free from albedo discontinuities and luminance variations, and hence is more useful for surface analysis.

Decomposing an image into reflectance and shading is a hard visual task, that is often tackled with machine learning algorithms. Most of them are supervised, and hence they rely on ground-truth reflectance and shadowing images. Contrary to most computer vision datasets, for which ground-truth is annotated from the training samples, reflectance and shading ground-truth images require precisely generating values for every pixel, which is impossible for a human. Hence decomposition datasets are either formed by manually painting objects, and hence knowing the reflectance part beforehand, or by constructing synthetic datasets, in which case reflectance and shading can be automatically extracted from the 3D renders.

In this section, we are dealing with a much easier task: reconstructing the initial image from its reflectance and shading. To reconstruct the initial image we simply multiply the albedo and the shading components element-wise. The result for

`ball.png` is presented in Figure 20, and although it is visually identical to the original image, comparing the RGB values per pixel gives an error of $2.53 \cdot 10^{-6}$ per channel, which could be attributed to machine precision.

Figure 20: Original (left) and reconstructed image (right). In the middle we see the albedo and shading components.

The albedo component of the ball contains two values, namely [0, 0, 0] which is the background and [184, 141, 108] which is the true material color of the ball in the RGB space. To recolour the ball image with pure green, we just change the later values of the albedo to [0, 255, 0] and multiply element-wise with the shading component, which produces the recoloured image of Figure 21. Of course, due to the ball's shape and scene lighting, same parts of the ball appear shaded. Hence the ball colour is neither perceived purely green nor uniform.

Figure 21: Recolouring the `ball.png` image from [184, 141, 108] to pure green.

5 Color Constancy

Color constancy has extensive uses in the field of image processing and computer vision. It is a method by which the color is detected independent to the light source in the image. It is the goal of this to achieve illuminant-invariant description of the objects under the given scene. One such algorithm that does this is called Gray-World algorithm.

Gray World Algorithm: It is a classic white balance algorithm and assumes that average of 3 RGB channels must be equal.

The major drawbacks of the gray world algorithm is that it does not take into consideration all the variations of illuminations and its intensity across the whole image. Also, when new objects come into the picture it fails. Specifically in the case of moving video events such as sports events, as soon as there are new objects the overall resultant of the gray world algorithm is messed up. Additionally, whenever there is a large dominant color patch in the image, the gray world algorithms usually fails.

Figure 22: With reference to image from [5] a) contains the original images of 4 different scenes and b) contains the results obtained from gray world algorithm.

Images in the Fig. 22 have been taken from the ColorChecker image library from Simon Fraser University (SFU). Four scenes such as sky scene, grass scene, bright scene and common scene are taken into consideration. In the first three images it can be seen that that wherever there is partial color the gray world does not work as expected. In the first image of sky and the image of grass the gray world algorithm over adjusts the image that change the color tone of the image altogether.

New methods of color constancy:

To improvise the gray world algorithm we can make use of edge detector to segment the objects first and then apply the gray world algorithm based on the extracted regions [5].

[14] this method makes use of an efficient color constancy by making use of gray pixels. It is based on the theory that there are gray pixels in the every image that can be used for computing the color constancy that are atleast visible under the white light. However, since most datasets and images are usually encoded into the RGB space it can be a challenging task to find the grey pixels. Therefore, the paper [14] defines novel Illuminant Invariant Measure (IIM) as a metric to identify the gray pixels in the image. It calculates the IIM by the standard deviation of tiny image patches iteratively using a 3x3 kernel size window and traversed across every channel. When the gray pixels are found out, information can be extracted from them to calculate the color consistency. Fig 23 shows the algorithm that can be used.

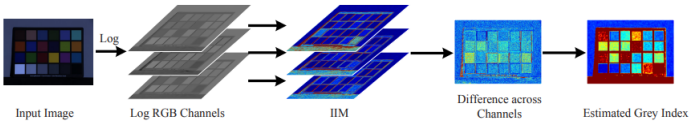


Figure 23: Flowchart of the algorithm proposed in [14]

Empirically, the algorithm is evaluated with state-of-the-art (SoTA) methods that are relevant by using datasets such as SFI indoor dataset and the Color checker dataset to obtain the minimum angular error when compared to other SoTA algorithms. Figure 24 shows the results on the datasets with the angular error.

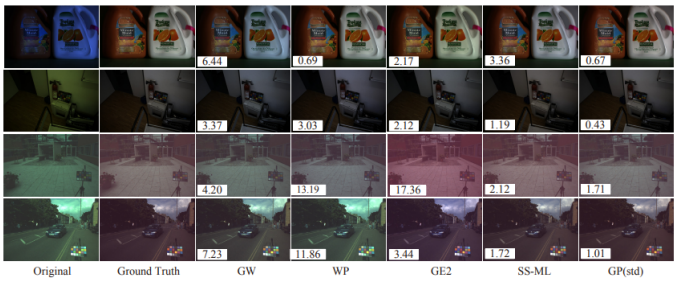


Figure 24: Flowchart of the algorithm proposed in [14]

6 Conclusions

In this report, we used the photometric stereo algorithm to reconstruct 3D objects from a collection of 2D images taken with different angled illumination sources. We noticed that the algorithm is not working accurately, probably due to formulation assumptions that do not hold in the real world. This was clear from the progressively decreasing performance of the 3D reconstruction when we moved from a simple sphere to a monkey and then to a real-world face images. Besides the imperfect results, the algorithm requires an empty background and precise knowledge of the light sources, which make it employable only in a studio set-up. However, this algorithm still provides an easy and comprehensible solution to an otherwise hard problem, and in turn provides more of an approximation algorithm.

Additionally, we observe that in color spaces, although RGB is the most widely used color space it alone does not serve the best for all purposes such as printing the image, thresholding the image, segmenting objects, storing the information, representing the image, transmission of the image. We can easily convert the given RGB image into color spaces like HSV, YCbCr, etc. to achieve a better representation of the objects. Through the experiments with colr spaces we observe how every color space can be viewed in different components that may sometimes directly lead to segmentation of objects via different channels or easy image thresholding processes. This can serve as an important pre-processing step for image analysis using image thresholding techniques, image edge detectors, object segmentation, object localization.

Another useful trick for easier image analysis is decomposing an image to its intrinsic parts, namely reflectance and shadowing. We explored the application of this technique for object recoloring, although we did not see how is such a decomposition perfomed (which is a considerably harder task).

Furthermore, in [4] different color spaces have been made use of to train existing deep neural classification networks and they obtain different accuracies on popular vision datasets like Imagenet, Cifar. By converting to a different color space; Interestingly, some images of several classes led to better visualization/representation because they were converted into different color space that may have aided the learning part for the neural network. Experiments also show that combining several color spaces result sin better accuracy than just using RGB and may make use of too few parameters to obtain SoTA results when compared to existing RGB models that make use of large number of parameters. However, it does require separate compute power and time both to convert to the required color space first and then use them for training the network. In the color constancy, we come across methods that help us in correcting in the images that is similar to the manner we humans correct the colors while perceiving the scene/object infront of us. Gray world algorithm albeit simple does an effective color correction for some images. It fails whenever the image has varied color patches as mentioned in the section, and this drawback is being overcomed by new SoTA methods that verify quantitatively on standard color constancy datasets.

References

[1] M. Bratkova, S. Boulos, and P. Shirley. orgb: a practical opponent color space for computer graphics. *IEEE computer graphics and applications*, 29(1):42–55, 2008.

[2] G. Evangelopoulos and P. Maragos. Image decomposition into structure and texture subcomponents with multifrequency modulation constraints. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[3] D. A. Forsyth and J. Ponce. A modern approach. *Computer vision: a modern approach*, 17:21–48, 2003.

[4] S. N. Gowda and C. Yuan. Colornet: Investigating the importance of color spaces for image classification. In *Asian Conference on Computer Vision*, pages 581–596. Springer, 2018.

[5] C. Guanghua and Z. Xiaolong. A method to improve robustness of the gray world algorithm. In *4th International Conference on Computer, Mechatronics, Control and Electronic Engineering*, pages 250–255, 2015.

[6] J. Huang, B. Kong, B. Li, and F. Zheng. A new method of unstructured road detection based on hsv color space and road features. In *2007 International Conference on Information Acquisition*, pages 596–601. IEEE, 2007.

[7] P. Joshi, D. M. Escrivá, and V. Godoy. *OpenCV By Example*. Packt Publishing Ltd, 2016.

[8] S. Kolkur, D. Kalbande, P. Shimpi, C. Bapat, and J. Jatakia. Human skin detection using rgb, hsv and ycbcr color models. *arXiv preprint arXiv:1708.02694*, 2017.

[9] R. V. Kumar, K. P. Raju, L. R. Kumar, and M. Jogendra. Grey level to rgb using ycbcr color space technique. *International Journal of Computer Applications*, 975:8887.

[10] F. Liu and Y. Wan. Improving the video shot boundary detection using the hsv color space and image subsampling. In *2015 Seventh International Conference on Advanced Computational Intelligence (ICACI)*, pages 351–354. IEEE, 2015.

[11] T. Papadhimitri and P. Favaro. A new perspective on uncalibrated photometric stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1474–1481, 2013.

[12] S. Sec. Radiometry and sources, shadows, and shading.

[13] S. Sural, G. Qian, and S. Pramanik. Segmentation and histogram generation using the hsv color space for image retrieval. In *Proceedings. International Conference on Image Processing*, volume 2, pages II–II. IEEE, 2002.

[14] K.-F. Yang, S.-B. Gao, and Y.-J. Li. Efficient illuminant estimation for color constancy using grey pixels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2254–2263, 2015.