Single-Shot Accurate 3D Reconstruction Using Structured Light Systems Based on Local Optimization

by

Neda Aslsabbaghpourhokmabadi

A thesis submitted in partial fulfilment of requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Creating 3D models of objects has been studied extensively for several decades. These models have different applications in different fields. As a result, creating 3D models is an interesting area in computer vision. There are many proposed methods for extracting 3D information of 2D images. One of the most common methods for 3D reconstruction is structured light methods. Although structured light methods can get valuable results of 3D reconstruction, they have limitations. For example, the structured light methods can get dense results on static scenes or get sparse results on dynamic scenes. In static scenes, the structured light method projects several patterns, and it results in dense models. However, in dynamic scenes, the structured light method projects just one pattern since the object is moving, and it results in sparse models.

The limitation of the structured light methods in dynamic scenes is the most important motivation for this thesis. In this thesis, a single-shot structured light method is developed to overcome the sparse results in dynamic scenes. In particular, the proposed method can obtain more accurate reconstruction with just one image of dynamic scenes than that of existing methods. The new method applies global and local optimizations to establish dense correspondences. The result of simulated experiments comparison with the ground truth demonstrates that the proposed method in this thesis achieves more accurate results than that of previous methods. Lastly, the technique developed in this thesis is applied to real data in order to obtain high quality 3D reconstruction results. The results of the new method are more accurate compared to previous methods.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Yang for the continuous support of my MSc study and related research, for his patience, motivation, invaluable support and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my MSc study.

I also thank my fellow labmates in for the stimulating discussions, helpful comments and their help. My labmates are Xida Chen, Yi Gu, Yiming Qian, Saeed Hojjati, Mohsen Taghaddosi, Jia Lin, Xiaoqiang Zhang, and G.M. Mashrur E Elahi.

Last but not the least, I would like to thank my family, my dear uncle, and my beloved husband for supporting me spiritually throughout writing this thesis and my life in general.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Creating a three dimensional (3D) model of an object from two or more images is called 3D reconstruction in computer vision. 3D reconstruction has been an active research area in computer vision for decades. It provides a valuable ability for visual-based machines to understand the environment. Additionally, 3D reconstruction has many practical applications in different fields. For example, sensFly, a drone company, [29] has designed flying robots with minimal impact on the environment to operate in remote areas that are hard to reach for humans, e.g. mountains or disaster areas. These robots capture images with attached cameras of these scenes. Then, computer vision algorithms create 3D models of the scene for scientists to study without having to go there. 3D reconstruction can also be used to preserve the heritage [35] by creating 3D models of historical objects and monuments. Since the models are digital, they will not suffer from erosion or weathering effects, and by using these models, historical artifacts can be preserved and passed from generation to generation. 3D reconstruction is used in biometric as well. For example, in head-scan [41, 26], the 3D shape and volume of a person is used to identify or authenticate a specific person. Surgery is also exploiting 3D reconstruction to recreate fractured or damaged organs [49]. Another recent application of 3D reconstruction is to let a person try on clothes virtually [25, 24]. Users can virtually see how the clothes would fit and look on their virtual 3D bodies.

Different approaches have been proposed for extracting 3D information of an object. There is similarity between these methods in extracting depth and human's perception of depth. Vision enables humans to perceive depth of objects in a scene. Perceiving depth in humans relies on the binocular stereopsis system in the eyes. Each eye, individually, provides a different view of the same object. When the two views are superimposed, the same object appears in different positions. This difference is called disparity. To imitate the human's binocular stereopsis system, computer vision algorithms use two or more cameras with different viewpoints or one moving camera to capture images of a scene from different viewpoints. This is the basis of most stereo methods.

In a typical stereo method, the algorithm finds corresponding points in different images of the same scene. Then, the matching points are used to get the 3D information using a process called triangulation. These stereo methods are divided into passive and active methods.

❶ Passive methods use ambient light [27]. Many of these methods are listed on the Middlebury Multi-View Stereo site [30], which evaluates methods for their performance using a dataset. However, the major problem of these methods is with non-textured objects, because corresponding points are difficult to obtain in areas with no textures. Hence, passive stereo methods normally cannot reconstruct a dense 3D model.

❷ Active methods use projected patterns to overcome the aforementioned problems of the passive methods. The most popular group in this category is the structured light methods which project one or more patterns on the scene to create texture on the object. As a result of the projected texture, the number of correspondences between images increases. Projecting a single pattern is used for reconstructing 3D shapes of dynamic objects, and multi pattern projection is used for static objects. Because of their higher accuracy, structured light methods are often used to build the ground truth of depth maps for evaluating passive stereo methods [30]. Although active methods are more accurate, developing a practical and robust structured light method is still an open area of study in computer vision.

This thesis is focused on developing a single-shot structured light method to reconstruct an accurate and dense model of the objects. The developed idea has two approaches. One approach incorporates two methods in order to find the accurate and dense corresponding points between an image pair. The first method is to apply global optimization to get an initial set of dense corresponding points. The second method is to apply the winner-takes-all idea to refine the set of corresponding points in their locations. The second method results in more accurate matched points between an image pair. The other approach incorporates one global optimization to get an accurate and dense corresponding points. These approaches are described in details in chapter 3 and results of each one are presented in chapter 4.

1.2 Background

Existing structured light methods in literature [4, 5, 11, 15, 16] can be categorized into two groups based on how the pixel location is encoded: temporal and spatial methods.

Temporal methods project several patterns on the object. A scene point is encoded using a temporal sequence of patterns. Correspondence of two images is then obtained by finding pixels with the same temporal code. These methods obtain dense and accurate results, but they are not applicable to dynamic scenes because the scene will change frame to frame.

Spatial methods project only one pattern onto the object and use the neighboring patterns to encode scene points. Correspondence of two images is then obtained by finding pixels with the same spatial codes. The results of these methods are not as dense because the code uses more physical space to do encoding. However, they can be used in dynamic scenes because they project only one pattern. The main purpose of this research is to develop an accurate single-shot structured light method which is both applicable to dynamic scenes and is sufficiently accurate compared to results of existing temporal structured light methods.

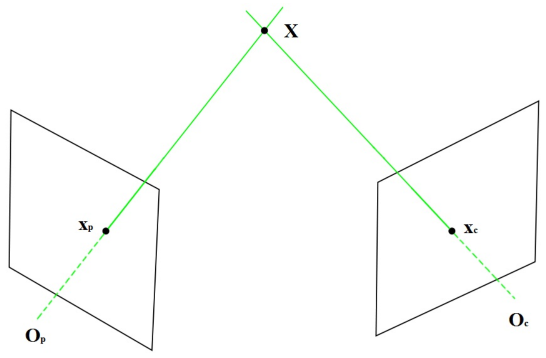


Figure 1.1: Triangulation with an established correspondence [3].

The extracted information of the scene from spatial and temporal methods is used for triangulation which results in 3D points. To have an accurate triangulation, camera-camera or camera-projector systems must be calibrated accurately. Camera calibration computes the internal information of the camera, e.g. focal length. However, because of using two or more cameras in structured light methods, the relative position of the cameras is also needed and must be determined.

Figure 1.1 shows the principle of triangulating and obtaining the 3D point when the correspondence between the camera-projector system is determined. It represents a camera-projector system where x_c and x_p are a pair of corresponding points between the camera and the projector, respectively; O_c is the center of the camera; O_p is the center of the projection; and X is the triangulated 3D point.

Figure 1.2 shows the setup of a typical structured light system. A projector projects a stripe pattern on the object, and the camera captures the scene. The projected stripes are captured in deformed shape, which is used to get the 3D information.

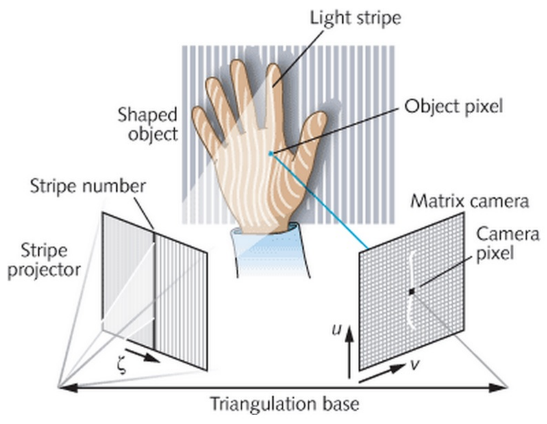


Figure 1.2: Configuration of a typical structured light system [31]

1.3 Thesis Organization

Chapter 2 provides an overview of structured light systems. It gives an introduction and categorization to existing structured light methods as well as their limitations.

Chapter 3 presents a novel structured light method that recovers a dense depth map using a single color pattern. Compared to existing methods, the advantage of this method is its ability of establishing dense and accurate corresponding points. Hence, it can be applied to dynamic scenes because it requires to project only one pattern to the scene.

Chapter 4 presents results of proposed methods, and comparison with existing methods for both simulated data and real world data.

Finally, chapter 5 highlights the contribution and provides the conclusions of the thesis and introduces future works.

Chapter 2

Related Work

Reconstructing 3D models using passive stereo methods has many challenges. One of the major challenges is the small number of corresponding points in texture-less regions of a scene which results in a sparse 3D model [34, 6].

Structured light methods can address and solve the aforementioned problem [46]. These methods are also referred to as the active stereo methods. They project light pattern(s) onto the scene. The pattern is projected by an active device such as a projector or a laser scanner. The projected pattern creates texture on the scene and increases the number of detected interest points and consequently corresponding points [45].

Structured light methods are categorized based on different aspects, for example scene applicability or coding strategy. Scene applicability represents whether or not a given pattern is suitable for measuring moving objects. Coding strategy specifies the type of codewords, periodic codification or non-periodic codification. Figure 2.1

shows the classification structured light methods [46].

The common steps among all the structured light methods are the following:

❶ Calibrating the camera-camera or camera-projector systems;

❷ Rectifying the captured images;

❸ Extracting distinctive features from rectified images;

❹ Obtaining corresponding points (❸+❹ is called decoding); and,

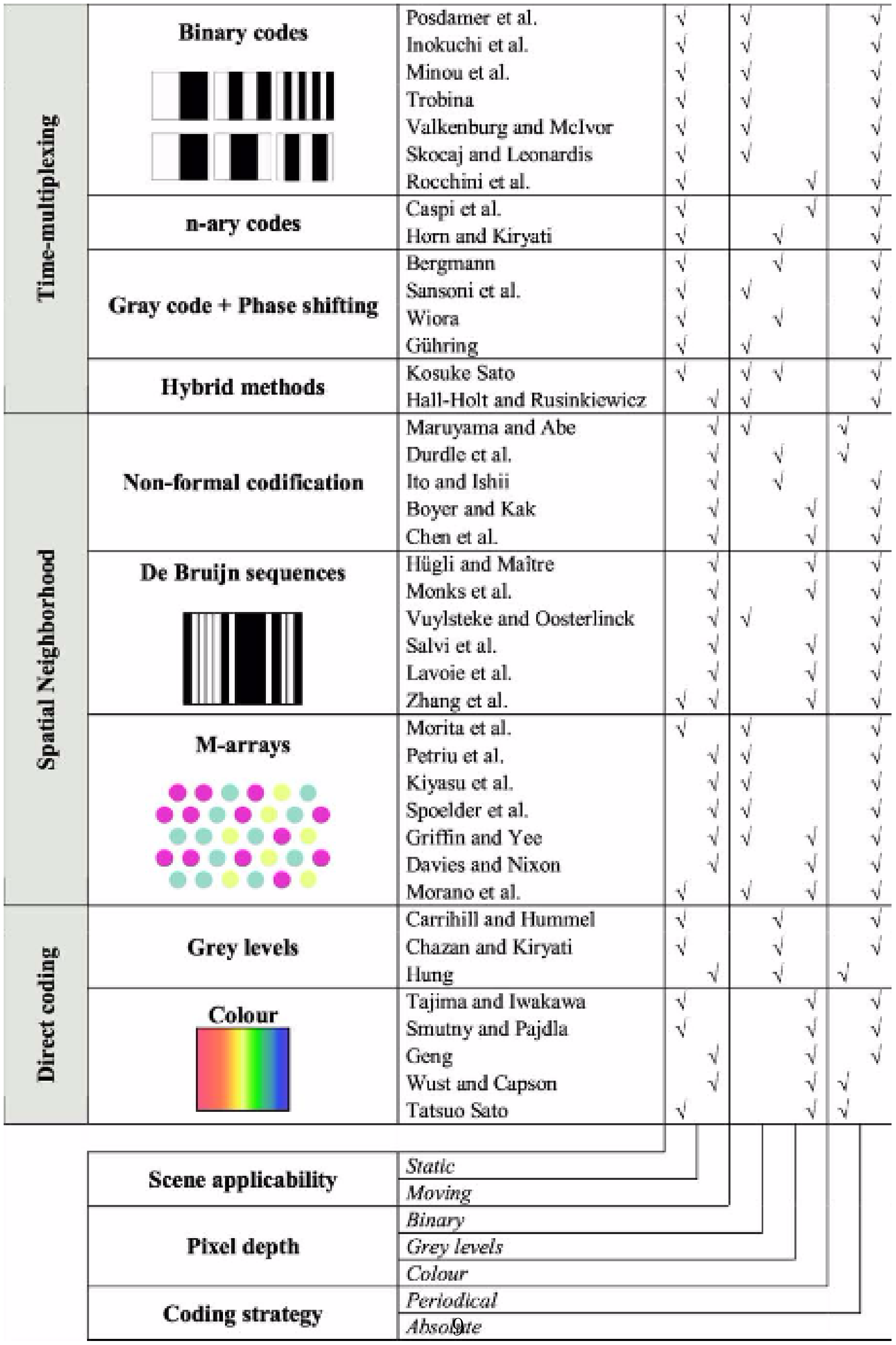
❺ Triangulating the corresponding points.

In the following section, calibration is discussed first. Then, some of the decoding methods are reviewed. These methods are used to extract features and distinguish correspondences between the images.

2.1 Calibration

Camera Calibration has been extensively studied in the past decades [8, 44, 55, 13]. It is needed in structured light methods because these methods assume the input images are rectified. The advantage of using rectified images is reducing the search space for establishing the correspondence points in the following steps from $O(mn)$ to $O(m)$, where the image size is $m \times n$.

To explain more, if we consider a pixel p_i in a captured image, the whole projected pattern is searched for finding the correspondence of p_i. Looking for a pixel in an image of size $n \times m$ takes $O(nm)$. However, if the images are rectified, the search size is reduced to $O(m)$, because corresponding points are on corresponding scanlines as it is shown in Figure 2.2. In other words, if a point p_i is located on the j^{th} row in



			Scene applicability		Pixel depth			Coding strategy	
			Static	*Moving*	*Binary*	*Grey levels*	*Colour*	*Periodical*	*Absolute*
Time-multiplexing	Binary codes	Posdamer et al.	√		√				√
		Inokuchi et al.	√		√				√
		Minou et al.	√		√				√
		Trobina	√		√				√
		Valkenburg and McIvor	√		√				√
		Skocaj and Leonardis	√		√				√
		Rocchini et al.	√				√		√
	n-ary codes	Caspi et al.	√				√		√
		Horn and Kiryati	√			√			√
	Gray code + Phase shifting	Bergmann	√			√			√
		Sansoni et al.	√		√				√
		Wiora	√			√			√
		Gühring	√		√				√
	Hybrid methods	Kosuke Sato	√		√	√			√
		Hall-Holt and Rusinkiewicz		√	√				√
Spatial Neighborhood	Non-formal codification	Maruyama and Abe		√	√			√	
		Durdle et al.		√		√		√	
		Ito and Ishii		√		√			√
		Boyer and Kak		√			√		√
		Chen et al.		√			√		√
	De Bruijn sequences	Hügli and Maître		√			√		√
		Monks et al.		√			√		√
		Vuylsteke and Oosterlinck		√	√				√
		Salvi et al.		√			√		√
		Lavoie et al.		√			√		√
		Zhang et al.	√	√			√		√
	M-arrays	Morita et al.	√		√				√
		Petriu et al.		√	√				√
		Kiyasu et al.		√	√				√
		Spoelder et al.		√	√				√
		Griffin and Yee		√	√		√		√
		Davies and Nixon		√			√		√
		Morano et al.	√		√		√		√
Direct coding	Grey levels	Carrihill and Hummel	√			√			√
		Chazan and Kiryati	√			√			√
		Hung		√		√		√	
	Colour	Tajima and Iwakawa	√				√		√
		Smutny and Pajdla	√				√		√
		Geng		√			√		√
		Wust and Capson		√			√	√	
		Tatsuo Sato	√				√	√	

Figure 2.1: Classification on the structured light methods [46].

the rectified captured image, its correspondence will be on the j^{th} row in the rectified projected pattern.

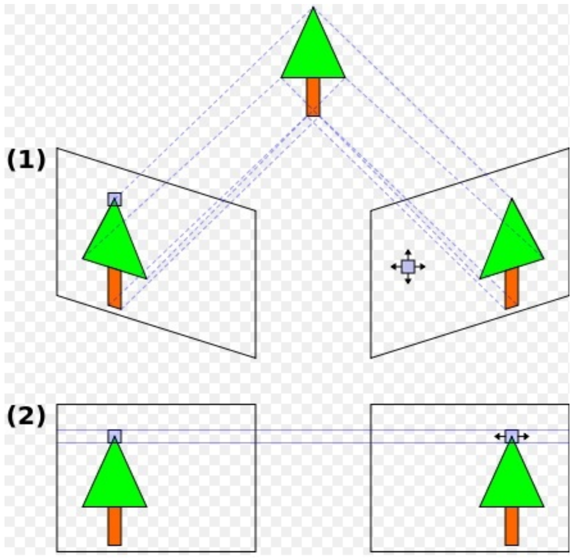


Figure 2.2: The search space between (1)original images and (2) rectified ones. [2]

Going back to the discussion of camera calibration, it estimates the intrinsic and extrinsic parameters of the pinhole cameras. The intrinsic parameters are the focal length, the principal point (i.e. image center), and the lens distortion parameters of the camera. The extrinsic parameters include the rotation and translation from the world coordinate system to the camera coordinate system. There is a publicly available camera calibration toolbox [8] which is implemented based on the method proposed by Zhang [55]. This toolbox is used in this thesis for calibrating the cameras. The algorithm of calibration can be described briefly as follows:

❶ Moving a planar checkerboard pattern and capturing images at different locations and orientations;

❷ Detecting corners of the captured images;

❸ Finding the corresponding points in different images;

❹ Using the closed form solution to get the intrinsic and extrinsic parameters; and,

❺ Refining the obtained parameters through maximum likelihood inferences.

In addition, there are some methods which use camera-projector systems. The camera-projector calibration method is an extension of camera calibration because the projector is counted as an inverse camera by projecting the images instead of capturing them. The method proposed in [28] utilizes a thoroughly flat surface placed at different locations with different orientations in front of the camera and the projector. The flat surface moves while it is in the focus range of the camera and the projector. A printed rectangular checkerboard is attached to the flat surface. Moreover, a pattern is projected on this surface. There should not be any overlaps between the projected pattern and the printed pattern, while both patterns are visible to the camera.

After capturing more than two images of the flat surface in different locations with different orientations, corners of the printed pattern are detected automatically using the camera calibration toolbox [8]. Then, the intrinsic and extrinsic parameters of the camera are estimated using a closed-form solution.

Once the camera's parameters are established, the calibration plane equation in the camera coordinate system is estimated. The plane is represented by specifying a non-zero normal vector to the plane, n, and a point on the plane, p. The first three elements of the third column of the rotation matrix represent n, and the three elements of the translation vector represent the coordinate of p (equation 2.1).

$$K_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

Based on equation 2.1, $n = [r_{13}, r_{23}, r_{33}]^t$ and $p = [t_x, t_y, t_z]^t$. Having the normal vector n and a point p, a plane is defined as shown in equation 2.2, where r is any point on the plane and "$\cdot$" represents the dot product,

$$n \cdot (r - p) = 0. \tag{2.2}$$

After defining the plane of the printed pattern, the corners of the projected pattern are detected using the camera calibration toolbox [8]. Then, a ray-plane intersection is required in order to obtain the 3D position of the corners of the projected pattern. Applying ray-plane intersection to different images produces a big number of non-coplanar 3D points for calibrating the projector. At the end, by applying Zhang's method [55] to the 2D to 3D corresponding points, the projector can then be calibrated. Then, the intrinsic and extrinsic parameters of the projector are estimated. As a result of calibrating the projector, the relative position between the camera and the projector are established.

2.2 Decoding

Structured light methods utilize different decoding methods. Some of the structured light methods use black and white patterns and some color patterns. Moreover, among existing methods, some project different patterns to deal with static objects, while others project just one pattern to deal with moving objects. The former group projects several patterns onto the objects (multi-shot), but the latter group projects just one pattern in each frame (single-shot). In this section, decoding methods are categorized based on the number of projected patterns and scene applicability. Those methods using multi-shots are applicable to static scenes, while those using single-shot are applicable to dynamic ones. Table 2.2 [46] shows these two categories and the most popular methods in each category. These methods are described in more detail in this chapter.

Methods applicable to static scenes	Binary Code [36, 48, 39] Gray Code [22, 7, 42, 20] N-ary Code [11] Phase Shifting [52, 23, 33, 32, 51, 53, 54]
Methods applicable to dynamic scenes	De Bruijn stripe pattern [18, 43, 9, 38, 50] M-array Patterns [17, 46, 21, 37, 16]

Table 2.1: Categorization of structured light methods [46].

2.2.1 Methods Applicable To Static Scenes

Methods applicable to static scenes usually project several varying patterns onto the objects. These methods are also called multi-shot methods or temporal methods. A scene point is encoded using a temporal sequence of patterns. By projecting enough number of patterns onto the objects, every pixel, which corresponds to a physical scene point, is encoded by a codeword, which is formed when all the patterns are projected. Hence, dense reconstruction results. There are several methods in this category which are explained below.

2.2.1.1 Binary Codes

The patterns in the binary code methods consist of black and white stripes. By projecting a sequence of black and white stripe patterns onto the object, each pixel is encoded in a sequence of black or white (0 or 1). The template of the pattern stripes follows a coarse-to-fine idea. In particular, if m is the number of binary patterns, at most 2^m stripes can be uniquely coded using the binary pattern methods. Some widely used binary code methods are proposed by Posdamer et al. [39], Minou et al. [36], and Valkenburg et al. [48]. These methods project black and white patterns and encode each point regardless of the object's color. Therefore, these methods are not dependent on object's color, and for this reason, they can be applied to most of the static scenes, including texture-less ones. After projecting a pattern, an edge detection method is applied to determine the transition between two adjacent stripes. Figure 2.3 shows binary patterns which contain 1024 different binary codes. Figure 2.3 has 1024 column and each column represents a specific binary code.

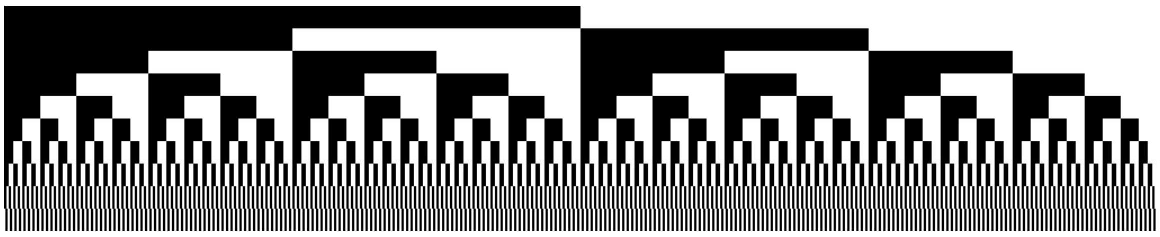


Figure 2.3: Binary patterns.

2.2.1.2 Gray Code

Gray code patterns (Figure 2.6) are very similar to binary code patterns where every pixel is uniquely encoded by a sequence of black and white. Since the number of stripes in the patterns increases by two, the length of the codeword is given by 2^m similar to binary codes, where m is the number of projected patterns. If m is 10, the width of image should be 1024 (2^{10}) to encode each pixel uniquely. Gray code methods are introduced by Guhring [22], and Bergmann [7]. These methods require a great number of projected patterns. To encode each pixel uniquely, some horizontal and some vertical patterns which follow the coarse-to-fine idea should be projected. This can be seen in Figure 2.4. These methods need enough time to project all the patterns. Hence, these methods are not efficient. Later on, various versions of gray code patterns have been developed where the number of projected patterns is reduced [42, 20].

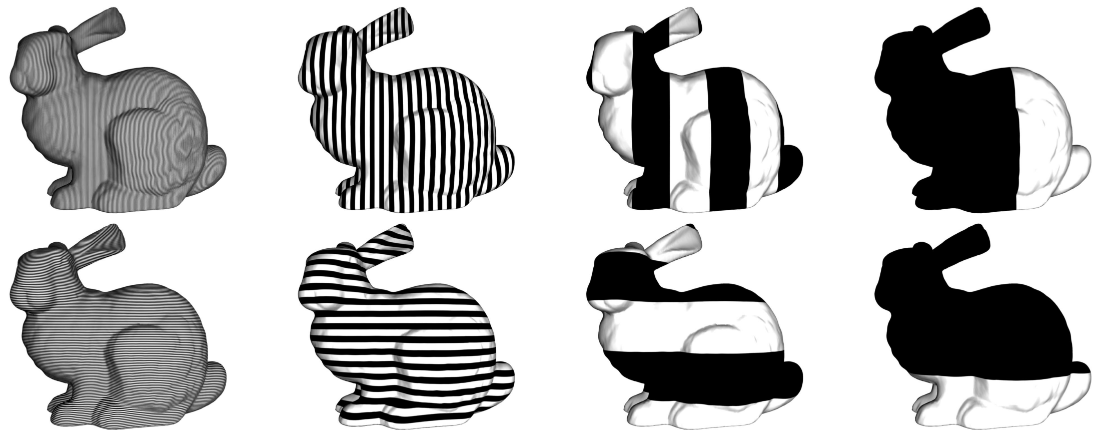


Figure 2.4: Some sample of gray code patterns [46], coarse-to-fine from right to left.

After projecting all the patterns, the values of pixel p_i in all the captured images are compared to a threshold successively. If the pixel value is higher than the threshold, it is coded to 1; otherwise, it is coded to 0. If there are m projected patterns, the length of codeword is m. Coding a pixel using the usual gray code pattern and a threshold is not easy because of the difference of surface albedo of a scene. Because of the difference of surface albedo of a scene, a scene point illuminated by a white stripe in a pattern may appear darker than another scene point illuminated by a black stripe. In this case, even using a threshold cannot help and may cause inaccurate and wrong coding. To overcome this issue and get more accurate results, inverse patterns are also projected. This doubles the number of required patterns, Figure 2.5. When the inverse patterns are projected, the brightness of each pixel in the original patterns and that in the inverse patterns are compared. Therefore, pixels are encoded based on brightness comparison between the original projected pattern and the inverse pattern. If a pixel value in the original pattern is brighter than the pixel value in the inverse pattern, the pixel is coded as 1; otherwise it is coded as 0.

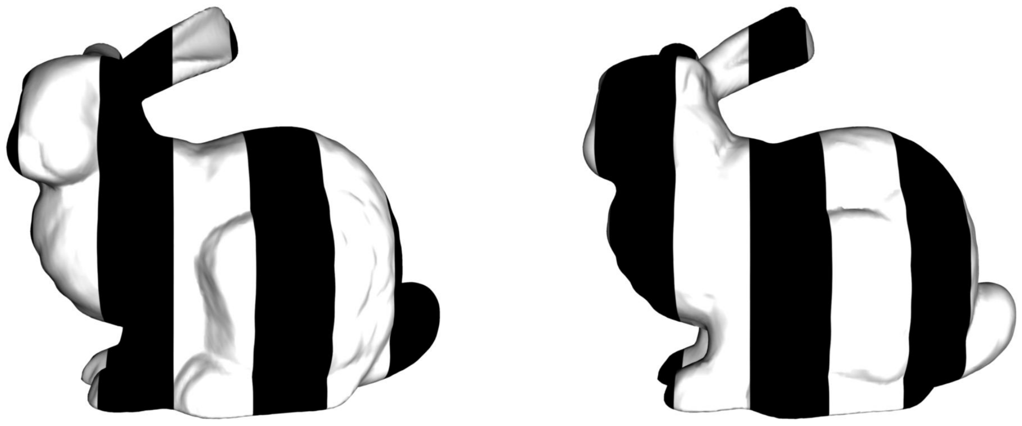


Figure 2.5: left: normal pattern, right: inverse pattern [46].

The difference between the binary code pattern and the gray code pattern is the order of the projected patterns. Figures 2.3 and 2.6 are visualizing the binary code and the gray code patterns. The difference in projection results an advantage for the gray code methods. The advantage of gray code is its applicability to objects with steep and bumpy surface. In gray code, if a pixel has only one bit of encoding error, then the code difference between this pixel and its two adjacent pixel is one bit. However, if a pixel has a 1-bit error using binary code, the code error between this pixel and its adjacent neighbors is more than one bit. For example, the code of the 512^{th} pixel in gray code is $(0,0,1,1,1,1,1,1,1,1)$ and the code of the 513^{th} pixel is $(1,0,1,1,1,1,1,1,1,1)$, and the difference between these two adjacent pixel is one bit. However, the code of the 512^{th} pixel in binary code is $(0,1,1,1,1,1,1,1,1,1)$ and the code of the 513^{th} pixel is $(1,0,0,0,0,0,0,0,0,1)$, and the difference between these two pixel is 9 bits.

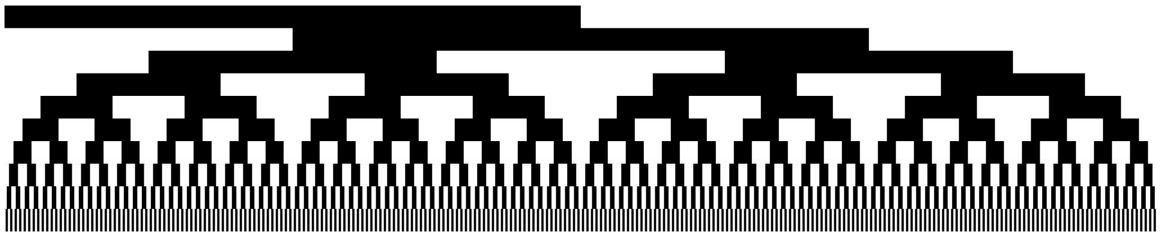


Figure 2.6: Gray code patterns.

2.2.1.3 N-ary codes

The methods in section 2.2.1.1 and 2.2.1.2 encode each pixel into two codes, in particular, 0 and 1. Coding into two codes causes these methods to be independent from the object's color. For this reason, these methods cannot preserve color information. N-ary methods, proposed by Caspi et al. [11], are color based patterns, where n^m

stripes are uniquely encoded in the RGB color space instead of black and white patterns, where n is the number of colors , and m the number of projected patterns. The parameters to set for designing N-ary patterns are: the number of colors (n), the number of projected patterns (m), and noise immunity factor (α). N-ary codes is an extension of gray codes. In the gray codes, there are just two colors, but in the N-ary codes there are n colors. For example, the N-ary codes with $n = 2$ and $m = 10$ is similar to the gray codes. The N-ary code methods project m patterns [11], and then encode each pixel using the Equation 2.3. Each color in projected pattern has a symbol. By solving Equation 2.3 for each pixel, corresponding symbol in the projected pattern is obtained, where $\overrightarrow{C}$ is the recovered projected color from the captured image; A is the projector-camera coupling matrix; K is the reflectance matrix; $\overrightarrow{P}$ is the non-linear and monotone transformation; and $\overrightarrow{C_0}$ is the reading of the camera under the ambient light. These parameters are known. And $\overrightarrow{c}$ is the correspondence of $\overrightarrow{C}$ for each pixel which is unknown and will be extracted. Each pixel is encoded by accumulating these symbols respectively.

$$\underbrace{\begin{bmatrix} R \\ G \\ B \end{bmatrix}}_{\overrightarrow{C}} = \underbrace{\begin{bmatrix} a_{rr} & a_{rg} & a_{rb} \\ a_{gr} & a_{gg} & a_{gb} \\ a_{br} & a_{bg} & a_{bb} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} k_r & 0 & 0 \\ 0 & k_g & 0 \\ 0 & 0 & k_b \end{bmatrix}}_{K} \overrightarrow{P} \underbrace{\begin{Bmatrix} r \\ g \\ b \end{Bmatrix}}_{\overrightarrow{c}} + \underbrace{\begin{bmatrix} R_0 \\ G_0 \\ B_0 \end{bmatrix}}_{\overrightarrow{C_0}} \tag{2.3}$$

N-ary code can be as accurate as the gray code methods while using fewer projected patterns. In other words, N-ary code methods reduce the number of required patterns significantly. Although these methods require fewer projected patterns compared to binary and gray codes, they do not reach the robustness of binary and gray codes because of their sensitivity to object's color since their patterns are designed in the RGB space.

2.2.1.4 Phase Shifting

Phase shifting methods are another kind of temporal methods for getting the 3D information of an object. In phase shifting, a sinusoidal shifting pattern is projected on a surface. The pattern is shifted and projected on the scene several times, and each projected pattern is captured. The surface of the object causes deformation in the captured patterns which is called phase deviation. The patterns used in these methods are reduced to 3 patterns and usually designed by equation 2.4, as proposed by Zhang et al. [52, 51, 53, 54], Han [23], and Haung et al. [33, 32] :

$$I(x,y) = I'(x,y) + I''(x,y)\cos\big(\phi(x,y) + \delta(t)\big), \tag{2.4}$$

where $I(x,y)$ is the intensity in the designed pattern; $I'(x,y)$ is the average intensity; $I''(x,y)$ is the intensity modulation; $\delta(t)$ is time-varying phase shift angle; and $\phi(x,y)$ is the phase to be solved for.

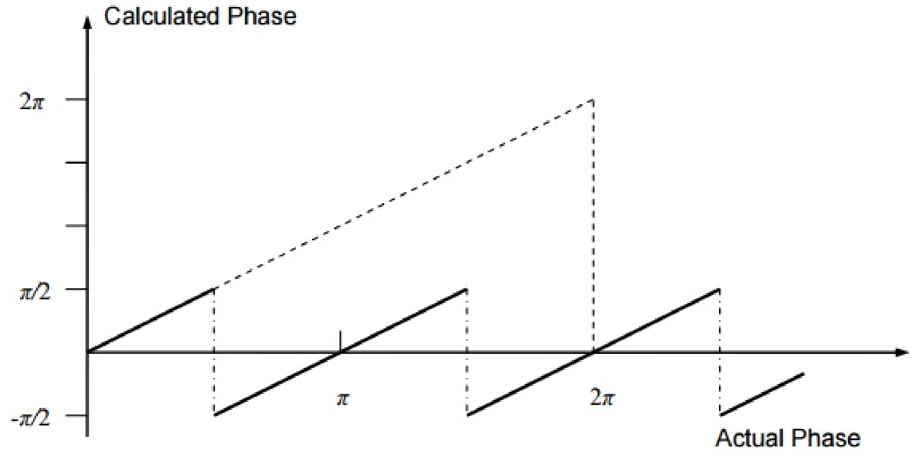
The common values for δ are 90°and 120°. When $\delta = 120$, the resulting symmetry is useful. Using equation 2.4 three patterns are generated as follows:

$$
\begin{aligned}
I_0(x,y) &= I'(x,y) + I''(x,y)\cos\big(\phi(x,y) - 2\pi/3\big) \\
I_1(x,y) &= I'(x,y) + I''(x,y)\cos\big(\phi(x,y)\big) \\
I_2(x,y) &= I'(x,y) + I''(x,y)\cos\big(\phi(x,y) + 2\pi/3\big).
\end{aligned}
\tag{2.5}
$$

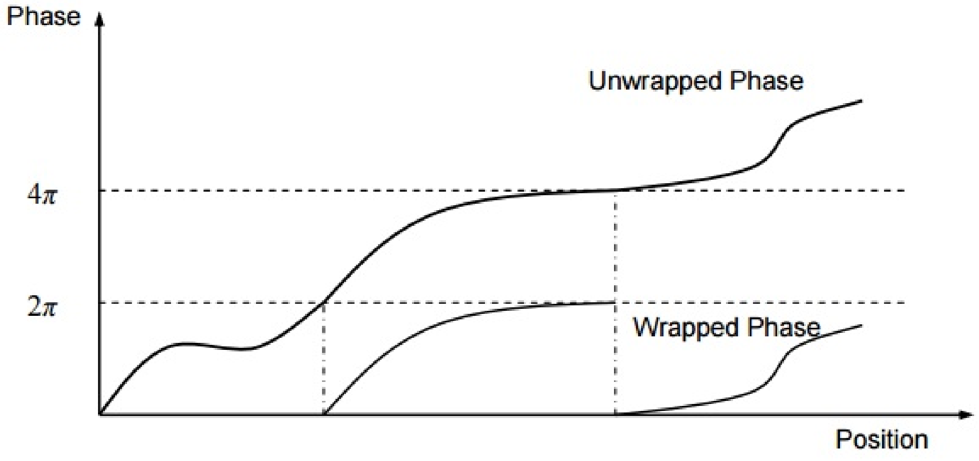
Solving the equations in equation 2.5 simultaneously, the phase value ($\phi(x,y)$) can be obtained as equation 2.6:

$$
\phi(x,y) = \tan^{-1}\left(\sqrt{3}\frac{I_0(x,y) - I_2(x,y)}{2I_1(x,y) - I_0(x,y) - I_2(x,y)}\right). \tag{2.6}
$$

The range of the values for tan^{-1} is defined from $\frac{-\pi}{2}$ to $\frac{\pi}{2}$. Hence, before using $\phi(x,y)$, the discontinuities that occur in the phase calculation must be corrected. A spatial phase unwrapping algorithm [52] can be applied to obtain continuous and absolute phase from the wrapped phase. The phase unwrapping is essential to detecting the 2π discontinuities and ambiguities, and to removing them by adding or subtracting multiples of 2π point by point . Figure 2.7(a) shows a wrapped phase, and Figure 2.7(b) shows an unwrapped phase.



(a)



(b)

Figure 2.7: (a) The recovered phase with ambiguity, and (b) The unwrapped phase without ambiguity. [23]

Phase shifting methods use fewer number of projected patterns; therefore, these methods are more efficient compared to binary or gray codes. Because of the few number of projected patterns, these methods can be applied to dynamic scenes for scenes with small motion. However, it is not recommended to use phase shifting for fast dynamic scenes. Also, the results of phase shifting methods are more accurate [33, 54]. Moreover, based on gray scale nature of the patterns, these methods are robust to object's color and less sensitive to ambient light and object's surface reflectivity variation [52]. The disadvantage of the phase shifting methods is their sensitivity to the projector defocus and noise [19].

To sum up, temporal coding strategies project multiple patterns and establish the corresponding points between a pair of images. The strategies of projecting multiple patterns are used for static scenes, and can achieve high accuracy in 3D reconstruction.

2.2.2 Methods Applicable To Dynamic Scenes

Unlike temporal coding, spatial coding projects just one pattern and creates identifiable features using spatial information. Spatial coding techniques are applicable to dynamic scenes, but they are not as accurate as temporal ones. There are different methods in this category which are explained below.

2.2.2.1 De Bruijn Stripe Pattern

One of the most popular patterns in spatial coding is the De Bruijn pattern which is discussed in this section, because it is used in the proposed method in this thesis. A De Bruijn pattern is designed based on the De Bruijn circular sequence [18]. A De

Bruijn pattern generation code is available on *"The combinatorial object server"* [43]. The first proposed De Bruijn patterns are found in *"Color-encoded structured light for rapid active ranging"* by Boyer et al. [9]. Later, various methods based on the De Bruijn patterns [38, 50, 12] are proposed. A De Bruijn pattern consists of 125 stripes in the RGB color space and has a window uniqueness property of 3. The window uniqueness property of n means that each sub-sequence of n consecutive stripes is unique within the entire sequence. An $n-ary$ De Bruijn sequence of order n is a circular sequence $d_0, d_1, ..., d_{k^n-1}$ containing each sub-sequence of length n, just once. In practice, only 125 stripes are needed and thus, the method worked with $k = 5$, and $n = 3$. Figure 2.8 shows a sample of the pattern which consists of 125 stripes.

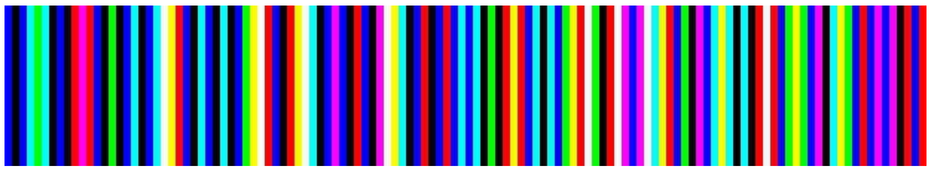


Figure 2.8: A sample of De Bruijn pattern.

There are different methods that use the De Bruijn patterns to get the 3D information of an object. Despite some differences, these methods follow a similar methodology as follows:

➊ Use the mentioned pattern in Figure 2.8, and project it on the scene;

➋ Detect the edges between the stripes in the captured images;

➌ Apply global optimization for matching corresponding points between a pair of images; and, ❹ Apply triangulation to get the 3D information of the object.

Differences between De Bruijn based methods usually happen in the first three steps by projecting different versions of De Bruijn patterns, using different image processing methods, and using different global optimization techniques. For example, Pages et al. [38] design a different pattern which combines multi-slit patterns and color stripe patterns properties. In multi-slit patterns, there are black gaps between the colored pixels. Therefore, it is possible to have similar colors in two consecutive slits. Having black gaps between color bands allows intensity peaks to be detected in the images. However, in color stripe patterns, there is no black gap between stripes. Therefore, two consecutive stripes cannot have the same color. In such patterns, edges between the stripes are detected in the image. Also, Chen et al. [12] design a Gaussian blurred De Bruijn pattern which has a window uniqueness property. Moreover, they use a pixel based matching method instead of edge based ones in the image processing step.

After projecting the De Bruijn stripe pattern and capturing the image, the image processing step is applied. This step requires detecting the edges of stripes (transition between two different adjacent color). The maxima and minima of the second order derivative of the captured images indicate edges [38]. After detecting the edges between stripes, global optimization using dynamic programming is applied to find the correspondences between images. Dynamic programming can work under the monotonicity assumption. Monotonicity assumption states that if a pixel, p_i, in a scan line have been seen before pixel p_{i+1} in the projected pattern, then the corresponding pixel c_i should be seen before c_{i+1} in the captured image. The dynamic programming part may have errors if the monotonicity assumption is violated. The monotonicity assumption can be violated in some cases, such as having occlusion as shown in Figure 2.9. The figure shows that pixel p_i appears before pixel p_{i+1} in the projector when we scan from left to right. But, in the camera c_i appears after c_{i+1}. Hence, the monotonicity assumption is violated. To overcome this problem, multi pass dynamic programming is developed by Zhang [50]. The dynamic programming fails to find the optimal path when there is occlusion, but in practice it identifies the monotonic solution. Although this solution is not the optimal path, it corresponds to a monotonic component of the optimal path. In other words, it identifies a sub path of the optimal path. For example, Figure 2.10 shows that the (F, G) surface is occluding the (A, B, C, D, E) surface, and there are 9 projected rays while 8 of them are captured. Figure 2.11 shows that dynamic programming can recover the (A, B, C, D, E) path in the first pass. The rest of optimal solutions such as sub path (F, G) also is a monotonic and can be identified by removing previously identified path from the grid. The previously identified pass (A, B, C, D, E) consists of columns $(1, 2, 4, 5, 6, 9)$ and rows $(1, 2, 5, 6, 7, 8)$ in the original grid. The procedure will continue until all columns and rows are exhausted.

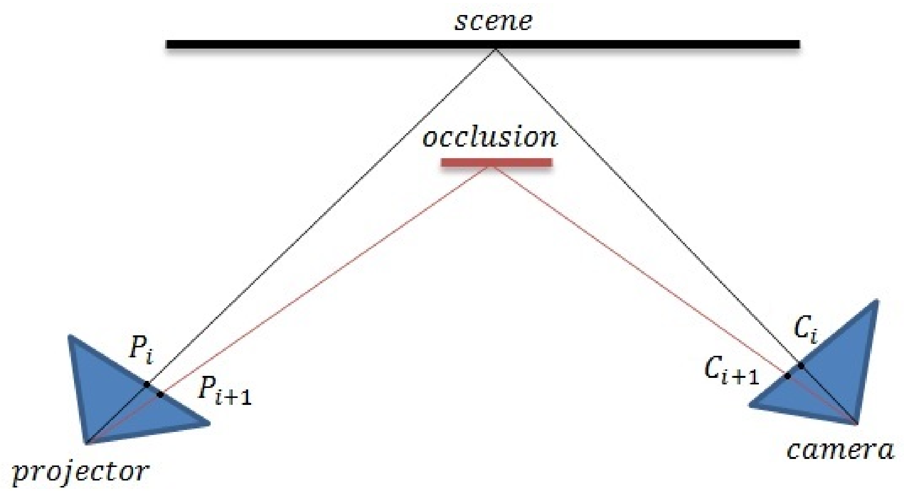


Figure 2.9: Violation of the monotonicity assumption. p_i is in the left of p_{i+1} when we scan from left to right in the projected pattern. But, in the captured image c_i is after c_{i+1}.

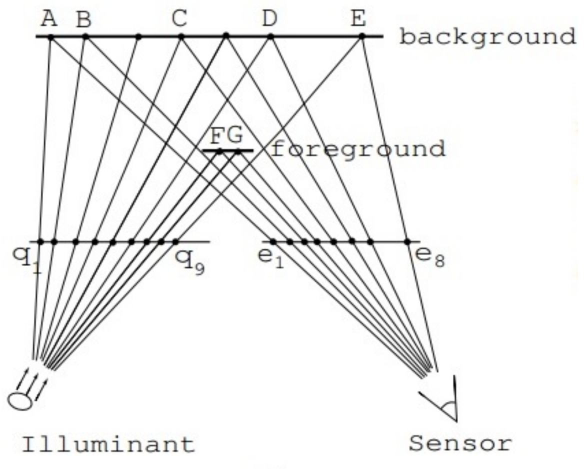


Figure 2.10: A thin (F, G) surface is occluding the (A, B, C, D, E) surface. There are 9 projected transitions while 8 of them are captured. [50].

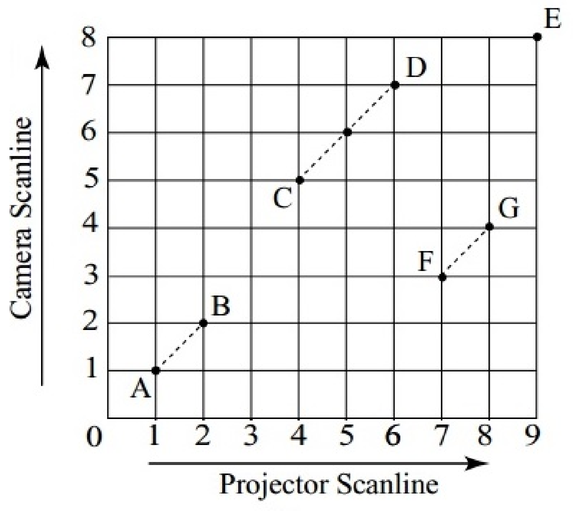


Figure 2.11: The multipass dynamic programming recovers the (A, B, C, D, E) path in the first pass, and then (F, G) path is recovered. [50].

2.2.2.2 M-array

The M-array (perfect maps) method is another class of spatial coding methods proposed by Etzion [17]. M-arrays are pseudorandom arrays of dimensions $r \times v$, where each submatrix of dimension $n \times m$ appears only once in the whole pattern (matrix M). An M-array of dimensions $n_1 \times n_2$ with a window property of $k_1 \times k_2$ requires a pseudorandom sequence of length $n = 2^{k_1 k_2} - 1$ (minus one is because zero submatrix is not permitted); where $n = n_1 \cdot n_2$, $n_1 = 2^{k_1} - 1$ and $n_2 = \frac{n}{n_1}$. Choosing an appropriate window property determines the robustness of the pattern to occlusions [46]. De Bruijn patterns are represented in one dimension (scan line), but M-array patterns are represented in two dimensions.

There are different proposed patterns using M-array which differ in the representation of their elements. Figures 2.12, 2.13, and 2.14 show different pattern representations. Figure 2.12 shows a binary M-array pattern proposed by Spoelder et al. [47]. Its pattern is a checkerboard, where the M-array elements are placed in the white squares and the black squares are used for pattern segmentation. The binary (0 and 1) symbols can be replaced by two different colors.

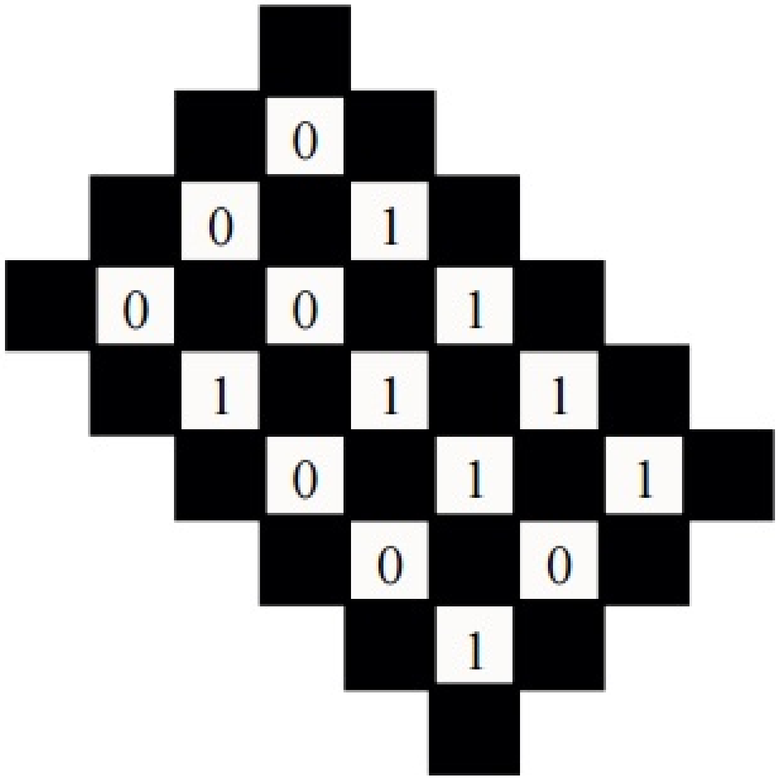


Figure 2.12: A portion of the M-array binary pattern [46].

Figure 2.13 shows a pattern proposed by Griffin [21]. This pattern is an array of four symbols $\{1, 2, 3, 4\}$. There are two different approaches of projecting this array. The first one is representing each symbol with a different color. The second approach is to represent each symbol with a shape primitive. Figure 2.13 shows an example of primitives for three symbols.

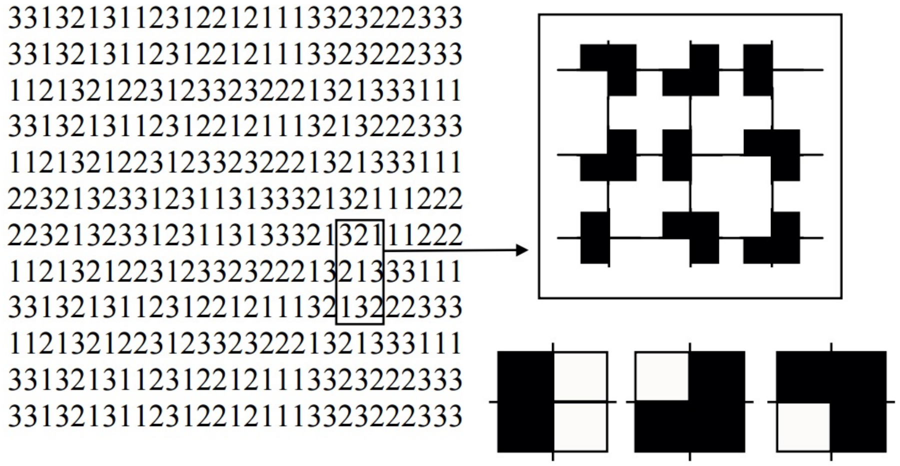


Figure 2.13: M-array pattern proposed by Griffin et al. [21]. Three shape primitives are used to represent the symbols of the alphabet $\{1, 2, 3\}$.

Figure 2.14 shows another pattern proposed by Morano et al. [37]. For generating this pattern, a brute-force approach is used. For creating an M-array with 3 different colors and a window property of 3×3, the following steps are performed: first choosing a subarray 3×3 randomly and placing it at the top left of the M-array. Second, consecutive random columns of 1×3 and rows of 3×1 are added to the right and below the initial subarray, respectively, while preserving the window property. Third, whenever the process reaches a state where adding a new column or row is not possible, the array is cleared, and the algorithm starts again with another initial random subarray. The property of this pattern is that every 3×3 array of color dots is unique and, it appears just once in the whole pattern. The designed pattern is applied by Desjardins et al. [16].

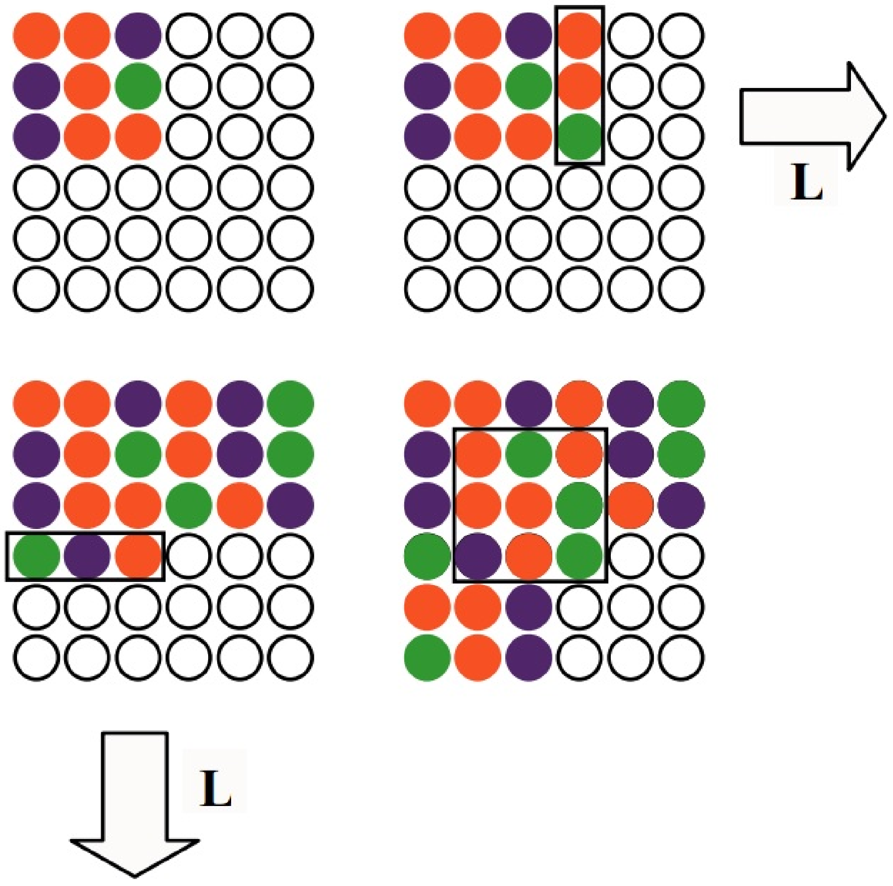


Figure 2.14: An M-array pattern with coloured dots [46].

Chapter 3

Proposed Method

In this chapter, a single-shot 3D reconstruction method is introduced. This gradient based method uses global optimization and local optimization, and the new method results in accurate and dense 3D models of dynamic scenes.

This method uses one projector and two cameras. The projector is employed in the system to superimpose texture onto the surface of the objects by projecting the De Bruijn pattern described in section 2.2.2.1. In addition, two cameras are used for capturing images from different viewpoints.

After capturing and rectifying images, image processing methods are required. They include color normalization, feature extraction, and establishing the correspondence points between images. This method uses two approaches to get a dense and accurate set of corresponding points. The first approach uses spatial information to detect interest points and uses global optimization to get an initial set of dense corresponding points. Then, by applying a local optimization, the method refines the corresponding points in their locations. The other approach uses spatial information to detect interest points, but is uses only one global optimization to get a set of dense and accurate corresponding points. At the end, 3D information is extracted by applying triangulation to the final set of corresponding points.

The steps of the proposed technique are presented in the following parts in more details:

- Calibrating the camera(s);
- Projecting a pattern;
- Normalizing color;
- First approach:

  Extracting dense corresponding point; and,

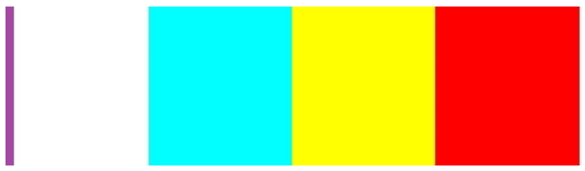
  Correspondence refinement.
- Second approach:

  Extracting dense and accurate corresponding points..

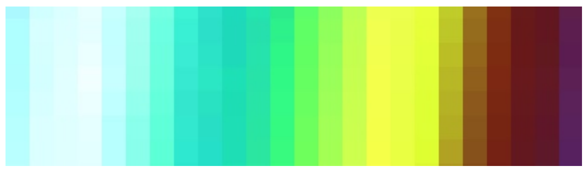
3.1 Calibrating The Camera(s)

There are two possible setups for structured light methods: camera-projector and camera-camera systems.

The camera-projector system consists of a camera for capturing the data and a projector for creating texture on the scene. The camera and the projector should be calibrated and their relative position should be known. The camera-projector system has some drawbacks. Consider the De Bruijn pattern explained in section 2.2.2.1. A portion of the De Bruijn pattern that the projector illuminates is illustrated in Figure 3.1(a). The pattern captured by the camera is illustrated in Figure 3.1(b). These two patterns seem very different. In other words, sometimes a broader spectrum of the projected color stripes appear in the captured image, Figure 3.1(b). This phenomenon, called color fringes, is a type of color distortion. Color fringes cause extra edges in the captured images compared to the projected pattern. These extra edges create extra color gradients in the captured images which cannot be matched to that of the projected pattern. Moreover, corresponding pixels belonging to the corresponding stripes between the projected pattern and the captured image have different color intensities, Figure 3.2. All of these disadvantages result in sparse matching points between the captured image and the projected pattern.



(a)



(b)

Figure 3.1: (a) Color stripes of the projected pattern, and (d) The captured image has color fringes.

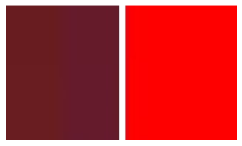


Figure 3.2: The intensity of a red pixel in the right image (projected pattern) is $(255, 0, 0)$, while in the left image (captured image) is $(104, 23, 17)$. These two pixels are representing two corresponding pixels, one on the projected image by the projector and the other on the captured image by the camera. These two corresponding pixels are supposed to have the same intensities but it does not happen in practice.

Because of the above discussed drawbacks, a camera-camera system is used in this thesis. In this setup, two cameras are used in a stereo rig. These two cameras are calibrated using a publicly available camera calibration toolbox [8], and there is no need to calibrate the projector because the projector is used to projecting the pattern to create texture on the object surface. The intrinsic parameters of each camera are obtained after calibration, and the results of the relative position between the two cameras are shown in Figure 3.3. Also, the error of camera calibration of each camera is shown in Figure 3.4 which causes error in rectification and the result of the next steps, since the calibration results are used to rectify the image pair. Using rectified images in 3D reconstruction methods reduces the search space for finding corresponding points between a pair of images in the following steps from $O(mn)$ to $O(m)$; where the image size is $m \times n$.

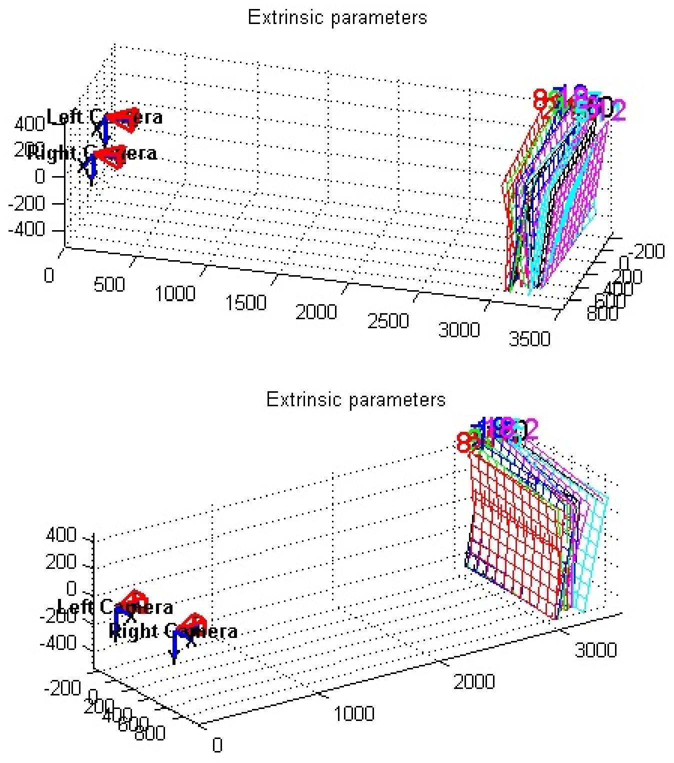


Figure 3.3: The extrinsic parameters. It shows the relative position between the left camera and the right camera.

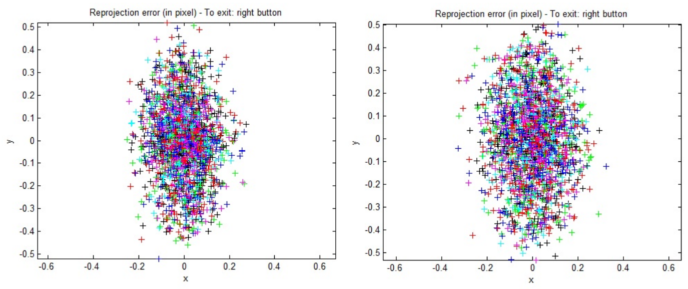
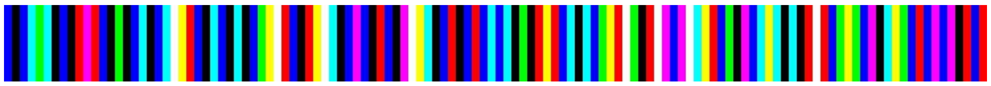


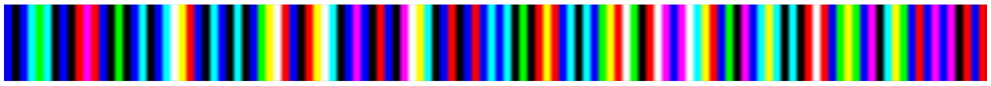
Figure 3.4: Camera calibration errors. (a) Left camera calibration error, and (b) right camera calibration error.

3.2 Projecting a Pattern

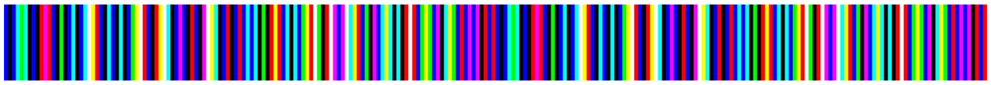
Structured light methods project a pattern onto the scene to increase the number of corresponding points. The projected pattern in this thesis is based on the De Bruijn pattern in section 2.2.2.1. Different versions of the De Bruijn patterns are generated in this thesis and visualized in Figure 4.1. Figure 3.5(a) shows a De Bruijn pattern with stripe width of 8 pixels, and Figures 3.5(b) shows a De Bruijn pattern which is blurred under a Gaussian filter (filter size is 5×5 and sigma is 2) with stripe width of 8 pixels. The Gaussian filter makes the De Bruijn pattern blurred. The blurriness creates intensity variation for pixels of a stripe. Also, it removes sharp edges between colors by adding a region with a broader spectrum of colors. Figures 3.5(c) and 3.5(d) show a De Bruijn pattern which is repeated two times and three times, respectively. The results of all the patterns are visualized and evaluated in chapter 4. Figure 3.5(e) shows the complement of the De Bruijn pattern. In Figure 3.5(f) a random binary matrix is combined with the De Bruijn pattern, and Figure 3.5(g) shows a De Bruijn pattern when the average of the color intensity value of each consecutive stripes is added in between.



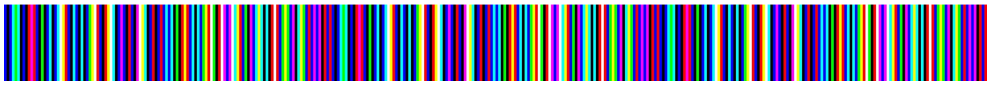
(a)



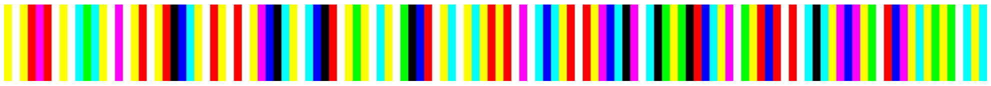
(b)



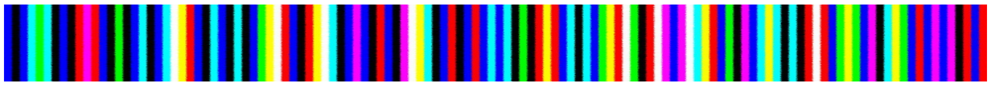
(c)



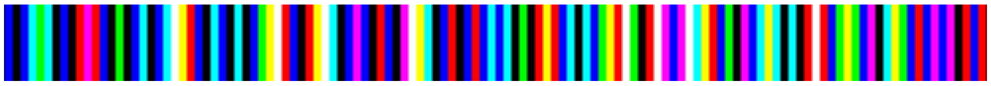
(d)



(e)



(f)



(g)

Figure 3.5: (a) The De Bruijn pattern with stripe width of 8 pixels, (b) the blurred De Bruijn pattern with a Gaussian filter and stripe width of 8 pixels, (c) two times repeated De Bruijn pattern, (d) three times repeated De Bruijn pattern, (e) complement of the De Bruijn pattern, (f) combination of a random binary matrix with the De Bruijn pattern, and (g) the De Bruijn pattern with average of color intensity value of each adjacent stripes in between39

3.3 Normalizing Color

In general, the distribution of color values in an image depends on the illumination and the color characteristics of the camera. Even under the same illumination, two cameras of the same model capture two images of the same scene often result in images in noticeable differences because of their differences in the color characteristics of their sensors. Since the proposed method uses two cameras and is based on color gradient, having similar color in two images is a pre-condition. Color normalization between two images is required to make the color distribution similar between them.

The color transformation method proposed by Reinhard et al. [40] uses simple statistical analysis to impose one image's color characteristics to another image in the Lab color space. The Lab color space is based on perceptual uniformity. Thus, it can be used to make accurate color balance corrections. In particular, both images are converted to the Lab space first, and are split to three channels L, a, and b. In the following, the color distribution of the target image, I_t, will be transformed to the color distribution of the source image, I_s, to have the similar color distribution. In addition, L_t, a_t, b_t are the split channels of the I_t, and L_s, a_s, and b_s are the split channels of the I_s. For purposes of transforming colors of the target image to the source image, computing the mean and the standard deviation along each of the three channels of both images is sufficient; $\bar{L_t}$, $\bar{a_t}$, $\bar{b_t}$, $\bar{L_s}$, $\bar{a_s}$, and $\bar{b_s}$ are the mean of L_t, a_t, b_t, L_s, a_s, and b_s, respectively. Also σ_t^L, σ_t^a, σ_t^b, σ_s^L, σ_s^a, and σ_s^b are the standard deviation of the L_t, a_t, b_t, L_s, a_s, and b_s, respectively. Then, two computational steps are required. First, the computations in equation 3.1 are performed:

$$\begin{aligned} L_t^* &= L_t - \bar{L_t} \\ a_t^* &= a_t - \bar{a_t} \\ b_t^* &= b_t - \bar{b_t} \end{aligned} \tag{3.1}$$

Second, scaling and shifting of L_t^*, a_t^*, and b_t^* are performed as shown in equation 3.2.

$$\begin{aligned} L_t^* &= (\sigma_t^L / \sigma_s^L) \times L_t^* + \bar{L_s} \\ a_t^* &= (\sigma_t^a / \sigma_s^a) \times a_t^* + \bar{a_s} \\ b_t^* &= (\sigma_t^b / \sigma_s^b) \times b_t^* + \bar{b_s} \end{aligned} \tag{3.2}$$

After the above transformation, L_t^*, a_t^*, and b_t^* will have the same distribution as that of the I_s. Then, color space conversion is needed to convert the target image from the Lab space to the RGB space.

3.4 First Approach

3.4.1 Extracting Dense Corresponding Points

After normalizing the images in the previous step, there are two color normalized images that are both rectified and have the same size. In this thesis, one image is denoted by I^R that is captured by the right camera and the other image is denoted by I^L that is captured by the left camera. The method looks for correspondences between I^R and I^L. Since the images are rectified, the corresponding pixels between images can be found on the same scanline. To find the corresponding points in the j^{th} scanline in both images, the method gets a score for each pixel pairs between I^L and I^R. The size of the score matrix for each scanline is $I^L_{width} \times I^R_{width}$, where I^L_{width} is the width of I^L, and I^R_{width} is the width of I^R. Scoring the pixels is based on their gradient, as shown in equation 3.3:

$$score(p_i^L, p_j^R) = \frac{1}{\sum_{k=r,g,b} \frac{(|p^L_{i(k)} - p^R_{j(k)}|)}{255} + 0.02} \tag{3.3}$$

where p_i^L is the i^{th} pixel in I^L, p_j^R the j^{th} pixel in I^R, $p^L_{i(k)}$ the intensity of the k^{th} channel of p_i^L, and $p^R_{j(k)}$ the intensity of the k^{th} channel of p_j^R. The more the similar between p_i^L and p_j^R, the higher the $score(p_i^L, p_j^R)$.

Then, the proposed dynamic programming method by Cox et al. [14] is applied to match the pixels between the two images. The dynamic programming (DP) provides dense correspondences for almost all illuminated pixels in the captured images if there are no occlusions. Algorithm 1 demonstrates the pseudo code of calculating and reconstructing the optimum match using the DP approach. The output of the Algorithm 1 is a set of matched points. Using DP, matches along each scanline in both images are considered. Since the whole scanline is used to find correspondences, DP is considered as a global optimization method.

Algorithm 1 Finding the matched points using dynamic programming

```
procedure MATCHING PROCEDURE
    $I^L_{width} \leftarrow$ the width of $I^L$
    $I^R_{width} \leftarrow$ the width of $I^R$
    $cost \leftarrow$ a zero matrix $I^L_{width} \times I^R_{width}$
    $cases \leftarrow$ a zero matrix $I^L_{width} \times I^R_{width}$
    for $i = 1$ to $I^L_{width}$ do
        for $j = 1$ to $I^R_{width}$ do
            $temp1 \leftarrow cost(i-1, j-1) + score(p^L_i, p^R_j)$
            $temp2 \leftarrow cost(i-1, j)$
            $temp3 \leftarrow cost(i, j-1)$
            $cost(i, j) \leftarrow$ the maximum value among $temp1, temp2, temp3$
            if $cost(i, j) = temp1$ then
                $cases(i, j) \leftarrow 1$
            else if $cost(i, j) = temp2$ then
                $cases(i, j) \leftarrow 2$
            else if $cost(i, j) = temp3$ then
                $cases(i, j) \leftarrow 3$
            end if
        end for
    end for
end procedure
```

Algorithm 2 Cont'd Algorithm 1

```
$lw \leftarrow I^L_{width}$
$rw \leftarrow I^R_{width}$
**while** $lw \neq 0$ **and** $rw \neq 0$ **do**
    **if** $cases(lw, rw) = 1$ **then**
        $lw$ is matched to $rw$
        $lw \leftarrow lw - 1$
        $rw \leftarrow rw - 1$
    **else if** $cases(lw, rw) = 2$ **then**
        $lw$ is not matched
        $lw \leftarrow lw - 1$
    **else if** $cases(lw, rw) = 3$ **then**
        $rw$ is not matched
        $rw \leftarrow rw - 1$
    **end if**
**end while**
```

3.4.2 Correspondence Refinement

An initial set of dense corresponding points are obtained using the method described earlier. As regards to dynamic programming, all illuminated pixels in I^L are matched to their corresponding pixels in I^R while their matched locations are integer values. For example, p^L_{100} is matched to p^R_{350} where both 100 and 350 are integer numbers. However, in the ground truth, the location of correspondence of p^L_i is a floating number. For example, in the ground truth, the p^L_{100} is matched to $p^R_{350.8}$. The winner-takes-all idea is applied to refine the locations of detected corresponding points. The process is described in detail in the following.

The idea considers a pair of matched points obtained by the previous step, p^L_i and p^R_j, and consecutive neighbors of p^R_j (p^R_{j-1} and p^R_{j+1}), as shown in Figure 3.6. In Figure 3.6, the (r, g, b) values of each pixel are shown inside of them.

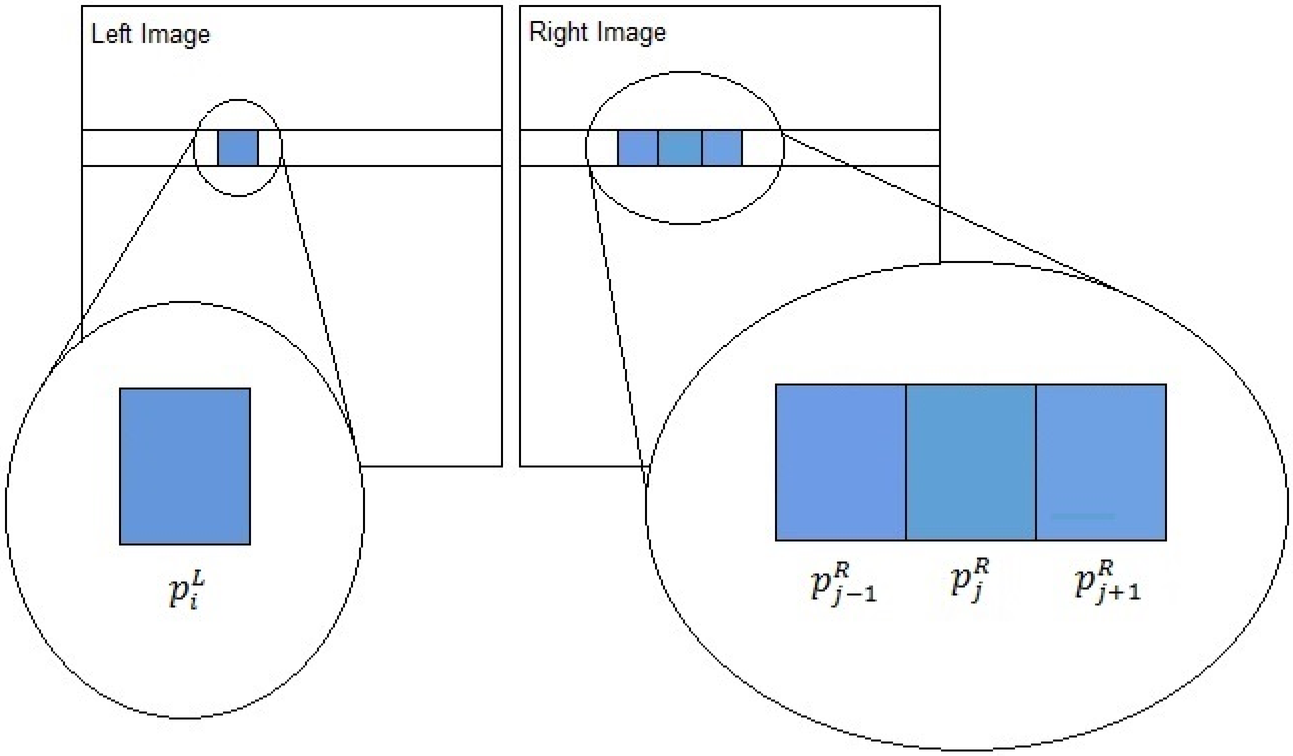


Figure 3.6: The p_i^L is matched to the p_j^R, because the $score(p_i^L, p_j^R)$ is higher than others. Moreover, p_{j-1}^R and p_{j+1}^R are the preceding and following neighbors of p_j^R. The intensity of each pixel is shown as the filled color.

Then, these consecutive pixels (p_{j-1}^R, p_j^R, and p_{j+1}^R) are linearly interpolated and n new pixels are generated between them, where n is the number of newly interpolated pixels between each two consecutive pixels. In particular, by linearly interpolating p_{j-1}^R and p_j^R, n new pixels are generated between them. Also, by linearly interpolating p_j^R and p_{j+1}^R, n new pixels are generated between them, too. After interpolation, there is a sequence of $2n + 1$ pixels between p_{j-1}^R and p_{j+1}^R; in other words,

$p^R_{1l}, p^R_{2l}, ..., p^R_{nl}, p^R_j, p^R_{1r}, p^R_{2r}, ..., p^R_{nr}$. The intensity of recently added pixels is based on the color intensity differences between p^R_j and its preceding and following neighbors in RGB color space. For example, the RGB color intensity of the p^R_{j-1}, p^R_j, and p^R_{j+1} are (r_l, g_l, b_l), (r, g, b), and (r_r, g_r, b_r), respectively. The color differences between the p^R_{j-1} and the p^R_j is (r', g', b'), and the color differences between the p^R_j and the p^R_{j+1} is (r'', g'', b''). In particular, the RGB color intensity value of pixels between p^R_{j-1} and p^R_j is computed using equation 3.4, and the RGB color intensity value of pixels between p^R_{j-1} and p^R_j is computed using equation 3.5, where $0 < k \leq n$.

$$p^R_k = (r_l + r'/(n+1) * k, g_l + g'/(n+1) * k, b_l + b'/(n+1) * k). \tag{3.4}$$

$$p^R_k = (r + r''/(n+1) * k, g + g''/(n+1) * k, b + b''/(n+1) * k). \tag{3.5}$$

After this process, the method finds the best matched pixel of p^L_i in $p^R_{1l}, p^R_{2l}, ..., p^R_{nl}, p^R_j$, $p^R_{1r}, p^R_{2r}, ..., p^R_{nr}$ by comparing each pair gradient. The minimum gradient indicates the best matched pair. Since the method is finding the matched pixel for one pixel (p^L_i) among $2n+1$ pixels, the size of the cost matrix is $1 \times 2n + 1$. In this section, inasmuch as a portion of a scanline is looked for the correspondences, the refinement process is considered as a local optimization method.

At the end, the location of p^R_j is refined to $p^R_{j^{new}}$ by equation 3.6, where k^{th} pixel in $p^R_{1l}, p^R_{2l}, ..., p^R_{nl}, p^R_j, p^R_{1r}, p^R_{2r}, ..., p^R_{nr}$ is matched to p^L_i.

$$j^{new} = j - ((n+1) - k)/(n+1). \tag{3.6}$$

Figure 3.7 visualizes the refinement process when the number of added pixels in each interpolation is 5 ($n = 5$). After adding virtual pixels, there are $2n+1$ (11) pixels. The p_i^L was matched to the p_j^R, but after the refinement process, the p_i^L is matched to the p_{4l}^R. The j^{new} is $j-((5+1)-4)/(5+1)$ and the p_i^L is matched to $p_{j^{new}}^R$.

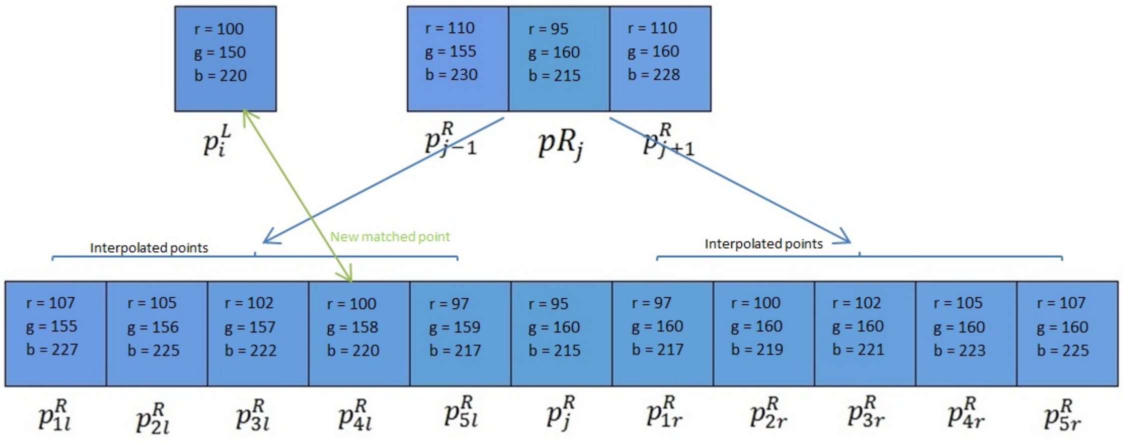


Figure 3.7: The p_i^L is matched to the p_j^R. Then the location of p_j^R is refined by adding some ($n = 5$) virtual pixels between p_j^R and its preceding and following neighbors. Then, the new matched point is obtained by comparing the gradient of each pair.

3.5 Second Approach

3.5.1 Extracting Dense And Accurate Corresponding Points

As it is described in section 3.4.1, one image is denoted by I^R that is captured by the right camera and the other image is denoted by I^L that is captured by the left camera. The method looks for correspondences between the I^R and the I^L. The I^L is considered as a reference image and the method is looking for each pixel correspondence in the I^R. Since the images are rectified, the corresponding pixels between images can be found on the same scanline. The disparity values of a matched pixel pairs using the DP are integer numbers and they are not accurate enough because the disparity values in the ground truth are float numbers. In order to obtain accurate disparity values, some virtual pixels are added before and after of each pixel in a scanline in the I^R similar to the described method in section 3.4.2. After adding virtual pixels to a scanline of the I^R, the number of pixels in the scanline is increased. If a scanline has m pixels and n virtual pixels between each two adjacent pixels are added, the scanline will have $m \times n$ pixels. Then the method gets a score for each pixel pairs between a scanline in the I^L and a modified scanline in the I^R, as shown in equation 3.3. The size of the score matrix for each scanline is $I^L_{width} \times I^R_{width} \times n$, where I^L_{width} is the width of I^L and number of pixels in a scanline in I^L, I^R_{width} is the width of I^R and number of pixels in a scanline in I^R, and n is number of virtually added pixels between each two adjacent pixels in a scanline in I^R. Then, the DP is applied to obtain the corresponding points between a scanline in I^L and I^R. Since the scanline of I^R has virtual pixels and high resolution, an indexing scheme is needed after the matching. For indexing, the location of correspondence of each pixel in I^L, which corresponds to a point in I^R, should be divided by n. For example, if the p_i^L is matched to the p_j^R, where $1 \leq j \leq I_{width}^R \times n$, by indexing p_i^L is matched to $p_{j/n}^R$.

Chapter 4

Results and Evaluations

Different versions of the De Bruijn patterns which are described in 3.2 are applied to several datasets to demonstrate the ability of the new structured light method. The method is implemented using C++ on a PC with 3.4GHz Intel Core i7 CPU and 16GB RAM and the running time is less than 6 seconds. For the simulated datasets, both quantitative and qualitative evaluations are provided. Also, for the real world datasets qualitative evaluations are provided.

4.1 Simulated Datasets

In the simulated experiments, a comprehensive 3D animation software Maya [1] is used to project different patterns onto the intended object and create the ground truth. The Stanford Bunny model is employed as the object in these experiments. In addition, two cameras in Maya are used to capture the bunny from different viewpoints. Figure 4.4 shows the captured images of the bunny from two viewpoints under the De Bruijn pattern. Figure 4.1(b) shows the ground truth of the depth map from one viewpoint obtained by Maya. Figures 4.1(c), 4.1(d), 4.1(e), 4.1(f), 4.1(g), 4.1(h), and 4.1(i) show the recovered depth maps by the new method(first approach) using patterns 3.5(a), 3.5(b), 3.5(c), 3.5(d), 3.5(e), 3.5(f), and 3.5(g), respectively. In the simulated datasets, it is observed that if the width of the stripes is less than 8 pixels, the pattern will be uniformly scaled by Maya to fit the display whereas the scaling can be modeled by a Gaussian filter. Since, the cameras in Maya are simulated, they both have the same color characteristics. Hence, they do not need color normalization.

In order to qualitatively evaluate the new method (first approach), the ground truth in the close up view of the part inside the red rectangle in Figure 4.2(a) are shown in Figure 4.2(b). The close up view of that part in the recovered depth maps with different patterns are shown in Figure 4.2. To make these figures more informative, the result of a traditional spatial coding method which is based on matching the edges is shown in Figure 4.2(j), and the result of a temporal coding method (gray code) is shown in Figure 4.2(k). Based on these figures, it is clear that the new method with almost all color patterns can reconstruct very fine details on the object surface and is very close to the ground truth. Moreover, Figures 4.2(c), 4.2(d), 4.2(e), 4.2(f), 4.2(g), 4.2(h), 4.2(i) show that the new method is not sensitive to the projected pattern. The proposed method can reconstruct the shape of object under different color patterns because the method is pixel based. Comparing different figures in Figure 4.2 indicates that the new method can reconstruct the shape better than traditional

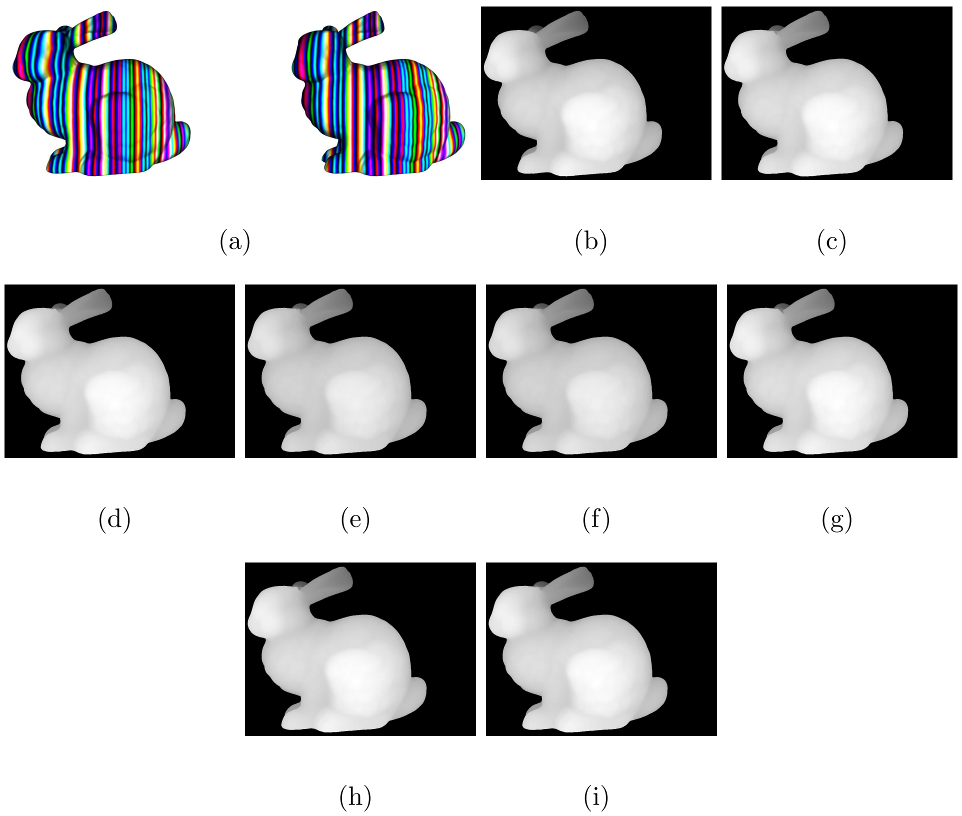


Figure 4.1: (a) Captured images of the bunny from two different viewpoints, (b) the depth map by Maya; (c), (d), (e), (f), (g), (h), and (i) are the recovered depth maps using the new method with using different patterns 3.5(a), 3.5(b), 3.5(c), 3.5(d), 3.5(e), 3.5(f), and 3.5(g), respectively.

spatial methods (edge based ones) though both have error around edges and discontinuous surfaces. Discontinuous surfaces cause problems because of small shadows and make retrieving surface information difficult. Figure 4.2(j) demonstrates the result of a traditional spatial coding method which is based on edge detection. As it is clear, it has large errors because it is based on edge detection and an error in edge detection causes further errors.

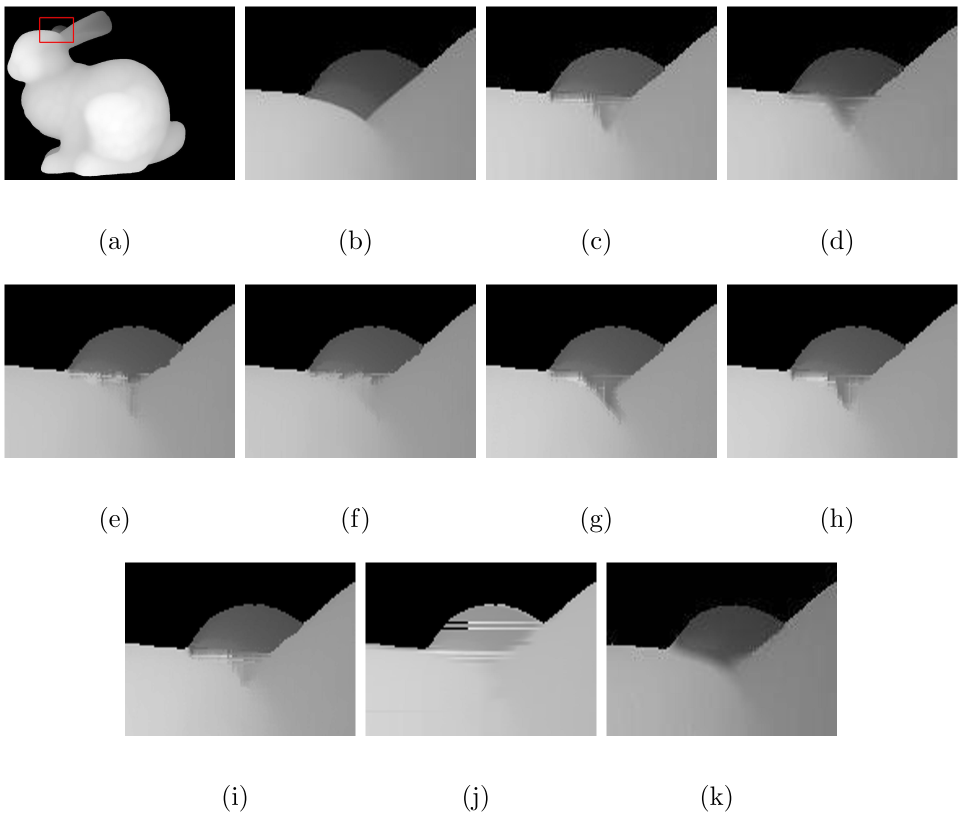


Figure 4.2: (a) The ground truth depth map, (b) the close up view of the area inside the red rectangle in (a); (c), (d), (e), (f), (g), (h), (i) are the close up view of the recovered depth map using the new method and different patterns 3.5(a), 3.5(b), 3.5(c), 3.5(d), 3.5(e), 3.5(f), and 3.5(g), respectively; (j) the recovered depth map using a traditional spatial coding method which is based on the edges, and (k) the recovered depth map using a temporal coding method.

Moreover, to quantitatively evaluate the new method (both approaches), two measures are used: recovery accuracy, and mean of disparity error.

❶ Recovery accuracy: This measure demonstrates the accuracy of the recovered pixels and it is denoted by $\frac{m}{n} \times 100\%$, where m is the number of pixels whose depth are established by the method; and n is the number of pixels who have depth values in the simulated ground truth.

❷ Mean of disparity error: This measure computes the error (measured in disparity units) between the recovered disparity map as denoted by $d_C(x,y)$ and the ground truth map as denoted by $d_T(x,y)$ for every pixels. The mean disparity error is defined as equation 4.1.

$$disparityError = \frac{1}{N}\sum_{(x,y)} |d_C(x,y) - d_T(x,y)|. \tag{4.1}$$

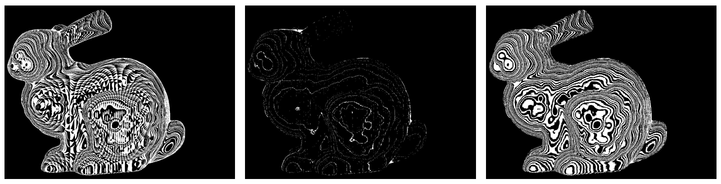
Table 4.1 shows the quantitative evaluation results for two patterns (3.5(a) and 3.5(b)). From the table it can be seen that the new method with both approaches using both mentioned patterns provides good results, with dense correspondences in a single-shot. It can retrieve dense correspondences between images because it is based on pixels, rather than edges as are used in traditional one-shot spatial coding methods. Also, the ground truth depth map has 352747 pixels, and the gray code method reconstruct 99% of these pixels while the new method using either the first approach or the second one can recovered 100%. Comparing the number of recovered pixels between the gray code and the new method indicates that almost all of the pixels are recovered by the new method as well as the gray code method while the new method projects just one pattern instead of 40 patterns by the gray code method. Moreover, the mean of disparity error of the gray code method is 0.57 which is higher than that of the new method. It is worthy to mention that the second approach of finding correspondences is not as fast as the first approach because of the score

	mean of disparity error
Gray code method	0.57
New method(1st approach) using 3.5(a)	0.355
New method(2nd approach) using 3.5(a)	0.353
[12] method using 3.5(a)	0.522
New method(1st approach) using 3.5(b)	0.345
New method(2nd approach) using 3.5(b)	0.339
[12] method using 3.5(b)	0.519
Traditional spatial coding method	0.91

Table 4.1: Evaluation of mean of disparity error.

matrix size. Since both results are similar, in the following only the results of the first approach are shown.

Figure 4.3 shows the difference between the recovered depth map by the gray code method, the new method, and the method in [12] using the blurred De Bruijn pattern and the ground truth depth map, respectively. The white pixels in Figure 4.3 represent pixels which their disparity value vary more than a 0.5 pixel from the ground truth disparity map. As shown in Figure 4.3, it is clear that the new method has a smaller error range compared with the other methods.

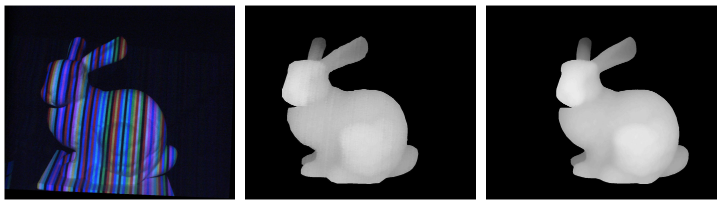


(a) (b) (c)

Figure 4.3: (a) Error map of the gray code method, (b) error map of the new method, and (c) error map of the method in [12].

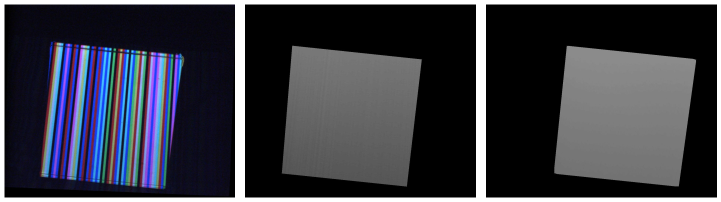
4.2 Real World Datasets

In the real world experiments, two Point Grey Flea3 cameras are used for capturing the images and a Dell M115HD WXGA LED projector is used for projecting patterns described in section 3.2. Also, a Kinect 2.0 is used for getting the depth of the objects and its results are compared with that of using the new method. The resolution of the cameras is 1224×1024, and the projector is 1024×768. It shows that even if the projector is out of focus, the new method can provide dense correspondences and accurate results. In these experiments, the blurred pattern is not used anymore. It is observed that when the projector is out of focus, the projected pattern is blurred. It is also pointed out in [10] that if the projector is out of focus and projects to a display surface, the display is uniformly blurred by a point-spread function which can be modeled as a $2D$ Gaussian filter. Therefore, instead of projecting the blurred pattern, the idea of setting the projector out of focus is used. The advantage of being able to work with defocused projector is that in real applications the projector is not able to focus on all the points of a scene. If the projector is out of focus, the temporal coding and traditional spatial coding methods may have problems in identifying the codewords and detecting edges, respectively. The experiments demonstrate that even if the projector is out of focus, the new method can provide dense correspondences and the results are visually pleasant. Since the new method does not use temporal information, it can be applied to both static and dynamic scenes. In the following experiments, the new method is applied to several different scenes with static and dynamic objects. The results of static objects are qualitatively compared to gray code method results, and the results of moving objects are compared to depth maps obtained using Kinect. The following figures show the result of the new method which are visually and qualitatively pleasant. Figures 4.4, 4.5, and 4.6 show the results of static objects. Figures 4.4(a), 4.5(a), and 4.6(a) are captured images of a static object illuminated under a projected pattern in a dark room. Figures 4.4(b), 4.4(c), and 4.4(b) are recovered depth maps by the new method. Figures 4.4(c), 4.5(c), and 4.6(c) are the recovered depth maps by the gray code method. Comparing Figures 4.4(b), 4.5(b), and 4.6(b) with Figures 4.4(b), 4.5(b), and 4.6(b) respectively indicates that the recovered depth maps using the new method are visually pleasant and similar to results of the gray code though the new method projects just one pattern. Figures 4.4 and 4.6 show how the new method can reconstruct amiable model of a curvy object.



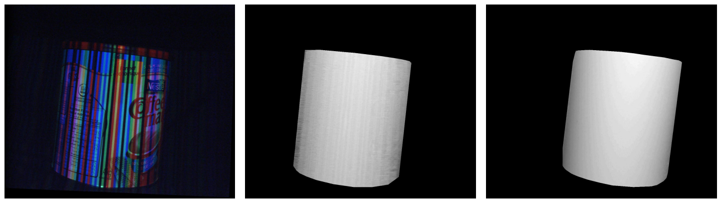
(a) (b) (c)

Figure 4.4: (a) A captured image of the bunny, (b) the recovered depth map by the new method, and (c) the recovered depth map by the gray code method.



(a) (b) (c)

Figure 4.5: (a) A captured image of the flat surface, (b) the recovered depth map by the new method, and (c) the recovered depth map by the gray code method.



(a) (b) (c)

Figure 4.6: (a) A captured image of the coffeemate box, (b) the recovered depth map by the new method, and (c) the recovered depth map by the gray code method.

Figure 4.5 shows that the new method can reconstruct a flat surface perfectly. Figure 4.7 shows the point cloud of the flat surface, and Figure 4.8(a) shows the normal map of the flat surface point cloud. Normal maps are used to store normal vectors. These vectors are encoded by colors in a specific way. Each vector has 3 components (x, y, z), and each component is represented by a specific color, (r, g, b). The x component is represented by the red channel, the y component is represented by the green channel, and the z component is represented by the blue channel. For example, if a pixel looks more bluish, its normal vector has higher value in the z component. By fitting a plane equation to flat surface point cloud, the error between the point cloud and fitted plane can be computed and compared to demonstrate the accuracy of the method. Figure 4.9 shows the error histogram between the point cloud and the fitted plane. On average this error is 0.001 for 461070 points. Also, Figure 4.10, and Figure 4.11 show the point cloud of the bunny and normal map of the point cloud, respectively.

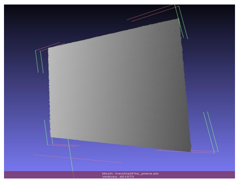
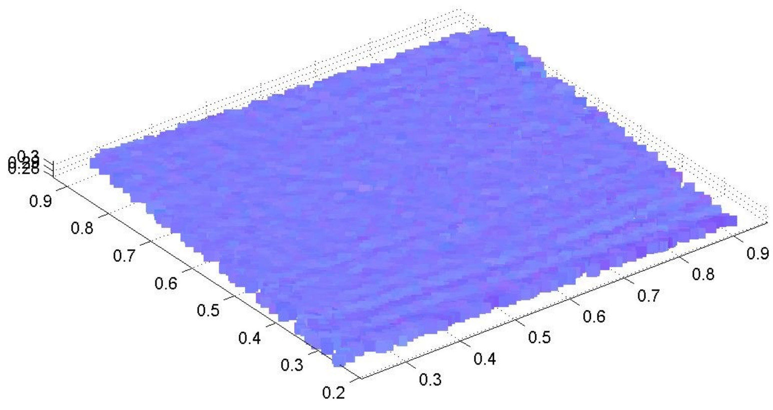
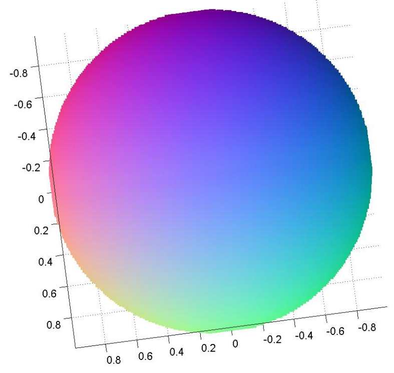


Figure 4.7: The point cloud of the flat surface.



(a)



(b)

Figure 4.8: (a) The normal map of the flat surface point cloud, and (b) the legend of the normal map.

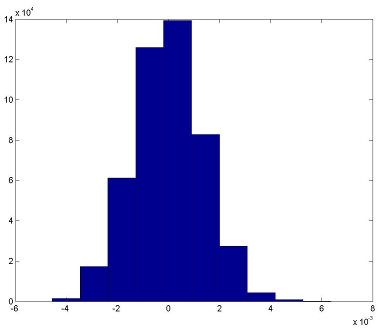


Figure 4.9: Histogram of errors between the fitted plane and the reconstructed flat surface.

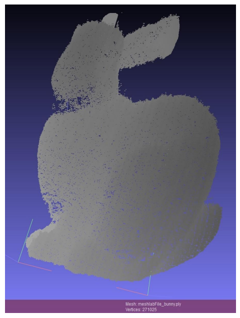
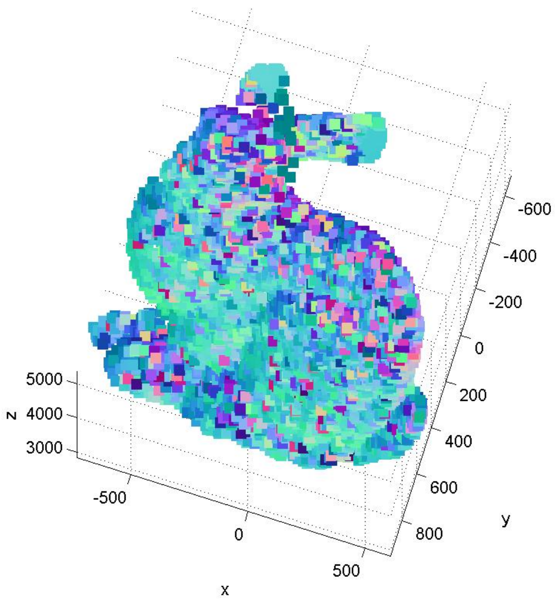
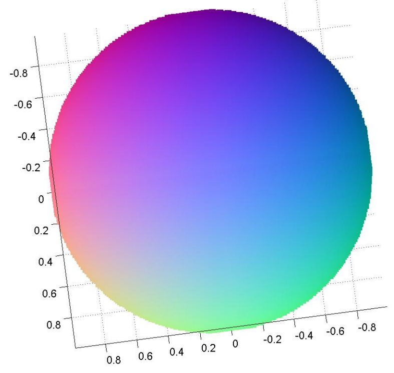


Figure 4.10: The point cloud of the bunny.



(a)



(b)

Figure 4.11: (a) The normal map of the bunny point cloud, and (b) the legend of the normal map.

Moreover, Figures 4.12, 4.13, and 4.14 show the results of moving objects. In these figures, the results of the new method are compared with the results of Kinect. The comparison between the results of the new method and Kinect demonstrates that the new method can get the depth map of a moving object without degradation of the quality of the depth map. Although the new method can reconstruct pleasant shape of the objects, it can not reconstruct regions illuminated under shadows.

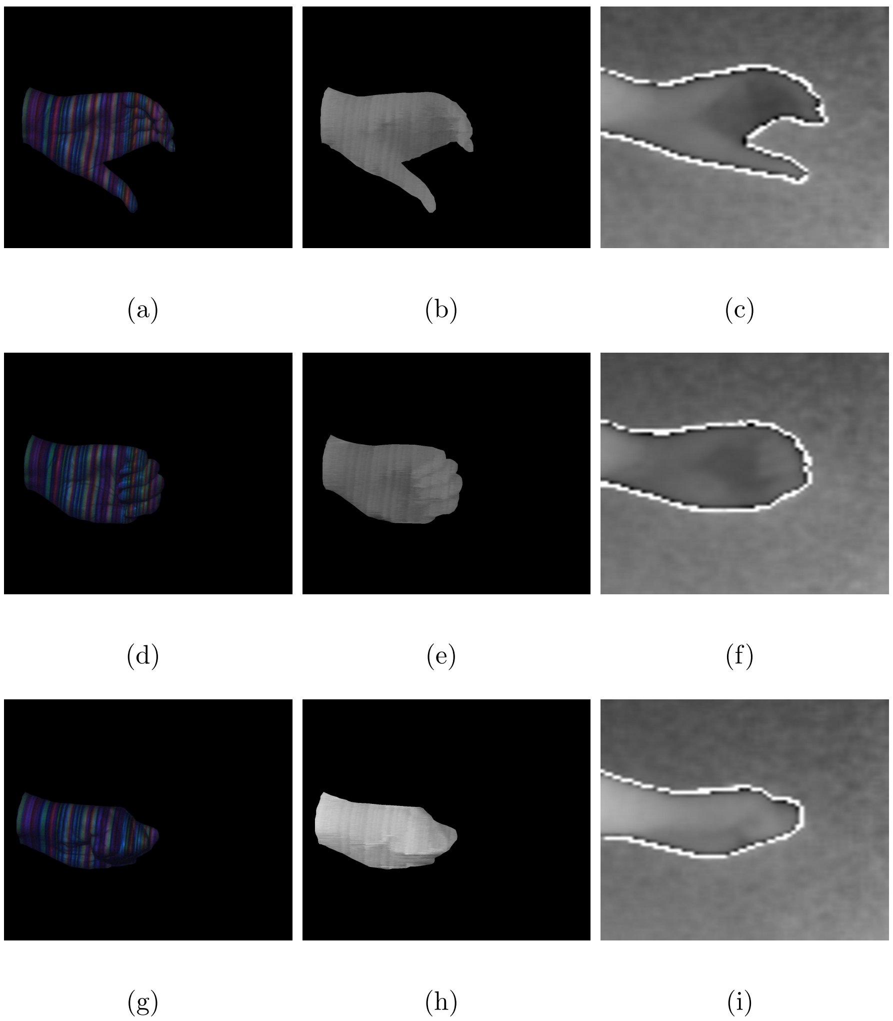
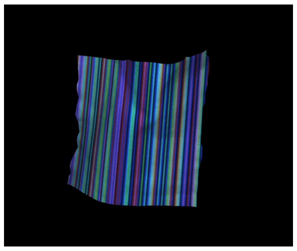
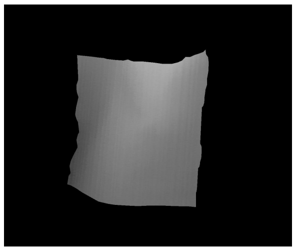
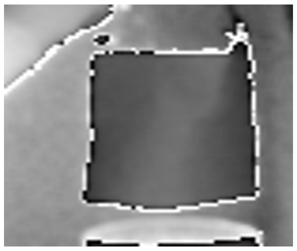
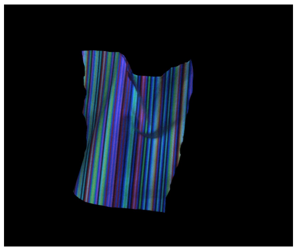


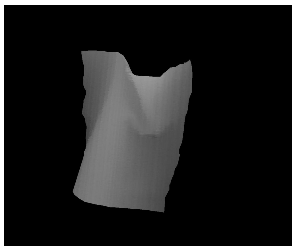
Figure 4.12: (a), (d), and (g) are captured image of the hand gestures; (b), (e), and (h) are the recovered depth map by the new method; and (c), (f), and (i) are the recovered depth map by Kinect.

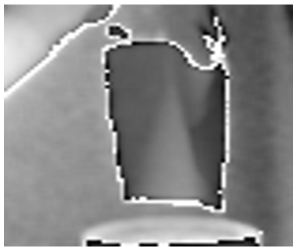

(a)

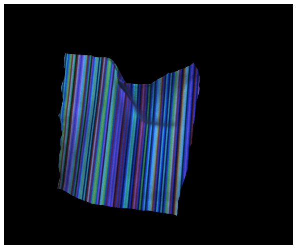

(b)

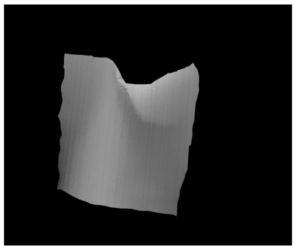

(c)


(d)


(e)


(f)


(g)


(h)

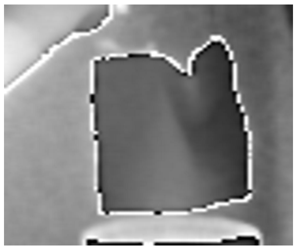

(i)

Figure 4.13: (a), (d), and (g) are captured image of a piece of cloth; (b), (e), and (h) are the recovered depth map by the new method; and (c), (f), and (i) are the recovered depth map by Kinect.

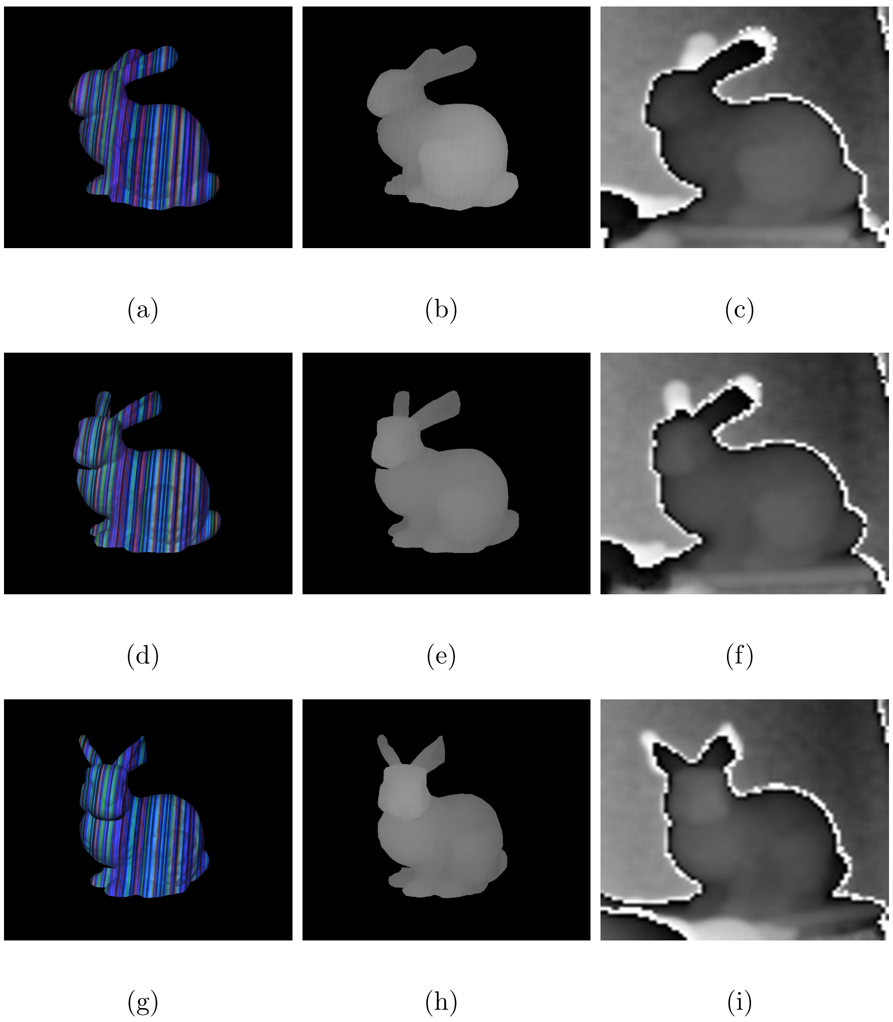


Figure 4.14: (a), (d), and (g) are captured image of a rotating bunny; (b), (e), and (h) are the recovered depth map by the new method; and (c), (f), and (i) are the recovered depth map by Kinect.

Moreover, the proposed method is tested under different indoor ambient lighting condition. In a room with typical and dimmer ambient lighting, the method performs as well as in the dark. Thus, this demonstrates that the proposed method is robust to changes in ambient lighting.

To sum up, a method is presented which recovers a depth map with one pattern only with a smaller disparity error and high pixel recovery accuracy. Moreover, the method can recover the 3D information of a moving object. In addition, the projector does not need to be focused since the method is applicable to the blurred pattern and does not need to detect edges. Having a defocused projector is more practical because it is not usually possible to focus the projector on all the scene points.

Chapter 5

Conclusion and Future Work

3D reconstruction has been an active research area in computer vision for decades. A new dense and accurate 3D reconstruction method is proposed in this thesis which is single-shot and applicable to moving objects. The proposed method has two approaches.

The first approach uses spatial information to detect interest points and uses global optimization to get an initial set of dense corresponding points. Then, by applying a local optimization, the method refines the corresponding points in their locations. This approach improves the resolution of a part of a scanline locally by adding virtual pixels. The major time consuming part in this approach is in the DP($O(I^L_{width} \times I^R_{width})$.

The second approach uses spatial information to detect interest points, but uses only one global optimization to get a set of dense and accurate corresponding points. This approach improves the resolution of a scanline globally. Unlike the first approach, it adds n virtual pixels before and after existing pixels. Having a high resolution scanline increases the time complexity to $O(I^L_{width} \times I^R_{width} \times n)$.

It is worthy to mention that the refinement in both approaches is done in one image only. The reason is that an image is considered as the reference image and the method recovers the disparity of each pixel of the reference image. Also, it is assumed that all the pixels are illuminated using the color pattern and the method is not able to recover pixels which are not illuminated but under shadow.

Moreover, different 3D reconstruction methods are reviewed in this thesis. In addition, calibration which is an inseparable part of 3D reconstruction methods is described.

5.1 Contributions

The goal of this research is to develop a new method to recover the 3D shape of an object. There are different methods to reconstruct the 3D shape, but they are not dense or accurate enough. In Chapter 3, a new structured light method is developed which recovers a dense and accurate depth map compared to previous methods. The new method establishes dense and accurate corresponding points by using a single-shot projected pattern unlike typical single-shot methods. The new method is applicable to dynamic scenes because it projects just one color pattern whereas a typical multi-shot method cannot. Also, the proposed method is not limited to the De Bruijn pattern because it is pixel based method and can recover the 3D models using different color stripe patterns as described in 4.1. Moreover, it can also work with color objects.

The proposed method is applied to both simulated data and real world data. In both cases, it performs better than other existing methods.

5.2 Future Work

Although the method can get accurate and dense results, there are two major directions for future works.

❶ The proposed method is able to work for images captured in the air, but it cannot work with underwater images because the assumption of having rectified images is violated in underwater images.

❷ The proposed method is not able to work perfectly for objects which have shiny parts because those parts are captured as a white area without any texture. In this case, the object is getting additional illumination from inter-reflection.

❸ The proposed method is limited to pixels that are illuminated under the pattern and not covered by shadows. The reason is that the shadow does not allow pixels to reflect the projected color pattern correctly and the method fails to find corresponding points for those pixels in shadow.

Glossary

Image Rectification

In stero, finding a corresponding point between two or more images can be difficult. Because finding a correspondence of a pixel in the other image requires a search in two-dimensions(image size). However, if the two cameras are calibrated and aligned correctly to be coplanar, the images can be rectified and the search space is reduced to one dimension - a scanline. As a result of rectification, if the location of a pixel in the left image is known, the correspondence of that pixel will be found in the right image by searching the same scanline as the left image.

Scanline

Each row in an image is considered as a scanline.

Surface Albedo

Surface albedo , or reflection coefficient of a surface, is the ratio of reflected radiation from the surface to incident radiation upon it. It is measured on a scale from 0 to 1. Zero for black surfaces with no reflection, and one for white surfaces with perfect reflection.

Bibliography

[1] Comprehensive 3d animation software. 51

[2] Image rectification. ix, 10

[3] Triangulation. ix, 5

[4] C. Albitar, P. Graebling, and C. Doignon. Design of a monochromatic pattern for a robust structured light coding. In *Proceedings of the International Conference on Image Processing*, pages 529–532, 2007. 4

[5] D. G. Aliaga. and Y. Xu. A self-calibrating method for photogeometric acquisition of 3d objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):747–754, 2010. 4

[6] J. Batlle, E. Mouaddib, and J. Salvi. Recent progress in coded structured light as a technique to solve the correspondence problem: a survey. 31:963982, July 1989. 7

[7] D. Bergmann. New approach for automatic surface reconstruction with coded light. In *Proceedings of Remote Sensing and Reconstruction for Three-Dimensional Objects and Scenes*, pages 2–9, San Diego, CA, 1995. 14, 16

[8] J. Y. Bouguet. Camera calibration toolbox for matlab. 2004. 8, 10, 11, 12, 35

[9] K. Boyer and A. Kak. Color-encoded structured light for rapid active ranging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9:14–28, January 1989. 14, 24

[10] M. Brown, P. Song, and T.Cham. Image pre-conditioning for out-of-focus projector blur. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:1956 – 1963, 2006. 57

[11] D. Caspi, N. Kiryati, and J. Shamir. Range imaging with adaptive color structured light. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):470–480, 1998. 4, 14, 18, 19

[12] X. Chen and Y. Yang. Recovering dense stereo depth maps using a single gaussian blurred structured light pattern. *International Conference on Computer and Robot Vision (CRV)*, pages 295 – 302, 2013. xii, 24, 25, 56, 57

[13] T. Clarke and J.G.Fryer. The development of camera calibration methods and models. *The Photogrammetric Record*, 16:5166, April 1998. 8

[14] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63:542–567, 1996. 42

[15] J. Davis, D. Nehab, R. Ramamoorthi, and S. Rusinkiewicz. Spacetime stereo: A unifying framework for depth from triangulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):296–302, Feb. 2005. 4

[16] D. Desjardins and P. Payeur. Dense stereo range sensing with marching pseudo-random patterns. In *Proceedings of the Fourth Canadian Conference on Computer and Robot Vision*, pages 216–226, 2007. 4, 14, 31

[17] T. Etzion. Constructions for perfect maps and pseudorandom arrays. *IEEE Transactions on Information Theory*, 34:1308–1316, September 1988. 14, 28

[18] H. Fredricksen. The lexicographically least debruijn cycle. *Journal of Combinatorial Theory*, 9(1):509–510, 1970. 14, 23

[19] D. C. Ghiglia and M. D. Pritt. *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software*. John Wiley and Sons, Inc, 1998. 23

[20] L. Goddyn and P. Gvozdjak. Binary gray codes with long bit runs. *ELECTRONIC JOURNAL OF COMBINATORICS*, 10, 2003. 14, 16

[21] P. M. Griffin, L. S. Narasimhan, and S. R. Yee. Generation of uniquely encoded light patterns for range data acquisition. *Pattern Recognition*, 25(6):609–616, 1992. x, 14, 29, 30

[22] J. Guhring. Dense 3-d surface acquisition by structured light using off-the-shelf components. In *Proceedings of Videometrics and Optical Methods for 3D Shape Measuring*, pages 220–231, 2001. 14, 16

[23] X. Han. *3D Shape Measurement Based on the Phase Shifting and Stereovision methods*. The Graduate School, Stony Brook University: Stony Brook, NY, May 2010. ix, 14, 20, 22

[24] D. Hirshberg, M. Loper, E. Rachlin, and M. Black. Coregistration: Simultaneous alignment and modeling of articulated 3D shape. In *European Conference on Computer Vision*, 2012. 2

[25] D. Hirshbergc, M. Loperc, E. Rachlinc, A. Tsoliac, A. Weissa, , B. Cornerb, and M. Blacka. Evaluating the automated alignment of 3d human body scans. 2011. 2

[26] http://arc-team-open-research.blogspot.com.br/2012/12/how-to-make-3d-scan-with-pictures and.html. How to make 3D scan with pictures. 2

[27] http://en.wikipedia.org/wiki/3D_reconstruction. Passive methods. 3

[28] https://code.google.com/p/procamcalib/. Camera-projector calibration toolbox. 11

[29] https://www.sensefly.com/home.html. sensefly. 1

[30] http://vision.middlebury.edu/stereo/. Middlebury stereo vision page. 3

[31] http://www.laserfocusworld.com/articles/2011/01/lasers-bring-gesture-recognition-to-the home.html. Photonic technologies and solutions for technical professionals worldwide. ix, 6

[32] P. S. Huang and S. Zhang. Fast three-step phase-shifting algorithm. *Applied Optics*, 45(21):5086–5091, 2005. 14, 20

[33] P. S. Huang, S. Zhang, and F.-P. Chiang. Trapezoidal phase-shifting method for 3-d shape measurement. *Optical Engineering*, 44(12), 2005. 14, 20, 23

[34] H. Kawasaki, R. Furukawa, R. Sagawa, and Y. Yagi. Dynamic scene shape reconstruction using a single structured light pattern. In *IEEE conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. 7

[35] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The Digital Michelangelo Project: 3D scanning of large statues. In *Proceedings of ACM SIGGRAPH 2000*, pages 131–144, July 2000. 1

[36] M. Minou, T. Kanade, and T. Sakai. A method of time-coded parallel planes of light for depth measurement. *Trans. Institute of Electronics and Communication Engineers of Japan*, E64(8):521–528, August 1981. 14, 15

[37] R. A. Morano, C. Ozturk, R. Conn, S. Dubin, S. Zietz, and J. Nissanov. Structured light using pseudorandom codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):322–327, Mar. 1998. 14, 30

[38] J. Pages, J. Salvi, C. Collewet, and J. Forest. Optimised de bruijn patterns for one-shot shape acquisition. *Image and Vision Computing*, 23(8):707–720, Aug. 2005. 14, 24, 25

[39] J. L. Posdamer and M. D. Altschuler. Surface measurement by space-encoded projected beam system. *Computer Graphics and Image Processing*, 18(1):1–17, 1982. 14, 15

[40] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. Color transfer between images. *Computer Graphics and Applications*, 21:34–41, September 2001. 40

[41] M. Rodrigues. Fast 3D reconstruction using structured light methods. In *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2011. 2

[42] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3d model acquisition. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 438–446, 2002. 14, 16

[43] F. Ruskey. The combinatorial object server. 14, 24

[44] J. Salvi, X. Armangue, and J. Batlle. A comparative review of camera calibrating methods with accuracy evaluation. 35:16171635, July 2002. 8

[45] J. Salvi, J. Batlle, and E. Mouaddib. A robust-coded pattern projection for dynamic 3D scene measurement. 19:10551065, September 1989. 7

[46] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37:827–849, 2004. viii, ix, x, 7, 8, 9, 13, 14, 16, 17, 28, 29, 31

[47] H. Spoelder, F. Vos, E. Petriu, and F. Groen. Some aspects of pseudo random binary array-based surface characterization. *IEEE Transaction on Instrumentation and Measurement*, 49:1331–1336, December 2000. 29

[48] R. Valkenburg and A. McIvor. Accurate 3d measurement using a structured light system. *Image and Vision Computing*, 16:99–110, 1996. 14, 15

[49] X. Wen, H. Wang, and W. Zhai. Medical image based 3d reconstruction and preoperative surgery-planning for microwave ablation. *IEEE International Conference onBioinformatics and Biomedicine*, pages 38–42, December 2013. 2

[50] L. Zhang, B. Curless, and S. M. Seitz. Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *Proceedings of the 1st International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 24–36, Padova, Italy, 2002. x, 14, 24, 26, 27, 28

[51] S. Zhang and P. Huang. High-resolution, real-time 3d shape acquisition. In *Proceedings of on Computer Vision and Pattern Recognition Workshop*, pages 28–, 2004. 14, 20

[52] S. Zhang, D. Welde, and J. Oliver. Superfast phase-shifting method for 3D shape measurement. In *Opt. Express*, volume 18. OSA, Apr 2010. 14, 20, 21, 23

[53] S. Zhang and S. T. Yau. High-resolution, real-time absolute 3-d coordinate measurement based on the phase shifting method. *Optics Express*, 14(7):2664–2649, 2007. 14, 20

[54] S. Zhang and S. T. Yau. High-speed three-dimensional shape measurement system using a modified two-plus-one phase-shifting algorithm. *Optical Engineering*, 46(11), 2007. 14, 20, 23

[55] Z. Zhang. A flexible new technique for camera calibration. 22:1330–1334, November 2000. 8, 10, 12