Geometry and Lighting from a single image of an indoor environment

A Project Report

submitted by

DESHMUKH ANEESH NARENDRA

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2013

THESIS CERTIFICATE

This is to certify that the Thesis entitled **Geometry and Lighting from a single image of an indoor environment** submitted by **Deshmukh Aneesh Narendra** to the Indian Institute of Technology Madras for the award of the degree of **Master of Technology** is a bonafide record of research work carried out by him under my supervision. The contents of this Thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. A.N. Rajagopalan
(Research Guide)
Professor
Dept. of Electrical Engg.
IIT Madras
Chennai - 600 036.

Place: Chennai
Date:

ACKNOWLEDGMENTS

I take this opportunity to express my deepest gratitude to my guide Dr. A. N. Rajagopalan for his constant motivation and timely guidance to help me see through getting this thesis in a very fruitful manner.

I would also like to thank the faculty at Electrical Department, who have imparted some valuable knowledge, academic insight and teachings obtained during various lectures, talks and interactions during my stay here. I would also like to thank the department to provide some excellent facilities.

I would also like to thank my lab mates for their discussions and interactions, which were helpful to broaden my vision in the field. A special mention for Natalia Neverova from University Jean Monnet, France who have helped giving some of the insights into her work in the same field. Also, I would like to thank my friends in campus who have made my stay a very memorable one.

Last, but not the least, I would like to thank my parents and my family members, who have been a constant support for me.

ABSTRACT

With the sale of smartphones touching 207.7 million, alone in the fourth quarter of 2012, the mobile boom has reshaped the way we interact with new technology. With greater processing power, multiple cores, sensors, displays and cameras being put into faster and efficient mobile phones, virtual reality has made a tremendous breakthrough. The most important things required for any virtual reality application to appear realistic and pleasing are geometry and lighting information.

This thesis explains about the geometry and the lighting information, all obtained from a single image of an indoor environment. The Manhattan-geometry which is very well observed in indoor scenes plays a pivotal role in inferring geometry from the scene. On the other hand, specularity found in a scene carries a lot more information about the lighting of the scene. Separating out specularity in a scene opens up its uses to estimate light chromaticity and finding the better albedo. However, the lack of knowledge of depth from a single scene is tough to handle. To provide a solution for this, our method uses a *Kinect* depth sensor provided by *Microsoft* to get the 3D information of the world. Using information about the 3D world and the normals, we present a simple approach to estimate the position(s) of light source(s).

The basic aim of this thesis is to provide a platform for augmented reality by providing information about the scene geometry, light chromaticity, correct albedo and light source positions to be able to render a scene.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATIONS USED

Notations Used

X, Y, Z	indicate world co-ordinate system
x, y	indicate image plane 2D co-ordinate system
v_x, v_y, v_z	vanishing points in the X, Y and Z direction respectively
$Sw(l_x, v_y, \alpha)$	set of pixels swept by an angle α
l or l_x	a line in 3D or a set of lines of a same orientation
$I(p)$	intensity of the image at pixel
p 2D	co-ordinate on the image plane
$D^0(p)$	diffuse component obtained from purely color based approach
$C^0(p)$	specularity obtained from purely color based approach
R, G, B	indicate three color channels
T_s	threshold to separate specularity
μ_s	mean of all minimum values among the three channels
σ_s	standard deviation of all minimum values among the three channels
R_s, G_s, B_s	light chromaticity in the three color channels
R_n, G_n, B_n	chromaticity normalized color channels
ρ	albedo (or reflectance)
S	shading
$\mathbf{r}$	mirror-like reflected line
$\mathbf{d}$	direction of light source
$\mathbf{l}$	line in 3D
$\mathbf{n}$	normal at a point
$\mathbf{v}$	viewing direction
m_b	diffuse shading distribution
m_{int}	specular shading distribution

CHAPTER 1

Introduction

1.1 Motivation

Sometime around 300 B.C., in Alexandria, in ancient Greece, Euclid laid down the principles of binocular vision, that each human eye sees a slightly different view of the same scene. Since then depth and three-dimensional information have been a part of our development (Sammons, 1992). Over years and centuries together, science and art have been applied together to amuse humans by creating 3D effects. However, it was only in the nineteenth century that there were significant developments in this field. More precisely in 1838, an English scientist Sir Charles Wheatstone (Sammons, 1992) developed a device for viewing a stereoscopic pair of separate images, depicting left-eye and right-eye views of the same scene, as a single three-dimensional image. This particular invention spurred a boom of three-dimensional revolution and lead various scientists, artists, photographers, cinematographers to present a perfect picture in 3D. What we see today available easily as a medium of entertainment, education and economy, 3D pictures, comics and films have indeed come a long way.

1.2 Overview of extracting 3D information

Inferring 3D from a scene has always been an important part of research in computer vision. Generating 3D models is very closely related to estimating depth. In fact, 3D and depth go hand-in-hand from a research point of view. From self-navigating robots, self-localizing humanoids to autonomously driven cars, 3D computer vision algorithms have always played a significant role. Depending on the data available, the environment, the requirement and the application, different depth estimation systems have been developed and used. Depth extraction techniques can be classified as depth from multiple-images or depth from single-image.

1.2.1 Depth and 3D from multiple images and videos

Depth information from multiple-images makes use of the point correspondences among the scenes captured from cameras at different locations. A classical example of this is the 3D reconstruction from stereo-vision systems (Hartley and Zisserman, 2004). The basic ideas behind these algorithms is disparity and parallax. More or less, all the codes follow a very similar approach. The important steps are feature detection and matching.

Also, depth information can be inferred from videos based on properties of optical-flow and camera motion. These are used to make stereoscopic-3D. There are some semi-automatic 2D-3D conversion systems which take manual inputs, such as multiple segmentation of semantic classes in the *key frames* of a video sequence, manual labeling of disparity in the *key frames* and deciding *shot boundaries*. Then using these inputs, they perform motion estimation for dense disparity propagation from *key frames*-to-*non-key frames*. One of the works which uses this is of (Cao *et al.*, 2011*b*). A similar approach based on estimating depth from a video sequence was given by (Huang *et al.*, 2009). His approach is to extract motion vectors from H.264 encoded video, generate initial motion based depth map, detect moving foreground objects and modify the depth map and extract depth based on geometrical perspective and perform depth map fusion using information from both the depth maps. All these algorithms and methods are directed more towards providing content for 3D TV. A general overview and comparison study of various 2D to 3D conversion systems can be found in the research assignment by Q. Wei (Assoc *et al.*, 2005). A critique of many such methods is described in a magazine article by (Cao *et al.*, 2011*a*).

1.2.2 Geometric reasoning from a single image

Depth information from a single image is a very challenging problem and there is as such no unique mechanism to infer depth from a single image. However, there have been various approaches to extract 3D information based on low-level image features such as edges, colors, vanishing points, textures, etc. These approaches use the knowledge of the three-dimensional world and try to use these features to make an estimate of the 3D world. An example is David C. Lee's work on geometric reasoning from a single image (Lee *et al.*, 2009), which is closely followed in this thesis. A very similar approach could also be found in (Delage *et al.*, 2005)'s work on automatic 3D reconstruction of indoor Manhattan world from a single image. However, researchers have shown that there have been breakthroughs in finding depth based on object-class and surroundings.

Some of the related works done recently in the context of 3D scene interpretation are as follows:

- **Geometric Context from a Single Image:** (Hoiem *et al.*, 2005)'s work is based on a learning model to infer properties of a 3D geometric structure using various cues for surfaces. To obtain useful statistics for modeling various geometric classes, they build the structural knowledge of the image from pixels to super-pixels (a fine way of image segmentation done in patches) and from super-pixels to multiple potential groupings of super-pixels. The various cues used are color, location, shape, texture, long lines, vanishing points, image gradients, etc. to classify ground, sky and vertical regions, which are further subdivided according to their planar orientation given by right-facing planes or left-facing planes or upright planes and non-planar regions as solid or porous regions. This approach is mainly for outdoor scenes.

- **Learning Depth from a Single Monocular Image:** (Saxena *et al.*, 2005) get depth information from a single image based upon training a collection of monocular images and their corresponding ground-truth depth maps. They use discriminatively-trained Markov Random Fields (MRF) that incorporate many local- and global-image features. Further, these features are used to predict depth map under a supervised learning scheme. His Make3D (Saxena *et al.*, 2009) also uses same approach to learn 3D structure from a single image.

- **Recovering the Spatial Layout of Cluttered Rooms:** (Hedau *et al.*, 2009) (Hedau *et al.*, 2010)'s works focus on getting a spatial layout of an indoor environment. Their approach is based on finding straight lines in the image and grouping them into three orthogonal sets corresponding to the 3D co-ordinate system to find the vanishing points for each set. From these vanishing lines and vanishing points, they rank the best possible candidate box for a scene based on learned parameters. The clutter in the room is then found using the ground-vertical line and the learned box model. The surface layout is further classified, similar to (Hoiem *et al.*, 2005)'s work.

- **Depth Information by Stage Classification:** (Nedovic *et al.*, 2007) have proposed scene categorization as a step to estimate depth. Few of the salient points from their paper is about the relation between image statistics, scene structure and depth pattern.

When the scene is small, larger surfaces merge into coarser structures showing finer details of the depth. As the scene depth increases, more and more objects are added to the scene and the texture of the various patches carry some information about the depth at those patches. Thus, that paper shows a way to simply categorize various elements in a scene and estimate depth information.

- **Others:** There have been many other approaches such as estimating average depth sense of the scene based on global Fourier transform and local Wavelet Transform (Torralba and Oliva, 2002), calculating distances between planes from a single scene and performing various geometric estimations mentioned in (Criminisi *et al.*, 2000), (Criminisi, 2001)'s works on single view metrology, etc..

In the later part of this thesis, the *Kinect* sensor (provided by Microsoft) (Microsoft, November, 2010) has been used to get the corresponding depth map of the scene. *Kinect* is a motion sensing input device having a color camera, an infrared projector and an infrared camera. More detailed discussions about *Kinect* are given in Chapter 4.

1.3 Importance of lighting information

Changing gears, light plays an important role in augmented reality. Augmented reality is about merging of two worlds - *real* and *virtual*. The "naturalness" of any scene can be taken off easily if the lighting of the scene is not properly known. Any lighting system can be quantified by the number of light sources, their corresponding chromaticity (i.e., color of light) and the position of these light sources in the real world. The interplay of these light sources with the objects and the surfaces in the scene presents a challenging scenario in computer vision and computer graphics. Convincing shadows, inter-reflections and proper illumination do make virtual objects in a scene or a video appear more realistic. Since, one of the basic aims of augmented reality is to make virtual world close to reality, lighting indeed becomes an important quantity.

In general, any computer graphics application requires the designer to specify the exact location of source of light, its shape, color, intensity, etc. to model its effects in a scene. However, the inverse problem of getting lighting information from a scene is not at all an easy task. A survey of light source detection methods (Funk, 2003) by Nathan Funk gives a good overview on the state-of-the-art light source detection methods.

1.3.1 Overview of different light estimation systems

There are many solutions proposed till date for light source detection. One of the first methods by Ullman (S., 1975) for light source detection was based only for images having light sources explicitly inside the image. But most of the light source detection methods are fine tuned for light sources which are not inside the image, but somewhere in the surrounding. Today many solutions make use of complex hardware setup such as high dynamic range cameras, high resolution cameras and light probe cameras to detect light sources. However, recently there have been significant advances in making light source detection possible with cheap hardware setups such as the *Kinect* (Neverova *et al.*, 2012) (Neverova, 2012). Following are some of the methods:

- **Image Based Lighting:** (Debevec, 2002) in his tutorial on Image Based Lighting brings in the flavor of computer graphics. He uses light probes such as spheres kept in the scene to estimate the light source and then renders the scene by a complete graphics based approach. This approach is widely used in movie productions and gaming where user satisfaction is more concerned at the cost of time and computational resources. Most of the graphics related methods involve ray tracing and ray casting algorithms, which are computationally very expensive.

- **Estimating the Natural Illumination Conditions from a Single Outdoor Image:** (Lalonde *et al.*, 2012) present a probabilistic approach in modeling the position of the sun in an outdoor scene. They also model the clouds which may occlude the sun's visibility. The probabilistic model is formed by a data-driven approach where parameters are combination of weak cues that can be extracted from different portions of the image: the sky, the vertical surfaces, the ground, and the convex objects in the image.

- **Markerless augmented reality with light source estimation for direct illumination:** (Frahm *et al.*, 2005) developed a system for inserting a synthetic object into a scene in real-time. Their approach uses an offline tracking of the scene with a normal camera and a fish-eye lens. The camera scans the entire space of the scene and the fish-eye lens captures the lighting in the scene. The light sources are estimated from the fish-eye lens. After aligning the images captured from two different moving cameras, a 3D model is reconstructed with position of light sources estimated and a synthetic object is rendered into a real scene.

- **Identifying Image Authenticity by Detecting Inconsistency in Light Source Direction:** (Lv *et al.*, 2009) developed a very simple approach based on detecting forgeries in the image by estimating the light source directions in the suspected patches. They use a blind identification process to detect the light source directions either at a finite or an infinite distance. They model the light source direction as a solution to a least-square method with error function being the measure between the actual light intensity and calculated light intensity.

- **Probe-free method:**Some other approaches are based on detecting shadows of objects of known geometry (Madsen and Nielsen, 2008) to make light source detection a probe-free method. As can be seen from few of these examples above, light source detection involves a lot of external hardware which may not be easily available. Hence, such methods are definitely useful.

1.4 Problem Statement

We believe that a single image does carry a lot more information than what meets the eye. Keeping the approach intact to a single image of an indoor scene, this thesis aims at inferring geometry and lighting accurately to the extent possible. Although depth information is not easy to derive from a single image, using depth sensor like the *Kinect*, 3D co-ordinates of the scene can be exploited to estimate the position of light source affecting the scene. The objectives of this thesis are:

1. to get 3D information about orientation of surfaces from a single image of an indoor scene,
2. to be able to detect the possible locations in a scene where the light sources are likely to affect; when no information about the light sources is available.
3. to estimate the correct chromaticity of light from a single image, normalize the image to get good color constancy, and in the process, remove inter-reflections in a scene,
4. to get the correct albedo of the scene and utilizing it for rendering a scene and
5. to use a cheap depth sensor like *Kinect* to get the depth map of the scene; since, depth is not easy to get from a single image, and use this depth to estimate the light sources which are affecting the scene.

All the algorithms are developed for indoor scenes where the indoor environment follows the Manhattan-world assumptions very closely. This paves the way to provide a platform for removing user interactions which are there in a recent work on rendering a synthetic object into a scene (Karsch *et al.*, 2011). Also, it provides insights into the amount of information which just a single image provides. This thesis presents a unique way of estimating light sources, different from most of the traditional approaches. The specularity in an image holds a key element for estimation of light source. The approach taken in this thesis closely follows (Neverova *et al.*, 2012) and (Neverova, 2012)'s works on light estimation in indoor environment. It would be relevant to quote from *The Merchant of Venice* by *William Shakespeare* - *"How far that little candle throws his beams! So shines a good deed in a weary world"* and yes, indeed this shine in the cluttered environment does *a good deed* for our algorithm. So, *let there be light!!!*

1.5 Organization of the thesis

Chapter 2 explains the method used to infer 3D geometry from a single image by getting possible line segment candidates, detecting vanishing points and estimating orientation of planes. This chapter also contains the results as well as the system limitations.

Chapter 3 deals with getting lighting information from a single image. It discusses about specularity as an important intrinsic image quantity. It also talks about Retinex theory and explains the methodology to detect light chromaticity and removing specularity and inter-reflections in the scene.

Chapter 4 brings in a new component to the thesis by using a depth sensor hardware as an image capturing device and getting the information of 3D world co-ordinate and normal at every pixel. This information about the geometry and the knowledge of depth helps in better understanding the lighting in the scene. The chapter discusses in detail the reflection model and exploits this information to detect the light sources in the image.

Chapter 5 throws some light on possible applications and extensions to this thesis along with conclusions.

CHAPTER 2

3D geometry from a single image

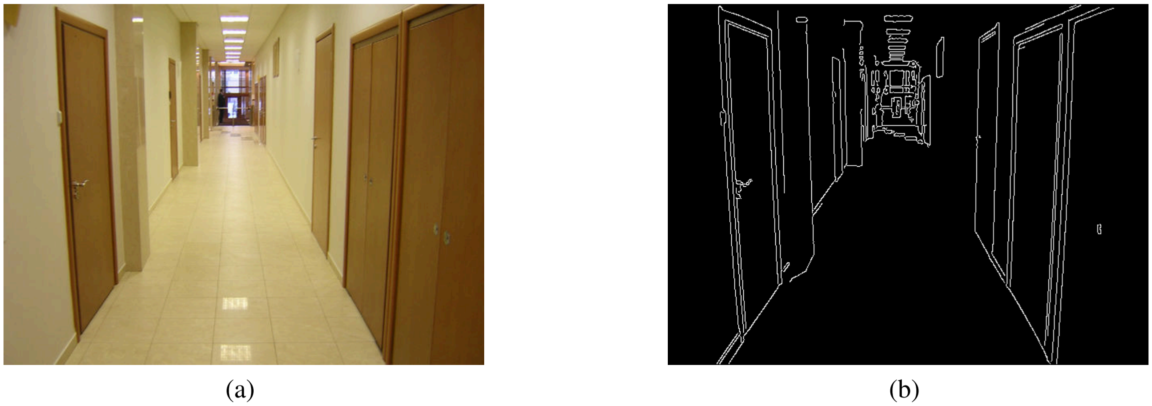
This chapter explains the process of generating 3D orientation estimates of a scene based on geometry. As explained earlier, the images considered in this thesis are only indoor environments. The indoor environments follow the Manhattan world very closely, by which, we mean that entirely all the architectural buildings follow the 3D co-ordinate system of vertical planes, ground and ceiling being orthogonal or parallel to each other. This scenario presents a unique way of interpreting scenes and geometry from just a single image.

This chapter closely follows the works of (Lee *et al.*, 2009), (Hedau *et al.*, 2010), (Rother, 2002) and uses some functions available in Kovesi's MATLAB toolbox (Kovesi, 2000). The entire algorithm is implemented in MATLAB. Few images of indoor scene were taken from the data set mentioned in (Hedau *et al.*, 2010) as well as on from NYU data set (Silberman and Fergus, 2011). These data sets provide images of various rooms, hallways, offices, corridors, etc.. Although clutter, inclined planes, undetected as well as false edges pose a challenge in estimating the 3D geometry of the indoor environment, the process followed in this thesis proves to be very helpful in understanding the surface orientations in the scene. A detailed approach is explained below.

2.1 Get possible line-segment candidates

The following procedure to get line candidates uses some standard functions from MATLAB toolbox given by (Kovesi, 2000) that are tweaked as per the requirement of the system. This is a very widely-practiced and effective approach taken towards classifying line segments.

Given any image, the first step of the algorithm is to convert it into a gray scale image. For getting a better contrast in the image in order to detect as many edges as possible, the



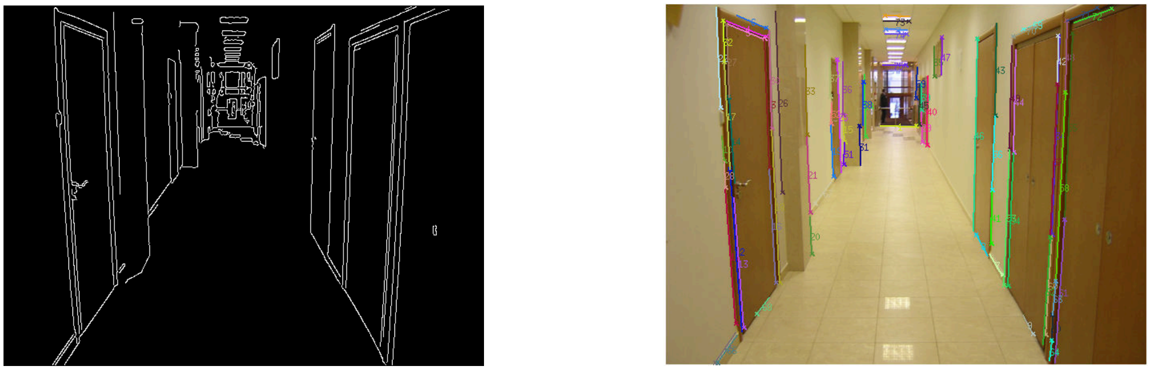
(a) (b)

Figure 2.1: Canny edge detection (2.1b) on the given image (2.1a) with threshold $[0.1\ 0.2]$ and Gaussian smoothing using variance 1.

image is histogram equalized. A Canny edge detector (Canny, 1986) proves to be a better candidate as compared to other edge detecting methods because of Gaussian smoothing operation, non-maximum suppression and double thresholding. For our purpose, the smoothing parameter is taken as 1, and the threshold $[0.1\ 0.2]$ works out decently for an indoor setup as shown in Figure 2.1.

The next step is to fit line-segments to an edge. A minimum length for the edge to be classified as a line segment is calculated and is a function of the image size, usually taken as $\frac{1}{30}^{th}$ of the diagonal of the image. It is necessary to discard short edges to avoid confusion to the algorithm. It is always the longer edges which have more significance in detecting the vanishing points. Given the image and the edge points, it is necessary to track all the edges. A raster scan through image is performed looking for edge points. Each edge point is considered as a starting point and edges are tracked in a particular direction. The connectivity for an edge is considered as its standard 8-pixel neighborhood. Till the point, such connected edges are found, the edges are stored in an array and labeled accordingly. When no more connected edges are found, the edge is tracked in the opposite direction from the previous starting point. This process is followed blindly for every edge point found. The edges whose length is shorter than the minimum length are discarded. At this stage, all the edges are stored in a form of list containing points corresponding to those edges and labeled accordingly.

Once all the edge arrays are listed, it is necessary to generate line-segments. This


(a)

(b)

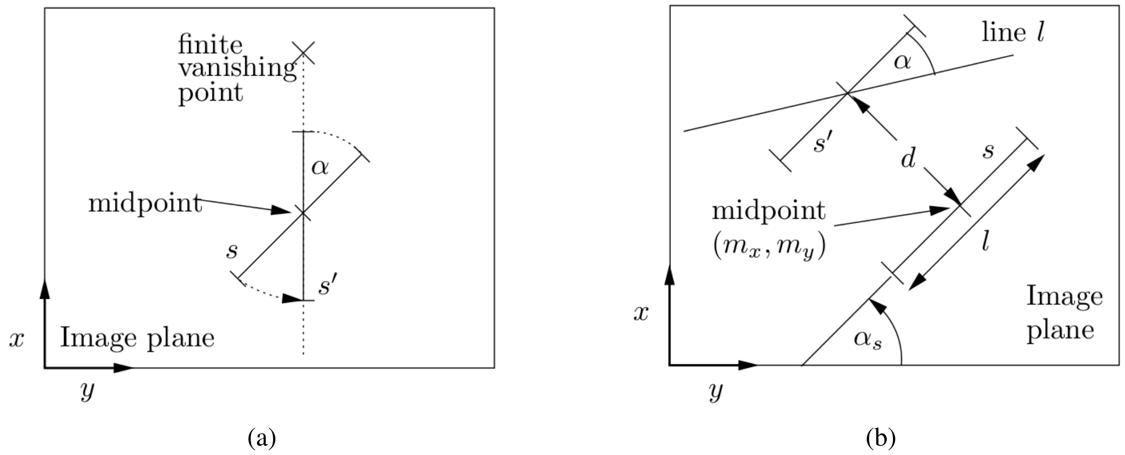
Figure 2.2: Line-segments (Figure 2.2b) are fitted onto the edge image (Figure 2.2a).

is performed by checking how straight a particular edge is. Given a particular edge, the deviation of the midpoint of that edge from the line joining these endpoints is checked. If this deviation is above a certain threshold, then the same procedure is followed; now considering the midpoint and the end points as two separate edges. In simple words, given any edge, we check how closely it can be imitated by various line-segments. Again the shorter line segments are discarded. At the end of this stage, all the possible line-segment candidates are available for further processing. Figure 2.2 shows the line-segments (Figure 2.2b) being fitted onto the edge image (Figure 2.2a) obtained by Canny edge detection (Canny, 1986).

2.2 Vanishing point detection

Most of the indoor images are taken upright with the vertical edges/walls appearing straight in the image. This proves to be an important starting point to classify line-segments which are vertical. All those lines which subtend a small angle (for our experiments, we consider 20° as the threshold) to a point (i.e., $[0,\ 10^5]$) very far away in the top of the image plane are classified as vertical lines. These lines are then used to find a common intersection point corresponding to these lines, which is nothing but the vertical vanishing point.

A RANSAC (Random Sampling Consensus) approach is used to decide which lines among those classified as vertical lines are used just to speed up the process. A vertical



(a)

(b)

Figure 2.3: Explanation of distance between a vanishing point and a line-segment given in Figure 2.3a; Figure 2.3b explains the distance between a line and a segment as used in the code (Image source: (Rother, 2002)).

vanishing point is decided depending on a voting algorithm explained in (Rother, 2002). The possible candidate gets votes which are decided by:

$$vote(a) = \sum_{\text{all s vote for a}} w_1 \left(1 - \frac{d\,(a, s)}{t_a}\right) + w_2 \left(\frac{\text{length of s}}{\text{max length of s}}\right) \tag{2.1}$$

where s is a line-segment which votes for a possible vanishing point candidate a only if the distance $d(a, s)$ is below a certain threshold t_a. In this notation, $d(a, s)$ is defined as the angle between the corresponding line-segment s and a point a. The concept of the distance between the vanishing point and a line-segment is shown in Figure 2.3a and the distance between a line-segment and a line is shown in Figure 2.3b. In this context, distance is calculated as the angle subtended by a point on the mid-point of the line-segment. Also more weightage is given if the length of the line-segment s is more. The weights w_1 and w_2 are taken as 0.3 and 0.7, respectively. The angular threshold t_a is taken as $10°$. Once the vertical vanishing point is decided, the set of lines which now subtend a small angle ($10°$) are classified as vertical lines. Others are used in computing the remaining two vanishing points.

The process of finding the second and third vanishing point uses the fact that these three vanishing points are orthogonal to each other in real world. Also in the image plane, from the properties of the vanishing points, they follow *orthogonality criterion* and *camera*

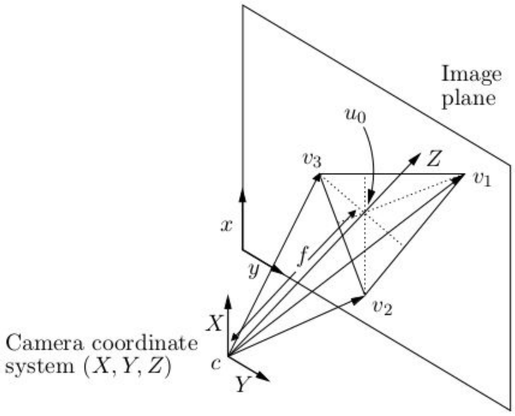


Figure 2.4: Explanation for the camera geometry and the orthogonality criterion (Image source: (Rother, 2002)).

criterion to compute the other two vanishing points. The *orthogonality criterion* is shown in Figure 2.4 and can be explained as

$$< cv_1, cv_2 >= 0, \ < cv_2, cv_3 >= 0 \text{ and } < cv_3, cv_1 >= 0 \tag{2.2}$$

The *camera criterion* makes use of the knowledge of principal point and focal length. For known camera intrinsic values, these are exact values, otherwise some empirically decided ranges and values for camera intrinsic parameters are used to decide which value makes the vanishing point orthogonal as explained in (Rother, 2002). The vanishing point may be at infinity in certain cases; but in our case we assume it to be finite for all the three directions. In this thesis, infinity is assumed as 10^7 (in pixels) for simplicity. A simple algorithm to explain the above express is explained below.

Algorithm 1 Find all vanishing point

```
Inputs: $v_1$ : Vertical vanishing point i.e., highest vote accumulator as explained before
Output: $[v_1, v_2, v_3]$: Orthogonal vanishing points
Use other lines which do not belong to vertical vanishing point $v_1$
Go through all pairs of possible vanishing points pair $v_i$ and $v_j$
if $(v_1, v_i)$, $(v_i, v_j)$ and $(v_j, v_1)$ satisfy vanishing line criterion then
    if Orthogonality criterion and camera criterion is satisfied by $(v_1, v_i, v_j)$ then
        Compute $vote = vote(v_1) + vote(v_i) + vote(v_j)$
    end if
end if
Take $(v_i, v_j)$ with the highest vote $vote$
```

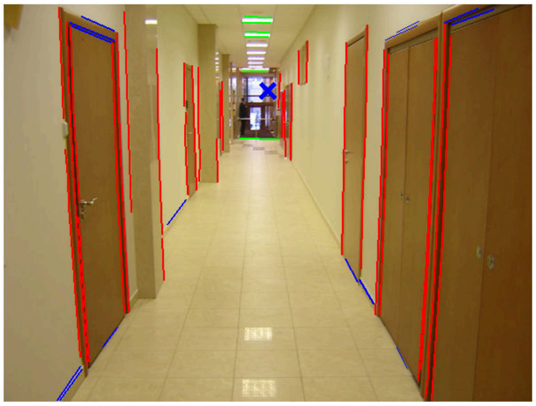
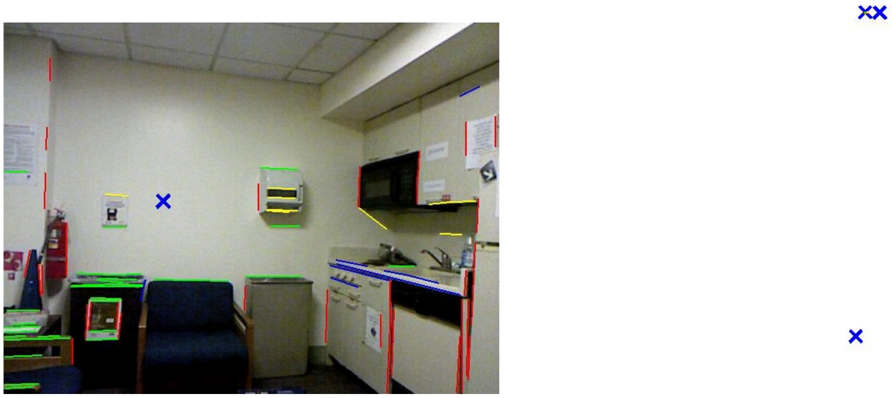


Figure 2.5: Detected line-segments, which are used to calculate the vanishing points. The three colors (red, green and blue) correspond to the line-segments for three vanishing point. One of the vanishing point is shown as a blue cross in the image. Other vanishing points are finite, but very far away in the image plane.



(a)

(b)

Figure 2.6: Another example showing the detected line-segments (Figure 2.6a) (yellow lines indicate outliers); corresponding vanishing points for the scene (Figure 2.6b).

The method to tackle vanishing point at infinity is explained in (Rother, 2002) but not considered in this thesis. This approach has been followed by (Lee *et al.*, 2009), (Hedau *et al.*, 2009), (Hedau *et al.*, 2010), (Karsch *et al.*, 2011) and (Neverova, 2012). Hence, this too becomes more or less a standard algorithm, when it comes to detecting vanishing points in indoor environment. The detected vanishing point for the image under consideration (Figure 2.1a) is shown in Figure 2.5. The vanishing point is shown as a blue cross in the image. Another example to show all the three vanishing points in the image plane is shown in Figure 2.6. The vanishing points are indicated as blue crosses.

2.3 Inferring orientation of planes

Once the vanishing points and the best set of lines corresponding to those vanishing points are decided, it is important to estimate what the combination of these lines means. This work was inspired from Geometry Reasoning for single image structure recovery (Lee *et al.*, 2009). The basic idea of this section is as follows: given that the surfaces are oriented along three co-ordinate axes, the set of lines which form a particular surface are very well defined and are unique for that particular orientation.

For example, consider Figure 2.7 below.

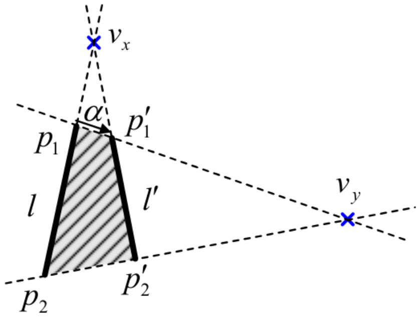


Figure 2.7: A surface inferred in between lines p_1p_2 and $p_1'p_2'$ (Image source: (Lee *et al.*, 2009)).

A line-segment p_1p_2 belonging to line l has the vanishing point v_x lying on it. Any surface can be formed by sweeping a segment about the vanishing point in a particular direction until it is restricted by another line belonging to the same vanishing point. In this case, that line is given by $p_1'p_2'$. As per simple logic, the other set of line completing the surface must belong to another vanishing point. As shown in Figure 2.7, the vanishing point v_y will give the line-segment p_1p_1'. The point p_1' can be given by

$$p_1' \;=\; p_1 \;+\; \alpha\,(\,v_y - p_1\,) \tag{2.3}$$

The point p_2' is given by the intersection of the lines v_xp_1' and v_yp_2. In a similar manner, we can sweep the surface in the opposite direction, where the sweep is given by β. Rest of the nomenclature remains the same. Thus, the set of pixels that is supported by all such lines can be given by $Sw(l_{x,i}, v_y, \widehat{\alpha_{x,i}})$ and $Sw(l_{x,i}, v_y, \widehat{\beta_{x,i}})$ ($l_{x,i}$ indicates all those lines belonging to vanishing point v_x) (Lee *et al.*, 2009), where

$$\widehat{\alpha_{x,i}} = \max(\alpha) \text{ and } \widehat{\beta_{x,i}} = \max(\beta) \qquad (2.4)$$

From this formulation, it is clear that the orientation of a plane whose line-segments belong to vanishing points v_x and v_y must be oriented in the direction of the other remaining vanishing point v_z. So the entire set of of pixels that is supported by all lines belonging to v_x and swept toward v_y in the orientation of Z is

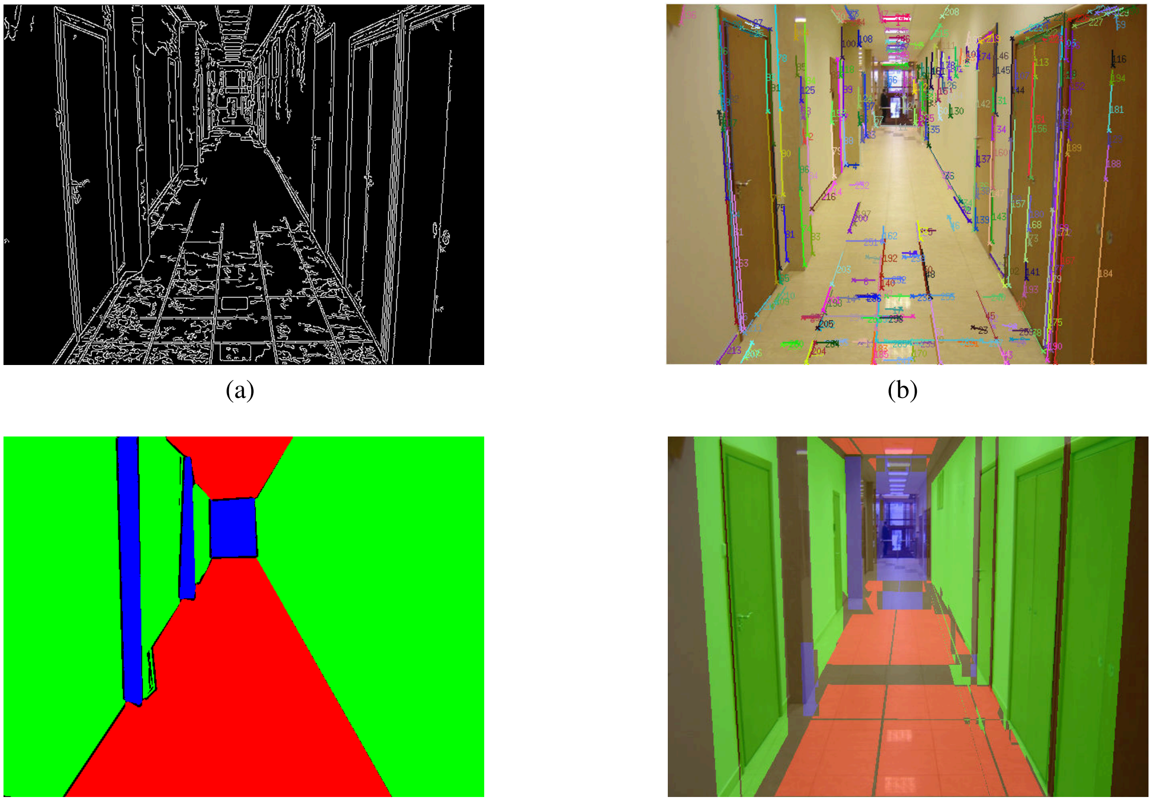
$$P_{xyz} = \bigcup_{\text{all } l_{x,i}} Sw(l_{x,i}, v_y, \widehat{\alpha_{x,i}}) \bigcup Sw(l_{x,i}, v_y, \widehat{\beta_{x,i}}) \qquad (2.5)$$

This is done for all the three vanishing points and using combination logic, the surfaces are inferred from these. The result of this stage is the estimated geometric surfaces from a single image (Figure 2.8). A simple algorithm is explained below.

Algorithm 2 Inferring orientation of planes

```
Inputs: Line-segments, Vanishing points
Output: Orientation map of the scene
Initialize a cell for all vanishing points and line classes.
Each cell corresponds to a image plane.
for Every line-segment do
    Find the line class i.e., which direction it belongs to.
    Find polygons which fit the plane as given by $Sw(l_i, v_j, \widehat{\alpha})$ and $Sw(l_i, v_j, \widehat{\beta})$.
    Assign that polygon a plane label.
end for
Use combination based on which plane will be formed by which set of vanishing point
and line class. Assign that as orientation map by combining all.
```

To be able to detect a good estimate of the orientation of the scene, more number of line-segments are required. Hence, Canny edge detector (Canny, 1986) with Gaussian smoothing parameter of 0.5 is used. Figure 2.8a is the resulting edge map and Figure 2.8b shows the line-segments fitted onto those edges. Figure 2.8d is the estimated orientation map of the scene while Figure 2.8c is the corresponding ground truth. More results follow.



(a)

(b)

(c)

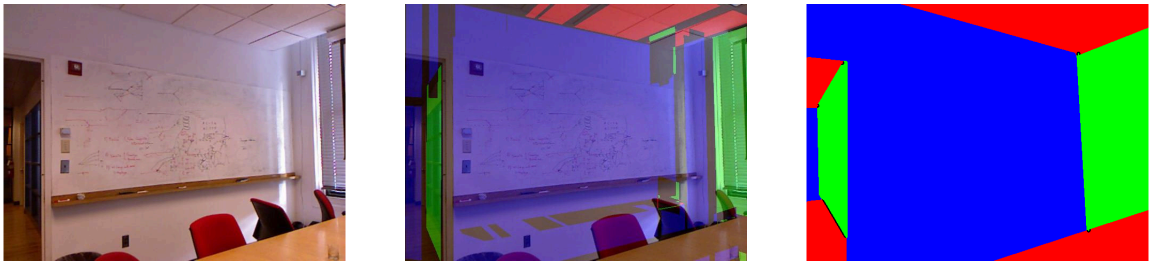
(d)

Figure 2.8: For detecting orientation map (2.8d), more lines (2.8b) are required which are obtained from a finer edge map (2.8a) and vanishing point information as explained in Section 2.2

2.4 Results

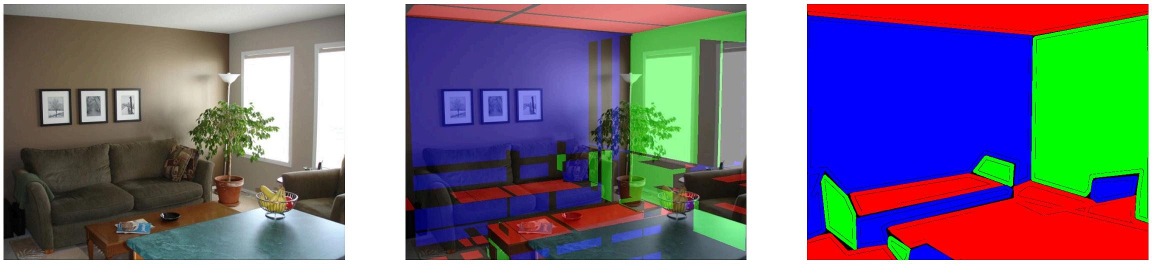
The images on which the experiments were performeded consist of various scenes. The images presents a very well balanced mixture of varying environments. Some results are shown in this section and they include cases where the estimated maps have come good as well as not that good. There are some failure cases as well. Also, an analysis of the results obtained is performed and follows the results. Later, some of the limitations and challenges are discussed in Section 2.5.

Figures 2.9a and 2.10a show the original images. The inferred orientation maps are shown in Figures 2.9b and 2.10b, respectively. The corresponding groundtruth orientation maps for these images are shown in Figures 2.9c and 2.10c, respectively. It can be observed that the geometry is very well defined in these scenes and the orientation maps have come



(a) (b) (c)

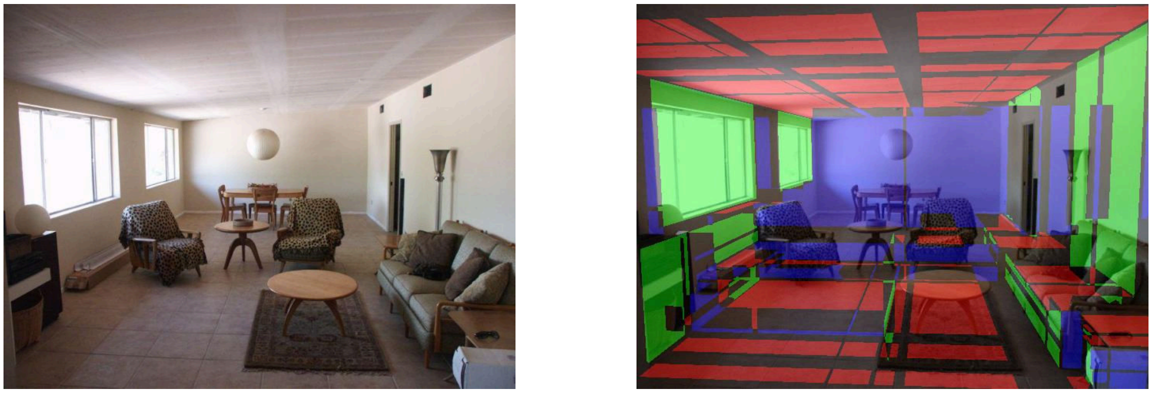
Figure 2.9: Estimated orientation map (Figure 2.9b) corresponding to image shown in Figure 2.9a. The groundtruth for the same is shown in Figure 2.9c.



(a) (b) (c)

Figure 2.10: Estimated orientation map (Figure 2.10b) corresponding to image shown in Figure 2.10a. The groundtruth for the same is shown in Figure 2.10c.

out really well. Very small portions are missing in the orientation maps due to presence of some object in the scene.



(a) (b)

Figure 2.11: Inferring orientation maps in presence of clutter. (Case: well defined geometry in the scene.)

In some cases there is a lot of clutter in the room and that poses a challenge to the algorithm. Yet, the geometry is inferred blindly (Figures 2.11b and 2.12b) to get the surfaces in between the detected line-segments. It can be seen that because of the presence of clutter in the scene, the orientation map do not come out continuous. In scenarios, where walls or edges are partially visible (Figure 2.12a), getting orientation map does not make much sense.

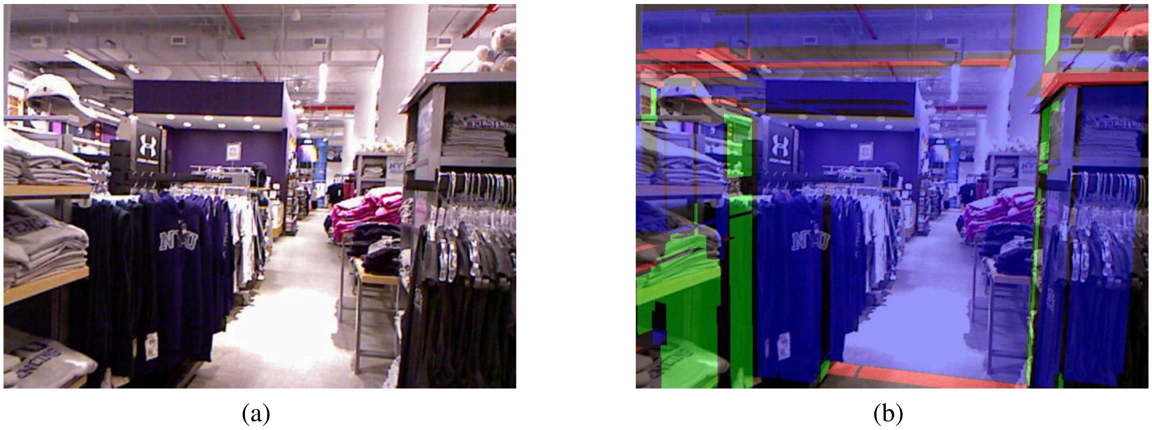


Figure 2.12: Inferring orientation maps in presence of clutter. (Case: not much geometry can be inferred anyway.)

However, there are few errors in detecting the planes because of wrongly detected line-segments as shown in Figures 2.13a and 2.14a. This results in getting false orientation in some patches. In some cases, due to absence of any strong lines or planes, the algorithm completely fails as shown in Figure 2.15b. Also, it can be seen from Figure 2.15a that because of false classification of line-segments in the scene, the algorithm fails and is not suitable to be used under such scenarios. Thus, to make use of this algorithm, 3D geometry of the scene should be perfectly captured in the scene.

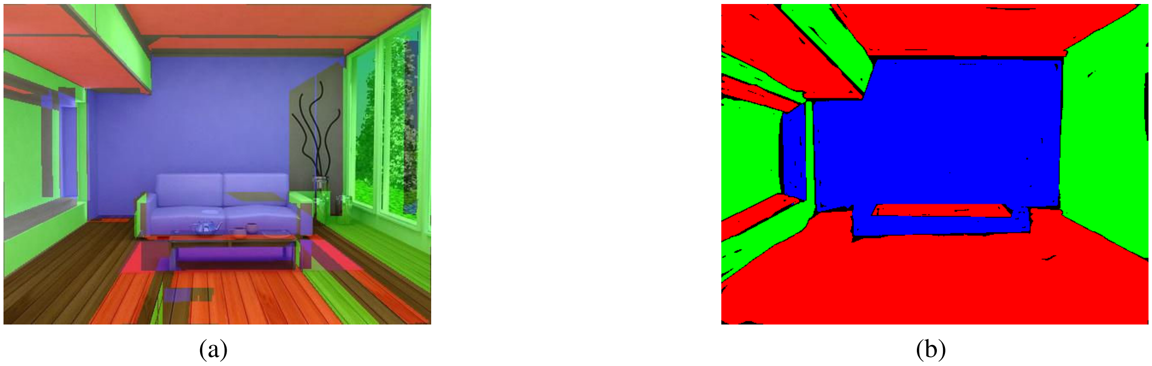
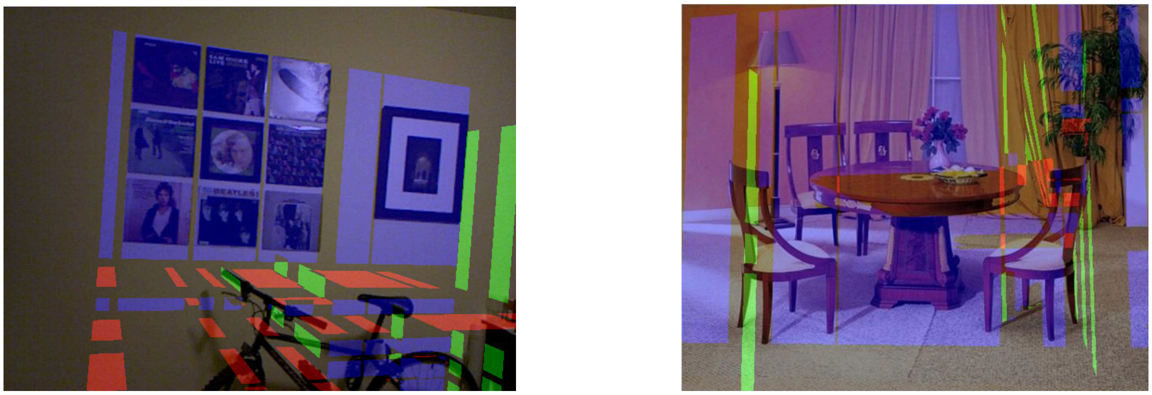


Figure 2.13: Example 1: Minor error due to wrongly classified line-segment.

(a) (b)

Figure 2.14: Example 2: Minor error due to wrongly classified line-segment.

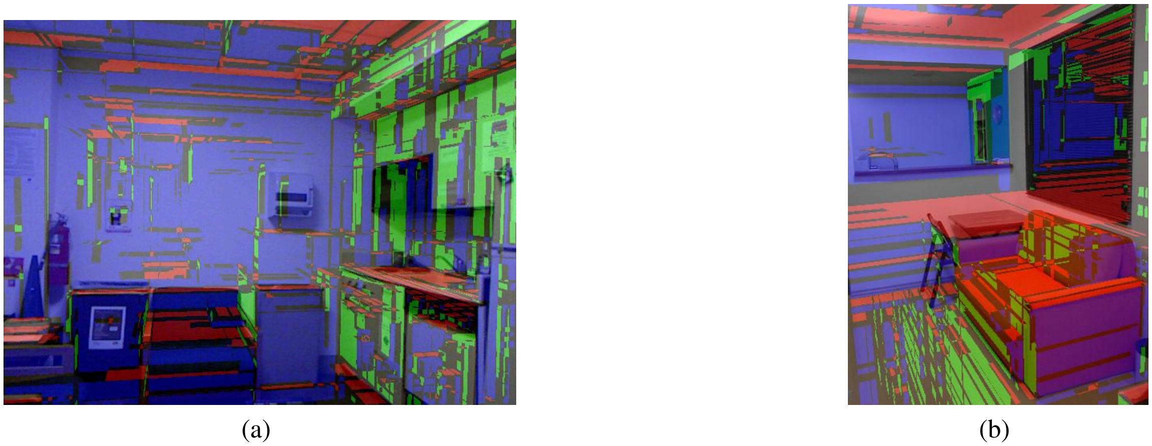


(a) (b)

Figure 2.15: Failure cases

2.4.1 Analysis of the results

The overall analysis of this algorithm is shown in Table 2.1. The result was obtained on 60 indoor scenes obtained from different datasets available mentioned in (Hedau *et al.*, 2009), (Neverova *et al.*, 2012). The images offer a great variety of indoor environments right from alleys, offices, kitchens, study rooms, shops, bedrooms, etc.. The analysis was done on two sets of results of orientation maps. In the first set of results, the minimum length of the line to be considered was $\frac{1}{30}^{\text{th}}$ of the length of the diagonal (in pixels) of the image for detecting the vanishing point and $\frac{1}{50}^{\text{th}}$ of the length of the diagonal (in pixels) for detecting the orientation map. The other set of result (shown in Figures 2.16a and 2.16b) was obtained using a fixed minimum length of 5 pixels for all the images, which is much shorter length than the previous case.



(a) (b)

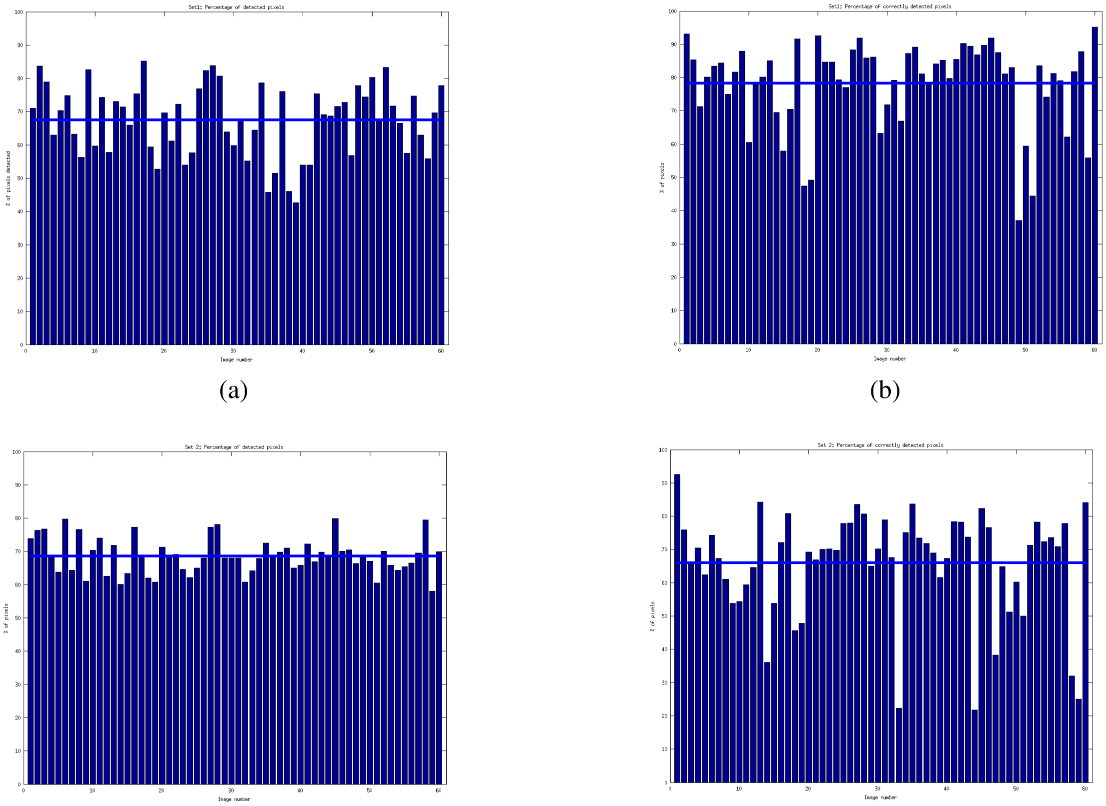
Figure 2.16: Results obtained allowing short lines.

To tell whether the orientation of a pixel is correct, a groundtruth orientation map was manually sketched using GIMP. We sketched the groundtruth and colored the planes in the scene according to the three colors (red - for floor and ceiling, green - for side walls and blue - for background) used to display our results. The groundtruth drawn neglected small clutter or objects in the scene and a coarse-groundtruth was only considered. Some of the examples of groundtruth are shown in Figures 2.9c, 2.10c, 2.13b and 2.14b.

The entire analysis done on the 60 images is shown in Figure 2.4.1. Figure 2.17a shows the percentage of detected pixels for all images in Set 1. Figure 2.17b shows the percentage of correctly detected pixels out of the detected pixels for Set 1. Similarly, Figures 2.17c and 2.17d show the percentage of detected pixels and correctly detected pixels among those for Set 2. In Figure 2.4.1, all the bar graphs have Y-axis as the percentage of detection and X-axis as the image number.

Another set of analysis was done on 39 images, where the principal point and the focal length were known and so, this knowledge was used to detect orientation of the plane. These images were obtained from NYU dataset (Silberman and Fergus, 2011). The analysis of the result obtained from these 39 images is shown in Table 2.2.

It can be seen from the tables that the number of pixels detected will increase when we allow shorter line-segments to decide the orientation maps. When compared to the coarser groundtruth maps, the accuracy will go down. However, the shorter line-segments capture few small planes very well which might be planes of objects in the scene. The images also



(a) (b)

(c) (d)

Figure 2.17: Analysis result for the 60 images.

consisted of some indoor scenes where geometry was difficult to tell.

Roughly, it can be said that the orientation map is detected for about 70% of the image and out of these detected pixels about 80% are correct. However, when images where geometry is very well intact, the detection rate can go upto 80% with an accuracy of $85 - 90\%$ correct detection.

For the first set of results, a visual inspection was performed. On a scale of $0 - 10$ (0 being the worst and 10 being the best), 60% of the images show a score of 7 and above and the average visual score for the 60 images is about 6.5. For the 39 images from NYU dataset, the average visual inspection score is 6.69 and more than 64% of the images have score of 7 and above. It can also be observed that perceptually, the results from Set 2 (Figures 2.16a and 2.16b) are not appealing because allowing short line-segments results in many smaller, thinner and discontinuous orientation planes, which are not likely to be found in an indoor scene. Such cases occur only when there is a lot of clutter in the room;

in which case, inferring geometry from such a scene becomes a difficult task.

Item	**Set** 1 **minlen1 =** $\frac{1}{30}$***diag** **minlen1 =** $\frac{1}{50}$***diag**	**Set** 2 **minlen1 = 5** **minlen1 = 5**
Average pixels detected	67.49% S.D. = 10.66%	**68.58%** **S.D. = 5.31%**

Accuracy of detection	**78.24%**
	S.D. = 13.03%
	MAX = 95.10%
	MIN = 37.1%
		65.93%
		S.D. 15.78%
		MAX = 92.6%
		MIN = 21.8%

Images having above average correct detection and their mean	$\mathbf{\frac{40}{60}}$**(66.67%)**
	MEAN = 85.65%
	S.D. = 4.33%
		$\frac{39}{60}$(65%)
		MEAN = 74.99%
		S.D. = 6.05%

Table 2.1: Analysis of 60 images.

Item	**Set** 1 **minlen1 =** $\frac{1}{30}$***diag** **minlen1 =** $\frac{1}{50}$***diag**	**Set** 2 **minlen1 = 5** **minlen1 = 5**
Average pixels detected	66.25% S.D. = 11.2%	**68.61%** **S.D. = 4.45%**
Accuracy of detection	**79.79%** **S.D. = 12.8%**	67.27% S.D. 15.37%

Images having above average correct detection and their mean	$\frac{25}{39}$(64.10%)
	MEAN = 86.42%
	S.D. = 3.45%
		$\mathbf{\frac{28}{39}}$**(71.79%)**
		MEAN = 74.63%
		S.D. = 4.88%

Table 2.2: Analysis of 39 images from NYU dataset.

2.5 Limitations and Challenges

The limitations and challenges faced by this approach are:

- The entire logic is based on the richness of the information provided by edges.
- If it is not possible to detect enough number of line-segments, the orientation map is coarser. To avoid this problem, the line detection algorithm is taken at two different stages. While detecting the vanishing points, longer edges are considered. However, while considering lines for orientation maps, short line-segments are considered.

- False line-segments detected due to various reasons (mainly, because of presence of gradient in the image itself) result into false orientation maps or result into small number of false planes in between bigger planes (Figure 2.13a, 2.14a).
- In places where there is no information about lines, the orientation map is missing.

One of the solutions is to correctly detect ground-vertical boundary and separate the ground plane (Delage *et al.*, 2005). Knowing the kind of corners formed by the connecting surfaces in an image (Lee *et al.*, 2009), reconstructing these corners in between the detected planes can rectify some of the problems faced by this algorithm. However, this is useful if 3D reconstruction of a scene is being considered, since in such a case the entire orientation of the room along with it's depth information is required to be known.

CHAPTER 3

Lighting information from a single image

As described in introduction (Chapter 1) of this thesis, there have been many methods to get the lighting information. However, the use of a very simple approach discussed in Shen and Cai (2009), Neverova *et al.* (2012) helps to estimate the light chromaticity in a given image.

3.1 Specularity: An important intrinsic quantity

Most of the computer vision algorithms assume that object surfaces are purely diffuse (or homogeneous) Shen and Cai (2009). However, there is no homogeneity in the real world. Many objects such as plastic, wood, human skin, textile, ceramics, etc. are not homogeneous and do appear frequently in the indoor environment. Hence it is important to consider both diffuse as well as specular reflections. This specularity carries a lot of information related to surface roughness, lighting direction or can be used for image-based scene synthesis Shen and Cai (2009).

For any material, the diffuse and specular components can be described as follows. Whenever a ray of light strikes a surface, some of the light ray is immediately reflected from the surface because of the different refractive indices between the air and the surface at that location. This part of light is called as specular reflection, which is dependent on the local orientation and the roughness degree of the surface Shen and Cai (2009). Thus most of the specular reflections i.e. "interface reflections" occur due to smooth surfaces. Since the reflection is almost mirror-like and occur in some small range of angles, it is safe to assume that specular reflections are caused because of the perfect mirror-like reflections.

Some of the light is absorbed by the body and then reflected back to the surroundings. This scattering of light depends upon the inside-structure of the object. Most non-homogeneous objects have a random structure inside them and hence it is safe to assume that the "body reflection" is uniform in all directions and this component is termed as *diffuse*.

The aim of this part of the thesis is to infer the light chromaticity from specularity obtained from a single image. This section provides a strong argument that it is the specularity in the indoor environment which is a strong cue for the presence of light source that is affecting the image. Knowing the light chromaticity correctly, a scene can be very well normalized to remove the effects of inter-reflections in the scene. This gives a better idea of the scene and a true specular-free image is obtained which can be then further decomposed into intrinsic images given by reflectance and shading, as described in Lightness and Retinex theory Land *et al.* (1971).

3.2 Methodology to separate specularity

This section explains the methodology used for getting a specular-free image and in the process estimating chromaticity of the light. This work has been referred from Shen and Cai (2009), Neverova *et al.* (2012). They present a very simple and efficient approach for removing specularity and estimating the light chromaticity from specularity. However, there are numerous methods available to separate specularity in an image. A brief survey on all specularity removal methodologies can be found in a survey by Artusi *et al.* (2011).

The approach is based on removing a minimum of the three color channels and then adding a pixel dependent offset. An intuitive explanation on why this algorithm works is based on the fact that diffuse component have higher chromaticity. Mathematically, it has been proved in Tan and Ikeuchi (2008). The following algorithm needs the light chromaticity to be known at first. So the same algorithm is performed in two iterations. From the result of the first iteration, the light chromaticity is estimated. Using this chromaticity, the image is normalized and then using this chromaticity-normalized image, specular reflections are found out. This proves to be a very simple and efficient approach to remove the inter-reflections and to get the correct diffuse and specular component of the image.

Given a image I, the diffuse component D^0 can be calculated as

$$D^0(p) = \begin{cases} I(p) & \text{if } \min[R(p), G(p), B(p)] < T_s \\ I(p) - k_s \min[R(p), G(p), B(p)] + k_s T_s & \text{otherwise} \end{cases} \tag{3.1}$$

where R, G and B are the three-color channels, p is the $2D$ pixel co-ordinate and the coefficient k_s is set experimentally ($k_s = 0.6$) and the threshold T_s depends on μ_s i.e., the mean of all the minimum values and σ_s^2 i.e., the variance of all minimum values. The threshold can be calculated as

$$T_s = \mu_s + \alpha\sigma_s \tag{3.2}$$

where α is again chosen experimentally. Changing the value of α changes the amount of specularity (or shine) as shown in Figure 3.1 below; the detected specularity goes on decreasing as we go from figure 3.1a to 3.1d with increasing α. Lesser the α, more is the speckle.

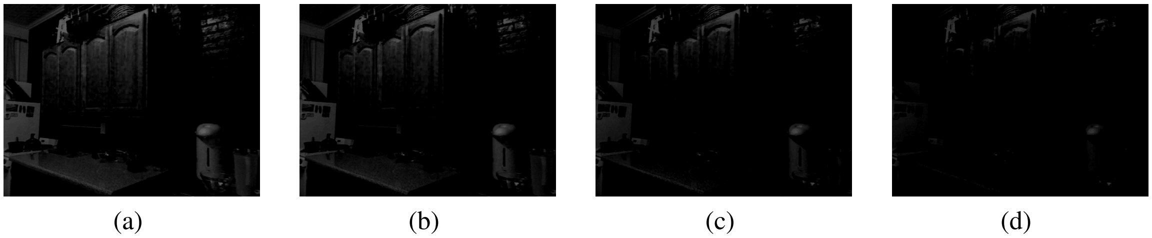

(a) (b) (c) (d)

Figure 3.1: Changing the value of α causes this effect. From left to right, the value of α is 0.2, 0.5, 1 and 1.5.

The specularity component C^0 can be obtained as

$$C^0(p) = I(p) - D^0(p) \tag{3.3}$$

The superscript 0 is used just to indicate that the results are obtained from a completely color-based approach.

At this stage, the specularity is separated from the image. From this obtained specularity component, the light chromaticity is estimated as the mean of all the specularity component. $[R_s, G_s, B_s]$ gives the mean chromaticity of the specular pixels. The normalized image is obtained as

$$\begin{aligned} R_n(p) &= R(p)\frac{R_s + G_s + B_s}{3R_s} \\ G_n(p) &= G(p)\frac{R_s + G_s + B_s}{3G_s} \\ B_n(p) &= B(p)\frac{R_s + G_s + B_s}{3B_s} \end{aligned} \tag{3.4}$$

where $[R_n, G_n, B_n]$ gives the normalized image. The same algorithm is then performed over this normalized image with a different value of α. The resulting specularity is the final specularity in the image. The result obtained is specular-free, chromaticity-normalized diffuse, which can be used for further processing. It is necessary to perform two iterations since the algorithm mentioned in (Shen and Cai, 2009) assumes that the specular component be pure white (Artusi *et al.*, 2011). The values of α used are 0.5 for first iteration and 1 for the second iteration. However, the experiment has also been performed with different sets of α's ranging from $0.6 - 2$ and the best possible combination, which appears convincing is selected. While deciding on the light chromaticity, the number of specular pixels for deciding the mean μ_s should be high enough to make a robust estimate. This also explains why a smaller value of α is used in the first iteration. If the number of specular pixels is very less, then the approach taken to estimate the light chromaticity is to calculate the mean intensity of the 10 brightest pixels in the image (Neverova *et al.*, 2012). This approach is very similar to maxRGB (Funt and Shi, 2010). Even if no specular component is found, the algorithm is not disturbed and the desired outputs are still available. It should be noted that the specular-free image obtained as a result of this algorithm is more close to the diffuse component of the image (Artusi *et al.*, 2011).

3.2.1 Intrinsic Images

In the Lightness and Retinex theory (Land *et al.*, 1971), it is shown that sensations of color show a strong correlation with reflectance. Although the amount of visible light reaching the eye depends on the product of reflectance and illumination, it is very easy for the visual system to separate the amount of flux and know exactly the reflectance. A very similar idea is also explained in (Tan and Ikeuchi, 2004)'s work on getting intrinsic properties of an image with highlights. A simple way to explain this is given by (Grosse *et al.*, 2009) as follows:

- The observed intensity at any pixel p in the image I can be written as:

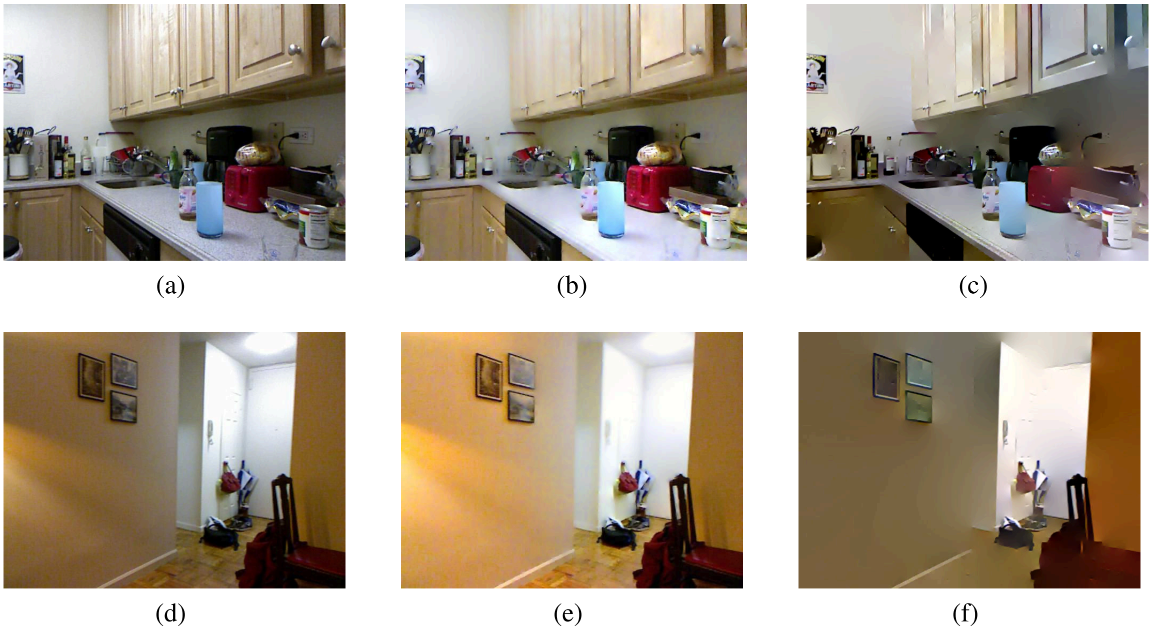
  $$I(p) \;=\; \rho(p)S(p) \;+\; C(p) \tag{3.5}$$

  where $\rho(p)$ is the albedo (or reflectance), $S(p)$ is the illumination (or shading) and $C(p)$ is the specularity.

- Since the specularity is removed in the previous step, it is safe to assume that the image I can be considered as a product of the shading image S and reflectance image ρ. It can be further expressed as:

  $$\begin{aligned} I(p) &= \rho(p)S(p) \\ \log I(p) &= \log\rho(p) \;+\; \log S(p) \\ \triangledown\log I(p) &= \triangledown\log\rho(p) \;+\; \triangledown\log S(p) \end{aligned} \tag{3.6}$$

  where $\triangledown$ is the gradient operator.


(a) (b) (c)

(d) (e) (f)

Figure 3.2: The different Retinex algorithm applied on images 3.2a and 3.2d. (Grosse *et al.*, 2009)'s results are shown in Figures 3.2b and 3.2e. (Limare *et al.*, 2011)'s results are shown in Figures 3.2c and 3.2f.

- From the retinex theory (Land *et al.*, 1971),(Grosse *et al.*, 2009), it is known that it is unlikely that significant shading boundaries and reflectance edges occur at the same point. Thus, a simplifying assumption can be made that every image derivative is either caused by shading or reflectance. This reduces the problem to that of a binary classification problem (Grosse *et al.*, 2009).

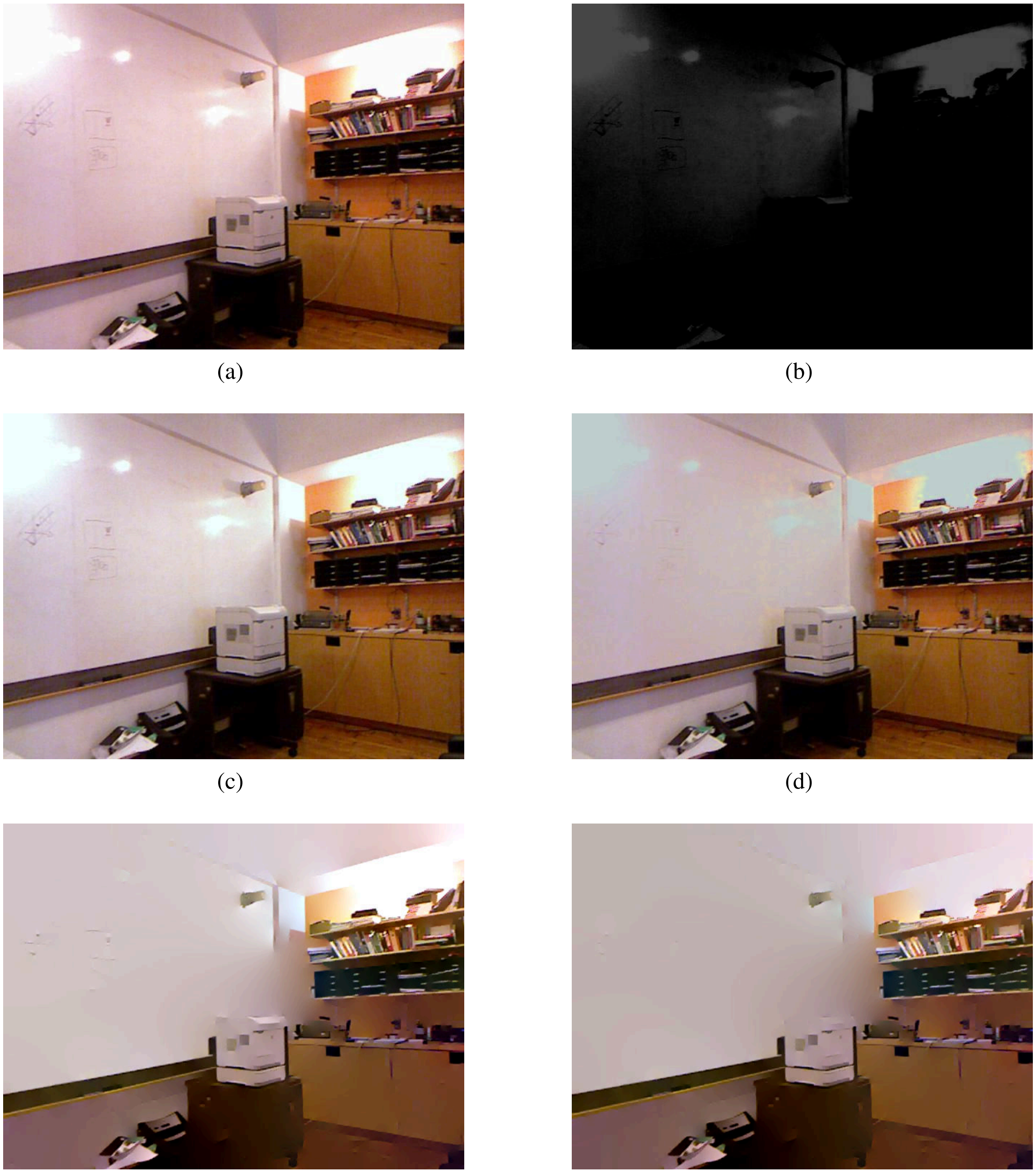
- The results obtained by (Grosse *et al.*, 2009)'s method are shown in Figures 3.2b and 3.2e. However, the albedo obtained is dependent on the shading obtained as the albedo and shading images occur as pairs.
- A better approach of getting reflectance only, completely independent of shading used in this thesis is to solve the discrete partial differential equation using Discrete Fourier Transform (Limare *et al.*, 2011). The results of this method are perceptually more convincing (Figures 3.2c and 3.2f).

3.3 Results

Following are some results and are shown in Figures 3.3 and 3.4. Figures 3.3a and 3.4a show the original images. The separated specularity for both the images is shown in Figures 3.3b and 3.4b, respectively. The chromaticity-normalized images obtained are shown in Figures 3.3c and 3.4c. It can be observed that a reddish tinge is removed from both the scenes. Removing specularity from the chromaticity normalized images give the diffuse images shown in Figures 3.3d and 3.4d. The albedos are estimated using partial differential equation (Limare *et al.*, 2011) and the results are shown in Figure 3.3e for Figure 3.3c, Figure 3.3f for Figure 3.3d, Figure 3.4e for Figure 3.4c and Figure 3.4f for Figure 3.4d. It can be observed that it is difficult to infer any color information at some location as it is completely lost due to specularity.

Few results on chromaticity-normalization are given in Figure 3.5. The first column of images shows the original image, the second column are the chromaticity-normalized images and the third column correspond to the diffuse component obtained.

To prove the point of better normalization, the algorithm was run on a dataset provided by (Gehler *et al.*, 2008), which included a Macbeth Color Checker in each image. The intensity of the white square in the Macbeth color checker was manually checked and the process of normalization agrees with the visual inspection results as well. Thus, this algorithm also provides tools for checking color constancy or getting the color constancy of the image.



(a)

(b)

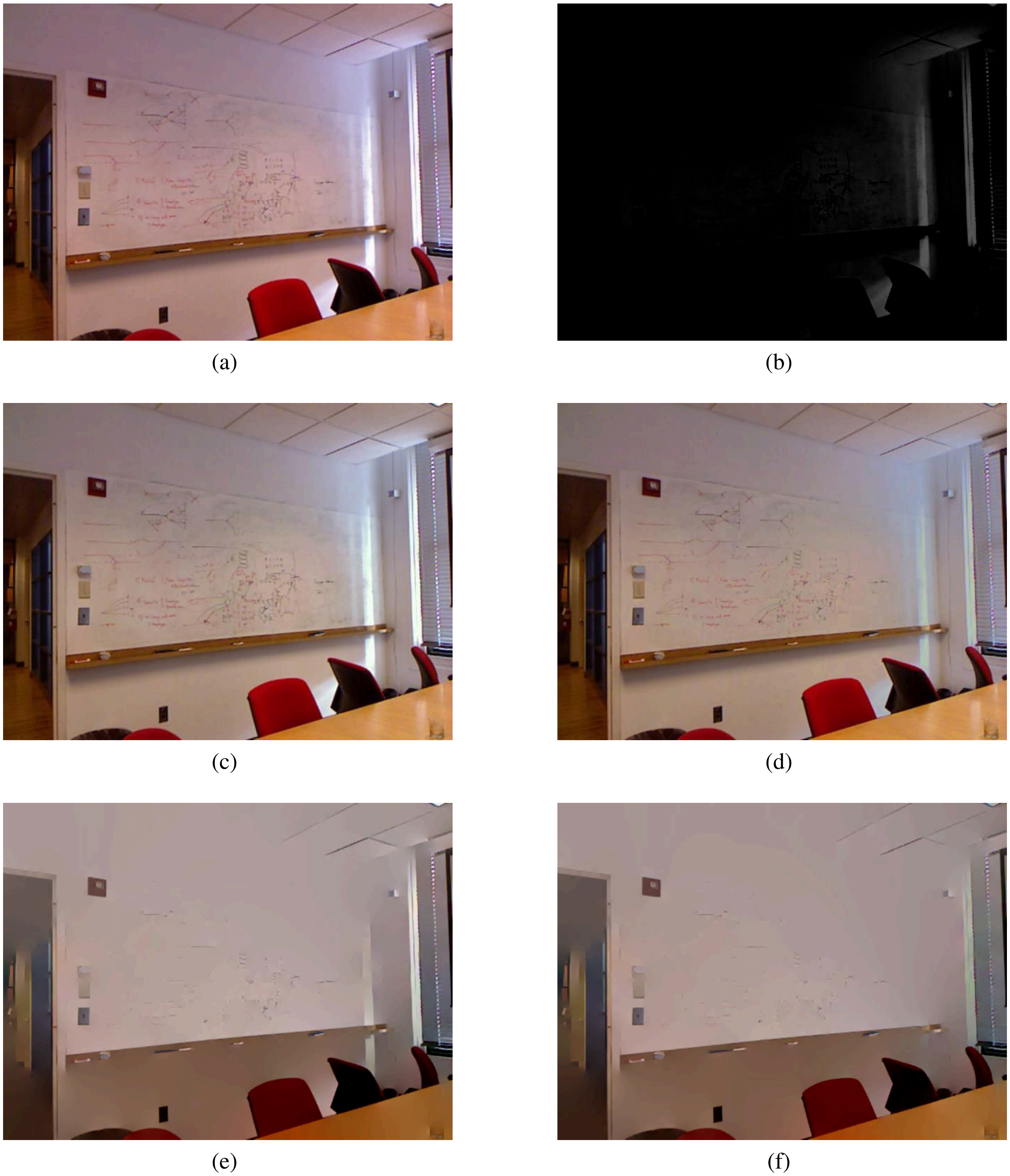
(c)

(d)

(e)

(f)

Figure 3.3: Estimated specularity(Figure 3.3b) from the image 3.3a. Chromaticity-normalized image is shown in Figure 3.3c. Diffuse image obtained is shown in Figure 3.3d. Estimated albedos Figure 3.3e for Figure 3.3c and Figure 3.3f for Figure 3.3d.



(a)

(b)

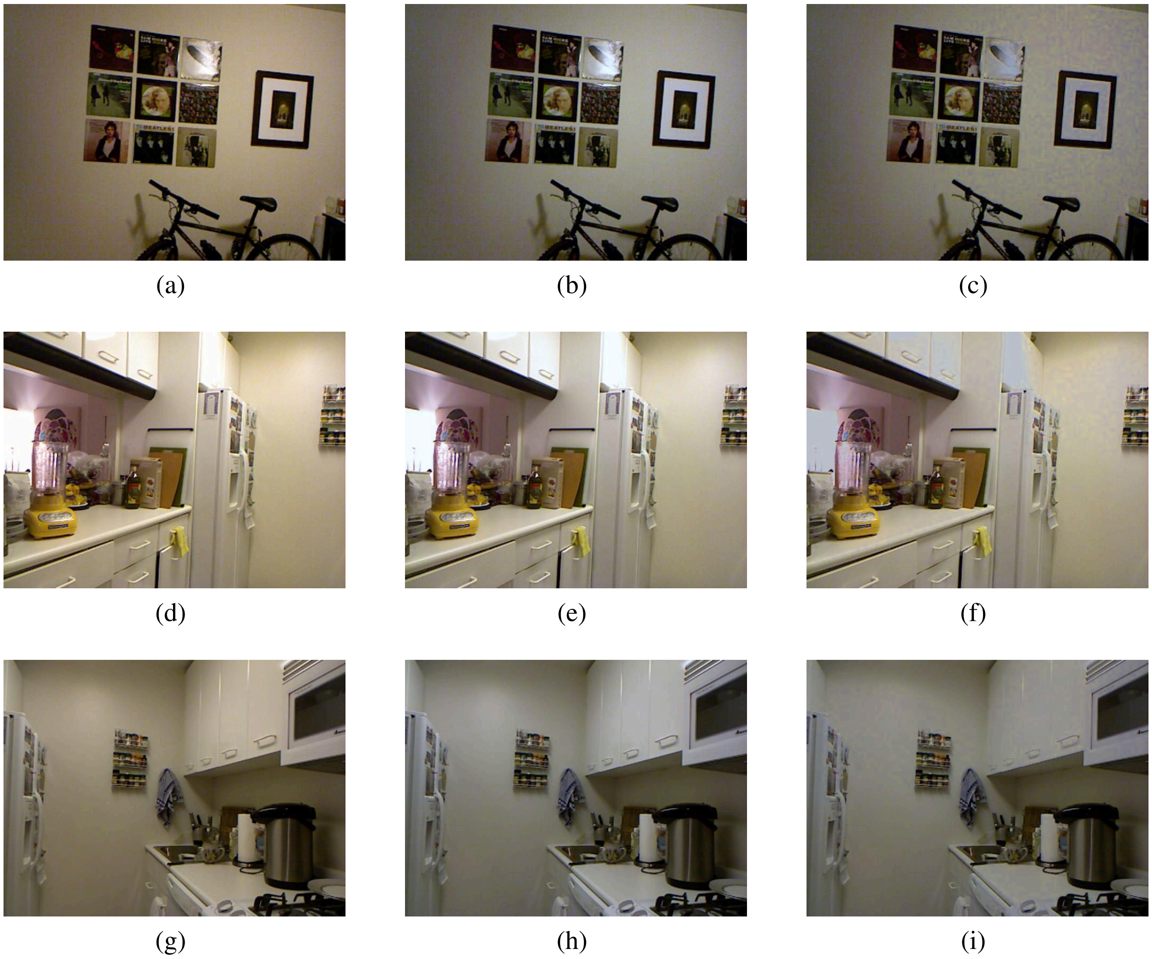
(c)

(d)

(e)

(f)

Figure 3.4: Estimated specularity(Figure 3.4b) from the image 3.4a. Chromaticity-normalized image is shown in Figure 3.4c. Diffuse image obtained is shown in Figure 3.4d. Estimated albedos Figure 3.4e for Figure 3.4c and Figure 3.4f for Figure 3.4d.



(a) (b) (c)

(d) (e) (f)

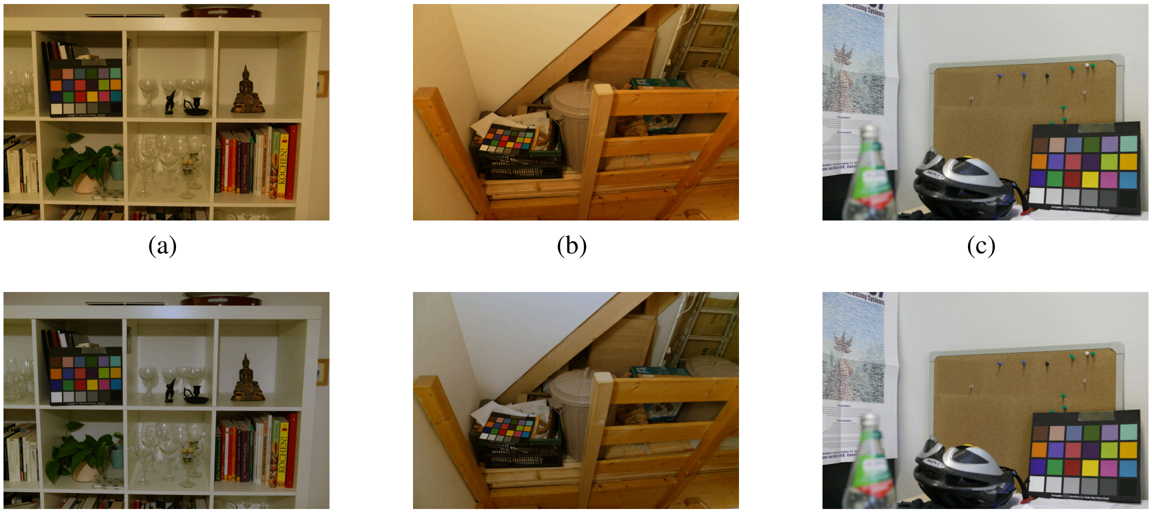
(g) (h) (i)

Figure 3.5: Original images are on the left, images in the middle column are chromaticity normalized and the corresponding diffuse components are on right.

Few of the results are shown in Figure 3.6 with first row (Figures 3.6a, 3.6b and 3.6c) displaying the original images from the dataset and the second row (Figures 3.6d, 3.6e and 3.6f respectively) displaying the results of the algorithm. The results from these images can be visually verified by observing the color of the white square appearing at the bottom left corner in the Color Checker.

3.3.1 Changing the light intensity

The separated specularity component, gives a simple tool to modify the light chromaticity of the scene. The specularity component tells the places where the light is affecting the intensity values. By asking a user to input which light chromaticity he wants, we can change



(a) (b) (c)

(d) (e) (f)

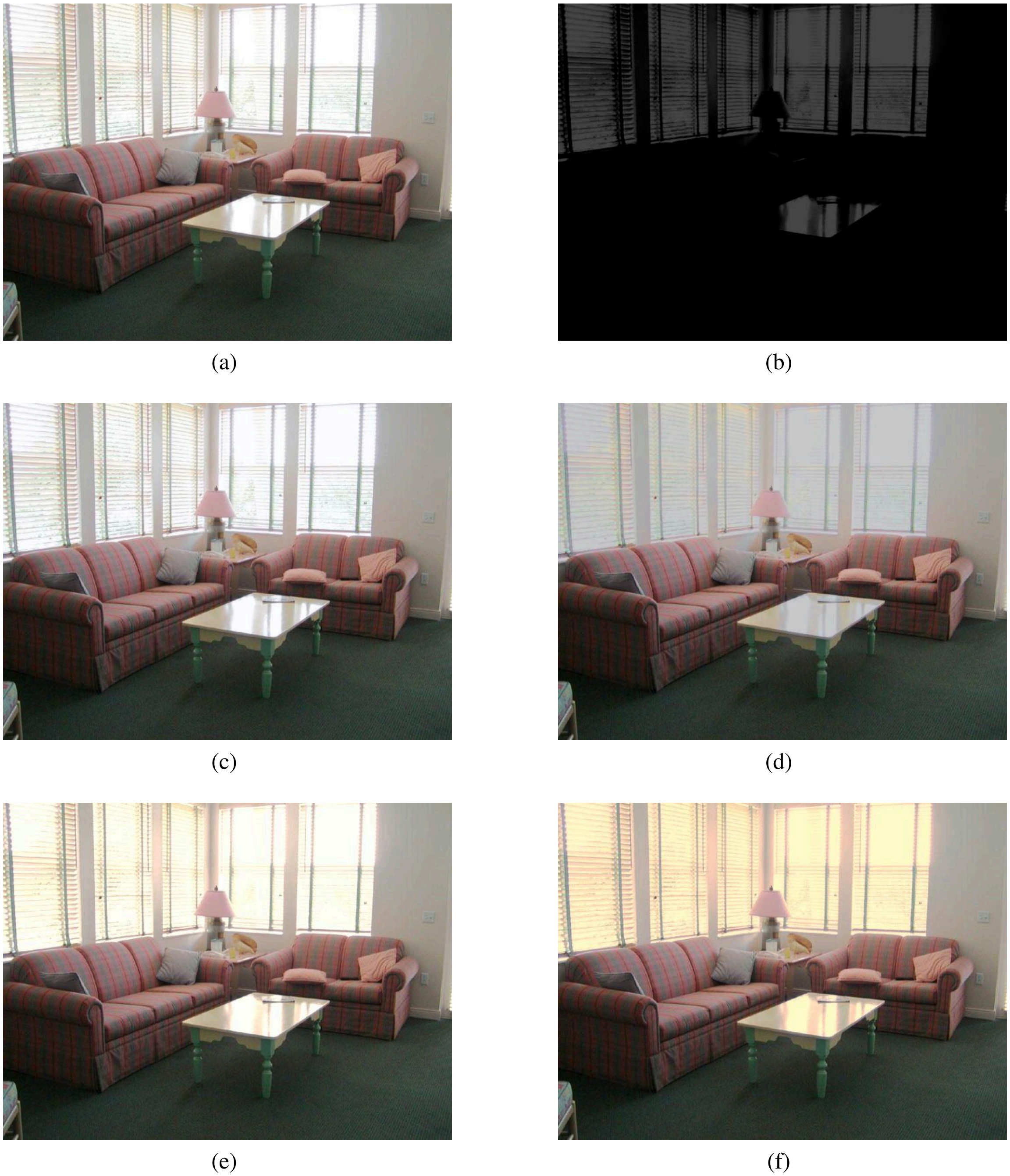
Figure 3.6: The first row shows the original images. The second row shows the better normalized images. Results are verified using the Macbeth Color Checker.

light colors as per our needs. This is done by just adding the chromaticity-normalized diffuse to the specular component enhanced by the input chromaticity from the user.

Some of the results of the scenes are shown in Figures 3.7e and 3.7f, corresponding to the image shown in Figure 3.7a. Before changing the intensity of light, it has been normalized as shown in Figure 3.7c. Another set of result on image 3.8a is shown. Lights of various intensities are shown in Figures 3.8b, 3.8c and 3.8d.

3.4 Limitations

- This thesis does not compare any different specularity removal method and solely depends on the result obtained by (Shen and Cai, 2009)'s algorithm.
- The images obtained were taken directly from the internet or from certain database, where the actual light intensity is not known.
- The algorithm explained in this thesis computes the results for checking color constancy using Macbeth color checker and relies on the fact that since, the colors in the Macbeth color checker are coming exactly as they should be after normalization, the light chromaticity must have been estimated correctly.



(a)

(b)

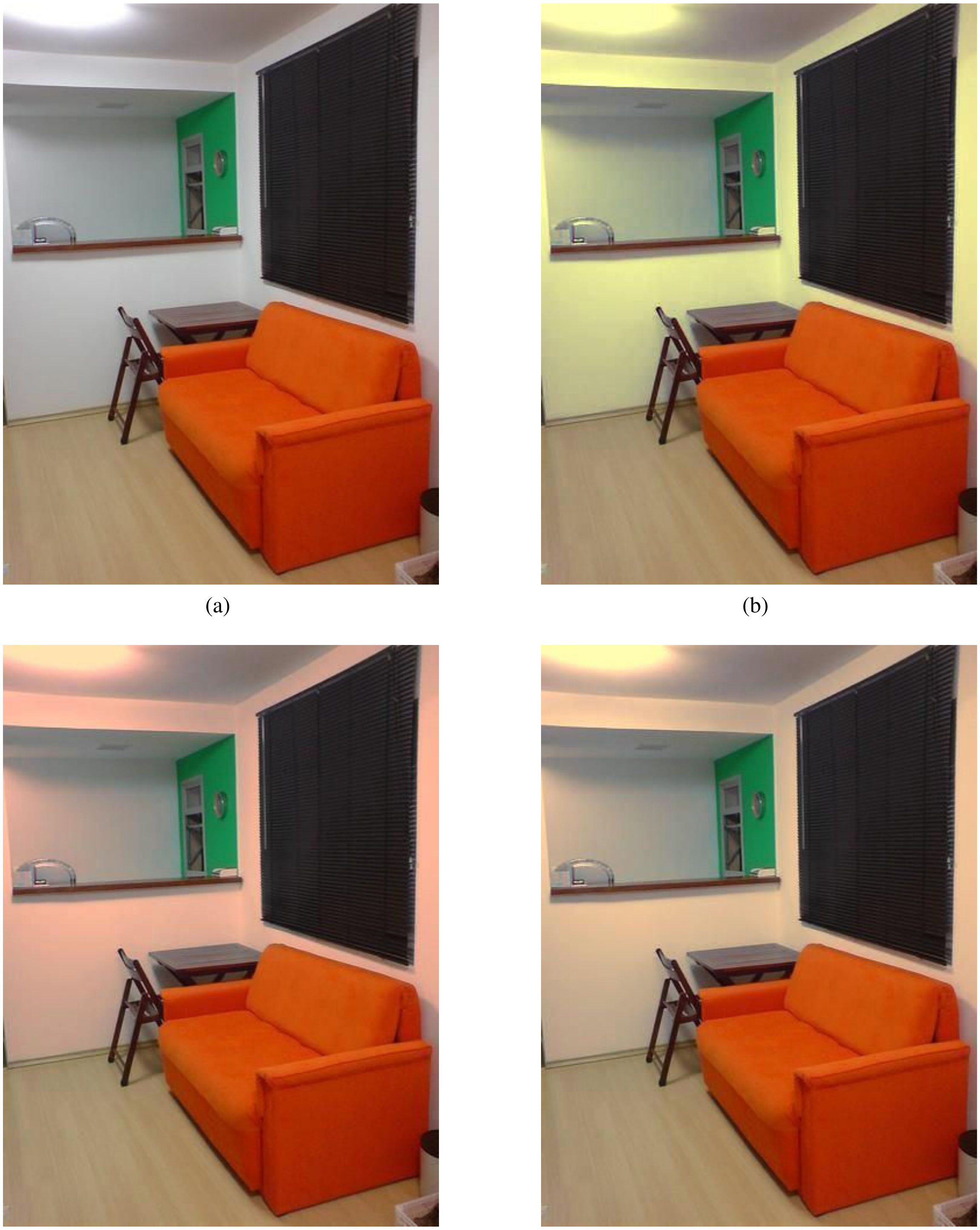
(c)

(d)

(e)

(f)

Figure 3.7: Changing intensity and color of light source. Image 3.7a is the original image. Image 3.7b shows the separated specularity. Image 3.7c is chromaticity-normalized image, whereas image 3.7d is the diffuse. Images 3.7e and 3.7f show the change of light color.



(a)

(b)

(c)

(d)

Figure 3.8: Changing intensity and color of light source. Image 3.7a is the original image. Normalization not required since only white light is present. Images 3.8b, 3.8c and 3.8d show the change of light color.

CHAPTER 4

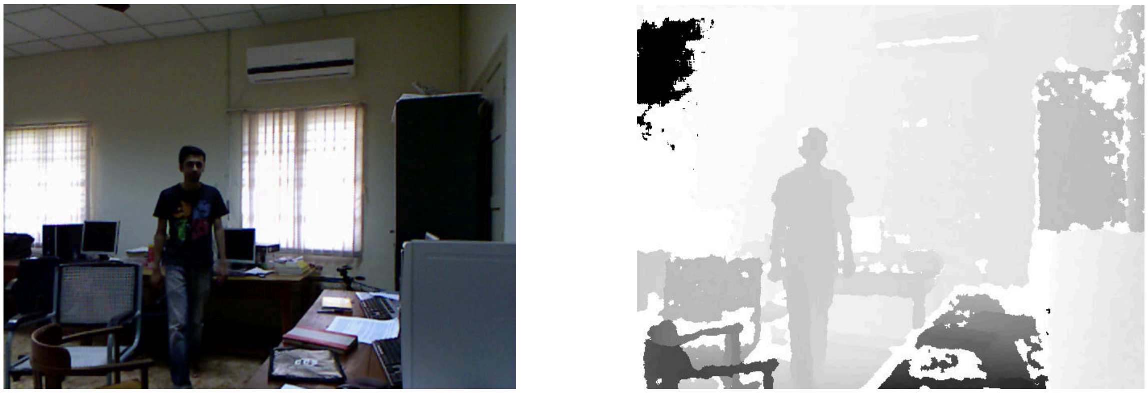
Lighting with normals

Till now, we have considered two independent single-image based approaches to get geometry and lighting information as discussed in Chapter 2 and Chapter 3, respectively. It is well known that geometry and depth go hand-in-hand. However, depth is difficult to estimate from just a single image and is an open-ended problem. In spite of inferring 3D geometry of the scene, it is difficult to give depth sense to a scene from just a single image. There have been learning based approaches (Hoiem *et al.*, 2005), (Saxena *et al.*, 2005), (Saxena *et al.*, 2009) as mentioned in Chapter 1. Inferring normals from a scene is also an extremely difficult task in computer vision. However, there have been works (Sun *et al.*, 2008), (Wu *et al.*, 2008) related to this also.

On the other hand, there has been a lot of development in the field of depth-sensing cameras and they provide a great platform for image processing engineers. In this thesis, we use *Kinect* as a depth-sensor camera to get the depth map of the scene. The depth, geometry and specular information helps to find the position of light source. In this thesis, we propose a method to find the single light source as well as multiple light sources. This chapter also tries to discuss some aspects related to lighting using dichromatic reflection model and geometry of the scene.

4.1 *Kinect*: An overview

The working principle of the *Kinect* is based on projection of a known IR-pattern which is detected by an IR CMOS camera. Calibration between the projector and the IR camera is done by the manufacturer. The depth map of a scene is calculated from the intensity of the IR light pattern observed by the IR camera. The depth information has a very high accuracy in the range of $0.8 - 1.2$m; it has a medium level of accuracy from $1.2 - 2$m range and beyond 2.0m, it has a low level of accuracy. The depth error increases quadratically from few millimeters at 0.5m distance to 7cm at the maximum operating range of 5m (Khoshelham and Elberink, 2012). A depth range of $0.4 - 4.0$m is considered to be acceptable range of operation. *Kinect* has a normal RGB camera also. The resolution of both the IR camera and the RGB camera is 640x480 @ 30fps with a point cloud density of about $300,000$. However, because of the spatially located cameras, the RGB image and the depth image need to be calibrated. Usually, the data obtained from Kinect without doing any processing appears like the one shown in Figures 4.1a and 4.1b. The correction of systematic errors is a prerequisite for the alignment of the depth and color data, and relies on the identification of the mathematical model of depth measurement and the calibration parameters involved (Khoshelham and Elberink, 2012).



(a)

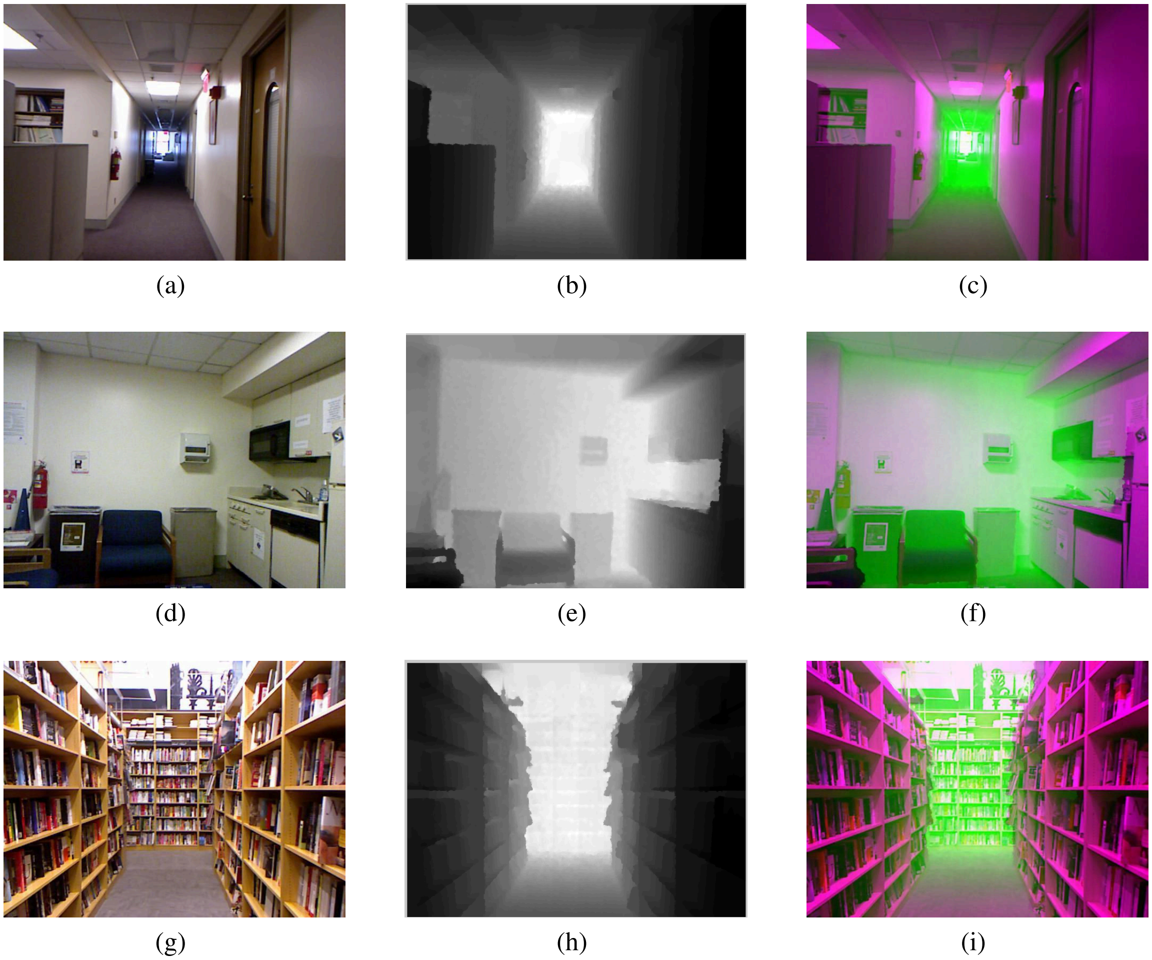
(b)

Figure 4.1: Raw result from the *Kinect*. 4.1a shows the color camera output. 4.1b is the depth captured from IR camera.

Normal calibration techniques cannot be used because of the presence of IR camera. For this purpose, *Kinect Calibration tool-kit* (Burrus, 2011) is used. Few of the problems related to *Kinect* are:

- Low resolution of depth and RGB image; while the region which overlaps in both color and depth image is further small compared to the total resolution.
- Non-linear accuracy of the depth data. The accuracy deteriorates drastically as the distance increases.
- The depth data is of low resolution and also is affected by the presence of occlusions since the camera is a single point of view.
- The pair of IR emitter-sensor fails to provide depth information in presence of light sources saturating the depth sensor, low albedo surfaces, object boundaries having high curvatures, specular reflections and transparent objects.

To have a good knowledge of the depth of the scene, it is important that the depth image is aligned with RGB image as well as the depth image is continuous (i.e., the holes occurring in a depth map are filled). To overcome all this, a dataset of indoor scenes provided by NYU (Silberman and Fergus, 2011) is used. The data set consists of 2284 indoor images having aligned and filtered depth-map consisting of various indoor scenes across NYU and also with different lighting conditions.


(a)

(b)

(c)

(d)

(e)

(f)

(g)

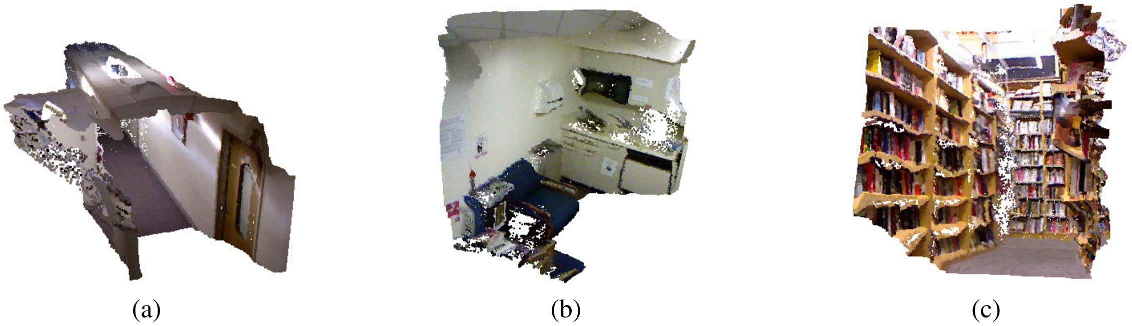
(h)

(i)

Figure 4.2: Examples from NYU dataset. First column shows the RGB color image from the *Kinect*. Second column shows the depth image. Third column shows the overlap of the color and the depth image.

Some of the examples are shown in Figure 4.2. Figures 4.2a, 4.2d and 4.2g shows the color images captured from *Kinect* sensor. Figure 4.2b, 4.2e and 4.2h show the corresponding depth maps. Figures 4.2c, 4.2f and 4.2i show the overlap between the color and the depth image confirming the alignment between the two images.

Figure 4.3 show the 3D models of the images. The figure shows the 3D models as viewed in *Meshlab*.


(a)

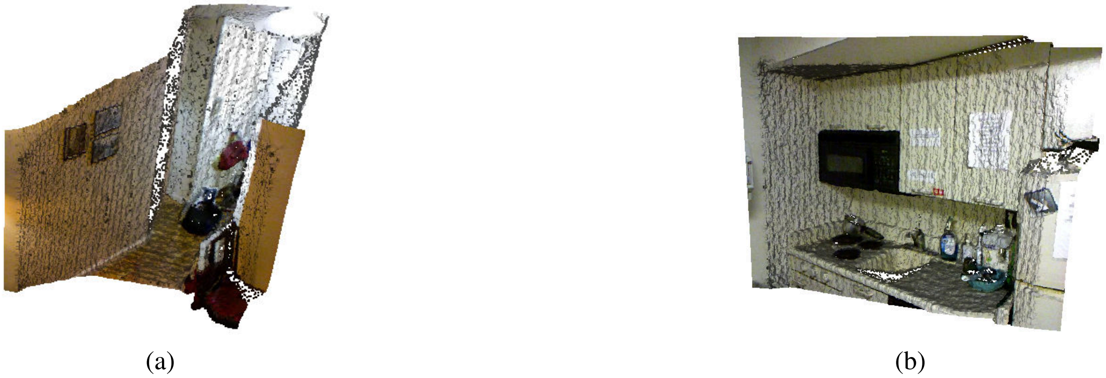
(b)

(c)

Figure 4.3: 3D models corresponding to images in 4.2a, 4.2d and 4.2g respectively.

Exploiting this depth information available from *Kinect* sensor, we generate the 3D model, calculate the normal at every pixel and then use this information to detect the source of light. While estimating the normal, the normal is calculated by considering neighboring points, estimating tangent planes and building Riemman graphs. This method is followed in *Meshlab* and MATLAB *surfnorm*. However, the normal does not come out smooth (refer to Figures: 4.4a and 4.4b). Hence, before using these normal, we use a function to smooth these normal and restrict it to be either one of these 6 orientations:

$$[\pm1, 0, 0]\,,\ [0, \pm1, 0]\,,\ [0, 0, \pm1]$$


(a)

(b)

Figure 4.4: 3D models showing the effect of light. The normals are actually wrongly oriented and hence those shadows are appearing.

4.2 Exploiting 3D information

In section 3.2, we separated the specularity from a single image. The approach taken in Chapter 3 was purely a color space based approach. Here, we bring in the information of geometry. The specularity at a certain point can also be modeled in the following way. Consider Figure 4.5.

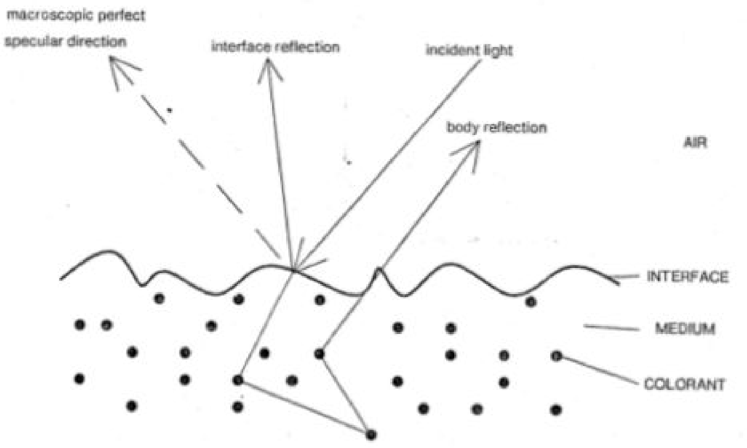


Figure 4.5: Reflection of light according to dichromatic reflection model (Image source: (Neverova, 2012)).

It is not always the case that the specular reflection occurs in a pure mirror-like reflection. For simplifying purpose, we can assume that the reflection is very likely in a small range of angles around the pure mirror reflection and hence, it can be modeled as follows:

$$\mathbf{r} \;=\; \frac{1}{\|\mathbf{d}\|}\left(\mathbf{d}_{\|} \;-\; \mathbf{d}_{\perp}\right) \;=\; \frac{\mathbf{d} \;-\; 2\mathbf{d}_{\perp}}{\|\mathbf{d}\|} \;=\; \frac{\mathbf{d} \;-\; 2(\mathbf{n},\mathbf{d})\mathbf{n}}{\|\mathbf{d}\|} \tag{4.1}$$

where $\mathbf{r}$ is the pure mirror reflection, $\mathbf{d}$ is the direction vector joining the point and the source of light and $\mathbf{n}$ is the normal at that point. Now whatever specularity is seen in the camera, it is because of the fact that this particular $\mathbf{r}$ is aligned with the viewing direction $\mathbf{v}$. Then, it can be shown from equation 4.1 and Figure 4.5 that whatever specularity is being observed in the scene, if back-projected at those specular points will give a set of vectors $\mathbf{d}$ pointing in the direction of incident lights.

$$\mathbf{r} \;-\; 2(\mathbf{n},\mathbf{r})\mathbf{n} \;=\; \frac{\mathbf{d}}{\|\mathbf{d}\|} \tag{4.2}$$

The intersection of all such $\mathbf{d}$'s gives us a potential light source. Writing the equation of a line in 3D as

$$\mathbf{l_i} \;=\; \mathbf{c_i} \;+\; t\mathbf{d_i} \tag{4.3}$$

where $\mathbf{c_i}$ is a specular point in the image, $\mathbf{d_i}$ is corresponding direction of light source (4.2) at that point and t is just a scalar quantity.

Using concepts from 3D co-ordinate geometry, it can be determined whether two lines are parallel or coinciding, or intersecting exactly in one point or are skew lines. It was found that in almost every case, the lines turned out to be skew. In general, it can be said that the light source in an indoor environment is a tube, or a cluster of tubes, or a bulb. It is very obvious from this fact that whatever light reflecting from any surface need not come from one single point. It is very difficult to model such a lighting system as a single point.

4.2.1 Getting candidates for detection of light source

The following is the method followed to get the possible light direction candidate.

1. As explained in the Section 3.2, we separate the specularity in the image. The specularity seen in an image is same for all the three color channels. And hence, only a gray scale (single channel) specularity is considered.

2. The specularity is normalized in the scale of 0 to 255 and only specular components having intensities above 100 are considered. Since, specular components occur very close to each other, mostly as connected components, it is safe to assume that they all lie on the same plane and hence can be clustered together as one. In a window of 3×3, the close neighbors are averaged for the ease of computation. Depending on how the specularity is observed in the scene, the size of this window is modified.

3. Light direction candidates at these specular locations obtained after filtering, are evaluated according to the formula given by equation 4.2. The equation of line passing at a particular specular point and having direction given by equation 4.2 is given by equation 4.3.

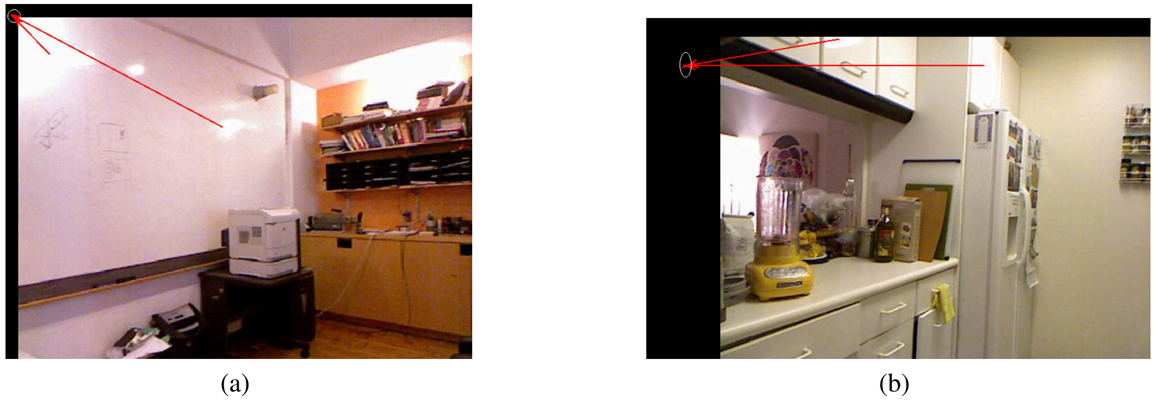
4.2.2 Single light source detection

In case if a single light source is affecting the scene, an approach similar to (Lv *et al.*, 2009) is used to estimate the source of light. Given that a single light source is present, that light source can be approximated as a point light source. As discussed previously, the specularity in the image holds information about the light sources affecting the image. The specular component separated from an image helps to find the direction of light source. The pixels which are classified as specular tell the locations. The locations $\mathbf{c_i}$s and the direction of light source i.e., $\mathbf{d_i}$s calculated at those points by using equation 4.2 can be written in the form of lines $\mathbf{l_i}$s given by equation 4.3. Since, all such lines must come from a single light source, they all will intersect at a point $\mathbf{S} = [S_x, S_y, S_z]$. This means that, ideally, the distance of all such lines from this point must be zero and hence, the sum of distances of all such points must also be zero.

Considering the arguments in section 4.2 that all such lines can be skew, a light source detection problem can be modeled as an optimization problem (refer to equation 4.4).

$$\begin{aligned} f(\mathbf{S}) &= f(S_x, S_y, S_z) \\ &= \sum_{i=1}^{k} [\text{Distance of point } S \text{ from } l_i] \qquad (4.4) \\ &= \sum_{i=1}^{k} \frac{\mid (L_{x,i} - L_{y,i})(c_{z,i} - S_z) + (L_{y,i} - L_{z,i})(c_{x,i} - S_x) + (L_{z,i} - L_{x,i})(c_{y,i} - S_y) \mid}{\|L_i\|} \end{aligned}$$

where k is the number of such lines, which means the number of significant specular pixels found in the image. Since this is a unconstrained optimization, MATLAB function *fminunc* is used to optimize equation 4.4 with respect to the variable $\mathbf{S}$ with initial condition as the origin (or camera location). The results obtained are convincing for the images where there is for sure a single light source present and the cost of the function remains very low. Although, this approach suffers from a drawback of getting stuck in a local minima, this gives a fairly decent idea of the light source direction correctly. The optimization is dependent on the initial condition and hence, the result can be improved with a better initialization of the light source. Few of the results are shown in Figures 4.6a and 4.6b.



(a) (b)

Figure 4.6: Light positions estimated using the logic explained in Section 4.2.2 and then projected on the image plane to display the result. The rays are drawn manually just to explain the concept.

4.2.3 Results for single light source detection

Since the images being considered for the experiment belonged to a dataset, no groundtruth about the position of light source was available. For single light source, the cost of the optimization was used as a measure to tell the accuracy of the algorithm. By dividing the cost into three classes as GOOD, OK and BAD, the result can be classified as shown in Table 4.1. The images under consideration consisted of those having not only a single light source but also multiple light sources, along with sunlight affecting few of the images. Also few of the images consisted of light source appearing inside the camera point of view and that meant that the light source was classified as specularity and was considered in the calculation for the cost function.

GOOD	OK	BAD
$\frac{14}{35} = 40\%$	$\frac{12}{35} = 34.29\%$	$\frac{9}{35} = 25.71\%$

Table 4.1: Analysis of the single light source estimates performed over 35 images with depth maps available.

4.2.4 Multiple light sources detection

As discussed in section 4.2, it is not proper to model a light source as a point light source. Even when there is only a single light source illuminating the image, it is hard to tell whether that light source is captured in the image, or it is outside the image and also it not known whether the light source is artificial or natural. There have been work on outdoor scenes to detect the source of light or direction of natural light (Lalonde *et al.*, 2012), (Frahm *et al.*, 2005). However, those algorithms are tailored only for outdoor purposes. By keeping the approach completely mathematical, a multiple light source detection algorithm is proposed.

1. The specularity obtained from an image is first normalized in the range 0 to 255. Using few morphological operations, the centroids of the contours formed is obtained. These centroids are probably the candidates to be considered for calculating light sources.

2. The distance between all possible pairs among the candidates is calculated given by the formula:

$$\begin{aligned} \text{Distance} &= \frac{\mathbf{c_1 c_2} \cdot \mathbf{N}}{\|\mathbf{N}\|} && (4.5) \\ \text{where } \mathbf{N} &= \mathbf{d_1} \times \mathbf{d_2} && (4.6) \end{aligned}$$

   where $\mathbf{c_i}$ and $\mathbf{d_i}$ are as explained in equation 4.3 and section 4.2.2.

3. The intersection point of only those pairs is considered for which the distance is below a user-defined threshold. For our experiments, we consider this threshold as 10cms. To find the intersection point, a least square solution is applied.

4. Once all such possible light source position candidates are obtained, we compute the Euclidean distance between all these points. Considering each candidate as a node in a graph, a link in the graph is defined only if the Euclidean distance is less than a certain threshold (same as above). A cluster of such connected nodes is then found to compute the light source positions.

4.3 Dichromatic Reflection model

Both the methodologies explained in sections 4.2.2 and 4.2.4 are crude. In a later section, we discuss about an optimization technique used to correct the position of light source as explained in (Neverova *et al.*, 2012), (Karsch *et al.*, 2011). The optimization technique discussed is based on rendering a scene knowing the geometry and the information of light source. In order to render a scene, the scene formation model need to be studied. Hence, we consider a simple dichromatic reflection model as explained in (Shafer, 1985) to see how a light source affects the scene. It is very similar to what is explained in Phong's model (Phong, 1975). With the 3D world co-ordinates and normals available now, the reflection model considered in this thesis brings in a new flavor to this problem.

Revisiting the reflection model, the intensity at any pixel in the image is because of the body

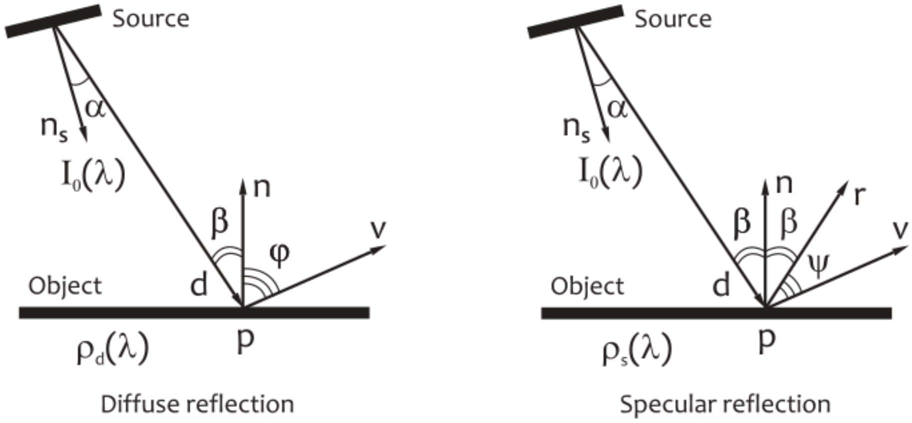


Figure 4.7: The diffuse component and the specular component (Image source: (Neverova, 2012)).

reflection or the interface reflection. Referring to the figure 4.7,

$$\begin{aligned} I(\lambda,\beta,\varphi,\psi,p) &= I_b(\lambda,\beta,\varphi,p) + I_{int}(\lambda,\beta,\psi,p) \\ &= m_b(\beta,\varphi,p)c_b(\lambda,p) + m_{int}(\beta,\psi,p)c_{int}(\lambda,p) \end{aligned} \tag{4.7}$$

where λ is the wavelength of light, $I(\lambda,\beta,\varphi,\psi,p)$ is the intensity at pixel p due to each spectral component (in simple words, color at p), m_{int} and m_b depend on the object and light source geometry and describe the intensity of the reflected light regardless of the light wavelength and material properties, c_{int} is the surface color under illumination and c_b is the color property of the material under the given illumination (Neverova *et al.*, 2012). Here c_{int} and c_b are the relative spectral distribution independent of the geometry. From equation 3.3, it is known that,

$$I(p) = D(p) + C(p) \tag{4.8}$$

where $D(p)$ is the diffuse component and $C(p)$ is the specular component. Equations 4.7 and 4.8 clearly show that

$$\begin{aligned} D(p) &= m_b(\beta,\varphi,p)c_b(\lambda,p) \text{ and} \\ C(p) &= m_{int}(\beta,\psi,p)c_{int}(\lambda,p) \end{aligned} \tag{4.9}$$

Also from photometry (Neverova, 2012), it is known that, for a single light source, the diffuse component can be expressed as: (refer to Figure 4.7 for notations)

$$\begin{aligned} D(\lambda,p) &= \rho_d(\lambda,p)I_0(\lambda)\frac{\cos\alpha\ \cos\beta}{\mathbf{d}^2} \\ &= -\rho_d(\lambda,p)I_0(\lambda)\frac{(\mathbf{n_s},\mathbf{d})(\mathbf{n},\mathbf{d})}{\|\mathbf{d}\|^4} \end{aligned} \tag{4.10}$$

where $\rho_d(\lambda,p)$ is the diffuse reflection coefficient, $I_0(\lambda)$ is the light source intensity on the given wavelength and in the direction perpendicular to the light source surface. If we consider N light sources, then the diffuse component can be expressed as:

$$D(\lambda,p)_N = -\rho_d(\lambda,p)\sum_{i=1}^{N} I_{0,i}(\lambda)\frac{(\mathbf{n_{s,i}},\mathbf{d_i})(\mathbf{n},\mathbf{d_i})}{\|\mathbf{d_i}\|^4} \tag{4.11}$$

The specular component due to a single light source can be expressed as:

$$\begin{aligned} C(\lambda,p) &= \rho_s(\lambda,p)I_0(\lambda)\cos\alpha(\cos\varphi)^k \\ &= \rho_s(\lambda,p)I_0(\lambda)\frac{(\mathbf{n_s},\mathbf{d})}{\|\mathbf{d}\|^{k+1}}(\mathbf{v},(\mathbf{d}-2(\mathbf{n},\mathbf{d})\mathbf{n}))^k \end{aligned} \tag{4.12}$$

where $\rho_s(\lambda,p)$ is the specular reflectance and k can be defined as a parameter set experimentally. (In original literature the value for k has been taken as $(0,...,10)$ (Phong, 1975), however, we take arbitrary value, typically $k = 55$ (Neverova *et al.*, 2012)).

For multiple light sources, specular component can be written as:

$$C(\lambda,p)_N = \rho_s(\lambda,p)\sum_{i=1}^{N} I_{0,i}(\lambda)\frac{(\mathbf{n_{s,i}},\mathbf{d_i})}{\|\mathbf{d_i}\|^{k+1}}(\mathbf{v},(\mathbf{d_i}-2(\mathbf{n},\mathbf{d_i})\mathbf{n}))^k \tag{4.13}$$

In practice, the specular reflectance is found to be constant for all pixels i.e. $\rho_s(\lambda,p) = \rho_s$. This also agrees with the fact that while normalizing the image in Chapter 3, we were normalizing the entire scene and not just the position where the specularity is present. Hence, from equations 4.8,4.11 and 4.13, the final light distribution which is observed from a camera can be written as

$$\begin{aligned} I(\lambda,p) &= -\rho_d(\lambda,p)\sum_{i=1}^{N} I_{0,i}(\lambda)\frac{(\mathbf{n_{s,i}},\mathbf{d_i})(\mathbf{n},\mathbf{d_i})}{\|\mathbf{d_i}\|^4} \\ &+\rho_s\sum_{i=1}^{N} I_{0,i}(\lambda)\frac{(\mathbf{n_{s,i}},\mathbf{d_i})}{\|\mathbf{d_i}\|^{k+1}}(\mathbf{v},(\mathbf{d_i}-2(\mathbf{n},\mathbf{d_i})\mathbf{n}))^k \end{aligned} \tag{4.14}$$

Equation 4.14 can be expressed in terms of it's three color components as

$$\begin{aligned}\begin{bmatrix}R\\G\\B\end{bmatrix}_{N,p} &= -\sum_{i=1}^{N}\begin{bmatrix}\rho_{d,r}(p)R_{s,i}\\ \rho_{d,g}(p)G_{s,i}\\ \rho_{d,b}(p)B_{s,i}\end{bmatrix}\left[\frac{(\mathbf{n_{s,i}},\mathbf{d_i})(\mathbf{n},\mathbf{d_i})}{\|\mathbf{d_i}\|^4}\right]\\ &+\rho_s\sum_{i=1}^{N}\begin{bmatrix}R_{s,i}\\G_{s,i}\\B_{s,i}\end{bmatrix}\left[\frac{(\mathbf{n_{s,i}},\mathbf{d_i})}{\|\mathbf{d_i}\|^{k+1}}(\mathbf{v},(\mathbf{d_i}-2(\mathbf{n},\mathbf{d_i})\mathbf{n}))^k\right]\end{aligned} \tag{4.15}$$

From equations 4.7, 4.14 and 4.15, the four components can be expressed as:

- Illuminant and material spectral characteristics:

$$c_{b,i}(p) = \begin{bmatrix}\rho_{d,r}(p)R_{s,i}\\ \rho_{d,g}(p)G_{s,i}\\ \rho_{d,b}(p)B_{s,i}\end{bmatrix} \tag{4.16}$$

- Diffuse shading distribution

$$m_{b,i}(p) = \left[\frac{(\mathbf{n_{s,i}},\mathbf{d_i})(\mathbf{n},\mathbf{d_i})}{\|\mathbf{d_i}\|^4}\right] \tag{4.17}$$

- Light chromaticity or specular reflecance coefficient:

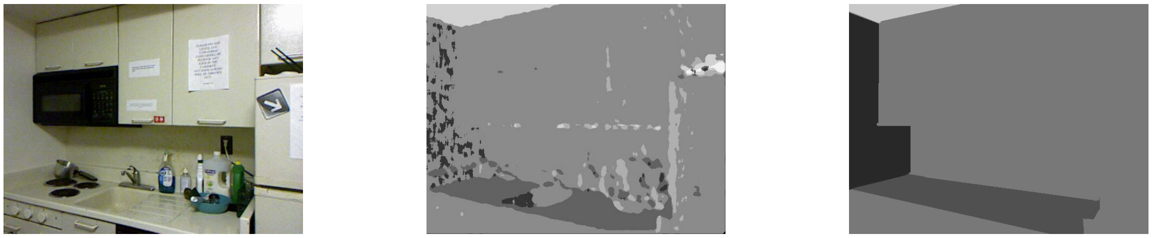
$$c_{int,i}(p) = \rho_s\begin{bmatrix}R_{s,i}\\G_{s,i}\\B_{s,i}\end{bmatrix} \tag{4.18}$$

- Specular shading distribution:

$$m_{int,i}(p) = \left[\frac{(\mathbf{n_{s,i}},\mathbf{d_i})}{\|\mathbf{d_i}\|^{k+1}}(\mathbf{v},(\mathbf{d_i}-2(\mathbf{n},\mathbf{d_i})\mathbf{n}))^k\right] \tag{4.19}$$

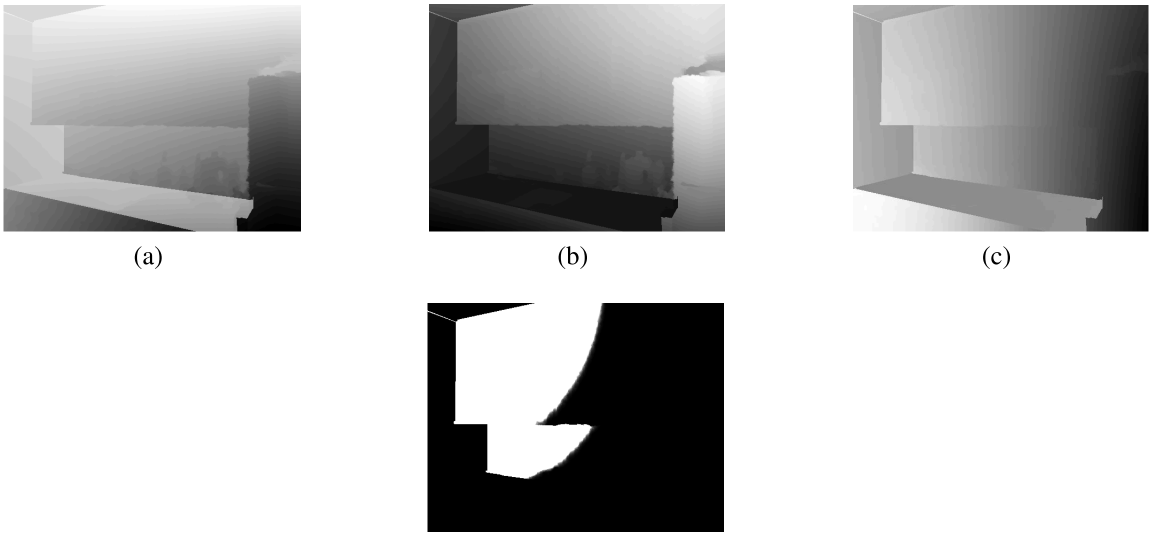
Using these formulae, one can recreate a shading as well as specularity knowing the 3D model of the scene. One such example (Figure 4.8a and its corresponding manually assigned normal map in Figure 4.8c) on which the Dichromatic Reflection Model was applied to generate the shading and specularity is shown in Figures 4.9a, 4.9b and 4.9c.

A point to be noted is that these results were obtained just by randomly changing the positions and direction of light source and considering only a single light source. But this exercise allows us to see how a position of light source is related to its shading and specularity (Figure 4.9d). This leads us to make use of an optimization technique to improve upon the estimate of light source based on shading and specularity obtained.



(a) (b) (c)

Figure 4.8: One example whose original image is shown in 4.8a, erroneous normal map obtained using *Meshlab* 4.8b and manually corrected normal map in 4.8c.



(a) (b) (c)

(d)

Figure 4.9: Various shadings (4.9a, 4.9b and 4.9c) obtained by changing the position and direction of light source. 4.9d shows a random specularity obtained in an image.

4.3.1 Rendering through Optimization

As explained in section 4.3, any image can be formed by considering equations 4.14 and 4.15. An optimization method to improve the estimated light source(s) based on the reflection model is explained in (Neverova *et al.*, 2012) and (Karsch *et al.*, 2011) is used. We optimize for the light source parameters by rendering the diffuse shading and specularity using the geometry and the current light source estimates (Neverova *et al.*, 2012).

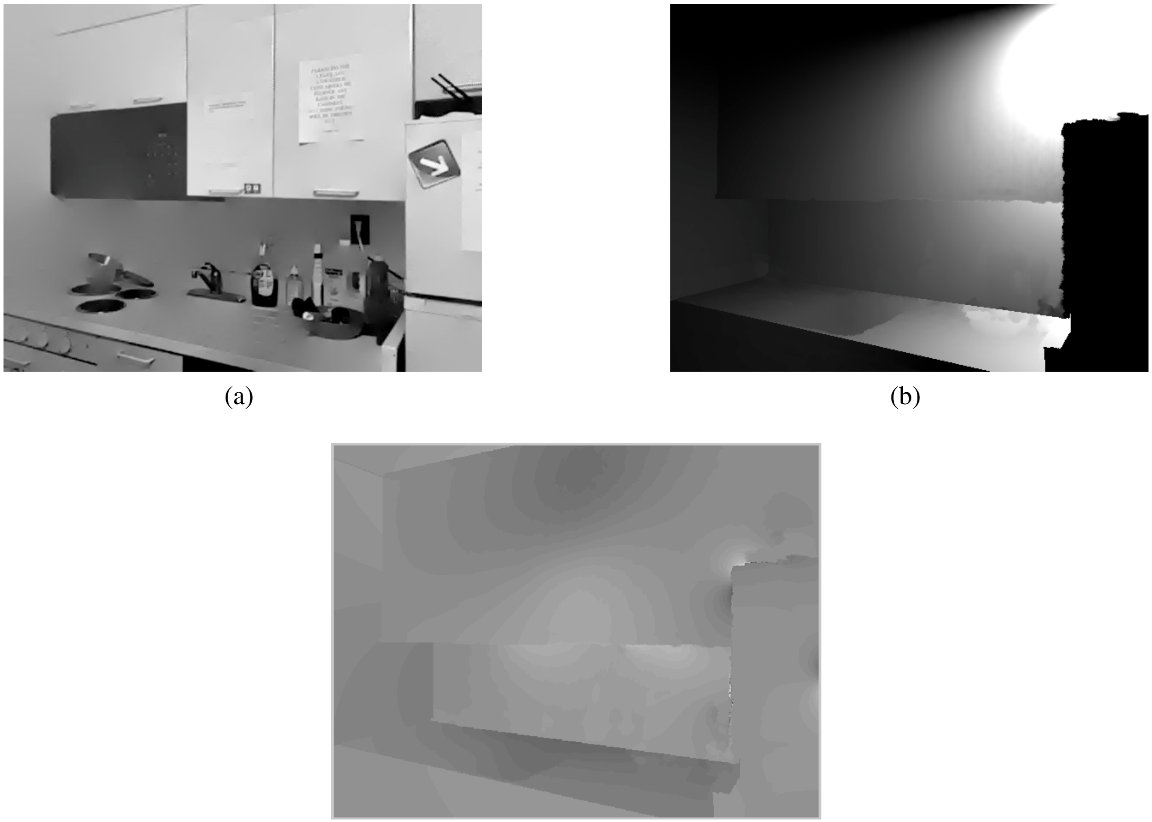
$$L = \text{argmin}_{L,C,S} \left\{ \sum_{p} \left[\alpha [C^0(p) - C(p)]^2 + \beta [S^0(p) - S(p)]^2 \right] + \sum_{i=1}^{N} \gamma [L_{0i} - L_i]^2 \right\} \quad (4.20)$$

where C^0 is specular shading or specularity obtained from pure image based approach, C is the specularity as calculated from the knowledge of 3D co-ordinates and normals, S^0 is diffuse shading obtained from pure image based approach and S is calculated by the knowledge of geometry, L_{0i} is the initial light source estimates obtained from sections 4.2.2 and 4.2.4, and L is the set of parameters, which we need to estimate. The dimension of L is $N \times 6$ where N is the number of light sources present and each light source has its 3 spatial co-ordinates and 3 parameters describing the lighting intensity. However, while performing experiments, we consider light chromaticity intensities to be constant for all the light sources in the scene and hence do not consider it as a metric to be changed during the optimization process. The third term in the optimization makes sure that the optimization doesn't diverge too much from the initial estimate. The basic idea is to make the rendered scene as close as possible to the original scene and in turn estimating the light source in a more reliable way. The quantities α, β and γ are weights and are decided experimentally to 1, 0.75 and $30 M_x M_y$ where M_x and M_y are the dimensions of the image (Neverova, 2012).

A result is shown in Figure 4.10 to explain how this optimization performs. Figure 4.10a shows the desired shading (i.e., the gray level of the albedo obtained). Figure 4.10b shows the shading obtained by using the single light source estimate as explained in section 4.2.2. Multiple light source estimates are obtained as explained in section 4.2.4. However, the result obtained for shading from those estimates does not convey much information and hence, not shown in this thesis. Giving these multiple light sources estimated as initial estimates for the optimization function, optimization is performed and the obtained light source positions are used to obtain the shading as shown in Figure 4.10c. The shading is obtained as a superposition of multiple shading images from different light sources.

4.4 Limitations and Shortcomings

- Equations 4.14 and 4.15 do not consider the shadows cast by objects on different parts of the scene. To make this possible, it is necessary to do cast detection for the scene. However, implementing cast detection in MATLAB was tried out and found to be not suitable computationally. It is completely a GPU based algorithm.
- There is no mathematical proof about the optimization (refer to equation 4.20) and it is not sure how much accurate and stable that system is going to be. It is purely based on the fact that lighting affects specularity and the diffuse shading of the scene and hence, this justifies the presence of these two terms in the optimization equation.
- All the graphics based approach like ray tracing, ray casting, which are used for rendering a



(a)

(b)

(c)

Figure 4.10: After optimization, the shading obtained using the corrected position of multiple light source is shown in Figure 4.10c. To illustrate, the shading obtained by a single light source is shown in Figure 4.10b. The original shading of the scene is shown in Figure 4.10a.

scene are not considered in this thesis. Also, in commercial rendering applications, off-the-shelf graphics tools are considered directly.

- Rendering a scene with a synthetic object or relighting of a scene have not been achieved completely although important steps in achieving so have been discussed and a platform for the same has been laid out.

However, the results do have significant importance as they have the capability to perform rendering of a scene just from the knowledge of geometry, lighting and depth and without using any rendering software. This in turn can help in performing rendering in scenarios without a GPU. With better knowledge about scene formation, the optimization function can be very well defined. Thus this particular topic has a huge potential for future work.

CHAPTER 5

Conclusions

In this thesis, an approach to get geometry and lighting information from a single image of an indoor scene was discussed. The approach was strictly restricted to a single image of an indoor scene. Relying on the fact that the architectural environment of the indoor environment has a very well-defined geometry, a simple algorithm was adopted from works by (Lee *et al.*, 2009) and others (Hedau *et al.*, 2009), (Rother, 2002), (Kovesi, 2000). Geometry estimate of indoor scenes from just a single image hints at what more can be done using just a single image. On the other hand, lighting information of the scene was also acquired from a single image by separating the specularity in the image as explained in (Shen and Cai, 2009) and (Neverova, 2012). Here, too, a very simple approach was used to tell us lot more information about the environment in which the photo was taken. With the kind of approach used, it was very easy to show that how a scene will appear in different lighting intensities.

Depth being a difficult task to obtain from a single image, we relied on using a depth sensor like *Kinect* to get the depth of the scene under consideration. This depth, when used with the previous results obtained, made way to exploit the geometry in the scene and use the knowledge of lighting information to predict the positions of light sources which are there around the scene. We proposed a simple way to do the initial estimates of a single light source (if present) and also, a mechanism for detecting multiple light sources as well as modeling multiple light sources as point light sources was proposed. Though these methods were very crude, they were important because of the nature of optimization which was used to refine these light source position estimates. Since, the optimization was unconstrained and had a high chance of getting stuck in local minima, the result obtained depended on the initial estimates. By using dichromatic reflection model and the initial estimate of light source, scene formation model was studied and using this a rendered scene was compared with original scene to correct for the light source positions. We believe that there can be further improvement to this optimization algorithm.

We could have obtain data from *Kinect* to perform our experiments and have the groundtruths available with us. But the problem with *Kinect* data was low resolution of depth image and other problems associated with it as explained in section 4.1. That is why we relied on *Kinect* data set provided by NYU (Silberman and Fergus, 2011), which was free of all the problems explained in that section. Thus, this thesis allowed us to look at further extensions which can be made by harnessing the *Kinect* data properly. One such possibility is that by registering the consecutive depth image taken by moving the *Kinect* across the indoor scene, one can not only obtain an increased point density, but also create a complete point cloud of an indoor environment possibly in real time (Khoshelham and Elberink, 2012).

Keeping the entire approach analytical and without considering any learning-based cues, the results obtained promise to provide a platform for numerous applications driven by augmented reality. Applications involving gaming via a *Kinect* sensor can be enhanced to make the gaming experience a very pleasing one. However, a simple mobile camera can also turn into a useful as well as a fun application using the geometry knowledge and lighting knowledge obtained just from a single scene of an indoor environment. One of the immediate offshoots of this project is in-painting the highlights inside the image and getting rid of the "flash-effect" when images are captured (Park and Lee, 2007). Also, if we do not restrict to a single image, using *Kinect*, amazing applications can be thought of. One such application is to take multi-view images of an indoor environment, build a dense 3D point cloud of the scene, estimate the entire geometry of the scene in 3D and estimate the position of the light sources in the scene (or affecting the scene.) Once all this information is obtained, a scene can be rendered with different light source positions and intensities. One such example is about how a room will appear over the course of the day. Also, inserting a synthetic object in a 3D world and then render a scene or a video, making it appear realistic, is also a possibility. These ideas require a lot of graphics-based methodologies and hence, these were not considered in this thesis. However, with good background in graphics and graphics based programming, these results can be obtained.

In this thesis, all the codes were implemented in form of MATLAB scripts and functions. From a commercial point of view, things can be implemented using OpenCV and C/C++ to get real-time computer vision applications. No standard graphical methods were used. However, the outputs from this thesis can be streamlined into graphics-hardware pipeline and using standard rendering tools, rendered images/videos may be formed as explained above. Although geometry and lighting information may not be an end product of any system, those are few of the most important parameters when it comes to augmented reality and some possible solutions to get those are studied and explained in this thesis.

REFERENCES

1. **Artusi, A.**, **F. Banterle**, and **D. Chetverikov**, A survey of specularity removal methods. *In Computer Graphics Forum*, volume 30. Wiley Online Library, 2011.

2. **Assoc, S.**, **P. Dr**, **I. E. A. Hendriks**, **D. Ir**, **P. A. Redert**, and **P. A. Redert** (2005). Title: Converting 2d to 3d: A survey author: Q. wei reviewers: E. a. hendriks (tu delft).

3. **Burrus, N.** (2011). Kinect calibration.

4. **Canny, J.** (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6), 679–698.

5. **Cao, X.**, **A. C. Bovik**, **Y. Wang**, and **Q. Dai** (2011*a*). Converting 2d video to 3d: An efficient path to a 3d experience. *IEEE MultiMedia*, **18**(4), 12–17.

6. **Cao, X.**, **Z. Li**, and **Q. Dai** (2011*b*). Semi-automatic 2d-to-3d conversion using disparity propagation. *TBC*, **57**(2), 491–499.

7. **Criminisi, A.**, *Accurate visual metrology from single and multiple uncalibrated images*. Springer Verlag, 2001.

8. **Criminisi, A.**, **I. D. Reid**, and **A. Zisserman** (2000). Single view metrology. *International Journal of Computer Vision*, **40**(2), 123–148.

9. **Debevec, P.** (2002). Image-based lighting. *IEEE Computer Graphics and Applications*, **22**(2), 26–34.

10. **Delage, E.**, **H. Lee**, and **A. Y. Ng**, Automatic single-image 3d reconstructions of indoor manhattan world scenes. *In ISRR*. 2005.

11. **Frahm, J.-M.**, **K. Koeser**, **D. Grest**, and **R. Koch**, Markerless augmented reality with light source estimation for direct illumination. *In Conference on Visual Media Production CVMP, London*. 2005.

12. **Funk, N.** (2003). A survey of light source detection methods.

13. **Funt, B.** and **L. Shi**, The rehabilitation of maxrgb. *In IS&T/SIDâĂŹs Color Imaging Conference*. 2010.

14. **Gehler, P. V.**, **C. Rother**, **A. Blake**, **T. Minka**, and **T. Sharp**, Bayesian color constancy revisited. *In IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2008.

15. **Grosse, R.**, **M. K. Johnson**, **E. H. Adelson**, and **W. T. Freeman**, Ground truth dataset and baseline evaluations for intrinsic image algorithms. *In Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009.

16. **Hartley, R. I.** and **A. Zisserman**, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, 2004, second edition.

17. **Hedau, V.**, **D. Hoiem**, and **D. Forsyth** (2010). Thinking inside the box: Using appearance models and context based on room geometry. *Computer Vision–ECCV 2010*, 224–237.

18. **Hedau, V.**, **D. Hoiem**, and **D. A. Forsyth**, Recovering the spatial layout of cluttered rooms. *In ICCV*. IEEE, 2009.

19. **Hoiem, D.**, **A. A. Efros**, and **M. Hebert**, Geometric context from a single image. *In In ICCV*. 2005.

20. **Huang, X.**, **L. Wang**, **J. Huang**, **D. Li**, and **M. Zhang**, A Depth Extraction Method Based on Motion and Geometry for 2D to 3D Conversion. *In 2009 Third International Symposium on Intelligent Information Technology Application*. IEEE, 2009.

21. **Karsch, K.**, **V. Hedau**, **D. Forsyth**, and **D. Hoiem**, Rendering synthetic objects into legacy photographs. *In Proceedings of the 2011 SIGGRAPH Asia Conference*, SA '11. ACM, New York, NY, USA, 2011.

22. **Khoshelham, K.** and **S. O. Elberink** (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, **12**(2), 1437–1454.

23. **Kovesi, P. D.** (2000). Matlab and octave functions for computer vision and image processing. *Online: http://www. csse. uwa. edu. au/˜pk/Research/MatlabFns/# match*.

24. **Lalonde, J.-F.**, **A. A. Efros**, and **S. G. Narasimhan** (2012). Estimating the natural illumination conditions from a single outdoor image. *International journal of computer vision*, 1–23.

25. **Land, E. H.**, **J. J. McCann**, *et al.* (1971). Lightness and retinex theory. *Journal of the Optical society of America*, **61**(1), 1–11.

26. **Lee, D. C.**, **M. Hebert**, and **T. Kanade**, Geometric reasoning for single image structure recovery. *In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009.

27. **Limare, N.**, **A. B. Petro**, **C. Sbert**, and **J.-M. Morel** (2011). Retinex poisson equation: a model for color perception. *Image Processing on Line*.

28. **Lv, Y.**, **X. Shen**, and **H. Chen**, Identifying image authenticity by detecting inconsistency in light source direction. *In Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*. IEEE, 2009.

29. **Madsen, C. B.** and **M. Nielsen** (2008). Towards probe-less augmented reality: a position paper.

30. **Microsoft** (November, 2010). Microsoft kinect. *http://www.xbox.com/en-US/kinect*.

31. **Nedovic, V.**, **A. W. M. Smeulders**, **A. Redert**, and **J.-M. Geusebroek**, Depth information by stage classification. *In ICCV*. IEEE, 2007.

32. **Neverova, N.** (2012). 3d scene reconstruction and augmenting using depth sensors.

33. **Neverova, N.**, **D. Muselet**, and **A. Trémeau**, Lighting estimation in indoor environments from low-quality images. *In ECCV Workshops (2)*. 2012.

34. **Park, J. W.** and **K. H. Lee** (2007). Inpainting highlights using color line projection. *IEICE TRANSACTIONS on Information and Systems*, **90**(1), 250–257.

35. **Phong, B. T.** (1975). Illumination for computer generated pictures. *Communications of the ACM*, **18**(6), 311–317.

36. **Rother, C.** (2002). A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, **20**(9), 647–655.

37. **S., U.** (1975). On visual detection of light sources.

38. **Sammons, E.**, *The World of 3-D Movies*. A Delphi Publication, 1992.

39. **Saxena, A.**, **S. H. Chung**, and **A. Y. Ng**, Learning depth from single monocular images. *In In NIPS 18*. MIT Press, 2005.

40. **Saxena, A.**, **M. Sun**, and **A. Y. Ng** (2009). Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **31**(5), 824–840.

41. **Shafer, S. A.** (1985). Using color to separate reflection components. *Color Research & Application*, **10**(4), 210–218.

42. **Shen, H.-L.** and **Q.-Y. Cai** (2009). Simple and efficient method for specularity removal in an image. *Applied optics*, **48**(14), 2711–2719.

43. **Silberman, N.** and **R. Fergus**, Indoor scene segmentation using a structured light sensor. *In Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011.

44. **Sun, X.-b.**, **J. Yin**, **D.-h. Li**, and **B.-l. Xiao** (2008). Point in polygon testing based on normal direction. *Optics and Precision Engineering*, **6**, 029.

45. **Tan, R.** and **K. Ikeuchi**, Intrinsic properties of an image with highlights. *In Meeting on image recognition and understanding MIRU*. 2004.

46. **Tan, R. T.** and **K. Ikeuchi**, Separating reflection components of textured surfaces using a single image. *In Digitally Archiving Cultural Objects*. Springer, 2008, 353–384.

47. **Torralba, A.** and **A. Oliva** (2002). Depth estimation from image structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**, 2002.

48. **Wu, T.-P.**, **J. Sun**, **C.-K. Tang**, and **H.-Y. Shum** (2008). Interactive normal reconstruction from a single image. *ACM Transactions on Graphics (TOG)*, **27**(5), 119.