uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Danick Desjardins
AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)
GRADE / DEGREE

School of Information Technology and Engineering
FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Structured Lighting Stereoscopy with Marching Pseudo-Random Patterns

TITRE DE LA THÈSE / TITLE OF THESIS

P. Payeur
DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

E. Petriu

Y. Ono

Gary W. Slater
Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Structured Lighting Stereoscopy with Marching Pseudo-Random Patterns

By:

Danick Desjardins

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements for the degree of

Master in Applied Science

Ottawa-Carleton Institute of Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-48598-9
Our file Notre référence
ISBN: 978-0-494-48598-9

Abstract

This thesis discusses the development of a 3D reconstruction system based on active stereoscopy with structured lighting. The proposed system accomplishes 3D reconstruction of objects using a combination of spatial-neighboring and time-multiplexing structured light patterns encoded with pseudo-random codes. This combination builds upon the concept of marching patterns that can adaptively increase the reconstruction resolution on demand. Original techniques are introduced to recover and validate pseudo-random codes from images using a combination of image processing algorithms. The validation of codes is performed such that potential outliers are minimized and the density of 3D point clouds remains fairly high in the models. Finally, the system simultaneously provides models with color and texture mapping using linear interpolation between vertices which enables a more accurate and intuitive representation of 3D objects. Experimental results presented demonstrate the various reconstruction densities possible with the proposed approach, its performance on objects with different surface properties, and the quality of the colored 3D models achieved.

Acknowledgments

I would like to thank my thesis supervisor Dr. Pierre Payeur for his direction, guidance, assistance and patience throughout my studies. Finally, I would also like to give thanks to my wonderful parents, family and friends for their support, patience and continued encouragement.

Table of Contents

ABSTRACT ii

ACKNOWLEDGMENTS iii

TABLE OF CONTENTS iv

TABLE OF FIGURES vi

CHAPTER 1 INTRODUCTION 1

1.1 CONTEXT 1
1.2 MOTIVATIONS 2
1.3 CHALLENGES 2
1.4 OBJECTIVES 3
1.5 PROPOSED SYSTEM 4
1.6 ORGANIZATION 5

CHAPTER 2 LITERATURE REVIEW 6

2.1 3D ACQUISITION PRINCIPLES 6
2.1.1 Passive Methods *6*
2.1.1.1 Stereoscopy 7
2.1.1.2 Other Passive Approaches 7
2.1.2 Active Methods *8*
2.1.2.1 Time-Of-Flight (TOF) 8
2.1.2.2 Active Triangulation 9
2.1.2.3 Structured Light 10
2.2 STRUCTURED LIGHTING PATTERNS 11
2.2.1 Time-multiplexing *12*
2.2.1.1 Binary Codes: Gray Code 13
2.2.1.2 N-ary Codes 14
2.2.2 Spatial Neighboring *17*
2.2.2.1 Pseudo-Random Codes 17
2.2.2.2 De Bruijn Sequences and Perfect Maps 20
2.2.2.3 Pseudo-random Binary Arrays 21
2.2.3 Direct Codification *25*
2.2.4 Comparison of Approaches *26*
2.3 IMAGE PROCESSING TECHNIQUES 27
2.4 3D RECONSTRUCTION 29
2.4.1 Epipolar Geometry *30*
2.4.2 Generalized Triangulation *34*
2.4.2.1 Middle Point Reconstruction 34
2.4.2.2 Optimal Polynomial Reconstruction 36

CHAPTER 3 SYSTEM DESCRIPTION AND IMPLEMENTATION 43

3.1 INTEGRATED 3D ACQUISITION AND RECONSTRUCTION SYSTEM DESIGN 43
3.2 CALIBRATION 46
3.2.1 Intrinsic Calibration Parameters *47*
3.2.2 Extrinsic Calibration Parameters *48*
3.3 PSEUDO-RANDOM CODE CONSTRUCTION 53
3.3.1 Generation of a Pseudo-random Array *53*
3.3.2 Pseudo-random Color Pattern Construction *57*
3.3.3 Color Palette Size *60*
3.4 PROJECTION AND ACQUISITION 60
3.5 IMAGE PROCESSING 65
3.5.1 Color Regions Segmentation *67*
3.5.2 Group Labeling *69*

3.5.3 Statistical Group Analysis *72*
3.5.4 Region Intensity Thresholding and Region Group Relabeling *74*
3.5.5 Post Processing *75*
3.6 CODE RECOVERY AND VALIDATION 76
3.6.1 Color Regions Ordering *77*
3.6.2 Code Validation and Confidence Map *78*
3.6.3 Point Extraction *81*
3.7 OUTLIERS REMOVAL 82
3.8 RECONSTRUCTION 83
3.9 MODEL COLORING 83
3.10 DATA FORMATTING AND VISUALIZATION 84

CHAPTER 4 EXPERIMENTAL VALIDATION 86

4.1 DESCRIPTION OF THE STEREO STRUCTURED LIGHTING SYSTEM 86
4.2 CALIBRATION ACCURACY EVALUATION 91
4.2.1 Best Fitting Plane *91*
4.2.2 Backprojection Error *93*
4.3 EXPERIMENTAL MODELING RESULTS 94
4.3.1 Simple Chair Object *94*
4.3.2 White Coffee Machine Object *100*
4.3.3 Highly Reflective Porcelain Vase Object *104*
4.3.4 Textured Box Object *108*
4.3.5 Bag Object *111*
4.3.6 Red Book Object *115*
4.3.7 Textured Chair Back Rest *120*
4.4 NATURAL LIGHTING EVALUATION 126
4.5 PERFORMANCE ANALYSIS 128

CHAPTER 5 CONCLUSION 130

5.1 SUMMARY 130
5.2 CONTRIBUTIONS 131
5.3 FUTURE WORK 132

REFERENCES 134

Table of Figures

Figure 2.1 – Active triangulation system principle. 9
Figure 2.2 - Generation of gray codes. 13
Figure 2.3 - Labeling of a location using gray codes. 14
Figure 2.4 - Circuit to determine a De Bruijn sequence. 20
Figure 2.5 - Building a PRBA. 23
Figure 2.6 – Constructed PRBA. 23
Figure 2.7 - PRBA pattern. 23
Figure 2.8 - Color slid grid using PRMVS. 24
Figure 2.9 - Grayscale pattern proposed by Carrihill and Hummel. 25
Figure 2.10 - Epipolar geometry. 30
Figure 2.11 - Middle point reconstruction. 35
Figure 2.12 - Epipolar minimization criteria. 37
Figure 2.13 – Parameterization of points. 39
Figure 3.1 - Overview of System. 45
Figure 3.2 - Analyzed checkerboard pattern. 48
Figure 3.3 – World point to camera reference frame transformation. 49
Figure 3.4 – Distance between calibration pattern squares. 50
Figure 3.5 – Corner extraction: a) in proper order, b) in reverse order. 51
Figure 3.6 – Series of calibration images. 52
Figure 3.7 – Pseudo-random code example. 54
Figure 3.8 – Color substitution. 55
Figure 3.9 – Pseudo-random code creation. 56
Figure 3.10 - Pseudo-random code with equivalent color pattern with dimensions. 59
Figure 3.11 – Projected pattern on various objects. 61
Figure 3.12 – Short exposure time. 62
Figure 3.13 – Long exposure time. 62
Figure 3.14 – a) Intensity image without the projected pattern, b) difference image from the scene. 63
Figure 3.15 - Marching pattern. 64
Figure 3.16 – Maximum shift. 65
Figure 3.17 - Image processing algorithm. 66
Figure 3.18 - Hue histogram for color segmentation. 67
Figure 3.19 – Color segmentation example. 68
Figure 3.20 – Structured light on an object with a complex surface and reflectance properties. 69
Figure 3.21 – Hue histogram of Figure 3.20 exhibiting 4 peaks for only 3 dominant projected colors. 69
Figure 3.22 - Labeling of color regions. 70
Figure 3.23 - Non-uniformed color distribution over a green region. 71
Figure 3.24 – Result from group labeling algorithm. 71
Figure 3.25 – Resolving the occurrence of holes in regions that get segmented in two different color masks. 73
Figure 3.26 - Overlapping color regions in the red mask image. 74

Figure 3.27 – Distribution of intensity in a single color segmented region. 75
Figure 3.28 – Result of the intensity thresholding and relabeling operations on merged color regions. 75
Figure 3.29 - Touching color regions. 76
Figure 3.30 - Sorting of gathered color regions. 78
Figure 3.31 – Extraction of 8 neighboring codes for validation with a confidence map. 79
Figure 3.32 – Code error correction with Hamming distance. 80
Figure 4.1 – Stereo structured lighting acquisition system. 87
Figure 4.2 – Projected pattern example. 88
Figure 4.3 – Resolution diagram. 90
Figure 4.4 - Simple chair. 95
Figure 4.5 – Simple chair with projected pattern. 96
Figure 4.6 – 3D reconstruction of a simple chair model with a single pseudo-random pattern projection. 97
Figure 4.7 - Simple chair reconstruction with 324 marching pseudo-random patterns. 98
Figure 4.8 – Simple chair colored reconstruction. 99
Figure 4.9 – Simple chair dimensions for evaluation of the scale factor in the model. 100
Figure 4.10 - Coffee machine with glass container covered with white paper. 101
Figure 4.11 – Coffee machine with projected pattern. 101
Figure 4.12 - Coffee machine reconstruction with 324 marching pseudo-random patterns. 102
Figure 4.13 – Coffee machine colored reconstruction. 103
Figure 4.14 - Highly reflective vase. 105
Figure 4.15 – Highly reflective vase with projected pattern. 105
Figure 4.16 - Highly reflective vase reconstruction using 324 marching pseudo-random patterns. 106
Figure 4.17 – Porcelain vase colored reconstruction. 107
Figure 4.18 - Textured box. 108
Figure 4.19 – Textured box with projected pattern. 109
Figure 4.20 - Textured box reconstruction using 324 marching pseudo-random patterns. 110
Figure 4.21 – Textured box colored reconstruction. 111
Figure 4.22 – Original brown bag. 112
Figure 4.23 – Brown bad with projected pattern. 113
Figure 4.24 – Brown bag point reconstruction. 114
Figure 4.25 – Brown bag colored reconstruction. 115
Figure 4.26 – Red book. 116
Figure 4.27 – Red book with projected pattern. 116
Figure 4.28 – Red book low resolution reconstruction. 117
Figure 4.29 – Red book full resolution reconstruction. 118
Figure 4.30 – Red book low resolution colored reconstruction. 119
Figure 4.31 – Red book high resolution colored reconstruction. 119
Figure 4.32 – Chair back rest. 121
Figure 4.33 – Chair back rest with the projected pattern. 121
Figure 4.34 – 3D reconstruction of the chair back rest with 25 marching pseudo-random patterns. 122
Figure 4.35 – Front view of the 3D reconstruction of the chair back rest with 324 marching pseudo-random patterns. 123

Figure 4.36 – Close up view of the high resolution reconstruction of the chair back rest with 324 marching pseudo-random patterns. 124
Figure 4.37 – Side view of the high resolution colored reconstruction of the chair back rest..... 125
Figure 4.38 – Front view of the high resolution colored reconstruction of the chair back rest..... 126
Figure 4.39 – Simple chair exposed to ambient lighting. 127
Figure 4.40 – 3D reconstruction of the simple chair in bright ambient light..... 127

Chapter 1 Introduction

This chapter gives an overview of the work and research performed within this thesis on the creation and use of active patterns for the 3D reconstruction of objects. The initial motivations and challenges faced in this thesis are mentioned to help in describing the process of 3D reconstruction in its entirety. This thesis aims at bringing new ideas to solve some of the common problems encountered in the subject matter.

1.1 Context

When some analysis needs to be performed on real 3D objects and their environments, a 3D reading needs to be acquired. 3D model reconstruction is a way to achieve this. 3D model data can be used for multiple things from simulations, virtual environments and accurate mapping of environments in robot navigation. There arises a need for innovative 3D model reconstruction systems since existing systems can be relatively slow, expensive and cannot easily adapt to multiple environments.

A complete system that can perform 3D model reconstruction of objects requires multiple modules including gathering data, calibration, reconstruction, data verification and data correction. Each module may have multiple possible implementations. These implementations will vary in effectiveness, performance, resolution and accuracy. An important common trade-off that must be considered for each module is the one between the quality of results and the time required to achieve them. Furthermore, some techniques are limited by their surrounding environment and the location of the objects of interest with respect to the reconstruction system. The final solution tries to be as general as possible with few limitations.

1.2 Motivations

Many techniques can be implemented to estimate the shape and position of objects in a scene. This thesis aims at researching and developing strategies that will be best suited for 3D modeling over different objects or scenes. The solution needs to be able to gather data with varying levels of density while only sacrificing processing time when it is deemed necessary or when needed. Many techniques will arguably produce superior results but will either require a higher amount of processing or more expensive setups and equipment to achieve them. The resulting 3D sensing technique should give results that are accurate and can be interpreted readily, and are achieved in an acceptable amount of time while operating from affordable off-the-shelf equipment, unlike ultra specialized 3D sensors.

The end goal of this thesis is to build a complete 3D reconstruction system that can be used for 3D modeling. The end system should be a first prototype of a completely integrated solution, from the pre-acquisition steps, such as calculating structured lighting patterns, to the visualization of the colored reconstructed models. The system should also be able to scan objects using different resolutions. Finally, it should not be greatly affected by varying levels of ambient light.

1.3 Challenges

As mentioned before, the main constraint and challenge of this thesis is to have a system that performs adequately given some timing constraints and from affordable equipment. This consideration constrains the design of all modules.

In the context of an open environment, the latter will greatly affect the results that can be obtained no matter which 3D reconstruction technique is used. Some realistic goals need to be set so they can be achieved. These goals include setting the maximum and minimum distances over which the system can model objects, the range of different reflective properties of objects that can be dealt with, the amount of textures on objects

required, the illumination level of environments and the depth range of 3D reconstructions.

Stereoscopic techniques have been well explored for 3D reconstruction. There are two types of stereoscopic techniques for the acquisition of points: active stereovision and passive stereovision. Active stereovision projects light onto objects and the acquisition is performed by capturing the effect of the projected light. The environment affects this technique, for example if the ambient light is of a higher intensity than the projected light. In this case, the projected light will not be able to be properly captured. Passive stereovision techniques require unique and distinguishable features on objects to perform well. Both techniques can be, in some cases, severely affected by the nature of their environment. These issues are investigated here and lead to the development of image processing strategies to reinforce the 3D vision system's robustness.

In the context of environment exploration, since no foreknowledge is available on the current surrounding environments, the system must be designed in such a way to be able to perform over a range of different scenes. In spite of this, there will always be some circumstances where no data can be extracted from a scene. The goal is to minimize these circumstances.

1.4 Objectives

This work addresses several issues encountered in the imaging and 3D reconstruction of objects in various environments. The proposed solution aims at:

1. Overcoming difficulties in the reconstruction of featureless objects with various reflective properties and of different colors.
2. Providing an alternative low cost 3D sensing approach that also offers high flexibility in the density of acquired points and performance desired.
3. Offering the ability to augment the resolution of scanned objects using an adaptive process.

4. Minimizing the effect of the environment's lighting on the system designing the system in such a way that it will reconstruct scenes that can vary over a certain range of illumination.
5. Properly identifying and removing outliers that typically appear in the 3D reconstructions from stereoscopy due to the difficulty in properly identifying matching points.
6. Integrating all suitable modules and algorithms to achieve a complete reconstruction system by selecting and expanding the best strategies for the final proposed system.
7. Enhancing 3D shape reconstruction with texture and color information to facilitate the visual interpretation of the models.

1.5 Proposed System

The research performed had as a goal to design a vision system suitable to acquire 3D data of objects in its immediate surroundings for exploration tasks. The implemented system is build around active stereoscopy. By definition, stereoscopy involves two capturing devices, matching of features and 3D reconstruction from the matching features. In active stereoscopy the features are created by projecting patterns or features onto the desired objects. The patterns have unique encodings to be able to distinguish locations accurately from the two capturing devices. Such a process is called structured lighting. Multiple types of patterns can be projected onto a scene. A pattern based on pseudo-random codes is used in this system due to its high flexibility and simplicity.

The system is composed of several modules. The first module calculates the calibration data of the acquisition devices. This is done using Zhang's calibration algorithm [1] and is only performed once or when required. The following module constructs pseudo-random patterns given certain parameters such as a Hamming distance and color restrictions. The next module projects the pattern, captures and processes the images of the scene. Color images of the scene are also taken at this step. The processing involves isolating the projected patterns and assembling codes contained in the projected pattern.

Once the codes are assembled, matches can be made on the two captured images. This list of matches is passed to a reconstruction module. The reconstruction module filters matches based on epipolar restrictions and surrounding codes to reduce the amount of outliers. Finally, it takes the valid matches and reconstructs 3D points. The 3D data is passed to a data formatting module which assembles the data into a model that can be rendered and interpreted. This module also extracts color information taken from images in a previous module and adds it to the 3D points. These color 3D models help users to see the various objects present in the scene more accurately.

1.6 Organization

This thesis is organized in five chapters. Chapter Two reviews the various techniques available for 3D reconstruction. It also evaluates several types of projected patterns that can be used in 3D active stereovision. The final section of the chapter gives a brief overview on triangulation algorithms. Chapter Three discusses in details the design and modules of the proposed system. It also discusses the rationale on the choice of algorithms. Chapter Four provides a list of results and comparisons achieved by the proposed system. Chapter Five brings forth conclusions and a list of possible improvements for future work.

Chapter 2 Literature Review

The acquisition of range measurements for the reconstruction of 3D models of real objects involves several aspects. These have been researched thoroughly in the literature, but current 3D acquisition and modeling systems still exhibit several limitations related to computational load, accuracy, or nature of the objects to be modeled and the environments in which they operate. This chapter proposes an overview of the available algorithms and techniques under the following four categories: 3D acquisition principles, structured lighting, image processing and 3D reconstruction.

2.1 3D Acquisition Principles

Measurement of 3D space is in general a complicated process. A simple yet slow process of acquiring data is to use coordinate measurement machines (CMM). A probe is moved along surfaces and the current coordinates of the CMM is measured at every point [2][3] or line by line [4][5]. The resolution of CMMs is governed by the mechanical aspects of the system, mostly the size of the increments and the error tolerance of the CMM. The range and the speed of the system are also limited and are not suited for fragile surfaces.

Optical measurements can also be used to gather 3D data. These can be classified in two groups. Passive methods that rely on existing ambient lighting present on the scene and active methods that use controlled external sources of light.

2.1.1 Passive Methods

These methods rely solely on features already present on the scene to compute 3D data of objects. No controlled external sources of light are used. This creates a limitation since the accuracy of the reconstruction is dependent on the scene. The scene must contain easily differentiable features with hard edges, corners or labeled regions. However, the setups needed for these methods are usually quite simple, easier and less expensive to build.

2.1.1.1 Stereoscopy

Stereoscopy is the most widely explored passive 3D acquisition principle in the literature. It is based on the principle of human vision [6]. 3D information can be reconstructed if the same objects or points are seen from two or more point of views. In computer vision, charged couple devices (CCD) cameras are used to capture scenes from different point of views. The images then need to be processed to extract and match the same objects in different views. Multiple difficulties arise in matching parts of images, as some feature points are only present in one image. In the second image, the features are either missing or are occluded due to the angle of view of the camera. There are also some difficulties in properly matching feature points in two different images due to similarities between different features and the characteristics in a typical scene. This makes it difficult to find complete and accurate matches between two images since it is not trivial to know what regions are present in both images. It is also difficult to match areas that are not rich enough in features and hence created multiple features of similar attributes.

2.1.1.2 Other Passive Approaches

Several other methods have been investigated to achieve passive 3D vision but did not reach the same level of popularity, mainly because of their inherent complexity or the specificity of the context in which they can be applied. These include photogrammetry, which is a more primitive form of stereoscopy. It relies on two or more photogrammes or photographs of a common scene between which features are matched to achieve reconstruction [7]. In its original form, it remains very sensitive to mismatched features and calibration inaccuracies.

Shape from shading is another passive technique that earned some credibility. It builds on the interpretation of the surface reflectance of objects to estimate the orientation of those surfaces and achieve 3D reconstruction estimates [8]. However, it usually assumes constant illumination from a point-light source, which is not adequate in a context of machine vision for exploratory tasks. Finally, structure from motion also falls in the category of passive 3D imaging techniques. It interprets the optical flow of perspective

projections of an object that moves in the scene while being observed from a static viewpoint, or vice-versa. It is based on the assumption of object rigidity [9]. While usually appropriate for imaging 3D objects, the need for motion imposed constraints on acquisition devices and setups.

2.1.2 Active Methods

Unlike passive methods, active methods rely on a controlled external light source to acquire 3D data. Active methods perform exceptionally well in areas where no features can be detected compared to its passive counterparts. Compared to passive methods that require sophisticated image processing, active methods involve processing that is simplified to detecting the projected light. Furthermore, since the processing is simpler, the total reconstruction time is shorter and thus makes measurements quicker to perform. However, active methods are highly dependant on the characteristics of the objects in the scene. They perform poorly on objects that are highly reflective, transparent and have non-uniform color characteristics. They are also limited by the intensity of the ambient light and the power of the projective source. The projective light needs to be seen and if the ambient light is too strong or the projective source too weak then the sensors cannot gather any information from the projected pattern. The most popular 3D active sensing techniques are time-of-flight, active triangulation and structured lighting.

2.1.2.1 Time-Of-Flight (TOF)

Time-of-flight is exploited in light detection and ranging (LIDAR) systems which mimic the well-known sonar devices [10]. A light beam pulse is emitted onto an object and a sensor measures the reflected portion of the pulse. The time difference is calculated between the sent pulse and the time the reflected pulse is received at the sensor. This method is more suitable for objects that lie at further distances since the speed of light is high and the TOF is often quite small. Since the TOF is small, to achieve accurate results the sensor needs to be sampled at high frequencies and the hardware to do so is

expensive. This approach is also sensitive to multiple reflections or diffusion on different surfaces. Sophisticated filtering is therefore required.

2.1.2.2 Active Triangulation

Active triangulation is also based on the projection of a focused light beam upon a surface [11]. The beam projected from the source is reflected by the object onto the image plane. The angle of the projected light beam with the location of the reflected beam onto the image plane can be used to calculate the 3D information of the point that the beam is reflected on. This method is highly dependant on the objects reflectivity properties since a minimum portion of the narrow light beam needs to be reflected to the sensor.

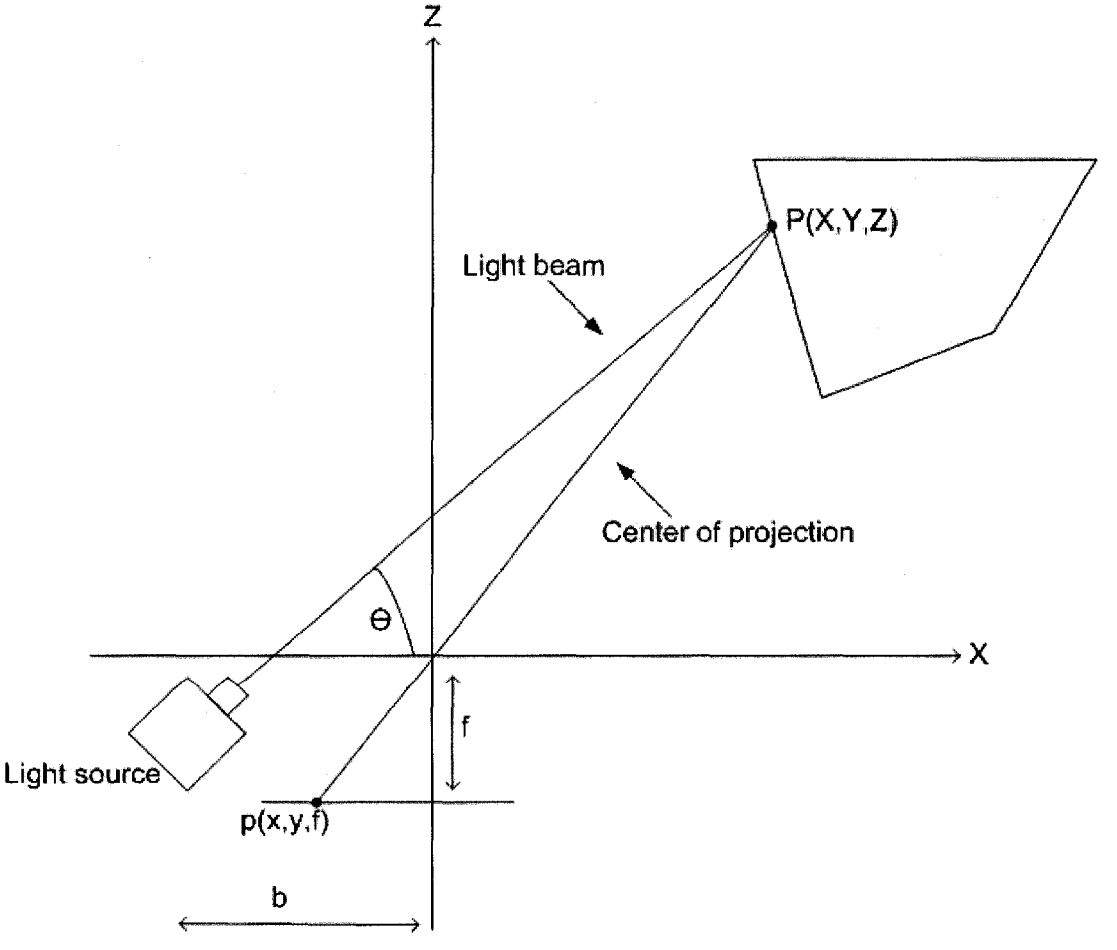


Figure 2.1 – Active triangulation system principle.

As shown in Figure 2.1, a light beam is projected from a light source and hits point P on an object. The light beam is then reflected onto a CCD sensor onto point p. The angle of projection θ, the point p, the distance b (the baseline) between the light source and the sensor and the focal length f are known. With this information it is possible to determine the coordinates of point P as follows:

$$Z = \frac{b}{f\cot(\theta) - x} f \tag{2.1}$$

$$X = \frac{b}{f\cot(\theta) - x} x \tag{2.2}$$

$$Y = \frac{b}{f\cot(\theta) - x} y \tag{2.3}$$

2.1.2.3 Structured Light

One of the simplest and robust methods used for 3D reconstruction is the structured light method. This method projects a known light pattern on a scene. These patterns can be light strips, grids, matrices or even a sequence of different patterns. Since the pattern is known, the segmentation of the captured images can be optimized to look for specific elements which make the process quicker. Furthermore, patterns can be built in a way so that every location is labeled uniquely. As with the other active methods, structured light is limited by the characteristics of the surface being reconstructed. Objects that have specular and highly reflective properties are hard to capture accurately on cameras. The ambient light and the light sources power also have a critical role in defining if the projected patterns can be acquired properly. On the other hand, a setup constructed for structured light has the advantage of being quite affordable since it is only composed of a projector and a CCD camera.

Given that this research work focuses on structured lighting approaches, this technology is explored further in the following sections where several types of patterns and structured light techniques are examined.

2.2 Structured Lighting Patterns

Structured lighting patterns create artificial features that are used to accurately identify locations on image planes that correspond to specific features points on an object. With locations accurately defined, it enables high accuracy when reconstructing the 3D information of points in space. A structured lighting setup requires a device that projects light, such as a projector, onto a scene and at least one acquisition device. With proper calibration between the projection and acquisition devices, full reconstruction is possible. Setups can vary in the number of acquisition devices. For example, an active stereoscopic system can contain a projector and two acquisition devices.

There are a variety of patterns that have been explored in the field of 3D imaging. These patterns can be classified into three main groups, as defined by Salvi *et al.* [12], which are time-multiplexing, spatial neighboring and direct coding.

Time-multiplexing patterns are temporal codes. A set of patterns is projected onto the desired scene in a timely fashion. The codeword for a certain position is composed of the illumination values of a particular pixel position throughout all the patterns and is unique for every location. These patterns, however, are not suited for moving environments since, as the name implies, the patterns are projected over time to create the unique codewords. In a moving scene, the location of a given object will change between successive acquisitions of the scene and locations will be wrongly labeled. Some of the most popular techniques under this category are gray codes and continuous patterns. They are examined in section 2.2.1.

Spatial neighboring encodes the location of positions by creating a code using neighboring information. Unlike time-multiplexing patterns, spatial neighboring patterns

perform well when motion is involved since only one pattern is needed to fully label positions. However, since only one pattern is used, precision is usually not as high as with time-multiplexing patterns. The precision is reduced since extra pixels are required to easily code and decode locations. There are different ways of creating these patterns. The most popular are patterns created from De Bruijn sequences or pseudo-random sequences. These have interesting properties that makes it possible to uniquely identify regions. A more flexible approach is to use pseudo-random techniques to build multi-valued matrices. This type of patterns are discusses in section 2.2.2.

Finally, a pattern could potentially uniquely label every visible pixel. This is known as direct codification. In order to achieve this, a large range of color values or periodicity will need to be exploited. A location is coded as a precise intensity level. These patterns, theoretically, could yield the highest precision possible for a given light source resolution. Practically this is not the case and these codes usually perform poorly. Since the codewords are now composed of single pixels it is difficult to label locations precisely. Any change in lighting across the scene or the texture and colors of objects being scanned can affect the intensity value being acquired. The location cannot be perfectly labeled and matches between two images become difficult to achieve. Patterns used in direct codification are covered in section 2.2.3.

2.2.1 Time-multiplexing

This type of patterns takes advantage of a system being able to project and acquire multiple images over time. The patterns are temporal codes since the images acquired are multiplexed over time to create the final codes. Since multiple images can be taken, the images projected can be simple and the primitives that need to be extracted from the scene are easy to detect. There are several types of time-multiplexing codes but only the most popular ones will be discussed, that is: patterns based on binary codes and patterns based on n-ary codes. These are considered since they summarize the main ideas behind this type of patterns.

2.2.1.1 Binary Codes: Gray Code

Binary codes only use two states of illumination, which are illuminated or non-illuminated, that is black and white stripes. Gray codes are a type of binary codes and will be used to discuss this class of codes.

In a gray code, two consecutive numbers only differ by one bit. This helps to minimize the errors in recuperating locations [13][14]. Figure 2.2 shows the derivation of a gray code for 16 possible positions (2^4) where C_x represents the encoding of a position and R_x represents successive rows that need to be used to encode positions. By correctly deriving the gray code for a given resolution, each row serves as a pattern to be projected. This results in bands of white and black colors for binary 0's and 1's. Images are saved after the successive projection of every row and the color value of each pixel is determined throughout all acquired images. Figure 2.3 shows an example where point *i,j* is illuminated with simple row patterns. By putting all the values of each projection together, the gray code *0111* is obtained and defines the point's position. Gray code projections are repeated vertically to identify the vertical position as well. The Gray code will give the position of a certain point from the projector's side, since the pattern is known, and will match this point accurately on the image side. To uniquely identify all pixels in a *NxM* image, $log_2(N)$ patterns are needed vertically and $log_2(M)$ are needed horizontally.

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}
R_0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
R_1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
R_2	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
R_3	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0

Figure 2.2 - Generation of gray codes.

Figure 2.3 - Labeling of a location using gray codes.

2.2.1.2 N-ary Codes

N-ary codes have the advantage of occupying the whole range of intensities available from the projector as opposed to only two states as in binary codes [15]. Since binary codes only utilize two states to fully define all positions in a scene, more images need to be taken given that each image contains less information than with n-ary codes. As a result, n-ary codes can theoretically label positions with fewer images or with a higher precision than their binary code counterpart.

2.2.1.2.1 Ramp Function

A ramp function can be used to create n-ary codes. The codes are calculated by projecting a pure white pattern, measuring the intensity values of the pixels, then projecting a gray level ramp function and again measuring the intensities. The position on the ramp is determined by the ratio of the two measurements. This technique, however, is highly dependent on low levels of noise and non-linearities. Furthermore, it has limited effective spatial resolution [13][15][16].

2.2.1.2.2 Frequency Varying Function

Rapidly varying functions can be used to augment the effective spatial resolution of patterns. These, however, introduce phase ambiguities meaning multiple positions will have the same associated phase. This can be resolved by projecting a series of periodic patterns to create an over determined system that can be solved with linear least squares.

Scharstein et al. propose continuous sin wave patterns [13]. The following function describes what is observed at the sensor:

$$I_{kl}(x,y) = A(x,y) + B_{kl}[\sin(2\pi f_k u + \phi_1) + 1)] \tag{2.4}$$

where $A(x,y)$ is the color of a given point without any projected pattern, B_{kl} the intensity of the (k,l)th projection, f_k and ϕ_1 its frequency and phase. $I_{kl}(x,y)$ is the observed intensity.

Instead of projecting multiple gray scale images of different sinusoidal phases, color images can also be used to create patterns. Color patterns take advantage of the added information that can be captured by color sensors and thus reduce the amount of images needed to solve the phase ambiguity. Zhang and Huang utilize the three-step algorithm [17]. The intensity observed at the sensor is modeled as follows for the three color channels with a phase shift of 120°:

$$\begin{aligned} I_r(x,y) &= I'(x,y) + I''(x,y)\cos[2\pi f + (\phi(x,y) - 2\pi/3)], \\ I_g(x,y) &= I'(x,y) + I''(x,y)\cos[2\pi f + \phi(x,y)], \\ I_b(x,y) &= I'(x,y) + I''(x,y)\cos[2\pi f + (\phi(x,y) + 2\pi/3)] \end{aligned} \tag{2.5}$$

where $I'(x,y)$ is the average intensity observed, $I''(x,y)$ the intensity of the projected light, f the frequency and $\phi(x,y)$ the phase to be determined.

Solving these three equations gives:

$$\phi(x,y) = \arctan[\sqrt{3}(I_r - I_b)/(2I_g - I_r - I_b)] \tag{2.6}$$

This equation however creates discontinuities in the 0 to 2π range. Unwrapping algorithms must be used to create a continuous 3D map phase. The drawback of using a single RGB image is that it is hard to perfectly segment the color components. In most sensors the response of RGB overlaps between each other. Additionally, the object being scanned needs to have a uniform color for it to work accurately. The same method can be used by projecting three separate grayscale images to produce the same effect.

Zhang and Huang also propose a method of trapezoidal phase-shifting [17]. They propose this method to be able to do reconstruction in real time which alleviates the calculation time of arc tangents seen in the three-step method. In this method, three phase shifted trapezoidal patterns are projected rapidly and sequentially on the scene. The patterns are broken down into N regions that can be uniquely identified. The difference with the sinusoidal method is that the intensity ratio is calculated and not the phase. This requires less processing to compute. The ratio is simply calculated as follows:

$$r(x,y) = \frac{I_{med}(x,y) - I_{\min}(x,y)}{I_{\max}(x,y) - I_{\min}(x.y)} \tag{2.7}$$

where $r(x,y)$ is the intensity ratio and $I_{\min}$, I_{med} and $I_{\max}$ are the minimum, median and maximum intensity values at point (x,y). $r(x,y)$ represents a triangular region which has a range from 0 to 1. It can be converted to a ramp function by the following formula:

$$r(x,y) = 2 \times round(\frac{N-1}{2}) + (-1)^{N+1} \frac{I_{med}(x,y) - I_{\min}(x,y)}{I_{\max}(x,y) - I_{\min}(x.y)} \tag{2.8}$$

where N is the region number, round is the rounding function to the nearest integer and the value of $r(x,y)$ ranges from 0 to 6. This value now specifically encodes regions. To increase the 3D resolution the pattern can be repeated. However, repetitions add ambiguity and must be dealt with.

2.2.1.2.3 Fringe Patterns

Fringe patterns are based on frequency time varying functions. Fringe patterns are usually used in conjunction with fringe pattern profilometry (FPP). Ronchi grating or sinusoidal grating is used to project fringe patterns onto a scene [18]. The result is a deformed fringe pattern that is compared to an original fringe pattern projected upon a base reference plane. Analysis is performed on the fringe patterns to find and label the deformations that are used directly to profile and extract 3D information from a scene. Fringe patterns

involve a complete process for extracting 3D information and does not necessarily define new patterns.

2.2.2 Spatial Neighboring

Unlike time-multiplexing codes, spatial neighboring is composed of the complete code in one projection. The positions are coded with the surrounding pixel information. The problem with this class of codes is that more information needs to be extracted accurately from the image to properly label positions. When only part of the code is extracted from the image, the position cannot be known with certainty. The main causes of incomplete codes are occlusions and projection effects associated with changes of the surface orientation that affect part of the code. Furthermore, additional space is usually needed to encode every position so that they can be easily extracted from the scene. This reduces the maximum 3D resolution that can be acquired. However, since only one pattern needs to be projected they are faster than time-multiplexing codes and can be used in dynamic environments.

2.2.2.1 Pseudo-Random Codes

Pseudo-random codes are composed of a series of pseudo-random arrays projected on an object. Each array, in the projected pattern, is composed of different intensities or colors and the arrays themselves only appear once in the pattern. The idea is that every point of interest is uniquely defined by its surrounding points. To make the correspondence problem more accurate, Hamming distances are computed on pairs of codes. Hamming distances assure that the difference between codes is at least a certain number which is important to avoid ambiguities in code recognition. Moreover, when Hamming distances are high between codes, mislabeled codes can be identified and corrected. For example, in a system where a given Hamming distance is present and a particular code is segmented but is not found in the list of existing codes, this erroneous code can be compared to the list of existing codes. The code that has the smallest Hamming distance with the mislabeled code has a high probability of being the correct code and can be substituted to replace the mislabeled code.

The code proposed by Morano *et al.* is a square pseudo-random matrix M, of size L ($k = l = L$) [19]. Each element in the array is assigned a color from a defined color palette A. The position of each element m_{ij}, except border elements, is determined by an *mxm* grid encompassing its surrounding neighbors. The value of m used in this case is three and can be expressed as a nine-element vector $V_{(ij)}$ where,

$$V_{(ij)} = \{m_{i-1,j-1}, m_{i-1,j}, m_{i-1,j+1}, m_{i,j-1}, m_{i,j}, m_{i,j+1}, m_{i+1,j-1}, m_{i+1,j}, m_{i+1,j+1}\} \tag{2.9}$$

$$V_{(ij)} = \{V_{(ij)1}, V_{(ij)2}, V_{(ij)3}, V_{(ij)4}, V_{(ij)5}, V_{(ij)6}, V_{(ij)7}, V_{(ij)8}, V_{(ij)9}\} \tag{2.10}$$

There must not be two $V_{(ij)}$ that are the same throughout the array. In fact every $V_{(ij)}$ must differ by a defined minimum Hamming distance h. The Hamming distance between $V_{(ij)}$ and $V_{(i'j')}$ is defined as follows:

$$H(ij, i'j') = \sum_{r=1}^{m^2} (\delta_r) \qquad \delta_r = \begin{cases} 0 \rightarrow v_{(ij)r} = v_{(i'j')r} \\ 1 \rightarrow v_{(ij)r} \neq v_{(i'j')r} \end{cases} \tag{2.11}$$

Thus the creation of a pattern depends on L, A, h and m. Unfortunately, it is impossible to always compute M based on those parameters since the size of the matrix chosen can be too large to uniquely label every position using a limited alphabet A. The choice of the Hamming distance might also make the construction impossible for a given L and A since either a smaller matrix or a bigger alphabet needs to be defined. There are however some construction methods available for certain values of L, A and h=1 that usually produce solutions of non-square window size. These methods are based on De Bruijn sequences and perfect maps. Since no direct solutions exist in general, a pseudo-random method can also be used to generate M.

Pseudo-random code generation consists of adding random colors from the A palette and verifying that each new $V_{(ij)}$ has a minimum Hamming distance of h with all other V vectors already included in M. Initially w x w intensities from the palette are randomly placed in the upper left corner of the M matrix. The algorithm then proceeds by adding

w x *1* random intensities downwards and then *1* x *w* random intensities to the right to create the first column and row respectively. Every time these are added a new *V(i,j)* is created and must be verified. If it does not satisfy the minimum Hamming distance, the newly added intensities, in this case *w* intensities, are replaced with new random values until the new *V(i,j)* is deemed unique. Once the edges are filled, the algorithm randomly adds one new color from the palette at a time starting from column *w+1* and row *w+1* and expanding until the end of the row. Every remaining row is filled following the same logic. Each new color added creates a new *V(i,j)* that is verified so that it passes all the given criteria. If the added color fails any criteria, a new random intensity is picked from the palette. At any stage, if all the intensities are tried from the palette and none passes the minimum Hamming distance criteria, the whole process is restarted.

Every vector *V(i,j)* contains nine elements and each of the elements, besides the middle element, will appear in eight neighboring vectors. This knowledge can be used to assign a confidence level on the vector that is identified. A confidence level is measured by the amount of neighboring vectors that are identified correctly from an image compared to the neighbors that originally appear in the pattern around the vector of interest. The confidence level is used to determine if reconstructed vectors have errors present and enable them to be dropped or repaired.

Vectors can only be corrected when the minimum Hamming distance between all codes exceeds the number of errors in reconstructed code by two. Therefore, a minimum Hamming distance of at least three is needed to be able to correct one fault since with lower Hamming distances it is only possible to detect errors but there is not enough information to be able to repair errors.

Albitar *et al.* suggest a similar code based on unique 3x3 symbols that have a Hamming distance of at least three or more [20]. The codes they suggest are based on monochromatic symmetric symbols. These patterns do not require any color information to decode and reduce the amount of information that is needed.

2.2.2.2 De Bruijn Sequences and Perfect Maps

Elements in a grid pattern can be constructed using a series of 1D De Bruijn sequences. These sequences are periodic sequences composed of elements from a determined alphabet. Every *v*-tuple of elements appear only once in a period. This property is important in 3D reconstructions since it can be used to construct pattern arrays where every location can be uniquely labeled.

As mentioned above, a De Bruijn sequence is defined from a specified alphabet n of order m [12]. This defines a circular sequence of length n^m and contains subsequences of length m that appear only once in the complete sequence. De Bruijn sequences are constructed by searching Eulerian circuits or Hamiltonian circuits. An Eulerian circuit is built by creating a path starting and ending at the same vertex by passing through all the edges exactly once. The label at every edge is assembled to create the De Bruijn sequence. A Hamiltonian circuit differs to the Eulerian circuit by passing through all the vertices exactly once but not through every edge. Figure 2.4 demonstrates a De Bruijn graph where m is equal to four.

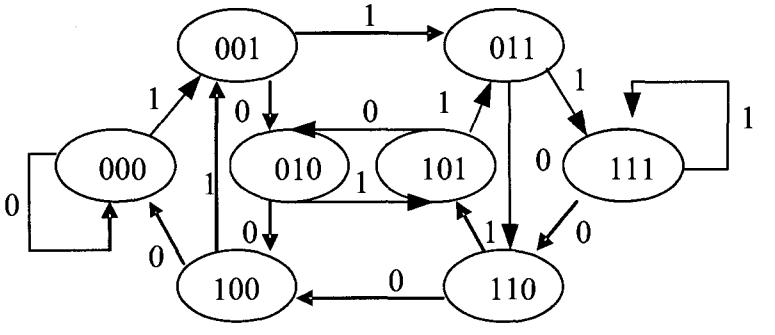


Figure 2.4 - Circuit to determine a De Bruijn sequence.

Traversing the graph as an Eulerian circuit at the vertex labeled *001,* the De Bruijn sequence created by appending the values of the edges traversed is S = *1000010111101001.* From *001* to *011* the edge labeled 1 is traversed hence *1* is added to the sequence. The next step *011* to *110* is traversed and the value of the edge is *0* hence *0* is added to the sequence. This is repeated until all edges are traversed and the first vertex

is reached again. In a similar fashion, the graph is traversed as a Hamiltonian circuit starting at the vertex *110* and an example of a sequence is $S = 00101110$ by traversing all vertices only once and ending up at the starting vertex.

De Bruijn sequences can be used to create perfect maps. Perfect maps are arrays of $n \times m$ elements where every possible subarray $u \times v$ is present exactly once. This definition can be relaxed so that every subarray may appear at most one time in the array or not at all. Unlike pseudo-random arrays, these arrays can be constructed in a deterministic way. De Bruijn sequences cannot however easily take into account other parameters such as Hamming distance or color restraints on the pattern.

De Bruijn sequences can be used to create a variety of different patterns. Some authors suggest using color slits. Hügli and Maitre add a constraint when building the sequences so that there are never two consecutive identical symbols in the sequence [21]. Vuylsteke and Oosterlinck use the sequences to encode a checkerboard pattern where the column of every grid point is encoded using two sequences [22].

Some researchers have also explored different ways of building decodable De Bruijn sequences. Mitchell *et al.* create De Bruijn patterns so that they can be easily decoded [23]. The observed patterns need to be decoded so that the location can be extracted. The brute force method simply compares the codes to a look up table that contains a list of all possible codes and their associated position. By providing a way to decode the sequence directly this step is accelerated and is not limited to the amount of memory needed for the look up tables.

2.2.2.3 Pseudo-random Binary Arrays

Lavoie *et al.* suggest encoding pseudo-random grids as patterns [24]. These are called pseudo-random binary arrays (PRBA). A PRBA is composed of $n_1 x n_2$ binary sequences such that any subset $k_1 x k_2$ is unique and defines a specific location. n_1 and n_2 must be

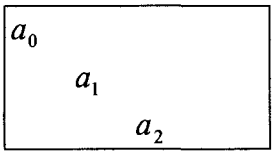
relatively prime. Two integers are relatively prime is they do not share any other common divider besides 1. n_1 and n_2 are defined as follows:

$$\begin{aligned} n &= 2^{k_1 k_2} - 1 \\ n_1 &= 2^{k_1} - 1 \\ n_2 &= \frac{n}{n_1} \end{aligned} \tag{2.12}$$

To generate the binary sequences the primitive polynomial modulo 2 method described by Press *et al.* is used [25]. Simply stated, primitive polynomial can be used to generate pseudo-random bits that will not be repeated until 2^n bits are generated where n is the degree of the primitive polynomial. In this case the degree of the primitive polynomial is $m = k_1 k_2$. Pseudo-random sequences (PRS) have the following important window property: a window of width m, where m is the degree of the primitive polynomial, can be slit on a PRS and the subsequence found in the window only appears once in the whole sequence. This enables the creation of PRBAs that uniquely define positions [26]. PRS are generated by using linear feedback shift registers (LFSR). The feedback equation is described by the primitive polynomial with coefficients from a finite Gallois field *GF(q)* where q is the power of a prime number. The binary sequence is of length n and can be used to construct a PRBA by folding the sequence to create a $n_1 \times n_2$ PRBA. The PRBA is created by first filling the main diagonal with the first entries in the sequence [27]. When the end is reached, the opposite side of the edge is filled until the array is full. As an example let us take $n_1 = 3$ and $n_2 = 5$ which creates the following sequence:

$a = 000100110101111$

This is then used to create a 3x5 array as shown in Figure 2.5.



Fill the main diagonal

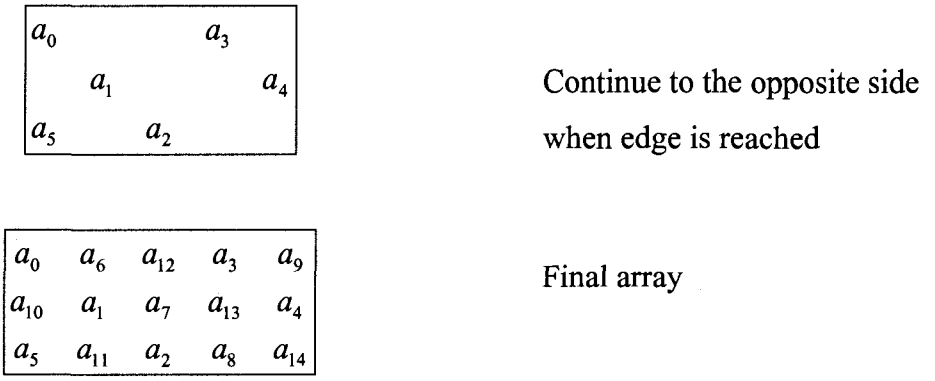


Figure 2.5 - Building a PRBA.

The final array in the example is the following:

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Figure 2.6 – Constructed PRBA.

The PRBA can then be used to create a pattern. Lavoie *et al.* create a line grid of $n_1 x n_2$. At the intersections of the lines full dots are created where ones are present in the array and smaller dots are created where zeroes are present. Figure 2.7 demonstrates such a grid.

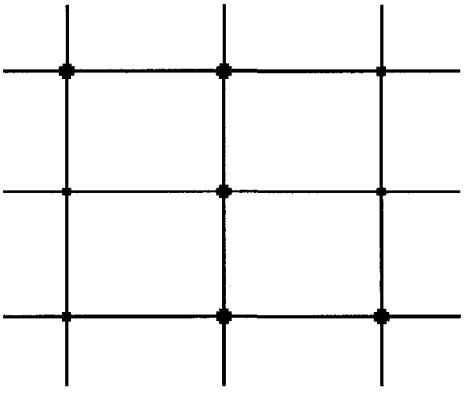


Figure 2.7 - PRBA pattern.

The method of segmentation proposed for this kind of code is heavily based on morphological operators. This method might suffer in special cases. For example, if any error in the detection of the lines occurs or if an object contains straight edges, problems could arise in labeling positions accurately.

A similar method suggested by Lavoie *et al.* can be used to create a pseudo-random multi-valued sequence (PRMVS) [28]. The values found in the PRMVS can be substituted by colors in a predetermined color palette. The sequence is projected as a series of vertical and horizontal color lines. Positions are uniquely defined by looking at the neighboring lines. As an example, if three is the primitive polynomial used to create the PRMVS, then to define a location, three lines need to be observed vertically and horizontally. Figure 2.8 demonstrates an example of a PRMVS.

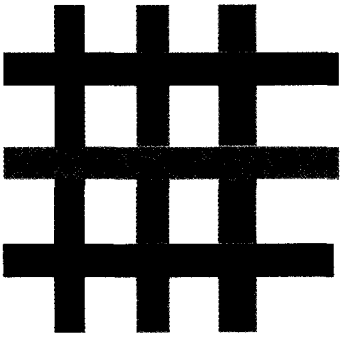


Figure 2.8 - Color slid grid using PRMVS.

The generation method of PRBAs is difficult to change to suit any additional requirements that might be required in the code. Pseudo-random codes do not have this limitation. PRMVS do not encode properly regions where horizontal and vertical color lines overlap and can create some holes. An alternative to that problem consists of searching for groups of four corners close to the color lines intersection, which also contributes to increase to density of points to be reconstructed in 3D [29]. Furthermore, if objects contain a high degree of curvatures it can be difficult to locate neighboring lines

as they then get heavily distorted on the image plane, and the encoding of the affected regions might remain unknown.

2.2.3 Direct Codification

Direct codification patterns encode each location uniquely. They are similar to the ramp functions but the latter usually only encode a finite number of sections and not every pixel as in direct codification. With direct codification, patterns are static, and positions are labeled by the intensity measured at the sensor. There are two predominant approaches for this encoding: codification based on grey levels and based on color. Carrihill and Hummel developed a system called *intensity ratio depth sensor* to find the correspondence of pixels [30]. The pattern projected consists of a linear wedge spread along the columns containing varying grey levels as shown in Figure 2.9. A ratio is then calculated between the intensity values of every pixel under the linear wedge and under a pattern of uniform illumination. The ratio is used to determine the position of every pixel. However it is quite difficult to accurately extract the proper ratio of locations due to the high sensitivity to noise and non-linearities of the device used.

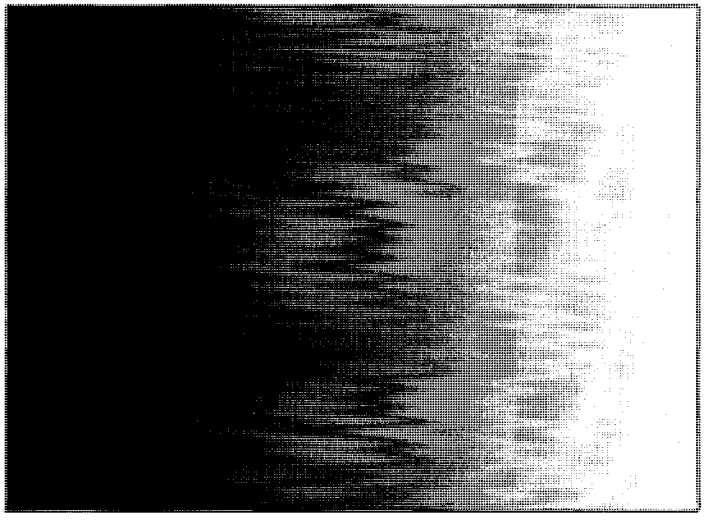


Figure 2.9 - Grayscale pattern proposed by Carrihill and Hummel.

Color codification follows a similar strategy. Tajima and Iwakawa suggest a rainbow pattern [31]. A large set of vertical slits are encoded using different wavelengths so that a

large sampling of the spectrum from red to blue is utilized. Two images of the scene are taken with different color filters. By calculating the ratio between the two images the unique code is found. This method does not depend on the illumination and on the color of the objects found in the scene.

Direct codification can theoretically acquire high resolution data but practically it is quite difficult to extract the correct intensity value of each location. The values of intensity surrounding a location are usually close in intensity and it is difficult to accurately label positions. Furthermore, different elements can affect the values observed. Noise, nonuniform lighting, object's textures and its specular reflection as well as sensors' characteristics are examples.

2.2.4 Comparison of Approaches

Scharstein and Szeliski argue that time-multiplexed binary code projections yield better results [13]. This is primarily due to the fact that this technique does not require estimations for the albedos of the scene and that it is less affected by non-linearities found in the camera, in the projector and from the interrelations in the scene when compared to continuous patterns. N-ary patterns could potentially yield more accurate results with the same amount of images as gathered for a gray code technique.

On the other hand, spatial neighboring can be encoded as grid patterns or line patterns [32]. These encodings suffer from discontinuities when matching lines. A priori knowledge of the object being scanned is required to pick a suitable line thickness. If the lines are too thin many discontinuities will arise; if the lines are too thick, then precision will be lost. Spatial neighboring patterns can also be encoded as pseudo-random arrays. This encoding does not need as much a priori knowledge of the scene and is relatively easy to capture and extract.

Finally, direct codification offers the simplicity of a static projected pattern. This class of patterns can, in theory, yield the highest level of density. However, in practice, it is well established that it is quite difficult to properly extract the codes from a scene. As a result, the data that is extracted is often ambiguous and unreliable.

2.3 Image Processing Techniques

The topic of image processing has been explored in great detail in the literature and hence there are a multitude of algorithms and techniques that are available. For structured lighting applications, a critical issue arises from the segmentation problem to retrieve the pattern on the image plane. With pseudo-random codes it is necessary to be able to clearly segment color regions. Furthermore, due to the nature of the application, this segmentation needs to be performed in a quick and efficient way. This section discusses some of the general classes of image segmentation algorithms that appear relevant for structured lighting stereoscopy and that have been considered in this work.

A class of segmentation algorithms to consider is based on edge detection [33]. Laplace and Sobel kernels can be used to detect gradients in images. These can then be used to segment regions based on the closed edges formed. Such algorithms do provide fast segmentations capabilities but suffer on some areas. Among other things, all color regions must be perfectly detectable. As mentioned in the previous sections, sometimes areas get merged together and an edge detection algorithm would only detect the whole merged regions since the gradients within the merged regions are negligible. Furthermore, closed regions must be found to be able to segment regions properly which is rarely the case. Even the Canny algorithm, which is intended to connect neighboring edges, will have difficulty in closing all regions.

Watershed transformation algorithms also require good gradient information between regions [33]. The algorithm is based on the idea of flooding regions with water. Water will accumulate in regions of low altitude and regions of high altitude will not gather any water. The first step is to calculate pixels that correspond to high gradients. These form watershed lines which represents the regions boundary between low and high levels.

Simulated water is then added and will follow the trajectory of the gradients and accumulate in areas of local low intensity which represent the segmented regions. This algorithm would need to be modified to fit the specific problem considered here since it is desired to segment different colors of similar intensities. The standard watershed algorithm would gather water in all color regions and offer no distinction between the projected colors. Furthermore, this algorithm is computationally intensive.

Region growing algorithms were also considered [33]. The seeded region growing algorithms require a number of seeds to be initially provided. The seeds represent the number of regions that will be segmented. A region is grown by comparing all of its surrounding pixels. Pixels are then added to the region based on a similarity measure. The most similar pixel of the surrounding pixels is added to the region. This is repeated for all regions until every pixel is assigned to a region.

The seeded region growing algorithms are suitable to segment various independent regions that are well seeded. However, they depend on the proper selection of seeds. The projected pattern is composed of several thousands of regions and will require as many seeds. This raises two important problems. First, it can be difficult to place so many seeds accurately. The second problem is caused by the computational strain of growing thousands of regions. These issues quickly determine that region growing algorithms are not suitable for the specifications and requirements of the system.

To solve some of the drawbacks with the seeded region growing algorithms, unseeded regions growing algorithms can be explored [33]. Unlike the seeded version, only an initial seed is needed. The algorithm will expand the region around the initial seed until a certain threshold of similarity is reached. Once this threshold is reached, the algorithm will create additional seeds and continue expanding regions. This is repeated until all pixels are assigned to a region. This technique is more suited to solve the required problem of segmenting the pseudo-random color regions. The only real drawback is again computational time.

Clustering algorithms, such as the K-means algorithm, are used to segment an image into K clusters [34]. The starting number of clusters is randomly chosen or can be chosen based on some knowledge of the image. Every pixel in the image is then assigned to a cluster based on minimizing a variance measure such as pixel color, intensity and texture qualities. The variance measure is performed between the pixel and the clusters center. The cluster center is then updated with the added assigned pixel values and the process is repeated between all new clusters. As with region growing algorithms, a good choice of initial clusters (seeds in the case of region growing algorithms) must be made. Even though in the pseudo-random color pattern the number of color regions is known, some might not appear in the image. This is due to various reasons such as occlusions, surface orientation or the various properties of the acquired objects. It also becomes difficult to accurately locate all the cluster centers. This is important because the further away the initial guess of the cluster centers are the more iterations the algorithms will need to perform before converging. This, again, is important because the desired processing must be done quickly.

The chosen segmentation algorithm based on histogram analysis is successful in solving the segmentation problem of the pseudo-random color pattern. It is fairly robust and is also efficient. Multiple segmentation techniques in the literature require multiple passes on every pixel to calculate various properties. However, the suggested histogram algorithm can require at best only one pass if no special segmentation cases are present in the image. Additionally, the suggested algorithm is simple yet still yields adequate results. This simplifies any changes and extensions that might be required to the algorithm. It also reduces the amount of parameters needed to properly segment an image and consequently simplifies and augments the possibility of good segmentation results over scenes that are not known a priori.

2.4 3D Reconstruction

When the acquisition principles of structured lighting are used to accurately label positions in captured images after image processing is performed, these labels are

required to reconstruct 3D information of a scene. Labels are defined in two dimensions and specify locations on a 2D plane, usually the image plane. A transformation between 2D information and 3D information, usually called the depth, must then be performed. Reconstruction is the usual method to accomplish this transformation. 3D data can be reconstructed if the same point is seen from multiple viewpoints and the relations between viewpoints are available. The following sections will review the epipolar geometry principle of stereoscopic systems. This geometry defines the relationship between calibrated acquisition devices and features that are matched in their respective image planes. The epipolar geometry of a system can be used to optimize reconstruction methods. Two reconstruction methods are also discussed in this section.

2.4.1 Epipolar Geometry

Matching points in the image planes are related by the epipolar geometry of a stereoscopic setup. The epipolar geometry facilitates the search for matching points since they provide a 1D search path. Figure 2.10 illustrates the epipolar geometry construction for two pinhole cameras.

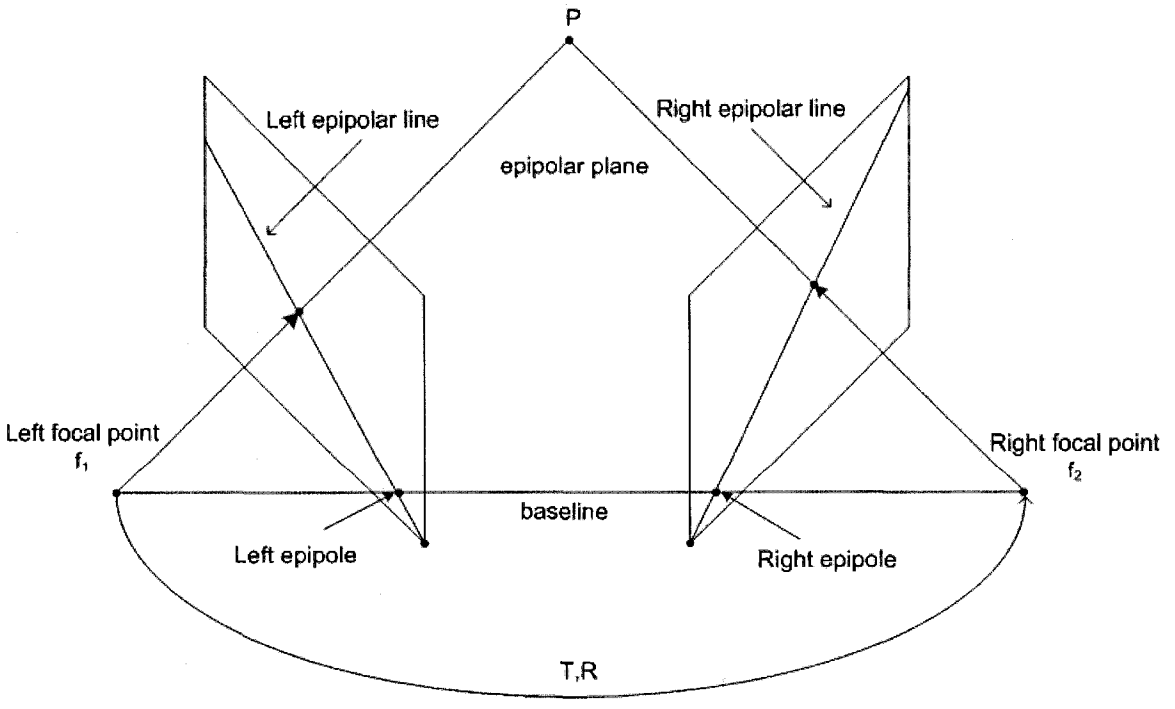


Figure 2.10 - Epipolar geometry.

The left epipole is the left image projection of the right camera's focal point and the right epipole is the right image projection of the left camera's focal point. Epipolar lines are defined as the vector from the epipole of a camera to the projection of point P. The epipolar plane is defined as a plane composed of point P and the two focal points of the cameras. It can be seen that any point that lies on an epipolar line will lie somewhere on the same epipolar line of the second camera hence limiting the search for matches to a 1D path.

With respect to the left camera's frame of reference, P_L can be defined as the vector from f_1 to P, T as the translation between the two reference frames and the vector P_L-T as the vector from f_2 to P [6]. These define the epipolar plane. Since these vectors are coplanar the following equation can be made.

$$(P_L - T)^T \cdot (T \times P_L) = 0 \tag{2.13}$$

The cross product yields a vector normal to the surface of the epipolar plane.

$$T \times P_L = \vec{n}_{epipolar} \tag{2.14}$$

Hence the following, since the two vectors are perpendicular to each other.

$$(P_L - T)^T \cdot \vec{n}_{epipolar} = 0 \tag{2.15}$$

The two reference frames are related by a rotation R and a translation T mentioned above.

$$\begin{aligned} P_R &= R(P_L - T) \\ P_L - T &= R^T P_R \end{aligned} \tag{2.16}$$

Replacing this equation in equation 2.13 we get:

$$(R^T P_R)^T \cdot (T \times P_L) = 0 \tag{2.17}$$

Developing $T \times P_L$ gives the following:

$$T \times P_L = \begin{bmatrix} T_y P_{Lz} - T_z P_{Ly} \\ T_z P_{Lx} - T_x P_{Lz} \\ T_x P_{Ly} - T_y P_{Lx} \end{bmatrix} = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \begin{bmatrix} P_{Lx} \\ P_{Ly} \\ P_{Lz} \end{bmatrix} = SP_L \tag{2.18}$$

Finally,

$$\begin{aligned} &(R^T P_R)^T \cdot (T \times P_L) = 0 \\ &P_R^T RSP_L = 0 \end{aligned} \tag{2.19}$$

The product of RS is defined are the essential matrix E in epipolar geometry.

$$E = RS \tag{2.20}$$

Also, P can be written as a projective transformation as follows:

$$P = \left(\frac{Zp}{f} \right) \tag{2.21}$$

Substituting P in (2.19) by its projective counter part in (2.21):

$$\left(\frac{Z_R p_R}{f} \right)^T E \left(\frac{Z_L p_L}{f} \right) = 0 \tag{2.22}$$

Finally dividing by $Z_R^T Z_L$

$$p_R^T E p_L = 0 \tag{2.23}$$

This equation represents how points are mapped between images on the epipolar lines. The essential matrix E encapsulates all the extrinsic parameters between two cameras which are R and T.

The essential matrix does not encapsulate the intrinsic parameters of cameras. It can be expanded to include these and results in the fundamental matrix. The latter maps pixel points and not 3D points to the epipolar lines between images. Since the mapping is from pixel the intrinsic parameters of the cameras need to be added to the essential matrix.

The intrinsic parameter matrix is defined as follows:

$$M_{\text{int}} = \begin{bmatrix} \frac{f}{s_x} & 0 & o_x \\ 0 & \frac{f}{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.24}$$

where f is the focal length of the lens, s_x and s_y are the size of the pixels on the CCD array of the camera, and o_x and o_y represent the center of the CCD array.

With the intrinsic matrix the image points p_L and p_R can be transformed into the pixel coordinates p_{pixelL} and p_{pixelR}.

$$\begin{aligned} p_L &= M_{\text{int}\,L}^{-1} p_{pixelL} \\ p_R &= M_{\text{int}\,R}^{-1} p_{pixelR} \end{aligned} \tag{2.25}$$

The essential matrix equation can be rewritten as follows:

$$(M_{\text{int}R}^{-1} p_{pixelR})^T E(M_{\text{int}L}^{-1} p_{pixelL}) = 0$$
$$p_{pixelR}^T (M_{\text{int}R}^{-1})^T E M_{\text{int}L}^{-1} p_{pixelL} = 0 \tag{2.26}$$

This gives the fundamental matrix F which is simply the essential matrix E multiplied with the intrinsic parameters matrices of the cameras.

$$p_{pixelR}^T F p_{pixelL} = 0 \tag{2.27}$$

2.4.2 Generalized Triangulation

Triangulation is essentially the procedure of calculating the intersection between two lines. Theoretically, two points that are matched on two camera planes, along with the knowledge of their intrinsic and extrinsic parameters, create a line with their respective base that intersects each other at a point in the world (epipolar constraint). Due to errors in digitalization and the difficulty in getting an accurate calibration, an actual intersection between two lines will rarely occur. Multiple methods exist to solve this problem. Two of these methods are examined in this section: the middle point reconstruction and the optimal polynomial reconstruction techniques.

2.4.2.1 Middle Point Reconstruction

Reconstruction is usually done by means of triangulation. To perform triangulation, calibration data between multiple viewpoints or cameras is needed with the location of a point of interest in the different image planes. A typical setup consists of at least two cameras both calibrated to a world reference frame. The matches found need to be with respect to the same reference frame so matches need to be transformed using the calibration parameters before being used in triangulation. The focal points of both cameras are converted to the 3D world reference frame coordinates. With these four points (the image plane matches and the focal points of the left and right cameras), triangulation is performed.

In the middle point algorithm, two rays are created between the focal point and the pair of matching points on the image plane of both cameras [6]. Ideally, with perfect matches, accurate calibration, and if the cameras had infinite resolution, the two rays would intersect in space. This intersection represents the 3D coordinates of the image point located in the image plane.

Since these circumstances never occur in practice, the middle point algorithm finds the minimum distance between the two rays and then takes the point in the middle of that line as an estimate of the 3D point. Figure 2.11 illustrates the concept.

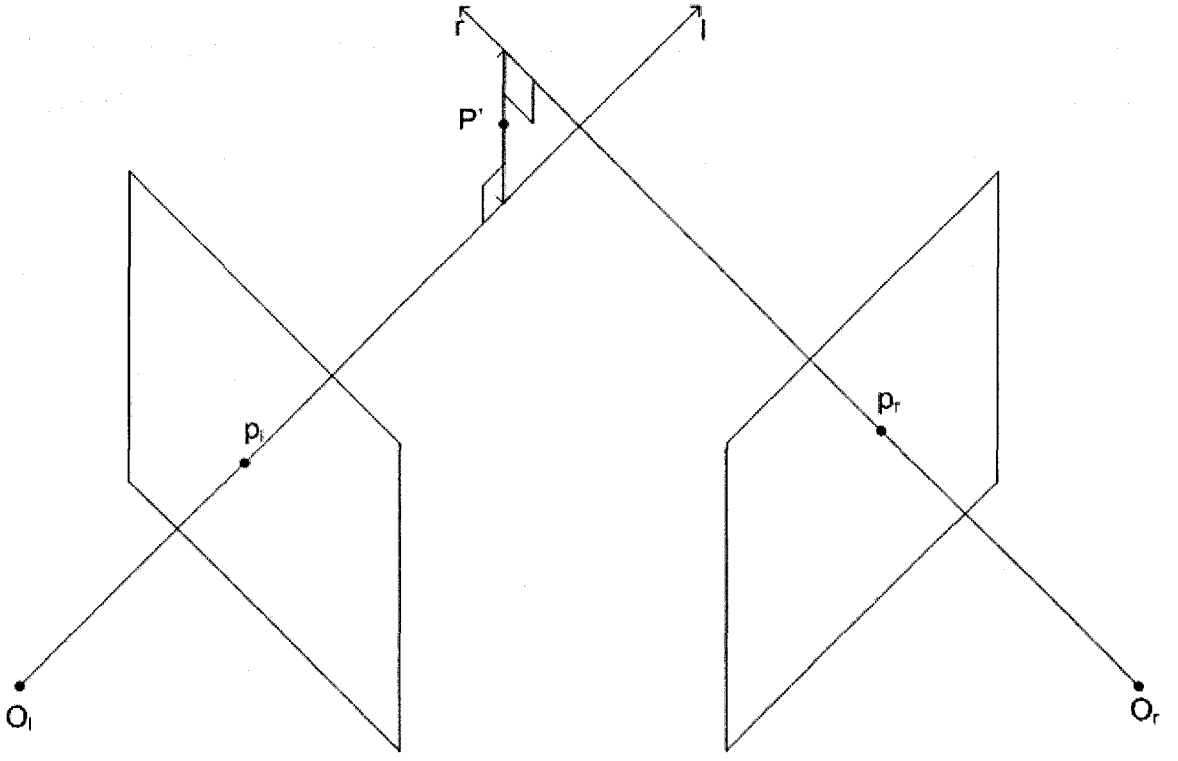


Figure 2.11 - Middle point reconstruction.

p_l and p_r are the 3D coordinates of the matches in the left and right camera reference frames respectively. O_l and O_r are the focal points in 3D of the left and right cameras respectively.

Let us define the following vectors: ap_l is the segment l from O_l going through p_l, $T + bR^T p_r$ is the segment r that goes through O_r and p_r, and finally w is a vector

perpendicular to l and r. To find the midpoint P', the problem can be represented as finding a segment that is parallel to w and that joins l and r. This can be done by solving the following linear equation.

$$ap_l - bR^T p_r + c(p_l \times R^T p_r) = T \tag{2.28}$$

where T and R are the translation and rotation between the two cameras.

The points defining the endpoints of the line intersecting l and r and that is parallel to w with respect to O_l are ap_l and $T+bR^Tp_r$. P' is calculated to be in the middle of the segment and is simply estimated by dividing the length between the two endpoints by two.

The middle point algorithm is simple and easy to compute. However, Hartley *et al.* demonstrate that it is a weak reconstruction algorithm compared to other approaches. This method is neither affine nor projective invariant and suffers poorly in those circumstances [35][36].

2.4.2.2 Optimal Polynomial Reconstruction

Hartley *et al.* developed a triangulation method using a minimization criteria based on the epipolar geometry [35][36]. In practice the matching pixel points x and x' will not satisfy to the epipolar constraint. The goal is to find two estimated points $\hat{x}$ and $\hat{x}'$ that are close to x and x' and satisfy the epipolar constraint $\hat{x}'^T F\hat{x} = 0$. This is showed in Figure 2.12.

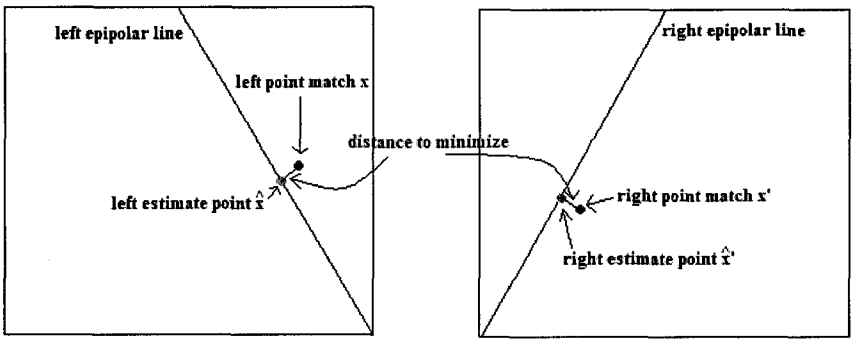


Figure 2.12 - Epipolar minimization criteria.

To find these points the geometric error cost function is minimized.

$$C(x,x') = d(x,\hat{x})^2 + d(x',\hat{x}')^2 \tag{2.29}$$

The estimated points must satisfy $\hat{x}'^T F\hat{x} = 0$ which results in finding matching epipolar lines. $d(x_1,x_2)^2$ is the squared Euclidean distance.

Any two points that satisfy the epipolar constraint will lie on the corresponding epipolar lines in both images. Thus point $\hat{x}$ and $\hat{x}'$ will lie on the epipolar line l and r respectively. The criteria can then be reformulated as the minimization of the sum of the squared Euclidean distance between the estimated points and their corresponding epipolar lines.

$$C(x,x') = d(x,l)^2 + d(x',r)^2 \tag{2.30}$$

where l and r range over all the choices of epipolar lines.

To minimize this distance the following must be performed [35][36]:

1. Parameterize the epipolar line by a parameter t to be able to represent the set of possible epipolar lines as a function of t such that l is represented as $l(t)$.
2. Use the fundamental matrix F to compute $r(t)$.

3. Express the cost function as a function of t.
4. Find the value t that minimizes the cost function.

Once the cost function is redefined, the problem is reduced to finding the minima of the cost function in t where the distance of the measured points is minimized between the best matching epipolar lines.

$$C(x,x',t) = d(x,l(t))^2 + d(x',l'(t))^2 \tag{2.31}$$

To simplify the analysis, both measured points x and x' are placed at the origin $(0,0,1)^T$ by applying a rigid transformation on the points. Furthermore, the epipoles are placed at the point $(1,0,f)^T$ and point $(1,0,f')^T$. Applying a rigid transformation does not change the outcome of the minimization of the cost function.

To move x to the origin a translation is needed. If x is the following, $x=(x_1,x_2,1)^T$, then the needed translation is as follows:

$$L = \begin{pmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -x_2 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.32}$$

such that:

$$Lx = (0,0,1)^T \tag{2.33}$$

The same calculations are performed on x' where $x' = (x_1', x_2', 1)^T$.

The next step is to transform the epipoles to $(1,0,f)^T$ and $(1,0,f')^T$ for the left and right epipoles respectively. The epipole requires a rotation around the origin by an angle θ. This rotation is represented by the following matrix.

$$R = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.34}$$

The epipole $e = (e_1, e_2, e_3)^T$ is translated and rotated on the x-axis such that:

$$RLe = (1,0,f)^T \tag{2.35}$$

Solving for θ yields the following:

$$\theta = -\arctan\left(\frac{e_2 - e_3 x_2}{e_1 - e_3 x_1}\right) \tag{2.36}$$

The same is applied to calculate the angle of rotation and the rotation matrix R' associated with the right epipole. Figure 2.13 illustrates the results after the transformation of an epipole e and a point x.

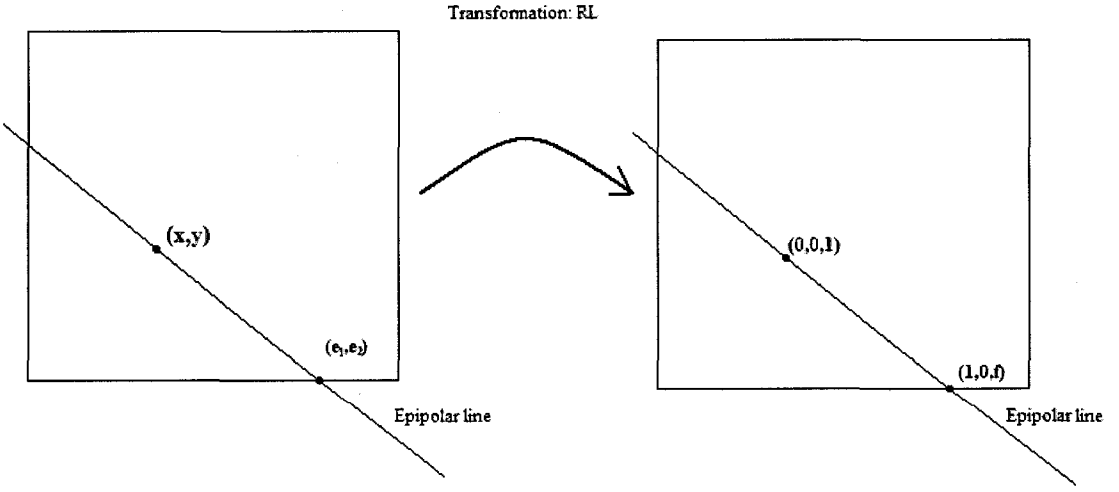


Figure 2.13 – Parameterization of points.

With the angle of rotation the R' and R^T matrices can be calculated and used to transform the fundamental matrix into the same reference frame.

$$F = R'L'^{-T}F_0L^{-1}R^T \tag{2.37}$$

where L and R are the transformation for the left image, L' and R' the transformations for the right image and F_0 is the original fundamental matrix.

Since $F(1,0,f)^T=(1,0,f')F=0$, the transformed fundamental matrix can be represented under the following form:

$$F = \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix} \tag{2.38}$$

where a, b, c and d represent the results of the modified fundamental matrix.

Next we need to define epipolar lines as a function of t and calculate the squared distance from a given epipolar line represented by $\lambda(t)$ and the origin. Taking the left epipolar line passing through the point $(0,t,1)^T$ and the epipole $(1,0,f)^T$. The line is represented by the cross product of these two points $(0,t,1)^T$ x $(1,0,f)^T = (tf,1,-t)^T$. The squared distance of the line to the origin is:

$$d(x,\lambda(t))^2 = \frac{t^2}{1+(tf)^2} \tag{2.39}$$

The right epipolar line can be computed by using the modified fundamental matrix and the vector representing this line is defined in a similar fashion as:

$$(-f'(ct+d), at+b, ct+d)^T \tag{2.40}$$

The squared distance of this vector to the origin is the following:

$$d(x',\lambda'(t))^2 = \frac{(ct+d)^2}{(at+b)^2 + f'^2(ct+d)^2} \tag{2.41}$$

Combining the two squared distances gives the new cost function to minimize, given as follows:

$$C(t) = \frac{t^2}{1+f^2t^2} + \frac{(ct+d)^2}{(at+b)^2 + f'^2(ct+d)^2} \tag{2.42}$$

where, a,b,c,f, and f' are from the modified F matrix.

To minimize the function, the first derivative is obtained and solved for zero.

$$\frac{dC(t)}{dt} = \frac{2t}{(1+(tf)^2)^2} - \frac{2(ad-bc)(at+b)(ct+d)}{((at+b)^2 + f'^2(ct+d)^2)^2} \tag{2.43}$$

To find the minimum the equation is manipulated to a common denominator and solved for the numerator equal to zero.

$$n(t) = t((at+b)^2 + f'^2(ct+d)^2)^2 - (ad-bc)(1+(tf)^2)^2(at+b)(ct+d) = 0 \tag{2.44}$$

This results in a polynomial of a sixth order therefore there will exist six roots and a maximum of three minima might exist. The roots are found by finding the Eigen values the characteristic polynomial $P(x) = \det[A - xI]$. The characteristic polynomial *mxm* companion matrix

$$A = \begin{pmatrix} \frac{-a_m - 1}{a_m} & \frac{-a_m - 2}{a_m} & \cdots & \frac{-a_1}{a_m} & \frac{-a_0}{a_m} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \tag{2.45}$$

is equivalent to the general polynomial equation

$$P(x) = \sum_{i=0}^{m} a_i x^i \tag{2.46}$$

By manipulating equation 2.44 into the general form polynomial equation this method can be used to find the desired roots [25].

Each minimum will have to be checked and the function should also be checked when $t \to \infty$.

Finally, once the minimum is established, t is substituted in the epipolar line equation to find the homogeneous point and then the inverse rigid transformation is performed on the points to find the real coordinates. From there the 3D coordinates can be calculated using a direct linear transform method.

Chapter 3 System Description and Implementation

Chapter 2 examined various methods that are available for solving the 3D acquisition and reconstruction problems. Chapter 3 discusses which methods are used and details how they are built in a working system. The proposed design results in a complete integrated system that covers every aspect of 3D reconstruction from pre-acquisition to full 3D color representation of scenes. Chapter 3 also explains the motivations behind the choices made and examines the fallbacks of certain methods and how these were solved or reduced. Furthermore, a detailed explanation of the steps followed by the system's algorithm is proposed.

3.1 Integrated 3D Acquisition and Reconstruction System Design

The suggested system uses pseudo-random codes as the projected pattern. This type of pattern has been chosen since it is easy to build and it is flexible. This aspect becomes important if a specific scene is error prone. The pseudo-random code can be adapted as required for added robustness or in certain circumstances to alleviate difficulties in the segmentation of the codes. Pseudo-random codes can be criticized on the length of time required to be generated. This might be an inconvenience if they are generated dynamically and used on the scene. However dynamic generation of patterns is neither required nor necessary. The pattern can be generated offline once and used multiple times when acquiring scenes. In some circumstances it may be beneficial to change the colors of the projected pattern. The colors can then be dynamically replaced with the offline generate codes and this process requires little processing time when the codes are already available.

Different physical setups exist to acquire 3D data. For example, the system can be composed of a single camera accompanied with a projector [37]. The calibration is performed between the projector and the camera and the reconstruction is performed with the calibration parameters of the system. There is however an important issue regarding

this type of setup when used for exploration purposes, for example on a mobile platform. Projectors tend to be significantly heavier and larger than cameras and hence are more likely to move and are more disposed to vibrate with the movements of the mobile platform. This can cause the projected pattern to be blurred when captured by the camera and furthermore its calibration parameters will tend to vary notably with time. Frequent calibration of the system would also be required as well as sturdy mounting brackets to assure optimal results. Moreover, calibration techniques between a pair of CCD cameras are well established, while the optical construction of LCD projectors makes the calibration procedures more complex and less accurate.

For the above mentioned reasons, the suggested setup uses two calibrated cameras and a projector. The projector is simply used to project light patterns and hence is not calibrated with any other part of the system. The cameras are mounted onto a fix bracket and will not, under normal circumstances, require multiple calibration sessions. The smaller size of the cameras relative to the size of the projector permits them to move less and to be mounted more robustly. Furthermore, mounting the cameras on a fixed bracket will prevent any calibration parameters to change between the cameras. The bracket itself might move but calibration between the cameras will not be affected. An extension to this system is to have the two cameras also calibrated with the projector. This could be used to acquire redundant data and perform data merging between the different sets of data. The redundant data could also be used to validate and correct erroneous information. However, it suffers from the same shortcomings found in the single camera and projector setup.

The two cameras and the projector create an active stereoscopic system. Active stereoscopy is used since it allows the gathering of dense 3D results on a variety of objects. It does not suffer from the same limitations as passive stereoscopic systems which are affected by the amount of features that can be matched between pairs of images. An active stereoscopic setup permits the generation of features using the projector. With these controlled features, a higher accuracy can be achieved when matching features between the two images.

The active stereoscopic system's calibration parameters are calculated before any acquisition of the scene. This enables a complete 3D reconstruction of the scene. Other systems that do not explicitly calculate the calibration parameters can only reconstruct a scene up to a scale factor or up to a projective transformation. These reconstruction techniques give a relative idea of the scene but do not give absolute coordinates which are often desirable.

The proposed system is divided in multiple logical modules that are presented in Figure 3.1

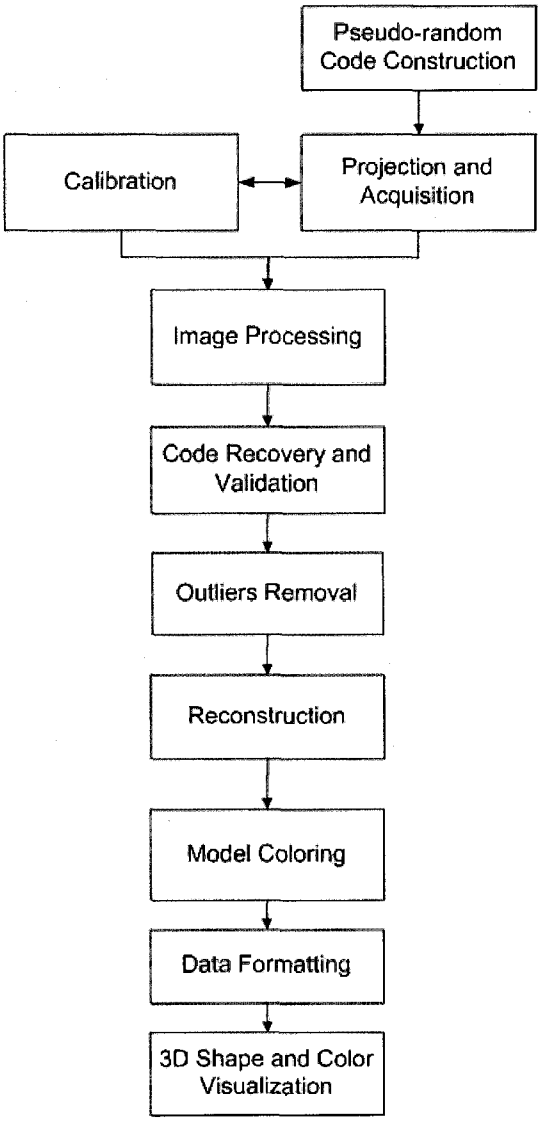


Figure 3.1 - Overview of System.

The system is composed of ten modules that include: pseudo-random code construction, projection and acquisition, calibration, image processing, code recovery and validation, outliers removal, reconstruction, model coloring, data formatting and 3D shape and color visualization. The pseudo-random code construction module builds the pseudo-random code used as a projection pattern onto desired scenes. It creates the projections using n colors and an optional minimal Hamming distance criterion. The projection and acquisition module displays the pseudo-random color pattern, acquires and saves images from the two cameras attached to the system. It is also the module that is in charge of acquiring marching patterns that are used to increase the resolution of an acquired scene. The calibration module, given a series of specific images, will analyze and estimate the intrinsic and extrinsic parameters of the two cameras. The image processing module segments the projected codes from the captured images. Code recovery and validation recognizes the pseudo-random codes from the segmented images and drops any inconsistent recovered codes. Using the calibration information the outliers removal module removes any codes that do not satisfy the epipolar constraint within a tolerance on the error. The 3D information about points marked by the pseudo-random codes is calculated in the reconstruction module. Model coloring gathers the color information from 2D images. The data formatting module organizes the data in various formats so that it can be used in different types of 3D data viewers or for further analysis. The final module handles the 3D shape color visualization. It takes as input the output of the data manipulation modules and renders to the screen a 3D color model of scanned scenes.

3.2 Calibration

A calibration module is necessary in the system since calibration parameters are required by the reconstruction module. As mentioned in section 2.4, the intrinsic and extrinsic parameters of the camera pair are required to perform triangulation.

The calibration module gathers the two sets of intrinsic parameters separately. This simplifies the process since the calibration patterns do not need to be completely visible by both cameras. Often some parts of the view of a camera are not seen by the second

camera. This limits the accuracy of the intrinsic parameters since, for example, the radial distortion cannot be modeled accurately. Radial distortion is more pronounced further away from the focal point of the camera and these are the regions that are usually not seen by other cameras.

Numerous calibration methods are available and some achieve more accurate calibration parameters. Some, however, require extensive setup and mechanically estimate the location of cameras with the help sophisticated measuring devices. The calibration of the system could potentially be performed numerous times and a simple flexible calibration process was required. This enables the system setup to be quick as needed for on-site system calibration.

3.2.1 Intrinsic Calibration Parameters

To calculate the intrinsic parameters, a series of checkerboard pattern images are taken for each camera independently. The series of images taken must cover most of the field of view of the cameras to accurately model the radial distortion and to have a complete solution when solving for the parameters.

Checkerboard patterns are used because of the simplicity in finding accurate points. The points that are used to calculate the various parameters are the corners of the black squares found in the pattern. In a checkerboard pattern every black square is surrounded by white squares. This creates a high level of contrast and black corners can be easily extracted. Furthermore, subpixel resolution can be used to enhance the accuracy of the matches. This can be accomplished by doing linear interpolation on the pixels that are identified as being part of a black corner. The points are extracted in a particular order so their relative order is always known as demonstrated in Figure 3.2.

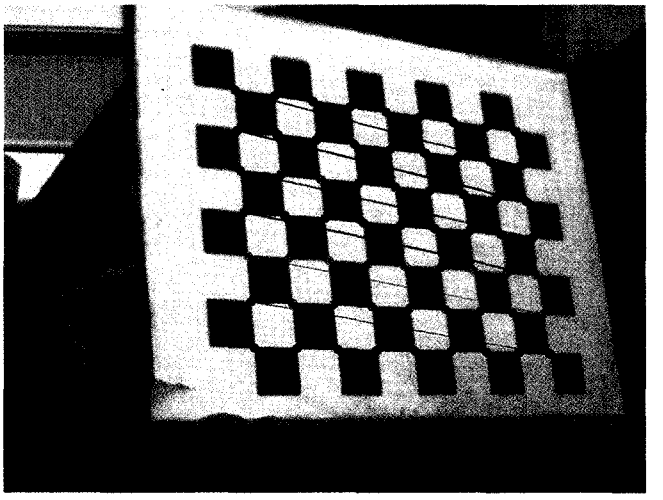


Figure 3.2 - Analyzed checkerboard pattern.

The list of extracted corners from the checkerboard pattern is passed to the Zhang's calibration algorithm to get the intrinsic parameters [1]. The intrinsic parameters calculated are the focal length of the lens, the size of the pixels on the CCD array of the camera and the principal point of the CCD array. The radial distortion is also estimated and used to rectify the images prior to the image processing step.

3.2.2 Extrinsic Calibration Parameters

With accurate intrinsic calibration parameters, the extrinsic parameters of the cameras can be calculated. These parameters are composed of a translation T and a rotation R between the two cameras.

In a stereo rig, the position of one camera with respect to the second camera is needed for reconstruction and not their absolute position with respect to a global reference frame. This assumption is used to simplify the stereo calibration process.

The following relationship can be defined for the left and right cameras with respect to a global reference frame mapping the 3D point P_W.

$$
\begin{aligned}
P_L &= R_L P_W + T_L \\
P_R &= R_R P_W + T_R
\end{aligned}
\tag{3.1}
$$

where P_L and P_R are the point P_W with respect to the left and right camera reference frames. R_L and R_R are the homogeneous rotations between the camera reference frames and the global reference frame while T_L and T_R are the translations.

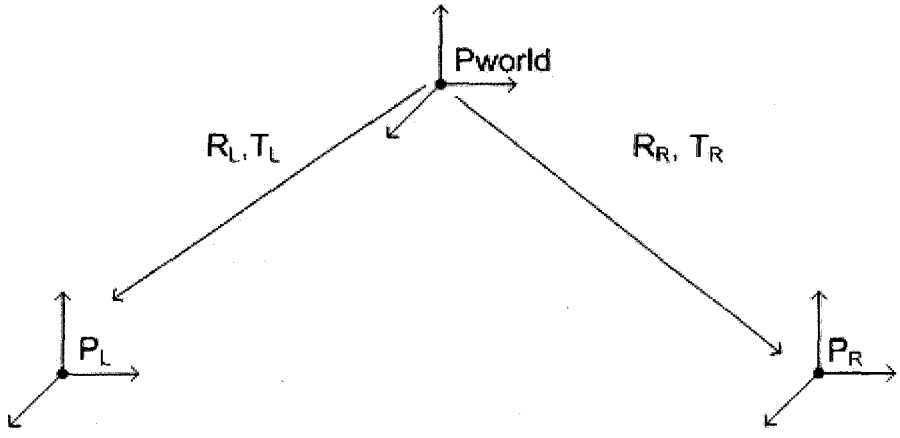


Figure 3.3 – World point to camera reference frame transformation.

With these equations, a relationship can be defined between the two cameras since the relations between the camera reference frames and the global reference frame are known. Assuming that P_W is defined with respect to the right camera reference frame

$$
P_W = R_R^T (P_R - T_R) \tag{3.2}
$$

the following relationship can be derived:

$$
\begin{aligned}
P_L &= R_L P_W + T_L \\
P_L &= R_L (R_R^T (P_R - T_R)) + T_L \\
P_L &= R_L R_R^T P_R + T_L - R_L R_R^T T_R
\end{aligned}
\tag{3.3}
$$

Rearranging the last equation, it can be showed that the rotation and the translation between the two cameras are:

$$R = R_L R_R^T$$
$$T = T_L - R_L R_R^T T_R \tag{3.4}$$

The rotation and translation between the two camera reference frames are what is required as extrinsic parameters. These parameters are calculated again with Zhang's algorithm.

The process is similar to that of the intrinsic parameters calibration. A series of checkerboard images are taken at different positions in the scene. An extra step is required to be able to reconstruct 3D points in world coordinates and not simply up to a scaled value. The horizontal and vertical distances between consecutive feature points on the checkerboard must be measured accurately and given to the calibration algorithm as shown in Figure 3.4. The horizontal and vertical lengths of the squares are the same across all squares. The measurements are performed using a ruler. With these distances, the calibration algorithms can calculate a complete set of extrinsic parameters needed to accomplish full reconstruction.

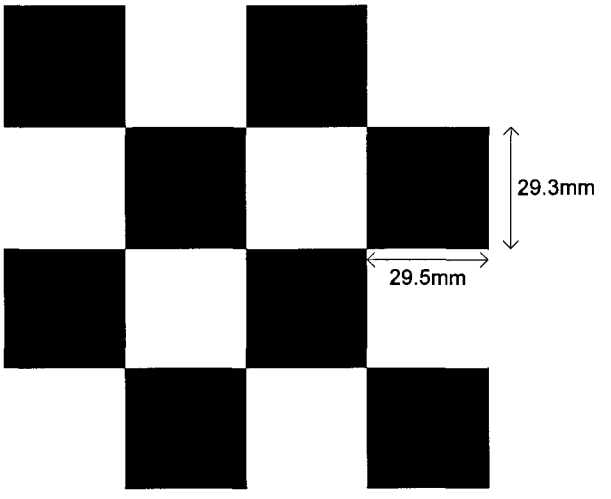


Figure 3.4 – Distance between calibration pattern squares.

The black corners on the checkerboard are extracted and verification is performed to assure that all points are extracted from the images of each camera. Since each camera has a different field of view this verification is required to assure that the points can be accurately matched between the pair of cameras. To facilitate the acquisition of valid images, the system is implemented with audio cues. Two audio sounds are used to help the user, one to indicate a good acquisition and another to indicate an invalid acquisition. This lets the user know when the checkerboard pattern is properly seen by the two cameras without needing to manually verify each acquisition.

OpenCV is used to perform the extraction of the corners [38]. However, the order in which points are extracted is not guaranteed to be identical. OpenCV's implementation does not always properly order the returned corners and that must be handled before calibration. The solution involves comparing the first corner returned with the last corner to determine which corner is in the most top left corner of the image. If the last corner meets this criterion, the list of corners must be reversed, as shown in Figure 3.5.

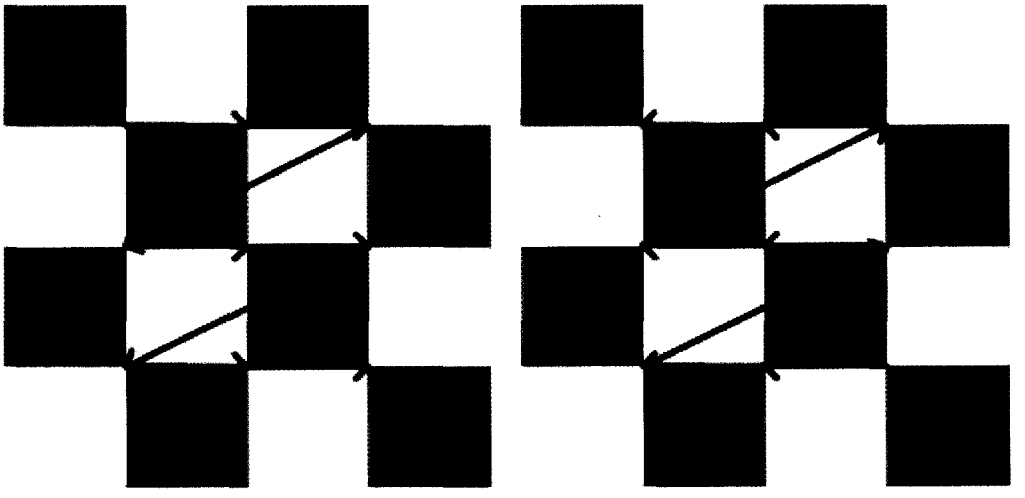


Figure 3.5 – Corner extraction: a) in proper order, b) in reverse order.

The matches from the left and right corners result in 2D pixel coordinates. The calibration algorithm requires that a depth be specified but the depth is not known. An arbitrary value of Z, usually zero or one, can simply be chosen for each pair of pixels

since for each pair of images, all the points lie in the same plane. The pairs of 3D points are finally passed to the calibration algorithm and the extrinsic parameters are calculated. Figure 3.6 demonstrates a series of calibration images taken from the left and right cameras.

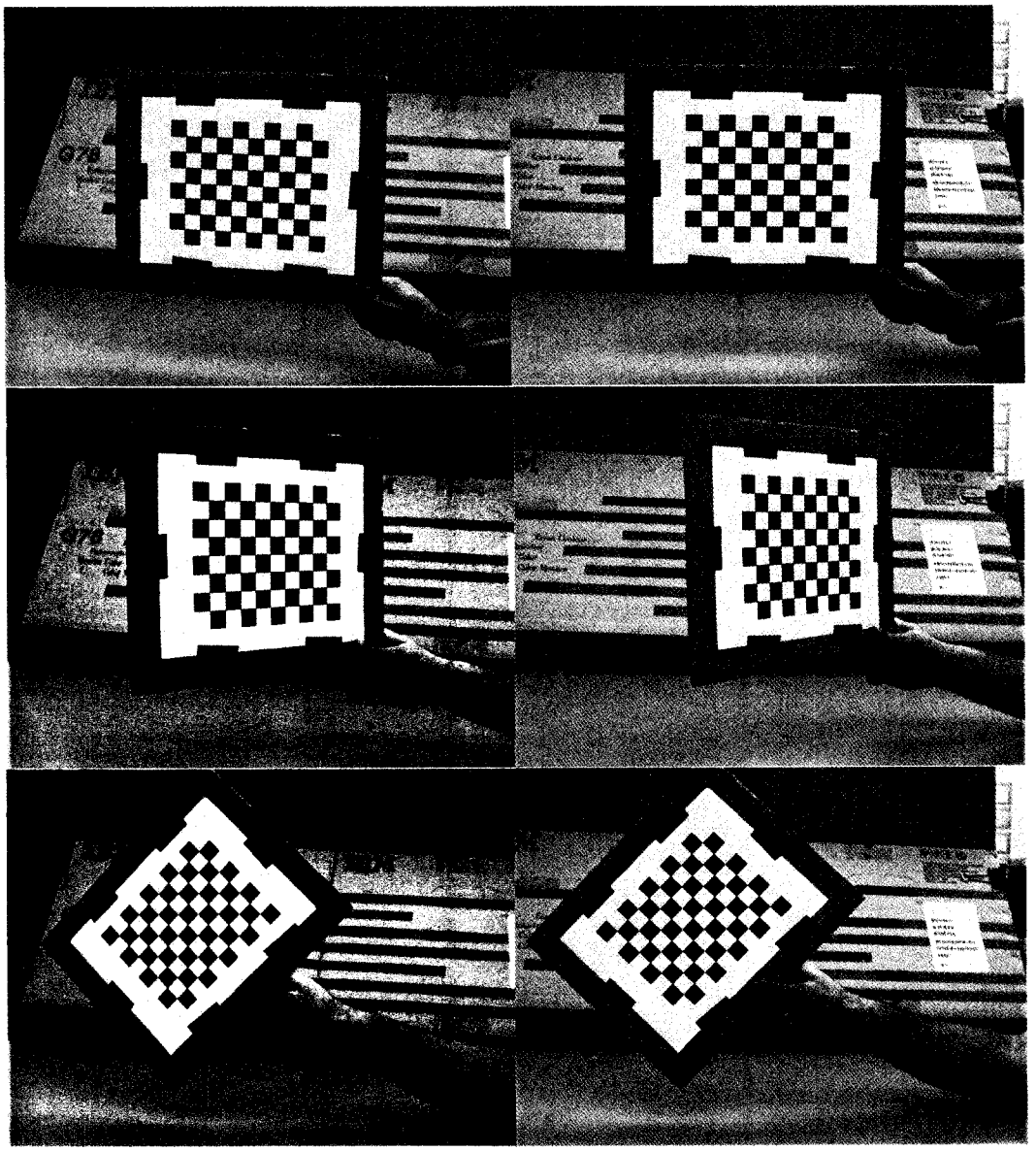


Figure 3.6 – Series of calibration images.

In practice, ten images of the checkerboard pattern are taken at various locations. This translates to 630 calibration points since the checkerboard pattern used consists of *9 x 7* corners hence 63 points per image.

3.3 Pseudo-random Code Construction

As explained in section 2.2.4, the chosen pattern in this work is a bi-dimensional pseudo-random pattern projected under the form of a grid of square color regions such that color patches are easily created on the surface with relatively high resolution using a standard LCD projector. Square color regions have been chosen since they reduce the ambiguity found with projected color bars. They can also be used to create bi-dimensional pseudo-random codes instead of a superposition of two uni-dimensional codes. Pseudo-random codes have been selected due to the simplicity of their generation and acquisition. Pseudo-random codes can easily be extended to have different criteria. For example the Hamming distance between codes in a pseudo-random pattern can easily be modified. Certain rules can also be implemented such as limiting the amount of identical consecutive colors that can appear in the pattern to simplify segmentation of the codes. De Bruijn sequences, however quick they are to generate, do not have this flexibility and require excellent understanding of the mathematical principles to be able to add additional requirements. Even though De Bruijn sequences require a short time to compute, patterns are only generated once offline and hence this is not a concern.

3.3.1 Generation of a Pseudo-random Array

The series of 3x3 codes that compose the pseudo-random array are generated off-line for a selected number of colors, k, following a pseudo-random iterative approach. The implementation is an adaptation of *Morano et al.'s* work [19]. The goal is to obtain a complete grid of square color regions where every 3x3 sub-array appears at most once in the complete pattern of NxM color elements. Sub-arrays of 3x3 are chosen since they are easier to group once they are extracted from the captured images. Furthermore, 3x3 sub-arrays contain enough information to be able to add extra criteria when they are built. Finally, this size is selected because sub-arrays of larger dimensions (4x4 or 5x5) revealed to create difficulties in the segmentation stage under certain circumstances. This problem arises when any of the color regions are not extracted properly. A missing or mislabel color region might cause the whole sub-code to be inaccurately extracted and

hence causes incorrect reconstructions. Limiting the size of the sub-arrays minimizes the amount of correct color regions that need to be segmented and grouped. Figure 3.7 shows an example of a completed color pseudo-random code in the case where k=3 colors are selected, that is red, green and blue.

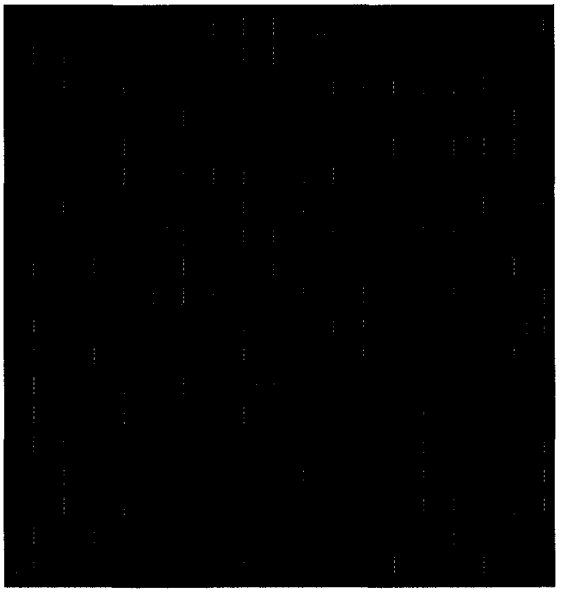


Figure 3.7 – Pseudo-random code example.

In practice, the array generated at this stage does not contain any real color information but simply different identification indexes from 0 to k-1 that represent possible color choices from an arbitrary color palette. These numbers are used in a later step to generate the color pattern that is projected onto the scene. This enables the dynamic substitution of colors to create the color pattern. This way the intensive processing is in the calculation of the initial pseudo-random code and not on the creation of the color pattern that is projected. Separating these two steps enables fast substitution of colors that are appropriate for a given scene. Figure 3.8 gives an example of a generated pattern and two possible color substitutions.

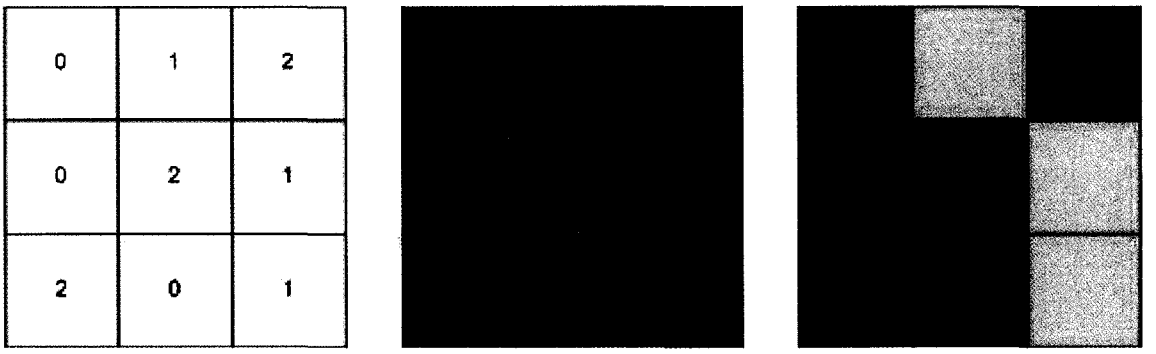


Figure 3.8 – Color substitution.

The pseudo-random grids use only a subset of perfect maps. Perfect maps state that every possible code will and must appear once in the pattern. Pseudo-random codes do not guarantee that all codes are present since it would make the generation of the grids prohibitive. A perfect map of 3x3 independent color-coded regions composed of k possible color elements offers the possibility for defining k^9 different color codes.

To compute the pseudo-random array, elements chosen from palette of size L are progressively added to an initially empty NxM array. The array construction starts off by adding one 3x3 code in the upper left corner. Every element of this 3x3 code is chosen randomly from L. The next step is to fill out the first three rows and then the first three columns of the array. This is done by randomly adding three elements in the current column until the width, M, of the array is reached or in the current row until the height, N, of the array is reached. Every time three new elements are added, a new 3x3 code is created and compared with all previously introduced codes to assure its uniqueness. If the newly created code is identical to a previous code, the three added elements are changed using a different possible permutation of elements. Since there are three elements added at a time, this makes k^3 possible permutations. This process is repeated until the new 3x3 code created with the three new elements is unique or until all possible permutations have been exhausted without creating a new unique code in the array. In the latter case, the procedure is restarted from an empty NxM array.

After this step, a 3xM and a Nx3 border of codes are defined on the top and left-hand side of the array. The remainder of the array is filled out one element at a time. This creates a

new 3x3 code for every entry. Before being accepted, the new code is compared to all existing codes in the array. Again, the list of permutations is kept and all possibilities are tried and compared. In this case, the number of permutations is reduced to k. This iterative process continues until a full array of unique 3x3 codes is created for a given number of k candidate elements. This process is performed only once for each value of k. The complete process is illustrated in Figure 3.9.

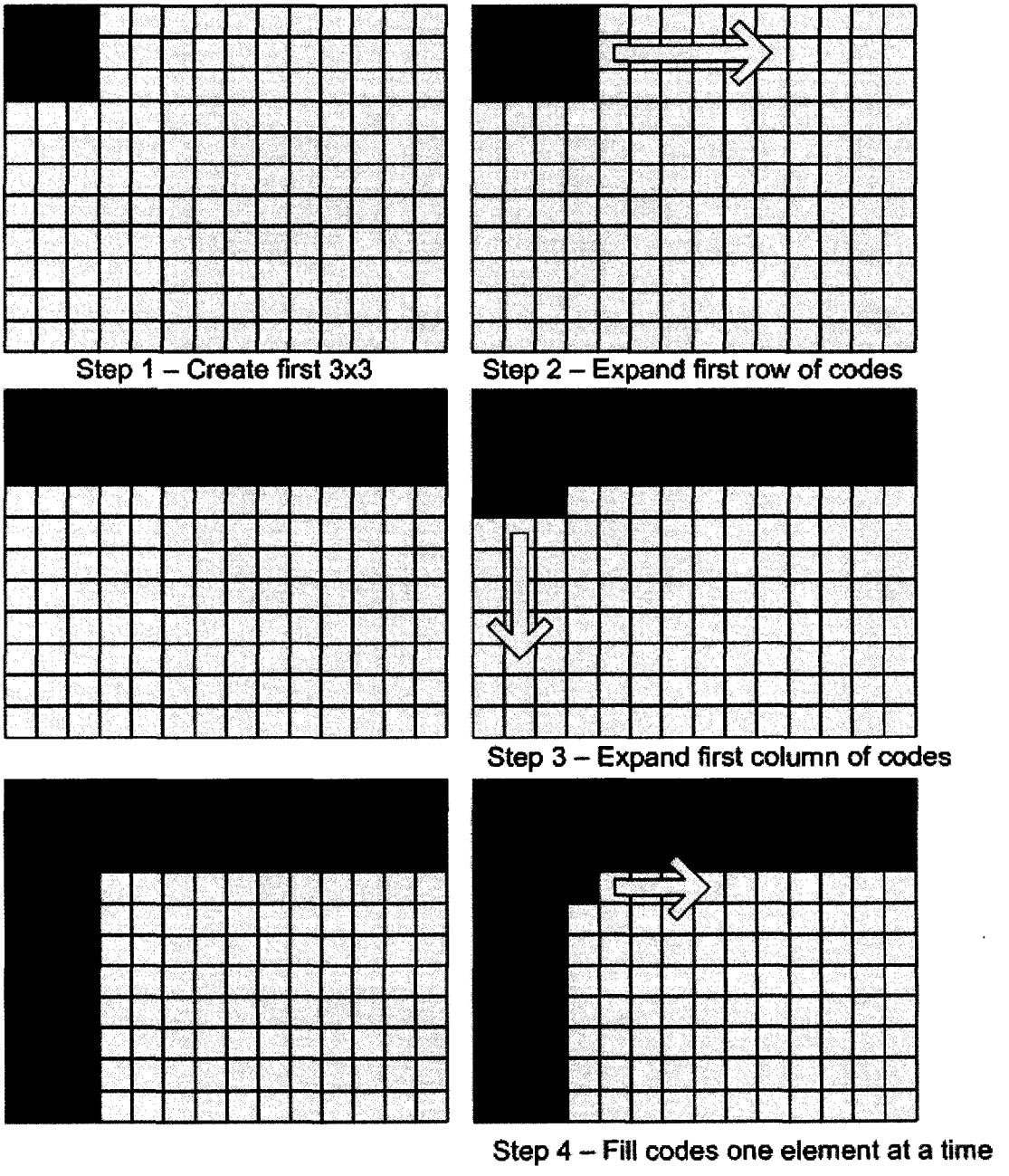


Figure 3.9 – Pseudo-random code creation.

It takes approximately 10 seconds to randomly generate a grid of 55x40 color codes given that only 10% of all possible permutations is used when k=3. This computation time drops drastically when k>3 as the number of necessary permutations is further reduced. Table 3.1 summarizes the percentage of all possible permutations required to create a pseudo-random array of 55x40 codes for different numbers of colors, k, as well as the estimated computation time. Once generated for a given value of k, the pseudo-random array definition remains constant and colors can easily be interchanged without compromising the uniqueness of color codes.

Number of colors (k)	Percentage of possible permutations required	Generation time (sec)
3	10%	10
4	0.8%	0.44
5	0.1%	0.34
6	0.02%	0.33

Table 3.1 – Permutations and time required to compute a 55x40 pseudo-random array with *k* colors.

3.3.2 Pseudo-random Color Pattern Construction

The previous step results in a pseudo-random array that can be used to create a color pattern that is projected upon a scene. The creation of the color pattern needs a color palette to substitute the element indexes with colors and then build a color pattern. Color values are chosen rather than grayscale values since they are easier to segment and contain more information. Color codes contain three levels of information from the sensor which are the intensity of the red, green and blue color components. A grayscale image on the other hand only contains information on the overall intensity. With more information it becomes easier to properly segment codes.

A general color palette is constructed from colors that are far apart in the hue space. This is important since colors are later segmented by using hue histograms and the overlap between colors needs to be minimized. This list is composed of the following six colors specifically chosen to meet this criterion: red, green, blue, magenta, aqua and yellow. The colors are chosen manually based on the segmentation results on a given scene. The

responsiveness of a certain color can be attributed to the different aspects of the scene. Some colors projected will not be reflected as much or the color projected might be a color that is dominant on the scene. The k colors that respond the most are chosen as appropriate and are used when building the color pattern.

Once the ideal k colors are determined the color pseudo-random pattern can be constructed with the pseudo-random array previously computed for k colors. The ideal colors are simply placed in a color array of size k. The pseudo-random pattern uses the indexes of 0 to k-1 and substitutes them by the colors found in the color array. The order of the colors in the color array must be in the order in which they appear in hue space. This assures the segmented codes are matched accurately to the original code which is needed for error detection.

The pseudo-random pattern is composed of NxM elements and is usually quite smaller than the pixel resolution of the projector. This means that there is extra resolution from the projector that can be utilized to create a color pattern that simplifies its segmentation. This is done by using the extra resolution as a sort of buffer zone. Each element can be represented by a color region instead of a single pixel and separated by a border. This is desirable since single projected pixels are almost impossible to accurately capture and segment. Moreover, there is often a difference between the cameras' resolution and the projector's resolution. Occlusions can also easily hide single pixels. Opting for multi-pixel color regions reduces such problems that characterize direct codification approaches as discussed in section 2.2.3. A border zone also simplifies the segmentation of the color regions since it is known that they are separated by a black border. Without a border, the color regions dimensions would need to be estimated for proper segmentation. The various properties of a scanned scene, such as varying depth and sharp inclinations, might result in localized and varying color region sizes which further complicates the segmentation process. Adding a border removes these complications and the segmentation only requires the determination and grouping of color regions.

The elements are projected as color regions of PxQ pixels. They are also separated by columns and rows of n pixels as shown in Figure 3.10.

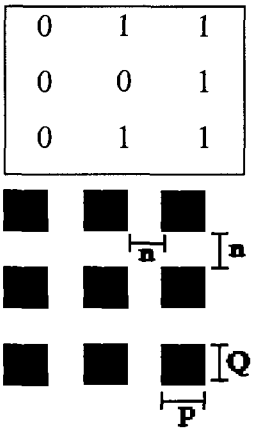


Figure 3.10 - Pseudo-random code with equivalent color pattern with dimensions.

The top matrix represents the results after building the pseudo-random array and the bottom image represents the color pattern built by assigning colors to the pseudo-random array elements. In this specific case red is associated to 0 and green to 1. The choice of P, Q and n varies depending on the resolutions of the projector and the cameras. The focal length of the lenses will also affect these parameters since they will change the size of the color regions that are captured. In essence, the color regions need to be clearly identifiable when they are captured by the cameras. The size of the pseudo-random pattern impacts the choice of P, Q and n since the color regions can only use the available extra resolution of the projector. If the color codes use more resolution than there is available, some color codes will be lost. The amount of pseudo-random codes in the array can simply be reduced if this is the case.

There is a dilemma between the amount of pseudo-random codes created and the choice of P, Q and n. Two circumstances can occur. First, a large array of pseudo-random codes can be created but only a limited resolution is available on the projector. Since a limited amount of extra pixels are available to create color regions and their separating borders, this will make the capture and segmentation of the color regions difficult. The second case occurs when a small number of pseudo-random codes are created and a higher resolution is available on the projector. Large color codes can be created since there is a

lot of extra resolution that can be utilized. However, this is inefficient because the density of the codes will be lowered and multiple images of a scene will need to be taken to acquire the desired resolution. A compromise must be met between the amount of pseudo-random codes that are generated and the size of the color codes to optimize code density and assure a robust capture and segmentation of color codes. For our experiments, a projector with a resolution of 1024x768 was used, P, Q and n, were all set to 9 pixels. A pseudo-random pattern composed of 55x40 color codes was found to be a good compromise.

3.3.3 Color Palette Size

Ideally the palette size should be as small as possible. Minimizing the palette size simplifies the color segmentation and also reduces processing time. However, a small palette size will limit the criteria that can be applied when the pseudo-random code is constructed such as the possible minimum Hamming distance and the size of the map. A small palette size also limits the possible maximum size of the pseudo-random code. Furthermore, a small palette will reduce the possible choice of Hamming distances that could allow erroneous code correction, as discussed in section 2.2.2.1. Therefore the size of the pseudo-random code and the desired Hamming distance in between the codes are inversely interdependent. A large pseudo-random code might only be constructed when the Hamming distance is small (minimum one) and a large Hamming distance might only be achievable for small dimensions of pseudo-random maps.

The chosen color palette size for the system is three. This proved to be an adequate size since the desired pattern size could easily be built and the segmentation of the color codes could still be easily performed.

3.4 Projection and Acquisition

In order to perform structured lighting in an efficient way, synchronization between the projection and acquisition of the pseudo-random color patterns must be achieved. The projection of the pattern is performed using an OpenGL context that is initialized to cover the whole display in Windows XP. The color pattern is saved as an image, copied to the OpenGL color buffer and swapped to the display. OpenGL is used in this case since it assures that the whole resolution of the display device is used without having any unwanted window artifacts projected as for example the windows taskbar. Examples of the projected pattern onto various objects are demonstrated in Figure 3.11 for a RGB pseudo-random color pattern.

Figure 3.11 – Projected pattern on various objects.

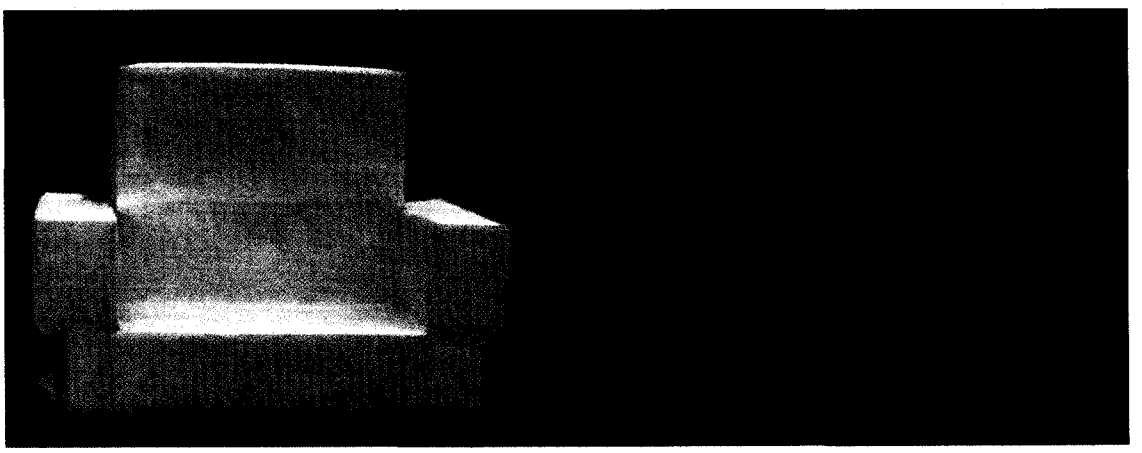
The projection and acquisition modules are coupled together since these two steps are always performed in sequence. This simplifies synchronization and assures that the projected patterns are quickly acquired. The different camera parameters are set in this module. The most important parameters are the gains and the exposure time. There is a gain parameter for each of the color components, red, green and blue. There is also an overall gain parameter that can be set for the cameras. These are set manually and depend on the type of scene being scanned. The correct setting of the exposure time is important since it defines how much of the projected pattern is acquired. When the exposure time is too short for a well lit scene, part of the pattern might not be captured since the ambient light dominates the light being captured. This is demonstrated in Figure 3.12.

Figure 3.12 – Short exposure time.

Long exposure times might cause too much captured light. This will cause the projected codes to merge together on the image plane because of excess bleeding and complicates the correct segmentation of codes as shown in Figure 3.13.

Figure 3.13 – Long exposure time.

At the beginning of every scene capture sequence, a pair of images of the objects is taken without any projected pattern with an exposure time that gives visually appealing images. These images are later used for the color reconstruction of the object. A second pair of images is taken with the same exposure time used to capture the patterns (that is with a setting that produces clear images of the projected pattern), but still without the pattern being projected. The latter images are used to calculate a difference image with the subsequent images taken with the projected pattern on the scene. The idea is taken from dynamic scene analysis algorithms that use background subtraction to isolate dynamic objects from the background [39]. As a result, the difference images are only composed of the projected pattern. This enables the different colors or elements of a scene to be ignored and will not need to be processed. A difference image is demonstrated on the right side of Figure 3.14.



a) b)

Figure 3.14 – a) Intensity image without the projected pattern, b) difference image from the scene.

The projection and acquisition module also performs specified shifts called a *marching* pattern. The same pseudo-random color pattern is projected onto a scene at different locations. More pictures are taken sequentially from the scene with the projected pattern at different locations, hence more codes are projected on the scene and this results in a denser 3D reconstruction of the scene. The pseudo-random color pattern is shifted from left to right by a specified Δx pixels for r increments and from top to bottom by a specified Δy pixels for s increments. The previous parameters can be adjusted to suit the

desired resolution of the reconstruction. The marching pattern algorithm is displayed in Figure 3.15.

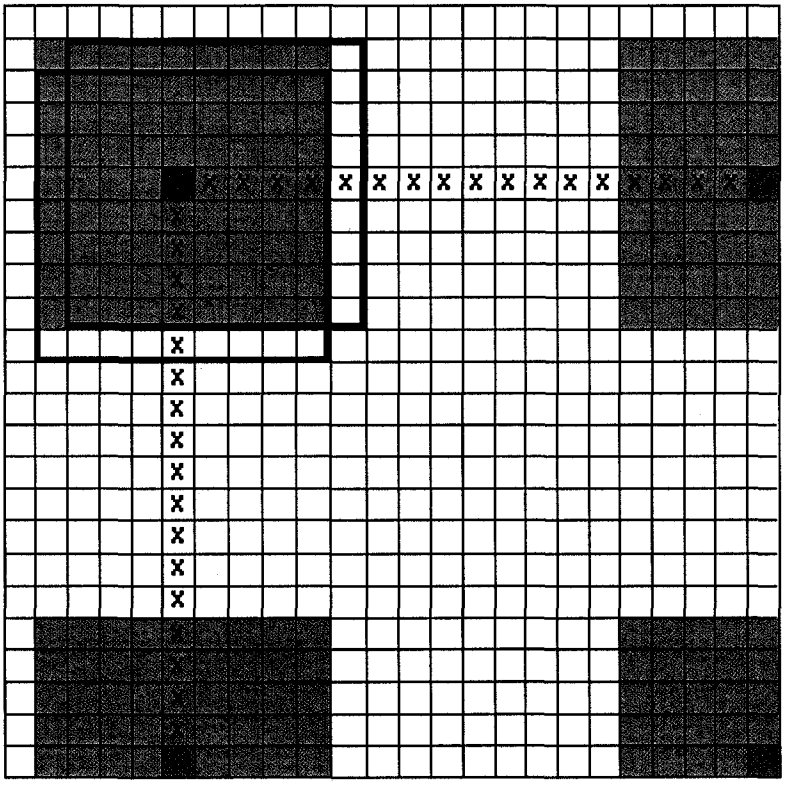


Figure 3.15 - Marching pattern.

The smallest possible shift of the pattern is of one pixel since the display device cannot shift patterns in subpixel resolution. Furthermore, the shifted patterns should not be shifted by more than the distance between the middles of two consecutive color regions. Shifting a pattern by a greater distance will only result in redundant data and a waste of processing time. Figure 3.16 shows an example of the maximum possible shifts on such a marching pattern. In our implementation, color regions contain 9x9 pixels and they are separated by a border of 9 pixels horizontally and vertically. As a result, 18 horizontal and 18 vertical shifts of the pattern are possible, which provides a maximum increase of the resolution by 324 times what is permitted by our initial 55x40 array of codes. Overall, a maximum of 712800 matching points car be reconstructed using the chosen pattern pixel dimensions.

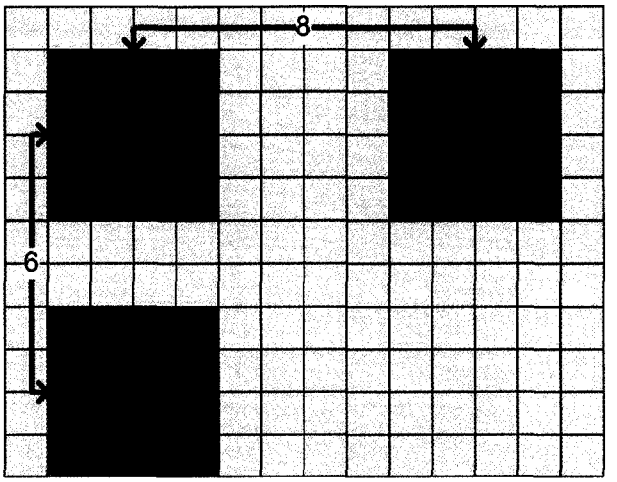


Figure 3.16 – Maximum shift.

The marching pattern algorithm enables a higher density acquisition while keeping the flexibility and advantages of the system. Its best comparison is with grey codes. Grey codes are known for their high density and accuracy. They however need to take multiple images of the scene to gather any data and are not suitable for dynamic scenes. Pseudo-random codes do not intrinsically require multiple images of a scene, but with the marching pattern approach the resolution can be augmented on demand. It represents an original combination of spatial neighboring and time multiplexing structured light techniques. The respective strengths of both techniques are combined in an efficient and integrated way, that is: the speed of acquisition characterizing spatial neighboring approaches, and the increased accuracy of the time-multiplexing strategies that can be used at a variable degree by selecting the desired number of horizontal and vertical shifts of the marching pattern, depending on the acquisition and computation time available for a given application.

3.5 Image Processing

The pseudo-random color pattern is projected on the scene and a calibrated stereo pair of images of the scene is taken. In order to extract feature points, the patterns must be reconstructed and this involves processing the images. Since the pattern is composed of color regions these require segmentation to eventually recover the original pseudo-

random code. The output of the image processing module is a list of color points that will be used to reconstruct the projected pseudo-random codes. The complete image processing algorithm is presented in Figure 3.17.

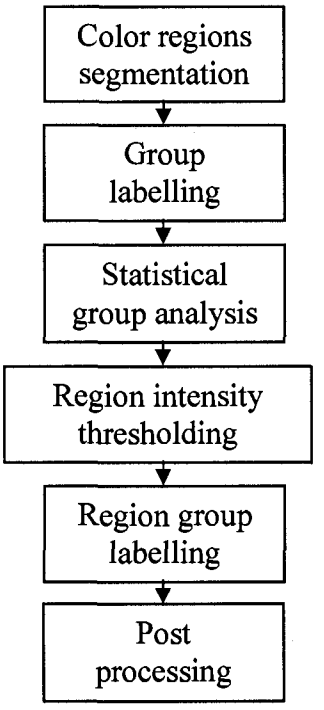


Figure 3.17 - Image processing algorithm.

The extraction of the pseudo-random codes is performed through a series of image processing steps. The first step consists of determining the dominant colors in the image and to isolate the corresponding regions into *n* different images. The second step labels all regions found in each of the *n* segmented images from step 1. The third step analyses the dimensions of each group and either drops small groups or tags groups that are larger than the average region in the image. The fourth step thresholds large groups into smaller groups. The threshold is performed using the average intensity found in the group. The final step labels the intensity threshold image and adds the regions to a group list. The result is *n* lists of groups, one for every major color found, and is then used to recover the pseudo-random codes.

3.5.1 Color Regions Segmentation

Segmentation of the color regions from the background subtraction image is performed in three separate steps over the left and the right images respectively. First, the color regions are segmented from the black background created by the pixel border between color areas. Images are then converted from the RGB to the HSV color space and an intensity histogram is computed on the hue dimension. The hue dimension is used to reduce the amount of parameters required to properly segment an image. With k colors present in the pattern, k peaks are normally formed if the color elements do not overly overlap in the hue space, as shown in Figure 3.18. The hue thresholds are derived by finding the local minima in the intensity histogram.

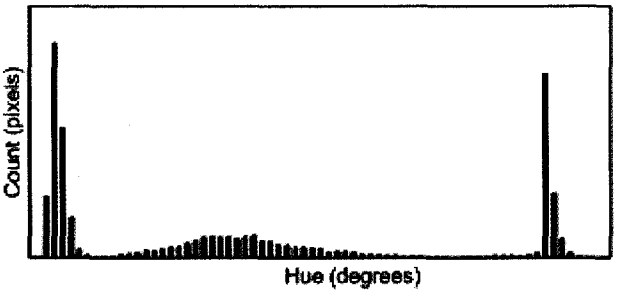
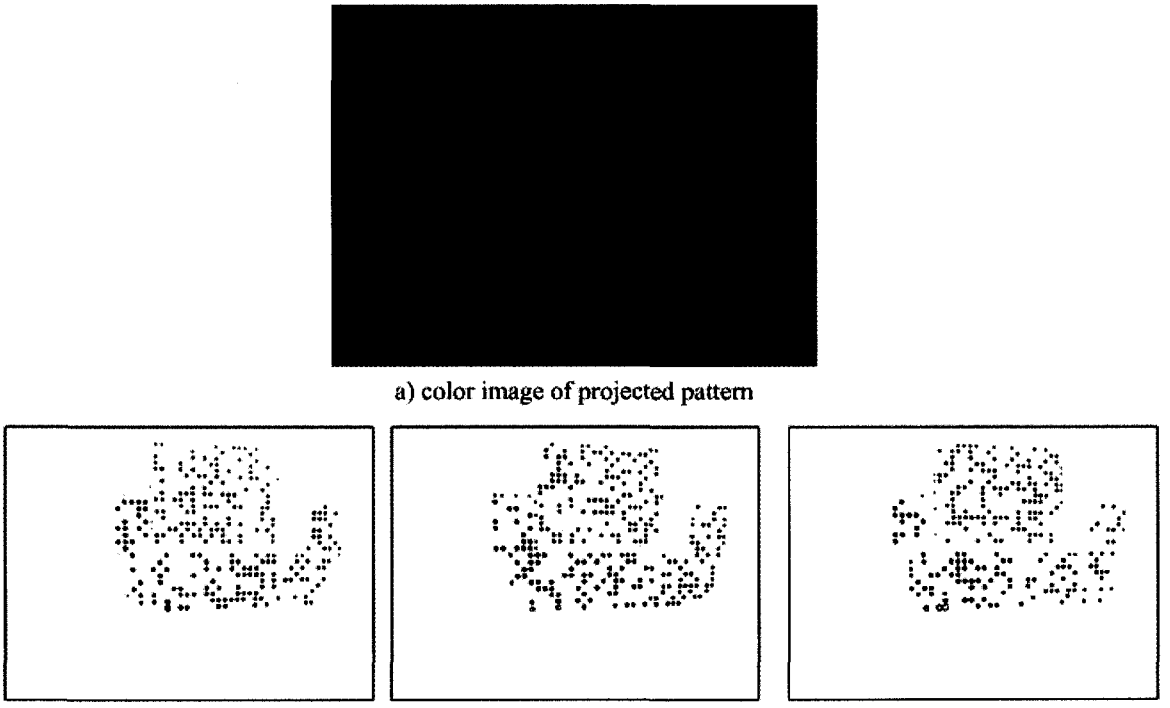


Figure 3.18 - Hue histogram for color segmentation.

A variation of the mode method mentioned by Jain *et al.* is used to find the local minima of the histogram [40]. This method evaluates the peakiness and valleyness of points. This is achieved by finding the highest point in a certain defined subregion. A subregion is used in case some local perturbations occur at a certain point in the histogram. This assures that these points are not mistaken as peaks. There are still some circumstances where some invalid peaks are detected. The algorithm takes advantage of the knowledge that is available about the number of colors, k, present in the projected pattern and assumes that only the tallest peaks are valid. For example, if k colors are projected onto a scene and $k + 3$ peaks are detected, the algorithm will use the k tallest peaks and drop all other peaks. Once the k peaks are found, the lowest points between two consecutive peaks are calculated. These are used as threshold values to segment the color image into k different parts.

The adaptive thresholding is performed on the hue channel and the pixels that fall in the threshold regions are stored in their respective image holder. These image holders only hold the image location of the pixels and not their intensities. The intensities do no need to be saved here because these image holders are used as masks and the intensity values can be retrieved from the original images. This algorithm results in k different binary images representing the different major colors present. An example of the segmentation is presented in Figure 3.19 where three colors are segmented in three different binary images each representing the major colors found in the projected pattern, respectively red, green and blue.



a) color image of projected pattern

b) regions with red pixels

c) regions with green pixels

d) regions with blue pixels

Figure 3.19 – Color segmentation example.

The robust histogram thresholding algorithm that is employed succeeds to properly segment valid regions even on histograms resulting from the mapping of complex surfaces as shown in Figure 3.20 which has for a histogram Figure 3.21. It demonstrates a case where three colors are projected onto the scene but four peaks are found in the histogram. It can be observed on Figure 3.20 that there is a small section, in the lower

right area, where the color codes are captured on a background object. These color codes appear differently in hue space and creates multiple peaks. A smaller peak is detected around the angle 55 on the x axis. The algorithm will consider the peak to be invalid since it is the smallest of the four peaks present and only three peaks are needed.

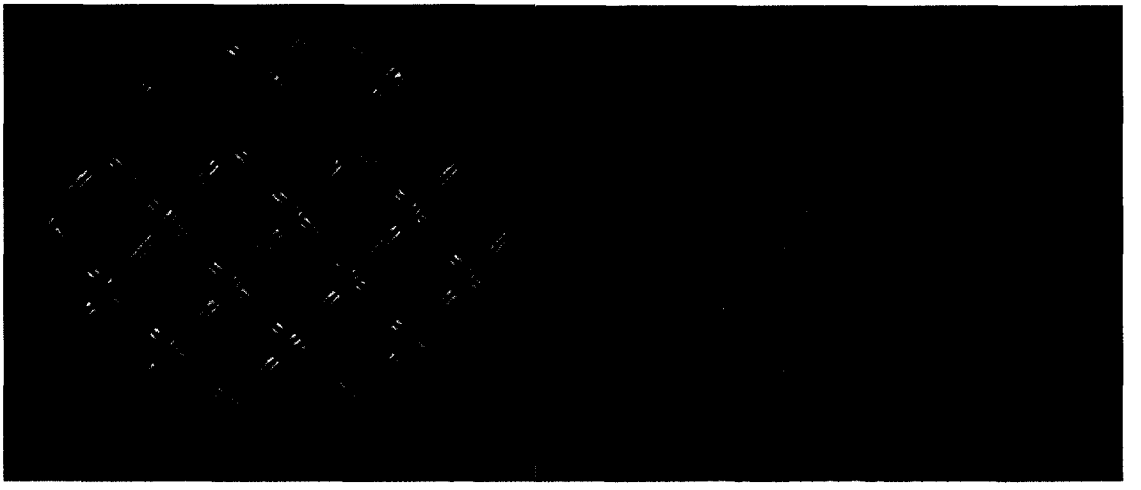


Figure 3.20 – Structured light on an object with a complex surface and reflectance properties.

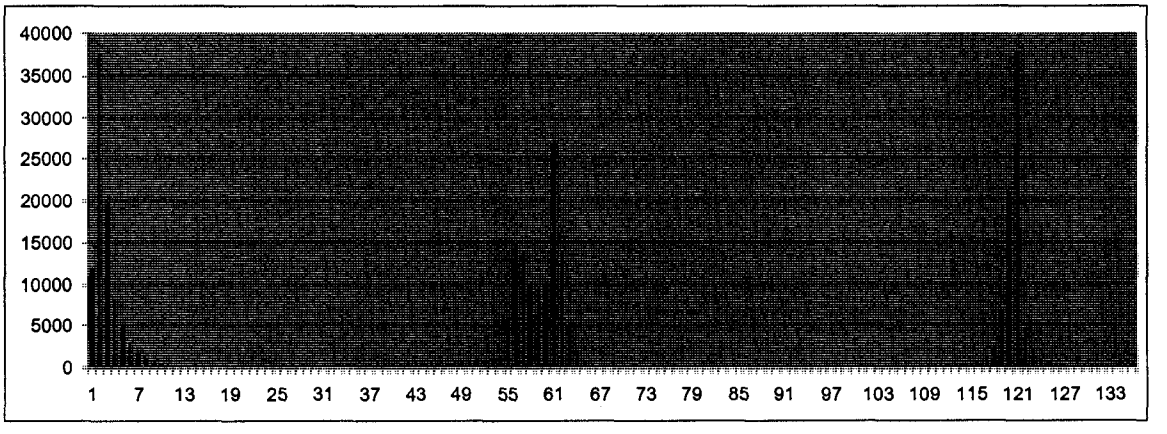


Figure 3.21 – Hue histogram of Figure 3.20 exhibiting 4 peaks for only 3 dominant projected colors.

3.5.2 Group Labeling

The second image processing step applies a labeling algorithm on each of the k mask images to locate the different regions and groups pixels in closed color regions. Labeling

is performed by scanning the mask image following a raster pattern. When a pixel of interest is detected, a new label is assigned only if none of the left, top left, top and top right pixels already have a label. Otherwise, the current pixel takes on their label. In some cases, as shown in Figure 3.22, relabeling is required. Here the top left and top right pixels have different values. The current pixel initially takes on the value of the top left pixel, but in this case it is also the pixel that joins regions 1 and 2. Therefore, the pixels labeled as region 2 are relabeled as region 1.

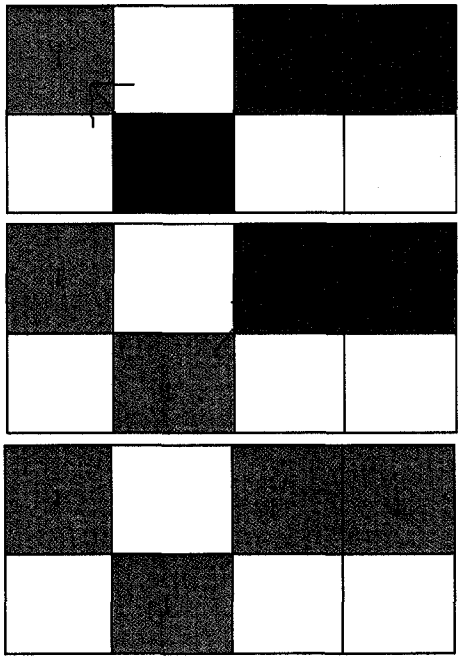


Figure 3.22 - Labeling of color regions.

Another issue arises from the fact that color thresholding is not always fully reliable since the same regions of color could themselves be segmented in different color regions. This occurs because the intensity of the captured regions is not distributed uniformly over the regions. Figure 3.23 demonstrates this observable fact. In this case the green color region tends to have different shades of green and the middle part might sometimes be segmented as shades of yellow while the rest of the region is green. The solution to this problem will be explored in section 3.5.3.

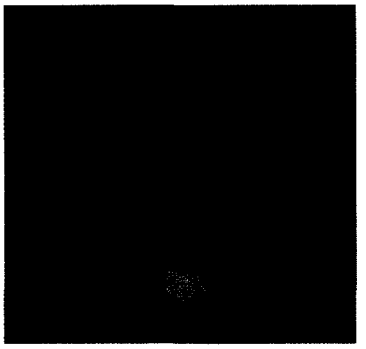


Figure 3.23 - Non-uniformed color distribution over a green region.

The group labeling algorithm labels groups of pixels together so that the groups can be used as color regions. Figure 3.24 demonstrates a series of color pixels. The group labeling algorithm will label these pixels in three groups as follows: group one is composed of pixels 1, 2, 6 and 7, group 2 of pixels 3, 4, 8, 11, 12, 15 and 16 and the last group of pixels 5, 9, 10, 13 and 14.

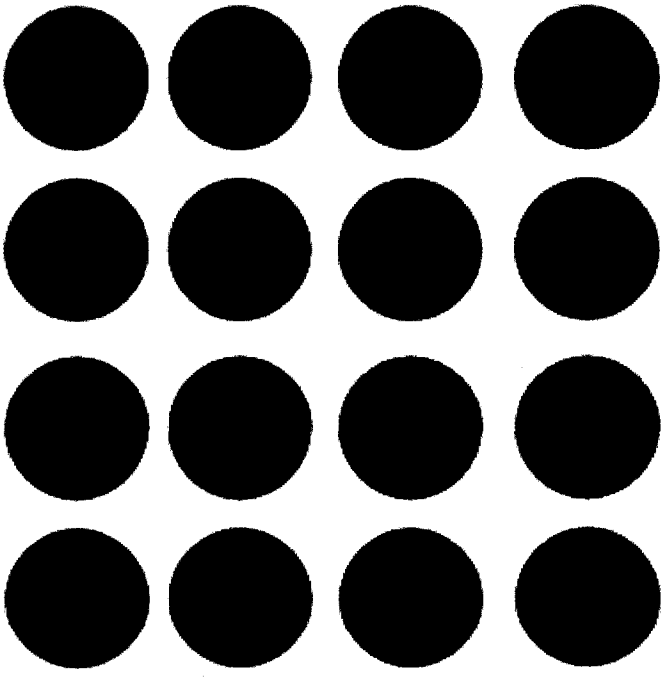


Figure 3.24 – Result from group labeling algorithm.

3.5.3 Statistical Group Analysis

In order to validate the labeled color regions, a statistical evaluation is performed. The average size and standard deviation of the pixel count of every labeled region is calculated.

$$avgSize = \frac{\sum_{i=1}^{N} groupSize_i}{N}$$
$$std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(avgSize - groupSize_i)^2} \tag{3.5}$$

where *groupSize* is the amount of pixels that form a group and N is the total number of groups.

A threshold is applied on the size of the regions. Regions that are larger than the average size plus one standard deviation are further segmented into multiple smaller regions. Regions that are under 5 pixels are considered as noise and are dropped.

As mentioned in section 3.5.2, some color regions can be segmented in two different color masks. This creates holes in one of the color masks and small color regions in other color masks. As a result, when the statistical group analysis is performed, the latter regions are processed as small regions and dropped when in fact they are valid regions.

The problem is solved by filling the holes and dropping the small regions in the other color masks. The pixels in a given region are tracked and a bounding box is calculated. All pixels inside the bounding box are added to the current region. Figure 3.25 demonstrates this procedure.

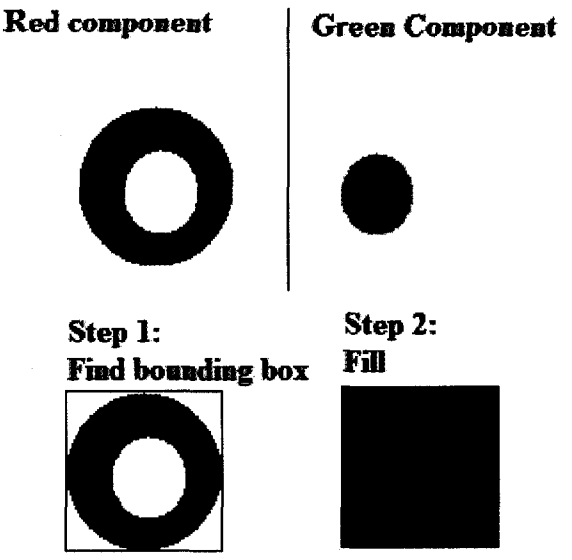


Figure 3.25 – Resolving the occurrence of holes in regions that get segmented in two different color masks.

The top part demonstrates how a color region can be segmented in two different color regions. This result occurs when the center of a color region is brighter than its surroundings. The region that contains a hole is considered to be the correct region while the small region is erroneous.

In the example, the red component might be dropped since it contains fewer pixels than the average. The same goes for the green component. The red component is however the correct color region that should be kept. To keep this region its bounding box is found and a filling operation is performed. Performing the filling will however change the location of the center pixel that is calculated in further steps but the change can be considered to be negligible. The center position change will always be within a couple of pixels from the true center. Furthermore, the optimal reconstruction algorithm performs an optimization on the center pixels and will correct any reasonable errors.

3.5.4 Region Intensity Thresholding and Region Group Relabeling

In some circumstances, two or more distinct regions are labeled as a single region, as shown in Figure 3.26.

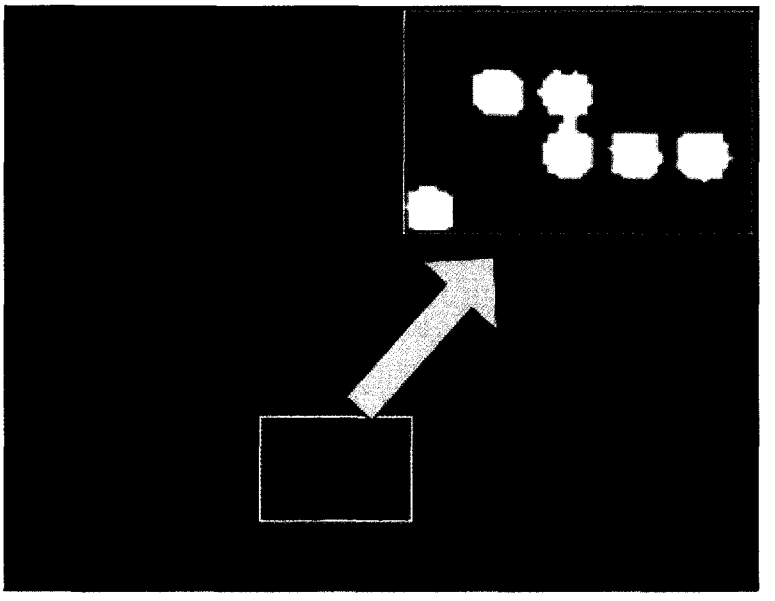


Figure 3.26 - Overlapping color regions in the red mask image.

This occurs mainly when the normal to the surface of some parts of the scene is close to the orientation of the image plane, as well as on highly reflective surfaces. To detect such situations, an assumption is made that all color regions should be approximately of the same size over a given area of the image. Any regions that are significantly larger are then further segmented.

The segmentation is based on the assumption that the intensity tends to be greater at the center of a region compared to its borders as observed in Figure 3.27. The average intensity (extracted from the *V* channel of the HSV color space) and its standard deviation are calculated over the entire area of the merged color regions. The addition of the average intensity and its standard deviation is used as a threshold that is locally applied to eliminate lower intensity pixels, typically surrounding the region, which leaves the desired disconnected regions.

Figure 3.27 – Distribution of intensity in a single color segmented region.

The labeling algorithm is executed a second time on the resulting image section. The initial large region is removed from the list of regions for that particular color. The new groups found are then added to the list. The result is illustrated in Figure 3.28.

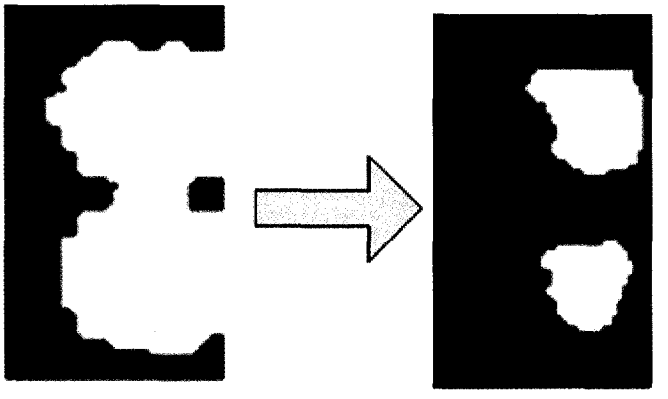


Figure 3.28 – Result of the intensity thresholding and relabeling operations on merged color regions.

3.5.5 Post Processing

A final post processing step is performed on the k segmented mask images. The angle of projection and the angle of the reflection from the object can create two regions of different colors that touch each other. The albedo of the object can also affect the reflection of the pattern to the capture devices. This case may also happen when the color regions segmentation step segments some colors in the wrong color regions. This emphasizes the importance of properly choosing colors that do not overlap in the hue space.

Color regions that are connected to each other create problems when the codes are reconstructed. Since two codes can be close to each other, there is no simple way to determine which code the colors belong to. A simple assumption is performed at this step to alleviate this problem. When two color codes touch it is assumed that the more dominant color group is the correct color. Its dominance is simply established by determining which of the two codes has the most pixels. The color region of the smallest size is dropped from the list of color regions. The center green color region in Figure 3.29 has a small blue color patch touching its right side. In this case it is obvious that the blue color patch is not part of the code and is simply removed since it is the smaller of the two touching regions.

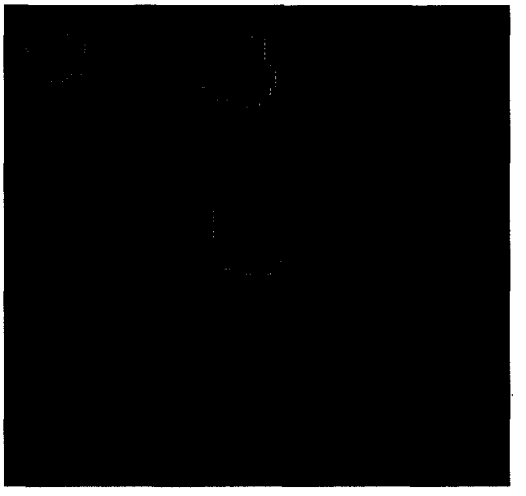


Figure 3.29 - Touching color regions.

3.6 Code Recovery and Validation

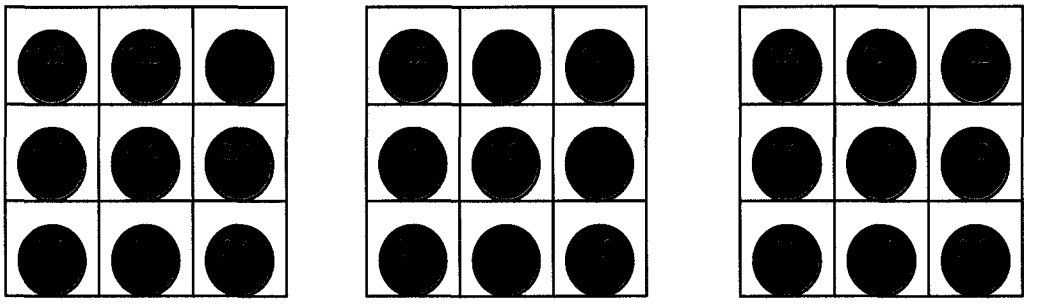
Once the color regions are properly segmented the pseudo-random codes can be recovered. This involves determining every 3x3 code as accurately as possible. This step heavily relies on the accuracy of the segmentation procedure. Any noise that might remain makes the reconstruction process difficult and inaccurate.

Given any color region, its eight closest neighbors are part of the same code. This assumption is only valid when all the color regions of a given code lie on a relatively flat

surface. Uneven surfaces might cause a color region from a different code to be closer than that of the correct color region. However, this circumstance typically occurs only on sharp edges and does not cover a vast part of the object. Furthermore, it can be argued that on these types of surfaces no algorithm can reconstruct the codes properly. However, means are proposed in this section to optimize the capability of code recovery.

3.6.1 Color Regions Ordering

Codes are recognized by traversing the list of color regions throughout the k color image masks and the nine nearest points of a given color region are saved. The nearest points are determined by calculating the square distance between the current color region and all remaining color regions. The nine color regions with the smallest square distance are determined as being part of a code. The result of this procedure is shown in Figure 3.30a. Since the nine closest color regions are not ordered they will not be in the correct spatial location. The correct location of each color code is indicated inside the color region by row and column respectively in Figure 3.30. To determine the proper code, these color regions need to be placed properly in spatial order. Two sorts are performed on the list. Initially, the color regions are sorted by vertical positions. This arranges the color elements so that the first three elements belong to the first row, the next three to the middle row and the last three to the last row as shown in Figure 3.30b. The second sort is performed horizontally on each row. This places the elements on each row in the correct horizontal position and results in the correct representation of a pseudo-random code at a given location as shown in Figure 3.30c.



a) Original order **b) After vertical sort** **c) After horizontal sort**

Figure 3.30 - Sorting of gathered color regions.

3.6.2 Code Validation and Confidence Map

Following the color regions ordering, 3x3 color codes can be recovered from separate color regions. However, in spite of a clear projection of the bi-dimensional pseudo-random pattern and robust image segmentation, some erroneous codes can still be detected. To alleviate this problem and minimize the number of false stereoscopic matches, a confidence map is computed for every detected code.

A code, C(x), is composed of nine elements, that is, a central color region and its eight neighbor color regions. A search for the eight spatially closest color regions is performed over the set of k segmented images. Every element in a 3x3 color code, besides the middle element, is part of another 3x3 code. This means that a given code will also have eight neighboring codes containing one of its elements, as shown in Figure 3.31, except on the borders of the pseudo-random pattern. These neighboring codes are extracted from the captured image and compared to the original pattern, which is known a priori.

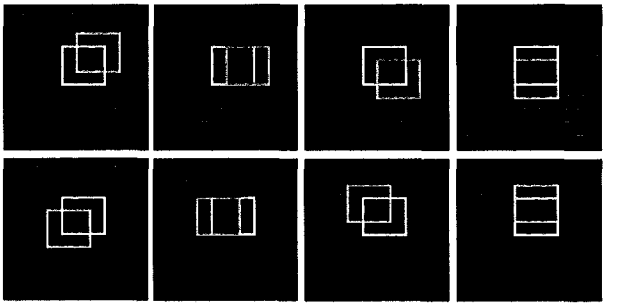


Figure 3.31 – Extraction of 8 neighboring codes for validation with a confidence map.

The number of neighboring codes that can locally be matched between the image and the known original map that was projected defines the confidence, S(x), attributed to the given code, C(x). Mathematically, the level of confidence for a code is defined as follows:

$$S(x) = \sum_{i=x-1}^{x+1} \sum_{j=x-1}^{x+1} [C_{image}(i,j) \wedge C_{original}(i,j)] \tag{3.6}$$

where ^ represents the intersection logical operator.

Ideally, all of the nine codes should match and the maximum confidence level of S(x) = 9 is reached for the central color code, C(x). But due to occlusions, changes in depth, borders, errors in acquisition and segmentation, a perfect confidence level is not always achieved even if the matches between C(x) and C'(x) from the left and right images are in fact correct.

The first step in code validation is to go through the list of gathered codes and identify the codes that are not present in the original projected codes. The identified codes are dropped from further processing. A second pass is done through the list to identify if any duplication of codes is present. The duplicate codes are then compared by there confidence level and the code with the highest confidence level are kept and the other is dropped. This enables an easy validation of invalid codes and reduces the amount of possible outliers in the reconstruction.

The system is build in such a way where code correction could be performed on codes that are deemed to be non existent compared to the original list of projected codes. These codes would usually contain some level of error and depending on the chosen Hamming distance between codes, code correction could be performed. The code correction can only be performed on a pseudo-random pattern that has a Hamming distance of at least three or more in between all valid codes present in the pattern. For example, if the pseudo-random pattern has a Hamming distance of three and there is one color that is incorrect in a code it can be determined that one of the 3x3 codes in the complete pseudo-random pattern will have a Hamming distance of one and all other codes will have a greater distance. The color code that has a Hamming distance of one can then be substituted to correct the error. This is demonstrated in Figure 3.32.

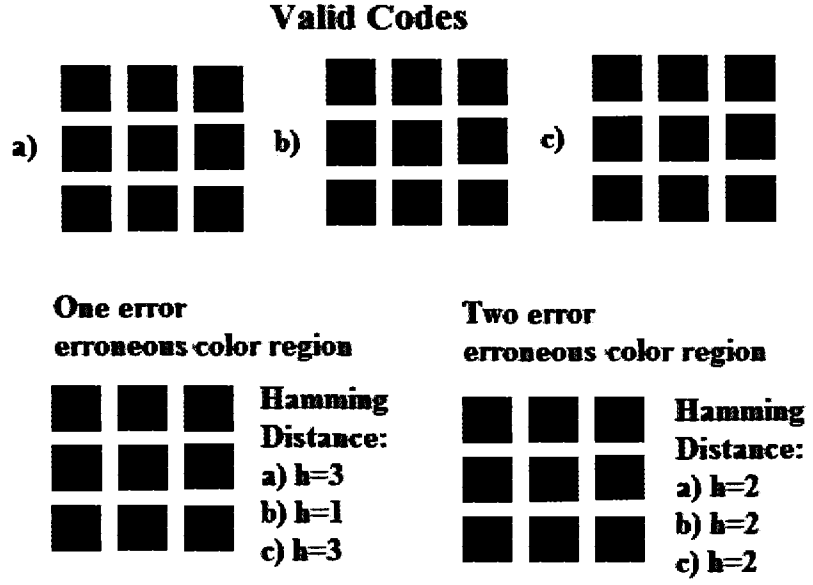


Figure 3.32 – Code error correction with Hamming distance.

At the top of the figure there are three codes that have a Hamming distance of three between each other. The reconstructed code below to the left has a single error. When the Hamming distance of the erroneous code is calculated with respect to the three other codes it can be seen that only one valid code has a Hamming distance with the erroneous code equal to one. Therefore the erroneous code can be repaired to match code *b* in the figure. In this example, if two errors are present, as shown in the bottom right code, it is

only possible to detect the error but it cannot be repaired. The Hamming distance between the erroneous code and all valid codes is two hence there is no way to know what the reconstructed code should be.

With higher Hamming distances between pseudo-random codes included in the projection pattern more errors can be corrected in the reconstructed codes. However, with higher Hamming distances it becomes more difficult to build complete pseudo-random codes given the same size criterion. The system currently does not perform any code correction but the pseudo-random construction module has been implemented in such a way to be able to produce codes with a desired Hamming distance.

3.6.3 Point Extraction

Once the codes have been recognized and validated, the 2D pixel points can be extracted. For any given code, the center color region is used to determine the 2D pixel point. The pixel point is calculated as a centroid from the list of pixels that compose the color group. The centroid is calculated by the following equation:

$$C_x = \frac{\sum_{i=1}^{N} x_i}{N}$$
$$C_y = \frac{\sum_{i=1}^{N} y_i}{N} \tag{3.7}$$

where N is the amount of pixels in a color group, C_x is the x coordinate and C_y the y coordinate of the pixel point.

This calculation is performed for every recognized and validated code.

3.7 Outliers Removal

There are always some circumstances where outliers will persist in the list of 3D points. A commonly used method to further minimize outliers is to use the epipolar geometry of the camera setup. The epipolar lines of two respective pixel matches are calculated for each camera and the distance between the matching pixel points and their epipolar line is calculated. Good matches will result in a minimal distance since the points should both lie near the matching pair of epipolar lines if they represent the same location in space.

The removal of outliers consists of determining a maximum acceptable distance in pixels from the epipolar lines as a threshold. Any matching pair that does not meet this criterion is dropped. In our experimentation, a minimum distance of three pixels has proven to show good consistent results. The algorithm used to perform the threshold consists of an implementation of a Random Sample Consensus (RANSAC) algorithm [41] by OpenCV. This algorithm takes as input a list of points from the left and right image pixel points, the desired level of confidence and the maximum acceptable distance from a point to an epipolar line in pixels. It calculates the fundamental matrix from the list of points using the RANSAC algorithm and outputs all the points that have passed the maximum acceptable distance criteria. The RANSAC algorithm starts by selecting a random subset of points and then performs the following checks:

1. Estimate the fundamental matrix with the subset of points.
2. Compare all other points to the estimated fundamental matrix and add those points to the subset that fall within the maximum distance from the epipolar line.
3. Re-estimate the fundamental matrix with the added points.
4. Calculate the level of confidence of the fundamental matrix.

This is performed for a fixed number of iterations until the desired error measurement is reached. In our imaging system, it proves successful at eliminating outliers and producing smoother 3D models.

3.8 Reconstruction

The list of pixel matches is passed to the 3D reconstruction module once the matches are validated and outliers are removed. The optimal polynomial reconstruction algorithm described in section 2.4.2.2 was implemented and is used to reconstruct the 3D data. The first part of the algorithm converts 2D matches into adjusted estimates. These estimates are calculated based on the best fitting epipolar lines for the two matches. The estimates are the closest points that lie on the best fitting epipolar lines from the original 2D matches. The second part of the algorithm uses the estimated matches and performs reconstruction following a direct linear transform method.

From our experiments, the optimal polynomial reconstruction algorithm has proven to be more stable and reliable than the middle point algorithm described in section 2.4.2.1. However, it does require more processing time compared to the middle point algorithm but since this part of the processing time is negligible compared to the whole system processing time it can still be used with negligible impact. More details about processing performance are provided in section 4.5.

This module produces a list of 3D points along with their corresponding pairs of 2D pixel matches. The 2D pixel matches are saved so that more information can be extracted from the images and used by different systems, including the color information that is useful to map texture on the 3D model.

3.9 Model Coloring

The projection and acquisition modules take specially suited images of a scene for model coloring. Once the reconstruction is done, color information needs to be added to all 3D points. At this stage, the pair of 2D points that were used for 3D reconstruction is still known. With these points, the left and right color images can be queried for there associated RGB color information. The final value of the 3D point's color information can be calculated in multiple ways. The simplest is to use the color of the left of the right image at the associated 2D points. An average could also be performed between the color

values of the left and right color images. Finally, a selected area around the 2D points can be used to calculate averages and variances as supplementary statistical data that might be saved and used in other systems. In this work, the average color intensities are calculated from the original images of the left and right cameras collected without pseudo-random pattern projection.

Once the final color of the 3D point is calculated, it is attached to the 3D point. In the following section, the visualizer will use this color as the color of the rendered vertex. Since a model is composed of multiple 3D points, OpenGL is used to linearly interpolate the colors between adjacent vertices. The interpolation fills the color data between vertices and also fills the color of missing regions. This enables a color reconstruction that is more robust when some 3D points are missing.

3.10 Data Formatting and Visualization

The final step of operation of the system consists of saving the 3D data into different formats that can then be used as the input to various visualization applications. The system saves in three different formats at the time being. The first two formats are in the Virtual Reality Modeling Language (VRML). This was chosen since it is a standard that can be loaded on different systems and platforms. Browser plug-ins are readily available for most popular internet browsers and makes the format quite portable for viewing. VRML is a 3D modeling language that permits display and navigation of 3D data. The first format is a list of 3D points that are placed in space. This gives an easy view of all the points but does not show well the characteristics of the modeled surfaces when only sparse points are recovered.

The second version of the VRML file groups the 3D data points into triangles using a Delaunay algorithm and writes these triangles in VRML script language. The Delaunay implementation used is from Sjaak Priester and is modified to improve performance [42]. The triangulated patches formed by the Delaunay algorithm make it easier to view certain characteristics of the surfaces. However, any remaining outliers worsen the visual quality of the 3D view. To minimize the effect of the remaining outliers, a simple filtering

technique is performed on the size of the triangles. The average size and the standard deviation of triangles are first calculated then all the triangles above the average size plus a standard deviation are dropped. This makes the resulting image more visually appealing and results in a rendering of triangle patches instead of mere 3D points. The visual quality of this step is dependent on the quality of the 3D data. The outliers, at this point, should be mostly dropped to finish with visually pleasing 3D representations.

Finally, the last data format that is saved corresponds to a proprietary format that was developed in our research group. The 3D points are saved along with the average color values calculated from the model coloring module. This data is saved to add extra dimensionality that helps for 3D model visualization and can be used to perform other manipulations, such as automatic inter-view registration [43].

A variation of the proprietary format is also used where the same triangulation data from the VRML format is saved. An OpenGL colored 3D model visualizer is built to view the data. It renders triangles calculated from the Delaunay triangulation along with the colors gathered from each vertex. OpenGL can automatically interpolate colors between vertices which enables a visually consistent model compared to the original images of the scene. It also performs well when some vertices are lost by linearly interpolating the missing color information. Finally, even with low resolution scans of a scene its color reconstruction can be visually appealing because of the interpolation. Examples presented in Chapter 4 are produced with software.

Chapter 4 Experimental Validation

The methods and algorithms chosen and developed are integrated to build a physical system to acquire clouds of 3D points from which colored 3D surface models of objects are computed. The following section discusses the physical specifications of the system. Following that, the characteristics of the system are explored such as its speed of acquisition and the adaptive density of points that can be used for the reconstruction using a pseudo-random marching pattern. The sensitivity of the range scanner to surface reflectivity and colors is also evaluated. Experimental results and analysis of performance are provided throughout this chapter.

4.1 Description of the Stereo Structured Lighting System

The stereo rig used for experimentation consists of two Lumenera LU135C color CCD cameras [44] with a 1392x1040 resolution and one Electrohome EPS1024 projector with a resolution of 1024x768, as shown in Figure 4.1. Cameras are selected to have a resolution at least slightly higher than that of the projector to provide a capture of sharp colored regions. Focal lens of the optics, here 8.5 mm, is preferably selected to cover a field of view only slightly larger than the projection to ensure that the pseudo-random pattern covers most part of the image plane.

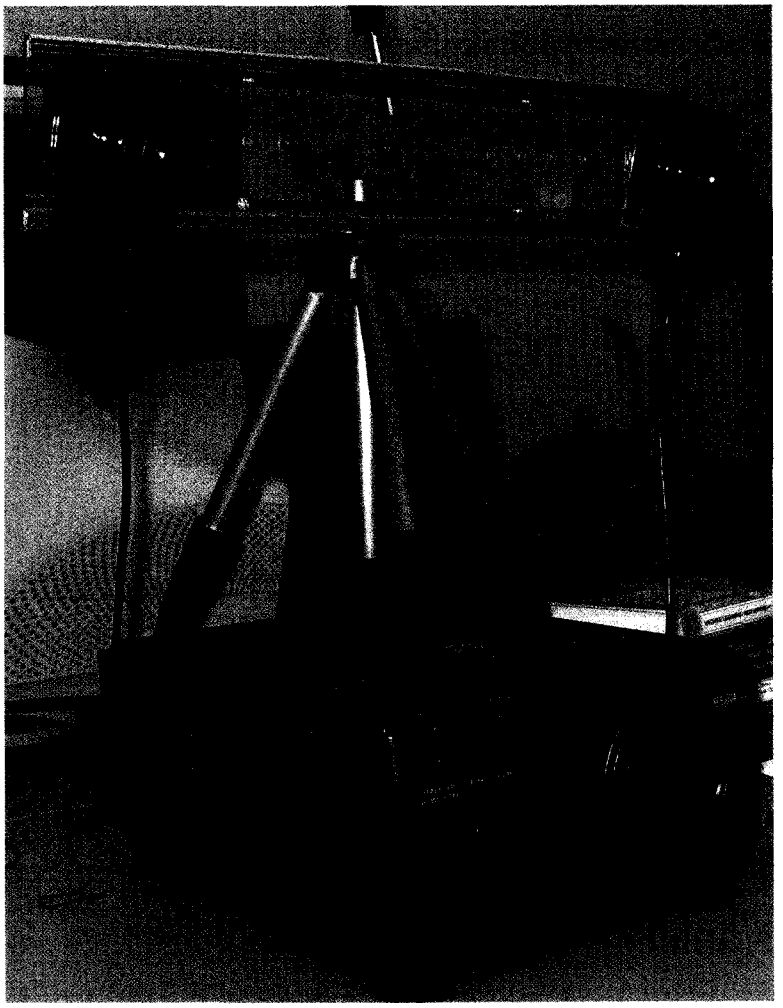


Figure 4.1 – Stereo structured lighting acquisition system.

One advantage of using a stereoscopic approach rather than a classical structured light system with only one camera is that no calibration is required between the projector and the cameras. Given that projectors are more cumbersome than CCD sensors, preserving the registration between the cameras over a long period of time is easier than with the projector. It also gives access to focusing, zooming and brightness adjustment functionalities of the projector to adapt to various operating conditions without influencing the calibration of the system. Moreover, very accurate inter-camera calibration schemes are widely available, while calibrating a projector with a camera still remains more challenging. Furthermore, this setup makes the stereo pair independent from the projector. This adds flexibility to 3D scanning since the projector could always

be placed in the optimal position relative to the stereo camera pair. Figure 4.2 illustrates the system in operation as it projects a pseudo-random pattern over an object.

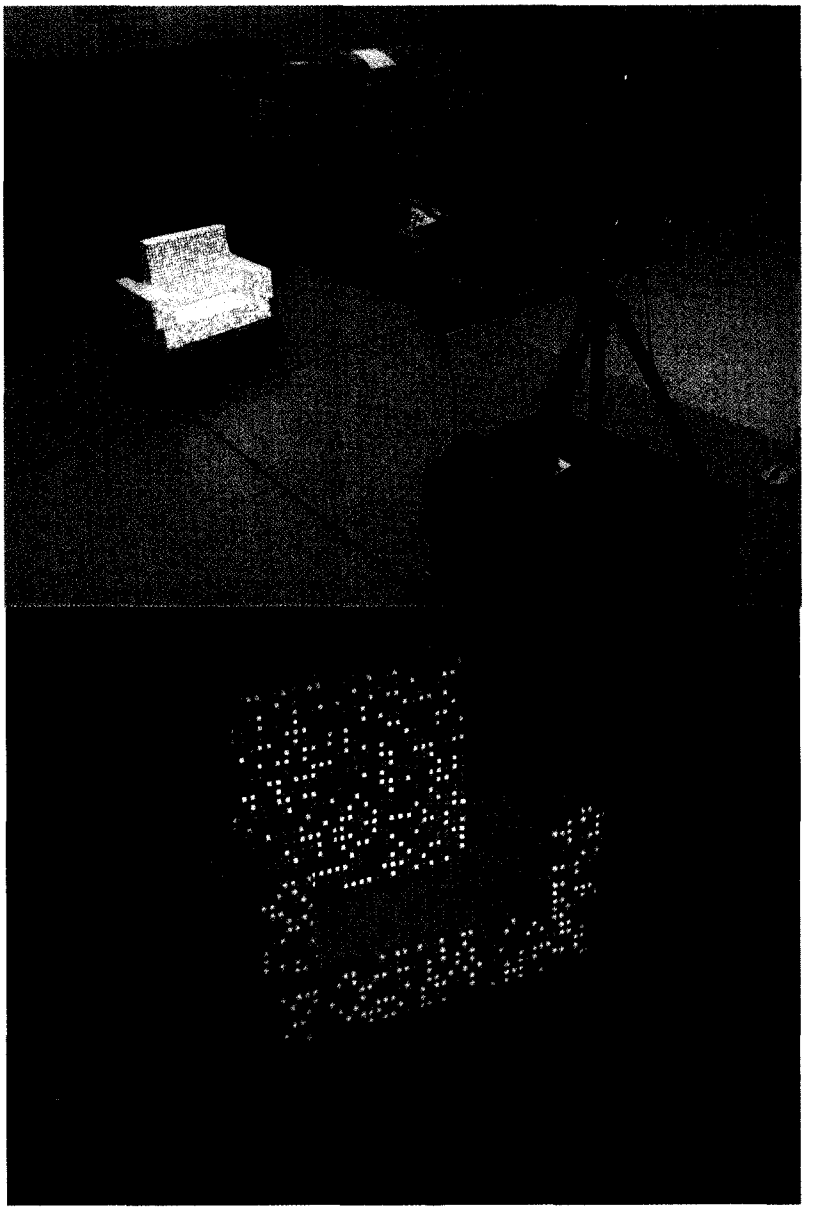


Figure 4.2 – Projected pattern example.

With the available zooming levels and the current power of the projector's light bulb, the minimum achievable depth of the system is 693mm and its maximum depth is 3130mm. The minimum depth is limited by the focus capability of the projector with the current setup. If the projector was to be placed further behind the cameras the minimum depth could be reduced. Two aspects limit the maximum depth of the system. The first being

the power of the bulb used with the projector. There must be enough light that can be projected onto a scene so that the features can be accurately seen at the cameras. The second aspect is the projector's ability to concentrate the pattern on a specific area. Further distances increase the area where the light is projected thus reducing the light intensity on the desired objects. With a projector that can better concentrate light the maximum depth distance could be increased.

The achievable precision of the system using marching patterns is defined as a function of distance. Closer objects can achieve high resolution and further objects allow a lower potential resolution. The highest resolution achievable is when marching patterns are shifted by one pixel and cover the whole scene. The difference in resolution is explained by the projective properties of the projector. A shift of one pixel in the projector's image plane on a near object will result in a smaller incremental distance on the object compared to the same shift of one pixel on an object that is located further away. This is demonstrated in Figure 4.3.

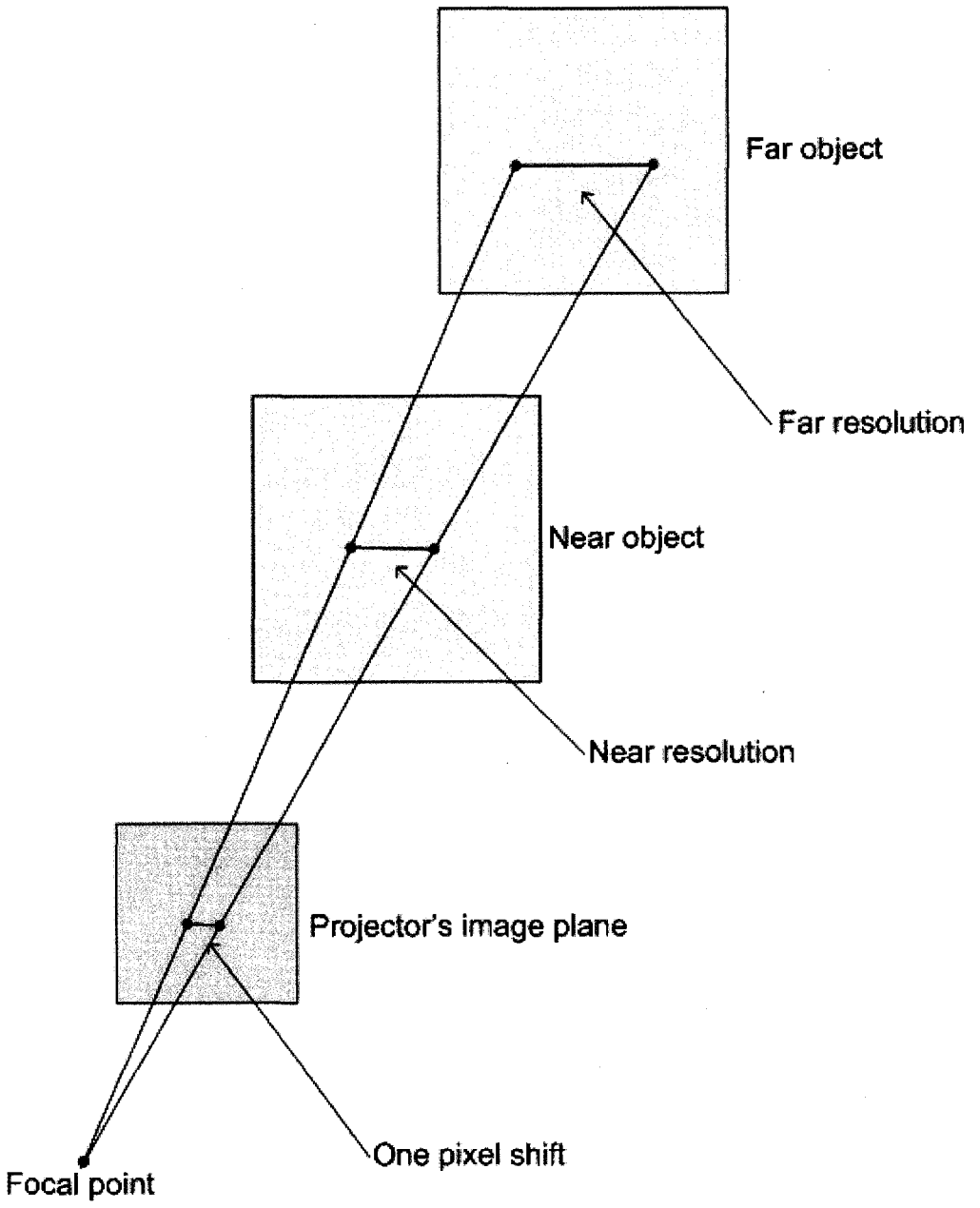


Figure 4.3 – Resolution diagram.

The maximum resolution achievable at the nearest distance is 0.61mm and the maximum resolution achievable at the furthest distance is 1.53mm.

4.2 Calibration Accuracy Evaluation

The calibration method used is an implementation of Zhang's algorithm in the OpenCV library. To demonstrate that the calibration performed on the system is accurate, two tests are conducted. The first test involves scanning a flat surface, calculating the best fitting plane on the reconstructed points and finally calculating the average distance between the points and the plane. The second test involves back projecting reconstructed points onto the image plane and calculating the distance in pixels between the original matches and the back projected matches. In both cases the distance measured should be minimized to demonstrate an accurate calibration.

4.2.1 Best Fitting Plane

The calibration is performed for a particular camera setup and multiple images of a planar surface are taken and reconstructed. A best fitting plane is calculated for a series of different resolutions of data from varying numbers of images taken from the marching patterns algorithm. The best fitting plane is calculated using the method of least squares.

The plane equation is defined as follows:

$$z = a + bx + cy \tag{4.1}$$

where *(x,y,z)* are the axis coordinates of the reference frame in which the plane is located, referred to the acquisition system reference frame, and a, b, and c are the unknowns for the plane equation.

Given a series of points on which a plane needs to be fitted based on the minimization of the error, defined as the squared distance between z_i and the perfect fitting plane, where i is defined from 1 to n, n is the number of points..

$$error = \sum_{i=1}^{n} \left[z_i - \left(a + bx_i + cy_i \right) \right]^2 \tag{4.2}$$

To find the minimum error three partial derivatives are performed on each variable and result in the following [45]:

$$\begin{aligned} \frac{\partial error}{\partial a} &= 2\sum_{i=1}^{n} \left[z_i - (a + bx_i + cy_i) \right] = 0 \\ \frac{\partial error}{\partial b} &= 2\sum_{i=1}^{n} x_i \left[z_i - (a + bx_i + cy_i) \right] = 0 \\ \frac{\partial error}{\partial c} &= 2\sum_{i=1}^{n} y_i \left[z_i - (a + bx_i + cy_i) \right] = 0 \end{aligned} \tag{4.3}$$

Expanding these equations yields three equations with three unknowns that can be solved easily using linear algebra. The solution to these equations yields the equation of the best fitting plane.

$$\begin{aligned} \sum_{i=1}^{n} z_i &= an + b\sum_{i=1}^{n} x_i + c\sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i z_i &= a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i^2 + c\sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i z_i &= a\sum_{i=1}^{n} y_i + b\sum_{i=1}^{n} x_i y_i + c\sum_{i=1}^{n} y_i^2 \end{aligned} \tag{4.4}$$

The average distance is calculated between the best fitting plane and all 3D points.

$$avgDistance = \sum_{i=1}^{n} \left(\frac{a + bx_i + cy_i - z_i}{\sqrt{a^2 + b^2 + 1}} \right) \tag{4.5}$$

where n is the number of points.

Table 4.1 demonstrates that for a planar object at three different depths from the sensor the average distance between the points and a best fitting plane is between 0.00733mm to

0.0507mm. These results should be sufficient in demonstrating the adequacy of the calibration technique.

Number of points	16634	9509	2191
Average depth of the plane to the sensor (mm)	756.57	1810.43	3093.72
Average distance between the points and the plane (mm)	0.00733	0.00435	0.0507

Table 4.1 – Evaluation of calibration accuracy with respect to best fitting plane.

4.2.2 Backprojection Error

Another measurement of calibration accuracy is to calculate the back projection error. This error is calculated by reconstructing 3D points using the calculated calibration followed by a backprojection of those the points onto the image plane using the same calibration parameters. Assuming a good reconstruction algorithm and near ideal matches between the stereo camera pair, the back projected points would have a minimum distance with the original matching points. The same planar object is used from section 4.2.1.

Number of images	1	25	100	324
Number of points	849	21780	84830	274440
Average left camera back projection error (pixels)	1.55	1.54	1.60	1.64
Average right camera back projection error (pixels)	1.64	1.63	1.76	1.84

Table 4.2 – Evaluation of calibration accuracy from backprojection error.

The amount of points reconstructed is below the amount of total pseudo-random codes projected. The reason being that not all the points are seen on the plane due to limitations

of the zooming and focusing parameters of the projector. The average pixel distance between the original matches and the back projected points vary between 1.55 to 1.84 pixels. Considering that the resolution of the cameras at 1392 horizontal pixels by 1040 vertical pixels, this corresponds to a physical error of roughly 0.85mm at a distance of 750 mm where the object is located. As such a backprojection error under 2 pixels can be seen as adequate for this prototype since it is targeting explorative 3D vision and not fine modeling or metrology.

4.3 Experimental Modeling Results

Results obtained on various 3D reconstructions of real objects are demonstrated in this section. Various types of objects have been chosen to demonstrate the overall capabilities of the system. These objects vary in specular properties, surface complexity, color distribution and color intensities. These properties offer a good sample of the types of objects that can be reconstructed and demonstrate how the system fairs on different capturing scenes. The reconstructions will be displayed in two formats. The first format will be as cloud of points and the second as a Delaunay triangulation and its colored reconstruction using linear color interpolation, as described in sections 3.9 and 3.10.

The same pseudo-random pattern is used across on all reconstructions. It is composed of 55 columns and 40 rows and a color palette of three colors which are red, blue and green. Each color region is nine pixels by nine pixels. Nine black pixels also delimit color regions from each other. These dimensions where chosen because they tend to give good results across different type of objects and for objects located at different distances from the stereo setup.

4.3.1 Simple Chair Object

The simplest types of objects that can be scanned are objects that have relatively flat surfaces, no variation in color and no texture. The first object considered is a miniature

chair made of foam and covered with white paper which represents a fairly good lambertian surface, as shown in Figure 4.4.

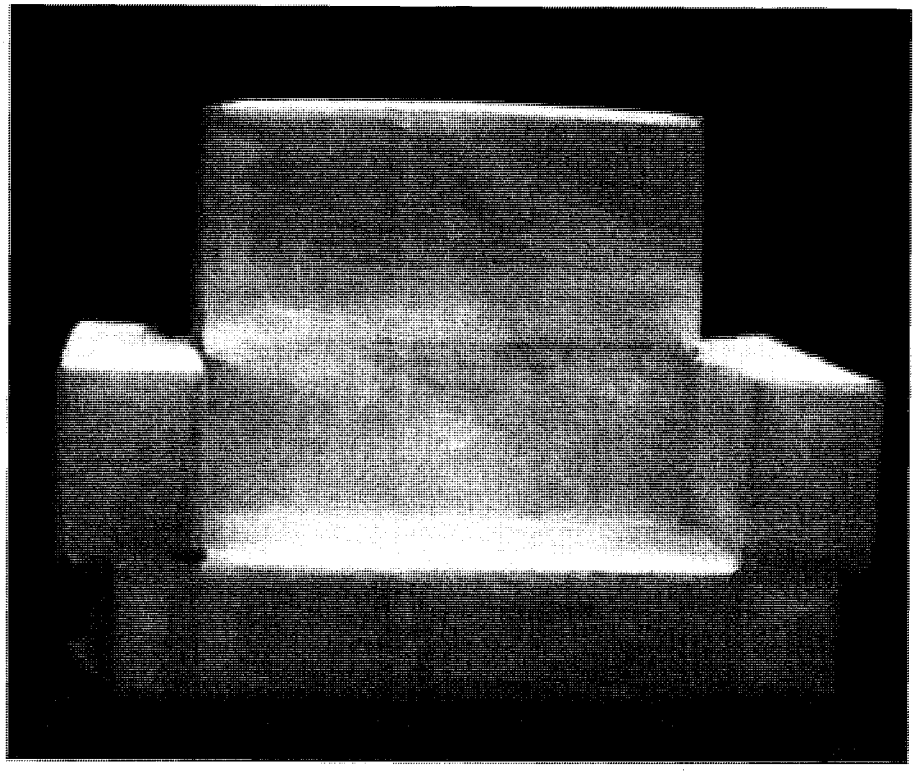


Figure 4.4 - Simple chair.

The projected pattern on the simple chair is demonstrated in Figure 4.5.

Figure 4.5 – Simple chair with projected pattern.

The first test is to reconstruct the simple chair using only one projection. The reconstruction yields 320 valid 3D points. Not all possible points are extracted from the projected pattern since only a part of the pattern is visible on the object. Some color regions are projected on the background surface, while others overlap the sides of the arms of the chair which have an orientation close to the optical axis of the cameras and are not clearly visible on the image planes. Also, in order to retrieve a code to create a matching point, a group of 3x3 color regions must be consistently retrieved. Given the relatively small size of the surfaces on this object, the number of complete codes that can be retrieved is also limited, which directly influences the number of possible matches and 3D reconstructed points available. The reconstruction results are shown in Figure 4.6. We can observe that from the number of 3x3 codes available from the images of the projected pattern, a high percentage are recovered, leading to a 3D point map with few missing data over the planar surfaces.

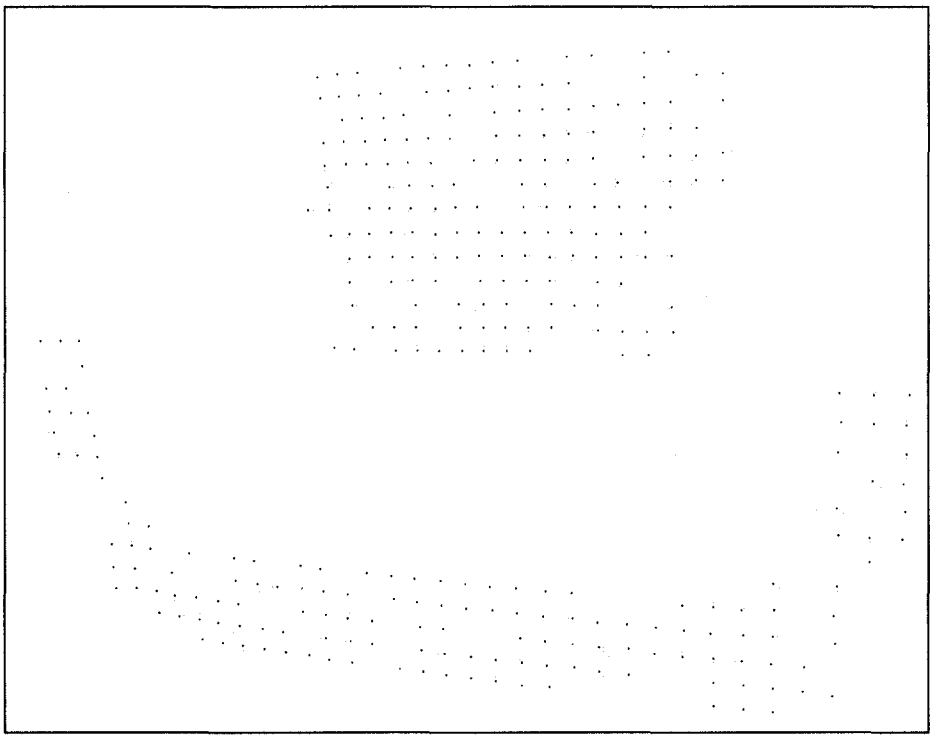


Figure 4.6 – 3D reconstruction of a simple chair model with a single pseudo-random pattern projection.

In order to increase the resolution of the reconstruction, the pseudo-random pattern can be projected multiple times onto a scene using the marching pattern principle. The next result is a reconstruction with 324 pair of images of the scene which is at the highest resolution possible. In this reconstruction there are 115891 reconstructed 3D points. The results are shown in Figure 4.7. We clearly observe a major increase in the density of points, as well as the planarity of surfaces in the model which demonstrates good performance on outlier removal.

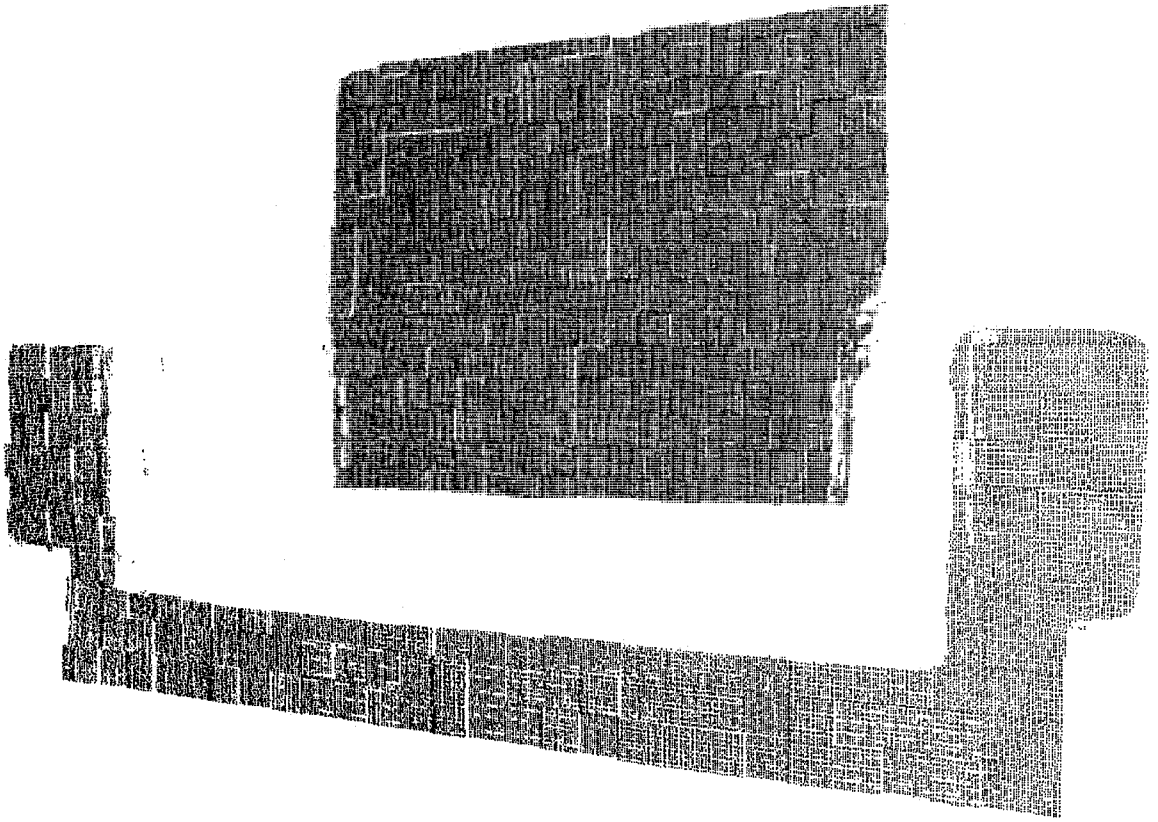


Figure 4.7 - Simple chair reconstruction with 324 marching pseudo-random patterns.

A colored reconstruction of the simple chair is also performed and shown in Figure 4.8. In this form, the reconstruction contains more surfaces that the original point cloud model because not every corner of the object can be captured with the chosen point of view. The triangulation interpolates between distance points surrounding the planar surfaces which creates surface patches that do not actually exist. These create an artificial blurring effect in the transition areas of the model.

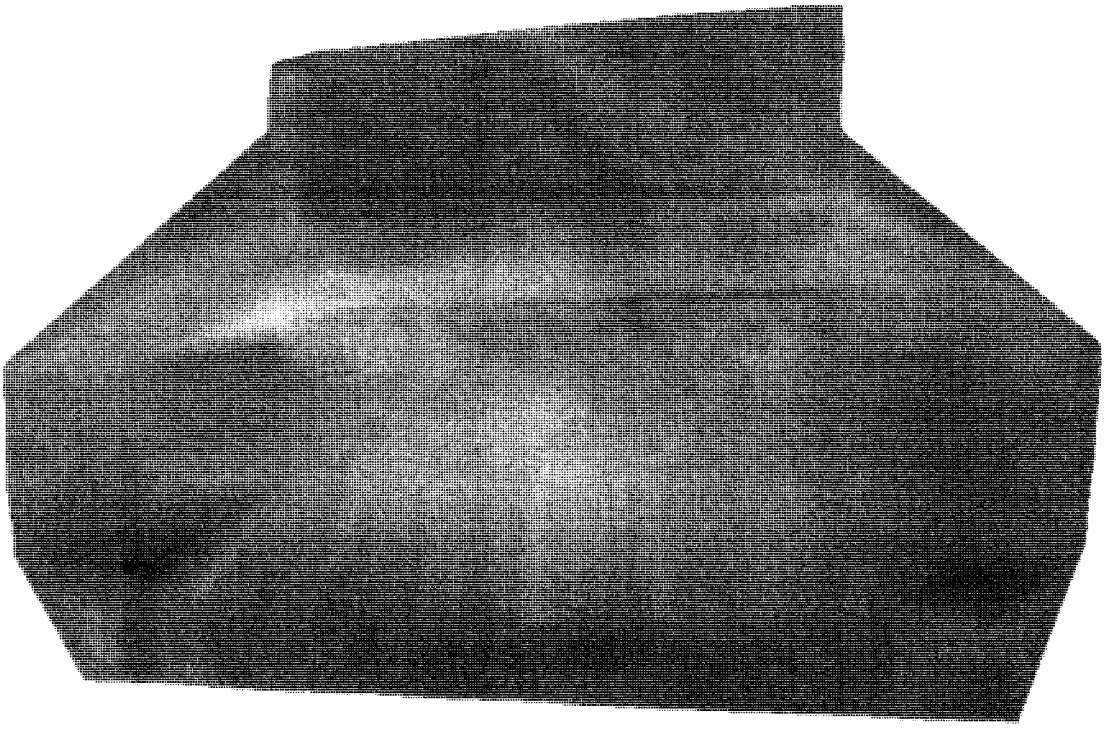


Figure 4.8 – Simple chair colored reconstruction.

In order to validate the scaling of the model, we take advantage of the geometric simplicity of this object to perform a comparison on the dimensions of the chair between the physically measured results and the reconstructed results. The reconstructed dimensions are estimated based on vertex picking functionality available in our OpenGL viewer. The picking function can only estimate the dimensions of the object since picking works by backprojecting the mouse coordinates into the 3D model. To be able to retrieve any coordinates, it is typical to backproject a small surface and not a single pixel. Thus, the OpenGL picking functionality returns a list of points that intersects with the backprojection and these are used to estimate the corners of the object used to calculate its dimensions. The distance is calculated between the returned vertices of two surface corners. This analysis is used as an approximation of the accuracy of the system only but still provides a good evaluation of the scale factor. The measurement can be seen in Figure 4.9.

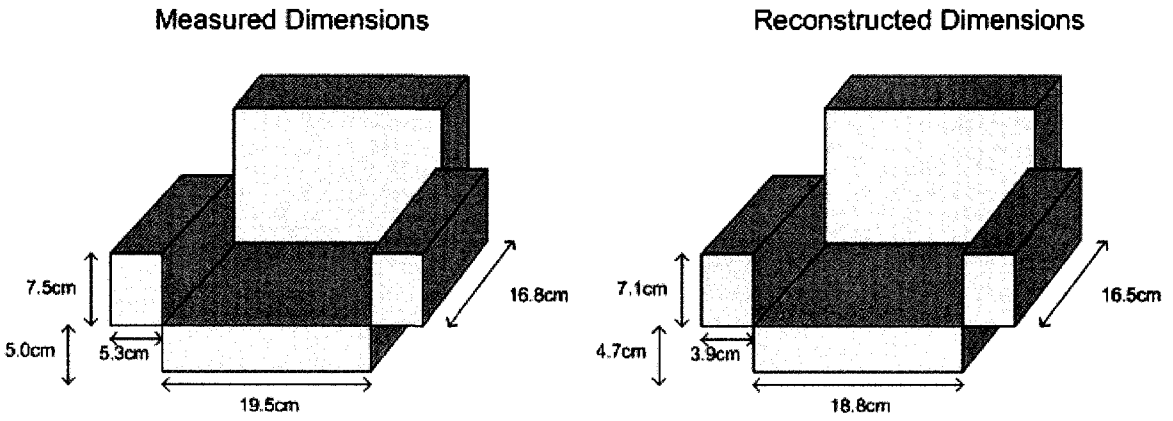


Figure 4.9 – Simple chair dimensions for evaluation of the scale factor in the model.

The differences between the measured and the reconstructed dimensions are contained within 0.5 cm on average when the object is located at about 75 cm in front of the sensor. This demonstrates that the system can reconstruct objects accurately enough for most applications in exploratory vision, such as robot navigation in remote environments.

4.3.2 White Coffee Machine Object

Unlike the simple chair that has relatively flat and lambertian surfaces, the next object features curvatures and a slightly more specular surface. It is considered to evaluate how well the sensor and reconstruction can perform on this type of surface. The chosen object is a typical coffee machine with white paper covering the glass pieces. The white paper is used to create a medium where light is reflected so that the projected patterns are visible. Without the paper the light would simply pass through the glass parts and no information would be gathered on the curved surface which is of interest. However the plastic coffee container is left uncovered which exhibits slightly higher reflectance than the white paper. The tiny lettering on the machine is also of interest to evaluate the capabilities of the model coloring scheme. The image of the modified coffee machine is shown in Figure 4.10 and the coffee machine with the projected pattern is shown in Figure 4.11.

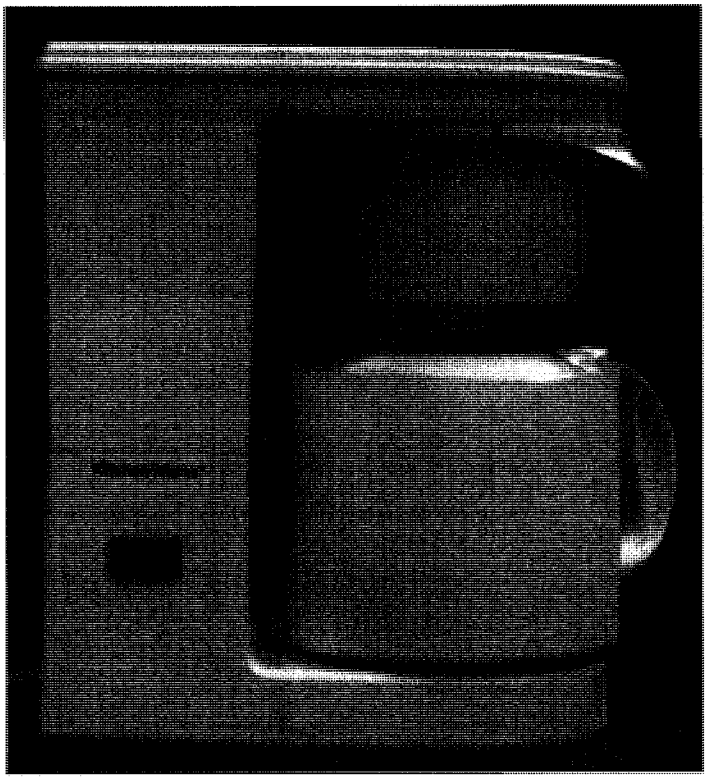


Figure 4.10 - Coffee machine with glass container covered with white paper.

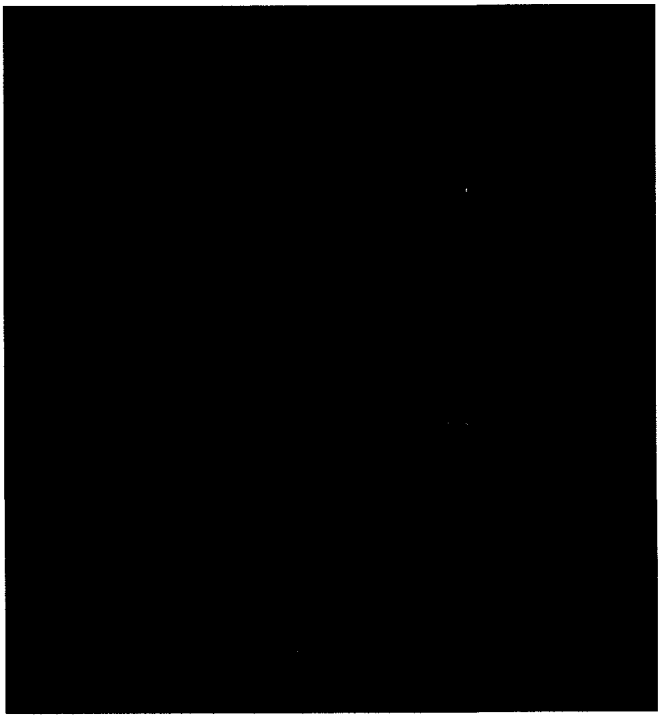


Figure 4.11 – Coffee machine with projected pattern

A full resolution scan composed of 324 pairs of images provided by the marching pseudo-random pattern is performed on the coffee machine and its 3D reconstruction result is shown in Figure 4.12.

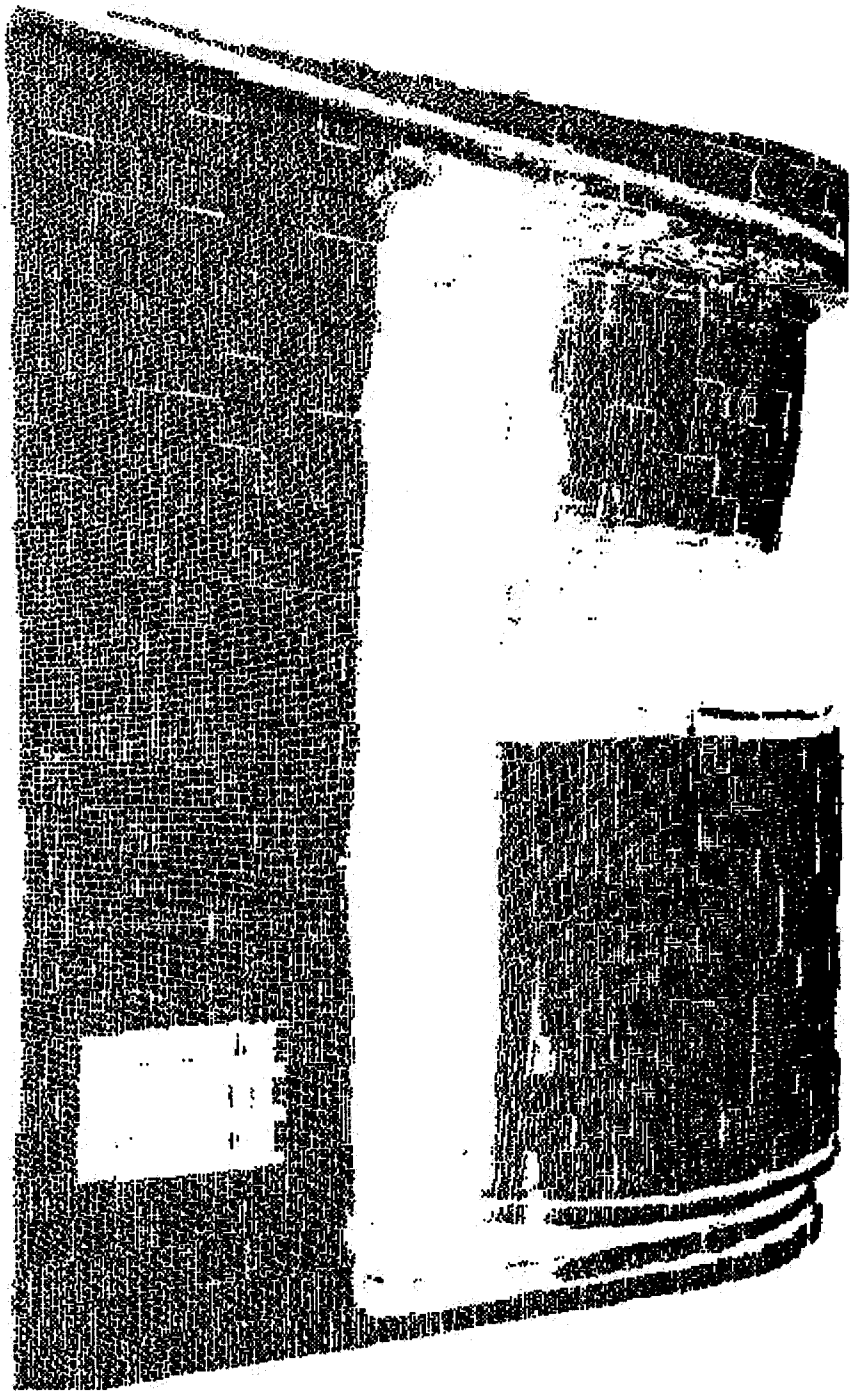


Figure 4.12 - Coffee machine reconstruction with 324 marching pseudo-random patterns.

The coffee maker is equipped with an on/off button on the lower left side that, in this case, is darker than the rest of the objects present in the scene. The exposure time of the cameras was then set so that just enough light is captured on the main sections of the object. When too much light is captured, this causes over saturation in the image and color regions tend to merge together, making segmentation of pseudo-random codes nearly impossible. When too little light is captured, low reflective parts of the object might not be seen as is the case here with the on/off button. A trade off must therefore be made on the exposure time and focus on the parts of the device to maximize the amount of relevant information collected on the scene. The colored reconstruction of the coffee machine is shown in Figure 4.13.

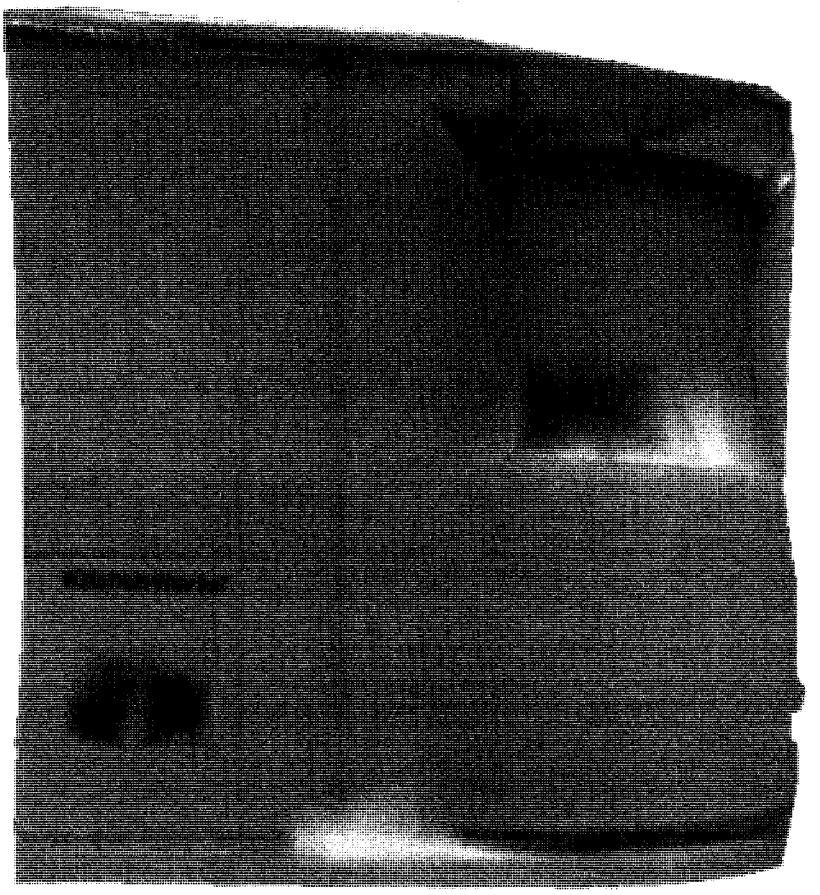


Figure 4.13 – Coffee machine colored reconstruction.

In the colored reconstruction we can observe that the on/off switch is not completely reconstructed, for the reasons mentioned above. However, it is interesting to notice that

the small lettering on top of the on/off switch "KitchenWorks" is properly reproduced on the colored 3D model. The reconstruction is accurate enough to recognize and render the writing on the coffee machine. Finally, the coffee pot and the coffee filter holder are clearly seen but some interpolation surface patches are inserted in the triangulation meshes that connect between the main surfaces. These patches are present here since no information is available from the bottom of the coffee filter and the top of the coffee pot, but are not included in the reconstruction composed only of 3D points, as shown in Figure 4.12.

4.3.3 Highly Reflective Porcelain Vase Object

Given that the system relies on projected light for creating virtual textures on surfaces, objects that are highly reflective and specular or object that do not reflect light perfectly onto the cameras are expected to exhibit less accurate reconstructions using this technology, in a similar way to numerous other range sensing approaches. However, an object was selected to evaluate these limitations. The object chosen for this test is a porcelain vase with a curvature, an uneven surface, dark colors and that is highly reflective. The object is shown is Figure 4.14 and the object with the projected pattern is shown in Figure 4.15.

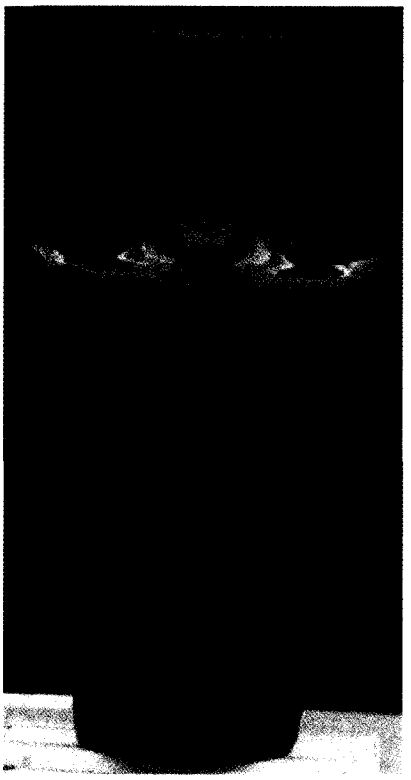


Figure 4.14 - Highly reflective vase.

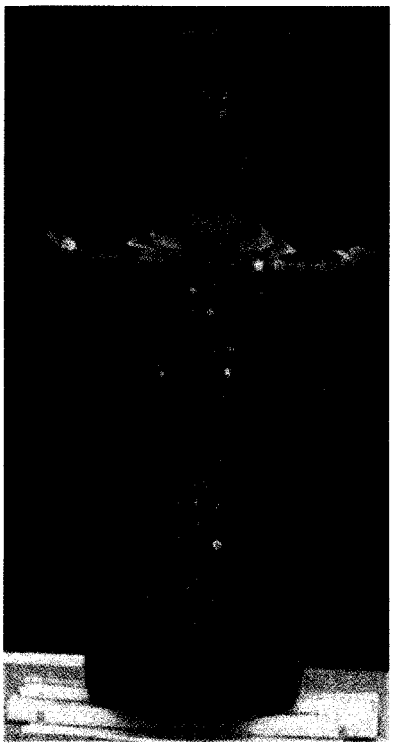


Figure 4.15 – Highly reflective vase with projected pattern.

A full resolution scan of the porcelain vase is performed and its 3D reconstruction result is shown in Figure 4.16.

a) front view b) lateral view

Figure 4.16 - Highly reflective vase reconstruction using 324 marching pseudo-random patterns.

Several points can be reconstructed on the object despite its complexity. The exposure time on the cameras had to be set longer than usual to be able to capture the patterns because less light was reflected to the cameras compared to the previous objects. This causes some added difficulty in the segmentation because some parts are over exposed

while others are still under exposed. On Figure 4.15, we also observe color regions that are imaged with combined color components in the center part of the vase, as described in section 3.5.3. This explains the slightly lower density of reconstructed points in the central part of the vase. Nonetheless, the vase can still be reconstructed correctly, and even some points located on the narrow upper part of the vase are successfully recovered. The upper section has a smaller radius which limits how many 3x3 pseudo-random codes can be properly projected onto the surface. This explains why mainly the central part of the tube is captured, independently from the high reflectivity (shininess) of this region. The colored reconstruction is shown in Figure 4.17.

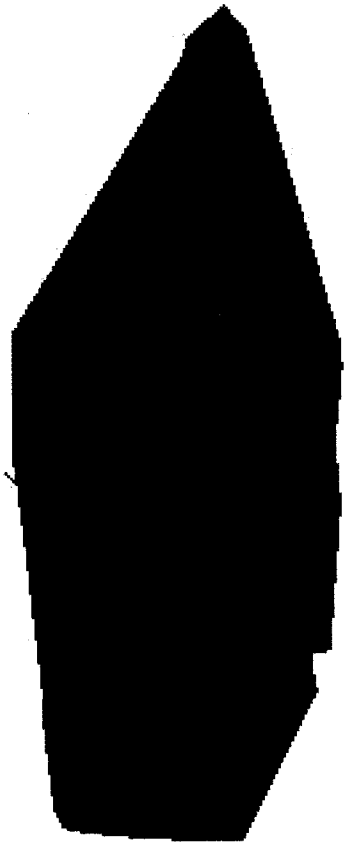


Figure 4.17 – Porcelain vase colored reconstruction.

4.3.4 Textured Box Object

Even though structured lighting is mainly designed to deal with surfaces that are textureless, such a sensor should be able to cope with more complex surfaces. That is inherent textures present on an object should not deteriorate the quality of the acquisition and reconstruction. Experimentation was performed to evaluate the proposed system's performance with an object that is rich in features. This object consists of a craft basket whose surface exhibits small variations in depth and contains reflective and non-reflective areas. The object is shown in Figure 4.18 and the object with the projected pattern is shown in Figure 4.19.

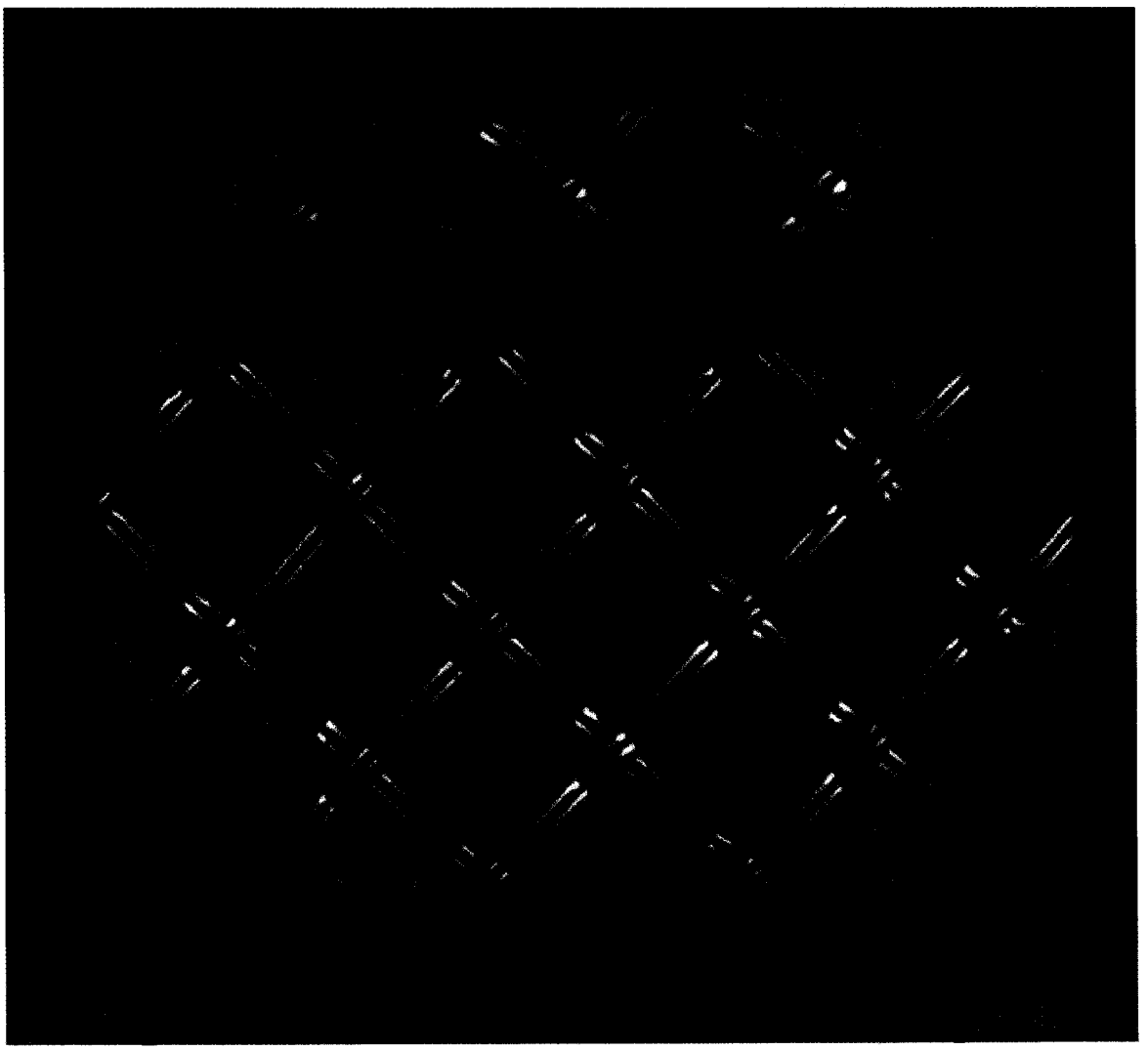


Figure 4.18 - Textured box.

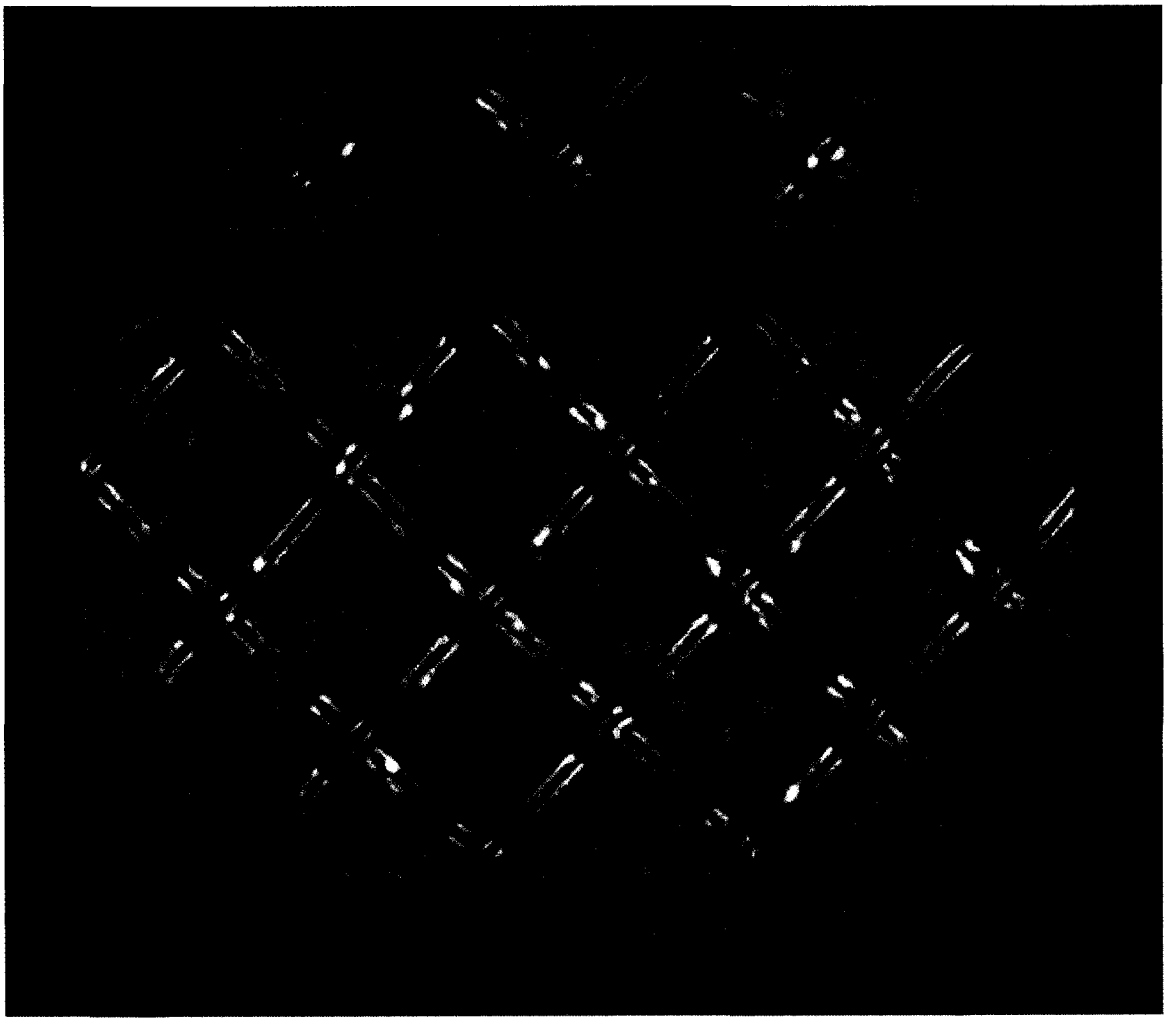


Figure 4.19 – Textured box with projected pattern.

A full resolution reconstruction of the object is performed and the result is shown in Figure 4.20. The colored reconstruction can be seen in Figure 4.21

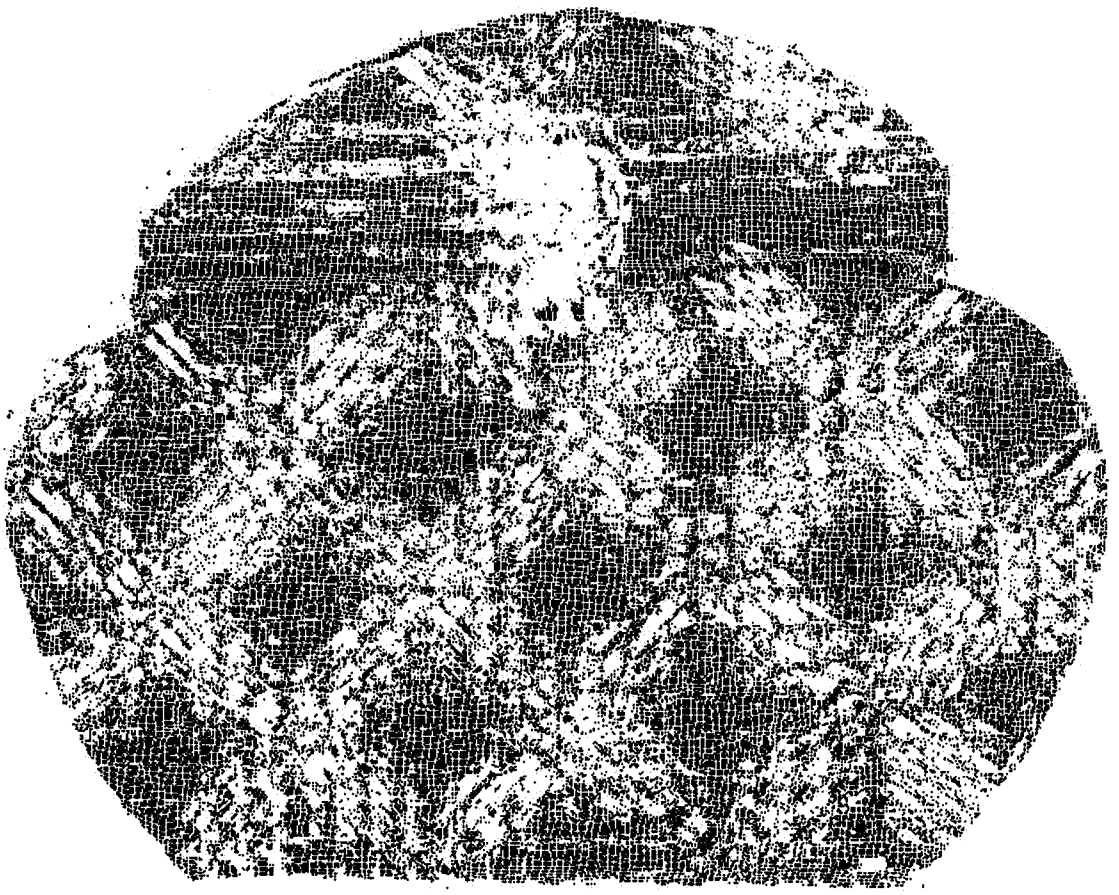


Figure 4.20 - Textured box reconstruction using 324 marching pseudo-random patterns.

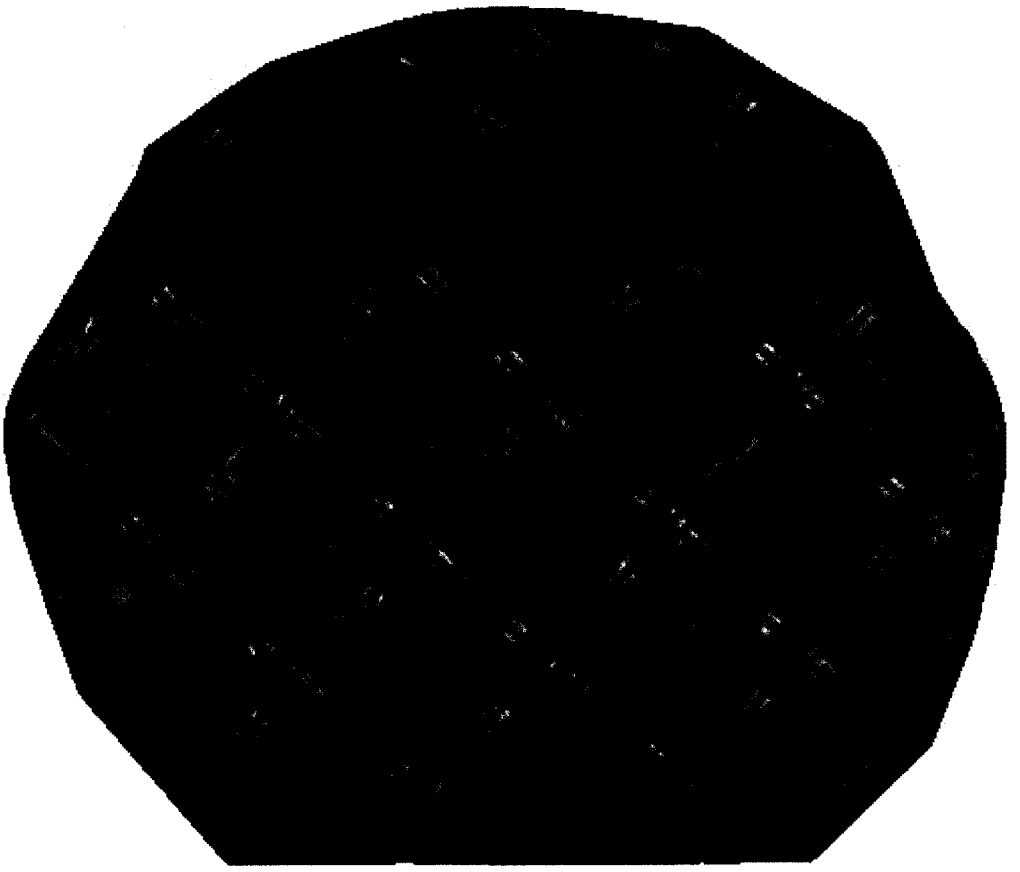


Figure 4.21 – Textured box colored reconstruction.

The results obtained are accurate considering the complexity of the textures present on the object. Only some points are lost along the reflective bands of the object and most points are reconstructed accurately on the non-reflective strands of the object. It can be noticed that the latch on the middle top of the object is not reconstructed accurately. This is explained by the fact that the latch is of a darker color than the rest of the object and cannot be seen in the images using the chosen exposure time. However, even with some points missing, the linear color interpolation clearly renders the missing information on the latch and on the strands.

4.3.5 Bag Object

The previous examples are all performed on surfaces where ambient lighting was relatively uniformly distributed. The next object, a brown bag, demonstrates an object that contains some irregular surfaces that create shadow effects and non planar

distribution and shows how the proposed system can still perform excellent 3D reconstruction. Figure 4.22 shows the original image of the bag and Figure 4.23 presents the object with the projected pattern.

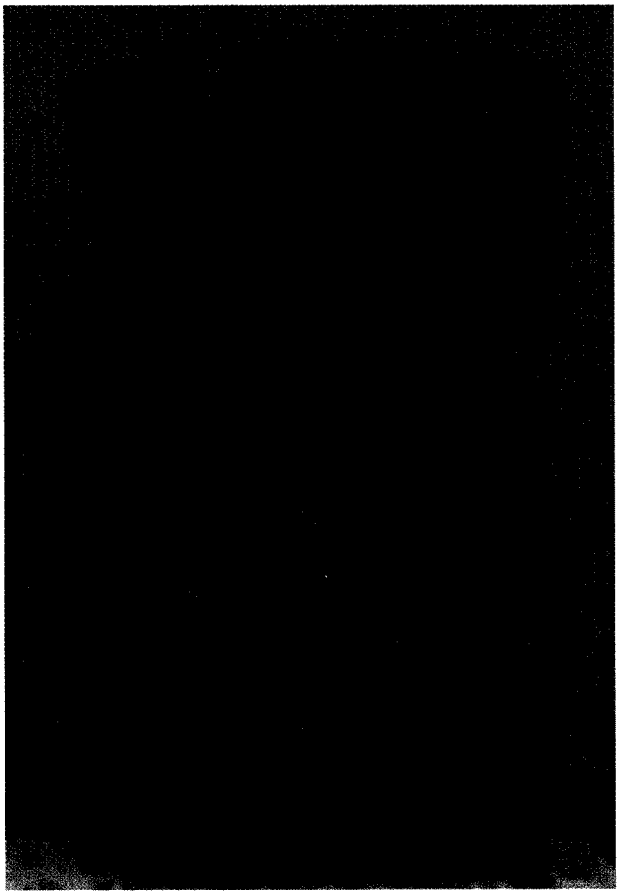


Figure 4.22 – Original brown bag.

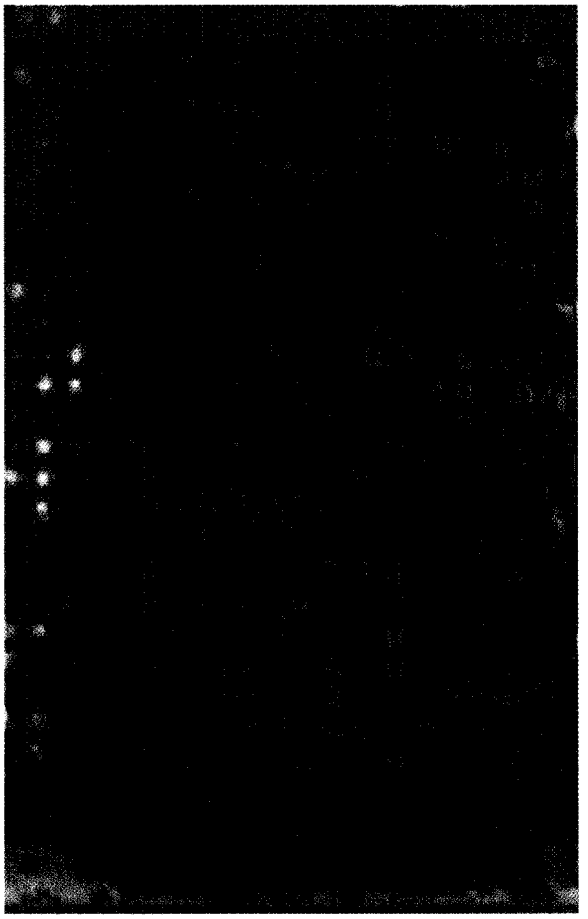
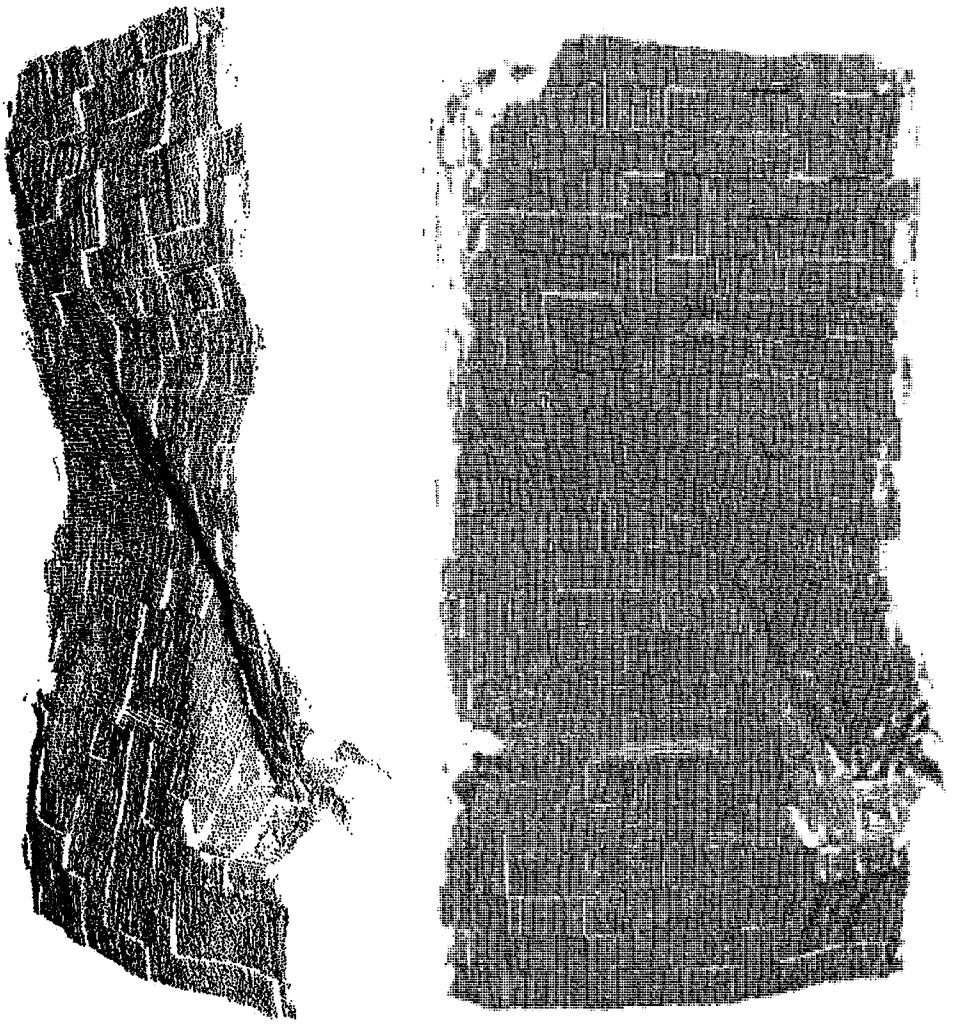


Figure 4.23 – Brown bad with projected pattern.

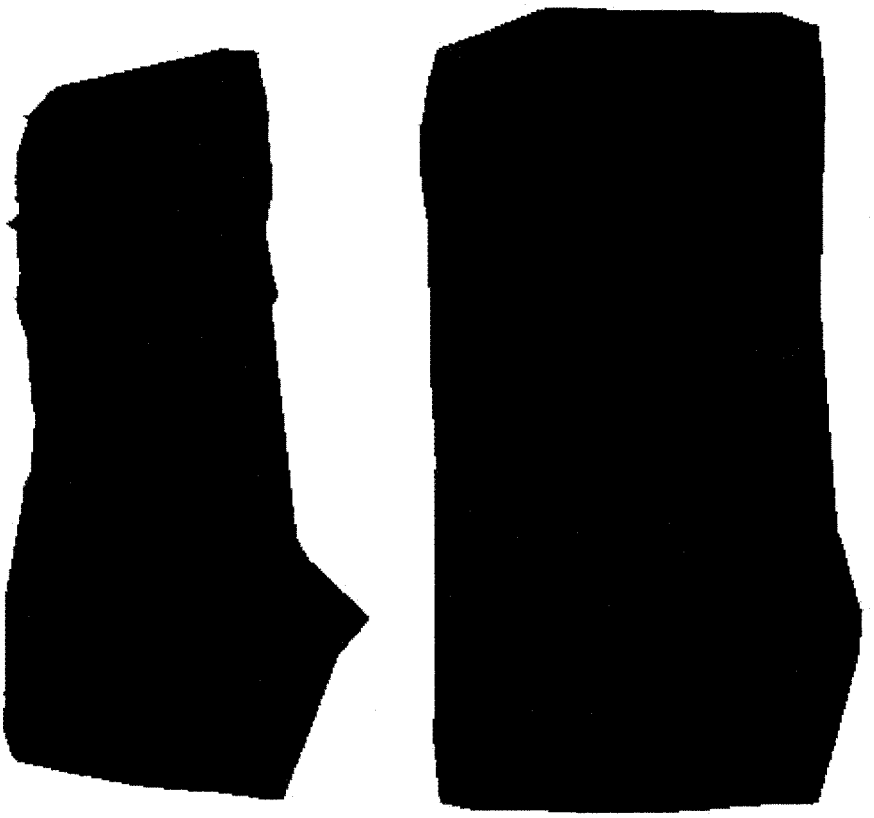
Figure 4.24 shows the point reconstruction of the bag from the front and side faces obtained with 324 marching pseudo-random patterns. The irregular surfaces are clearly visible and this demonstrates that the reconstruction is able to catch small deformations. A significant deformation in the bag can also be clearly seen on the middle right of the bag.



a) lateral view **b) front view**

Figure 4.24 – Brown bag point reconstruction.

Figure 4.25 shows the colored reconstruction of the brown bag for its front and side faces. Again the irregular surfaces of the bag can clearly be seen. The color reconstruction is close to the original image and visually accurate enough for further interpretation.



a) lateral view **b) front view**

Figure 4.25 – Brown bag colored reconstruction.

4.3.6 Red Book Object

In order to estimate the potential of the colored sensing and modeling mechanism to recover fine details, and to demonstrate the capability of the system to deal with surface colors that are similar to the colors found in the projected pattern, a red book with characters of various sizes was considered, as shown in Figure 4.26. Figure 4.27 presents the object with the projected pattern.

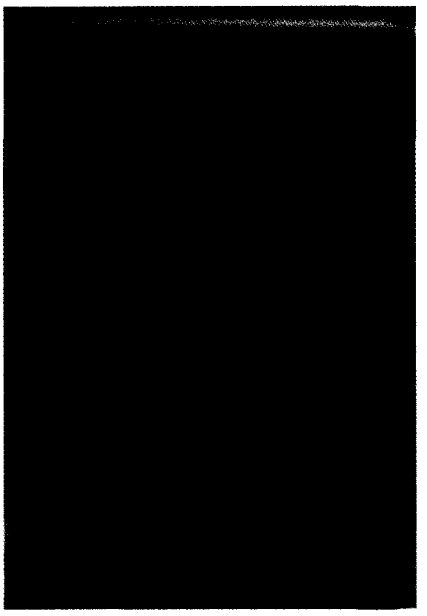


Figure 4.26 – Red book.

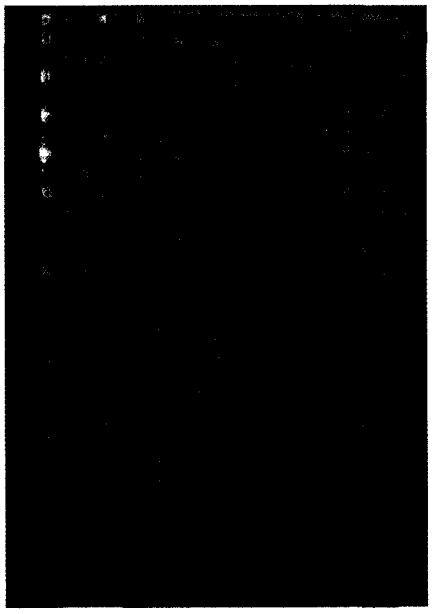


Figure 4.27 – Red book with projected pattern.

The reconstruction is performed twice, that is once with a single pattern projection, and a second time at full resolution with 324 marching patterns.

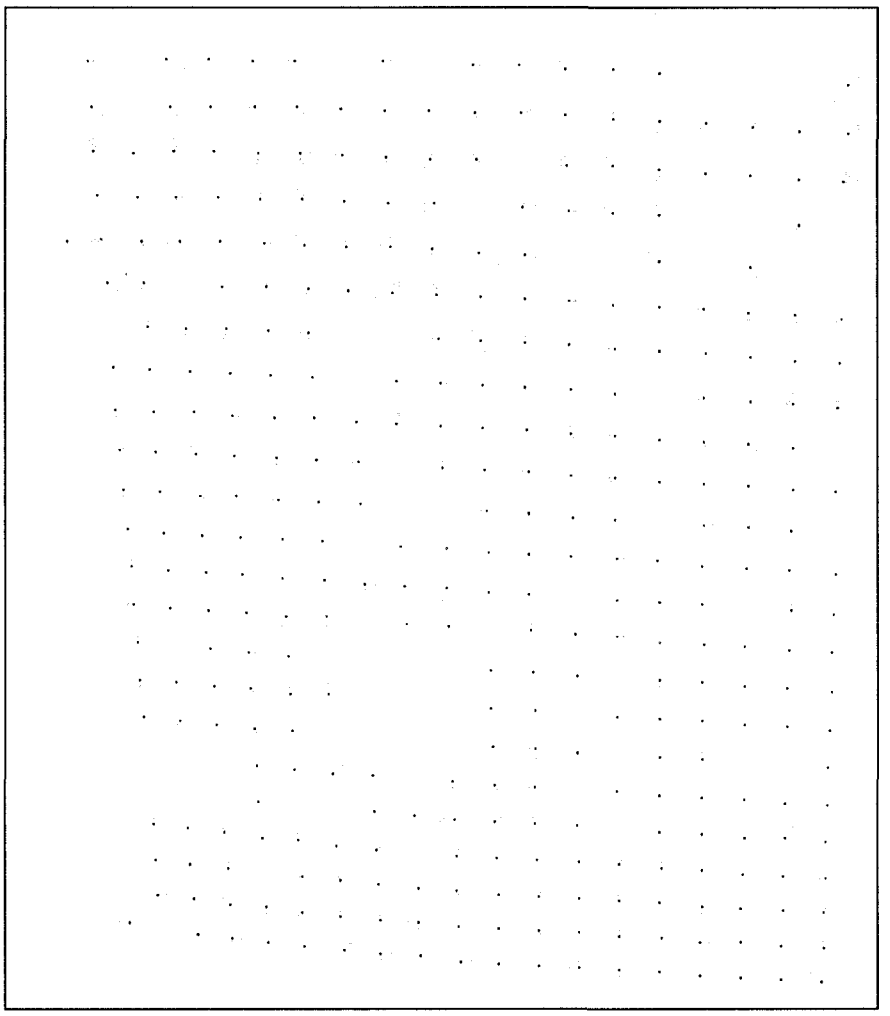


Figure 4.28 – Red book low resolution reconstruction.

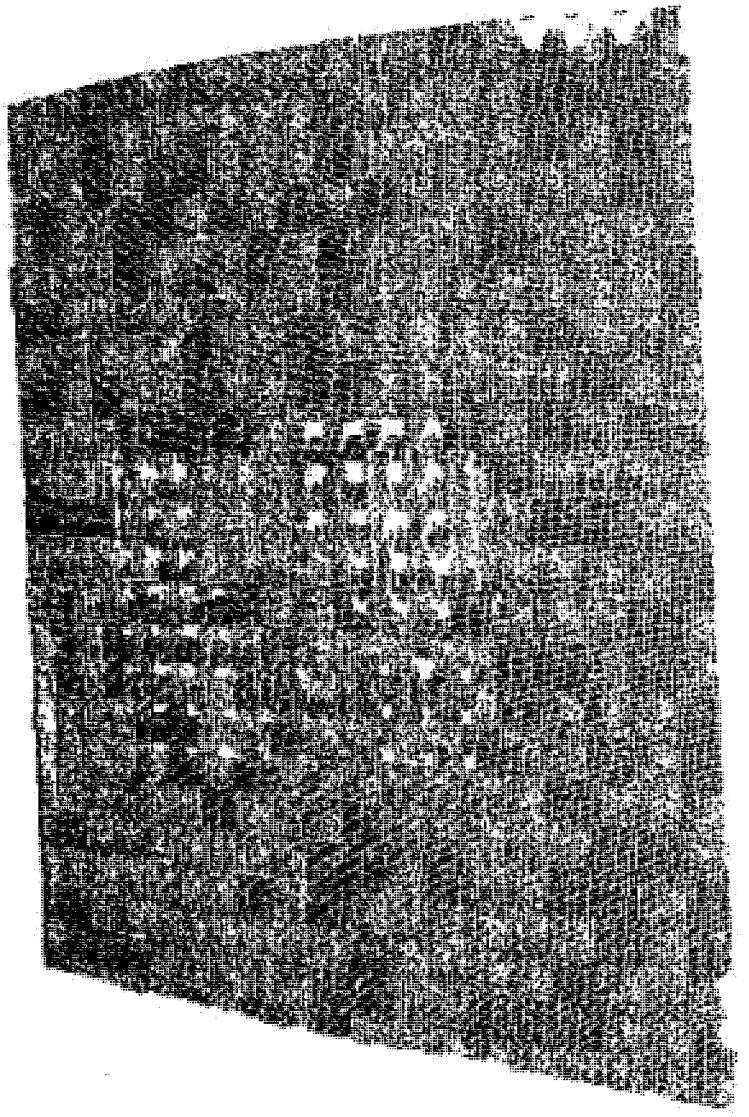


Figure 4.29 – Red book full resolution reconstruction.

Figure 4.28 contains 351 reconstructed points and Figure 4.29 contains 97069 reconstructed points. On the high resolution reconstruction it can be seen that fewer points are reconstructed around the black writing on the book. This is understandable since the reflective properties are substantially different between the writing and the cover of the book. These two 3D models are used to perform a color reconstruction of the red book and to demonstrate what visual quality each reconstruction can achieve.

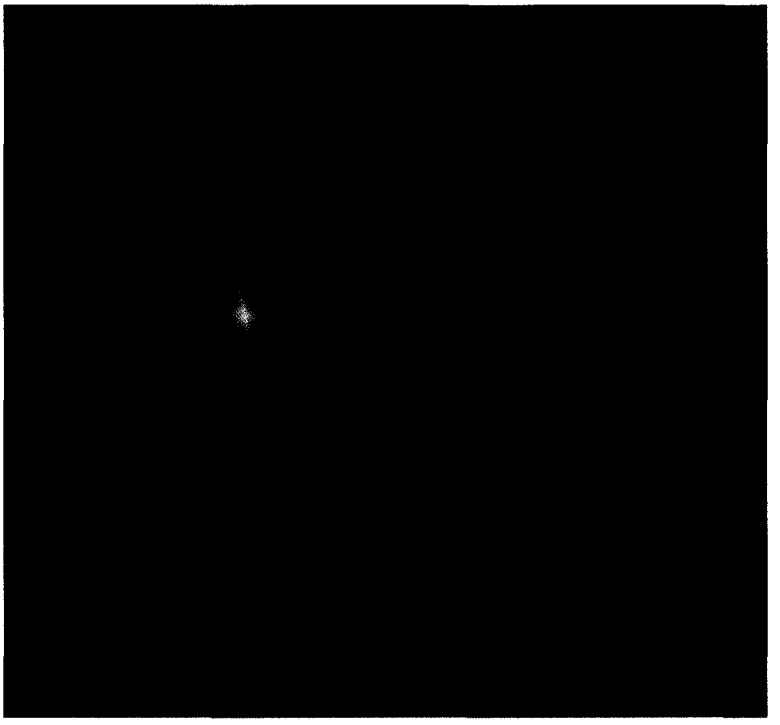


Figure 4.30 – Red book low resolution colored reconstruction.

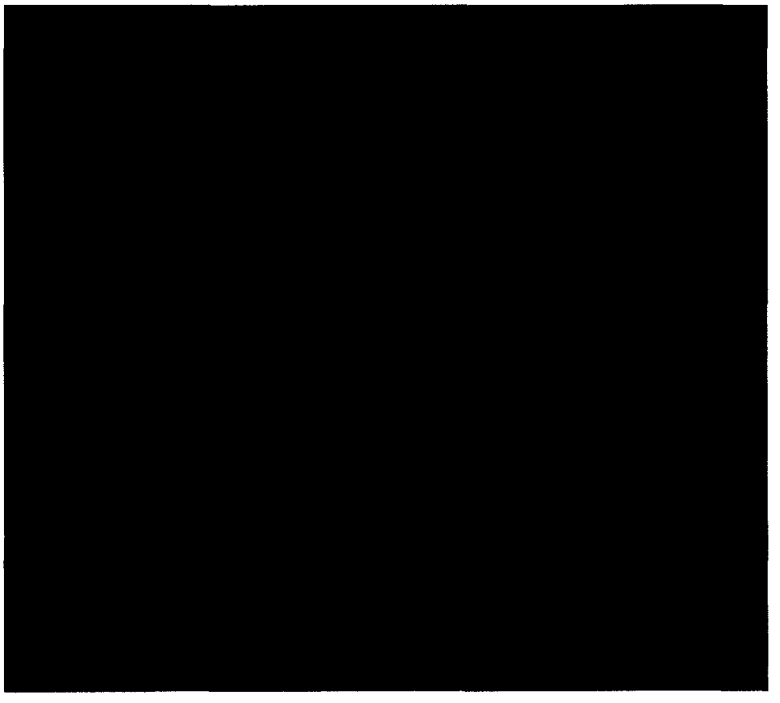


Figure 4.31 – Red book high resolution colored reconstruction.

The two color reconstructions can be seen in Figure 4.30 and Figure 4.31. Even at low resolution the black writing appears in the color reconstruction but is not accurate enough to be interpreted. When a reconstruction at full resolution is performed, the text is more legible. Even the smaller font text can be read. The text does however appear a bit blurry but demonstrates the precision of the color reconstruction. This experiment reveals the potential of marching pseudo-random patterns to produce on-demand adaptive resolution in both the 3D shape reconstruction and the color mapping on the model. It also demonstrates that even when an object is composed of a color that is part of the projected pattern, here red, reconstruction can still be performed.

4.3.7 Textured Chair Back Rest

This final reconstruction is performed on a chair back rest. This interesting object has important textures, varying degrees of reflexivity, non uniform colors and varying degrees of curvature. It was initially proposed as a sample object in the work of Kazantsev [29] and it is reconsidered here to provide a comparative basis of evaluation for the performance of structured light imaging systems on objects with complex textures. The object is shown in Figure 4.32 and the object with the projected pattern is illustrated in Figure 4.33.

Figure 4.32 – Chair back rest.

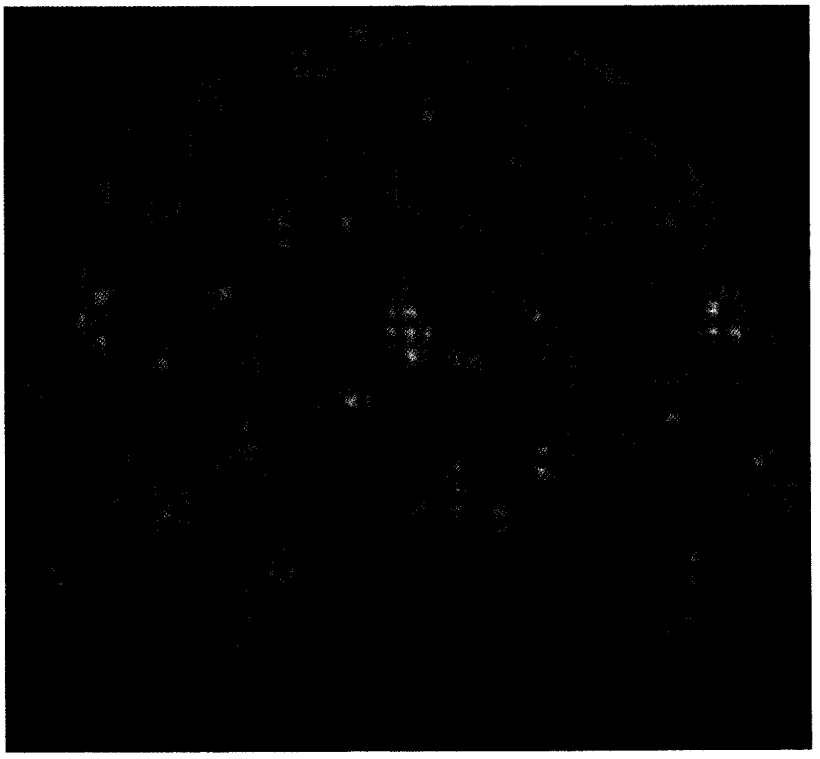
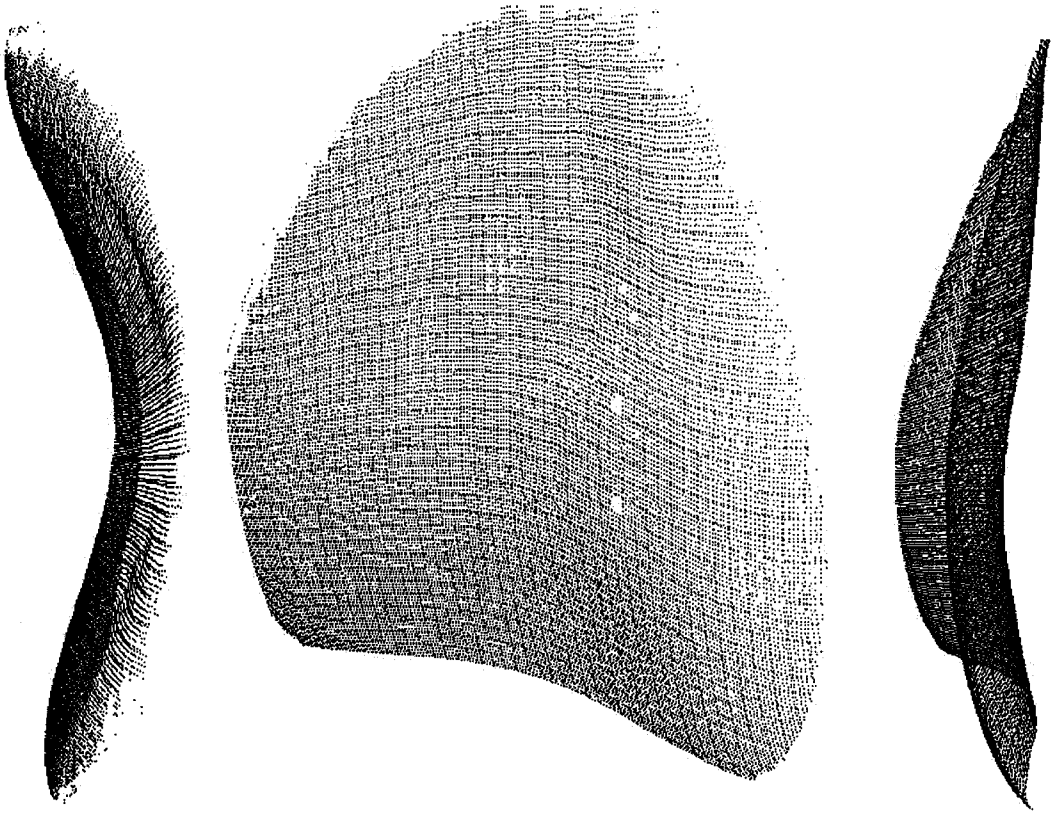


Figure 4.33 – Chair back rest with the projected pattern.

This object is scanned at various resolutions to demonstrate how the marching pattern scanning process can operate on such a surface. It must be noted that in this experiment the usual RGB pseudo-random pattern is projected on the scene and that no color selection is performed. Only the exposure time of the cameras is adjusted to obtain clear images. The results of scanning the object with 25 and 324 projected patterns are demonstrated in Figure 4.34, Figure 4.35 respectively, and in Figure 4.36, as point based-rendering.



a) top view **b) front view** **c) left lateral view**

Figure 4.34 – 3D reconstruction of the chair back rest with 25 marching pseudo-random patterns.

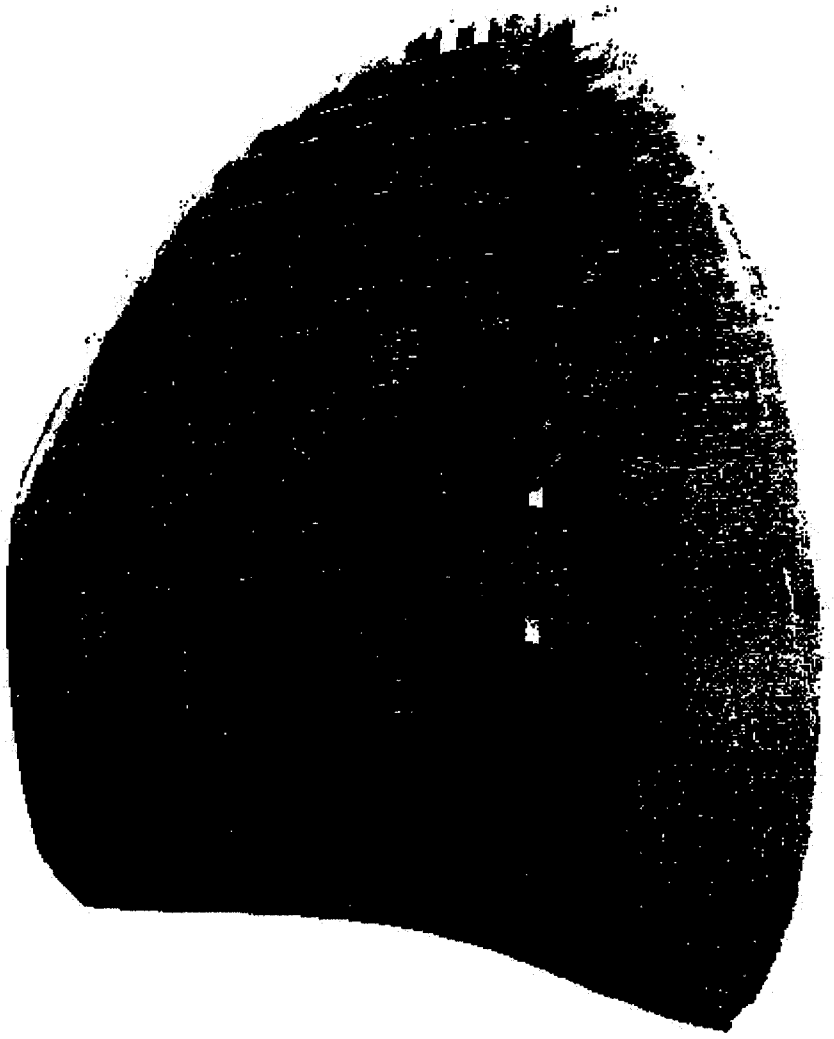


Figure 4.35 – Front view of the 3D reconstruction of the chair back rest with 324 marching pseudo-random patterns.

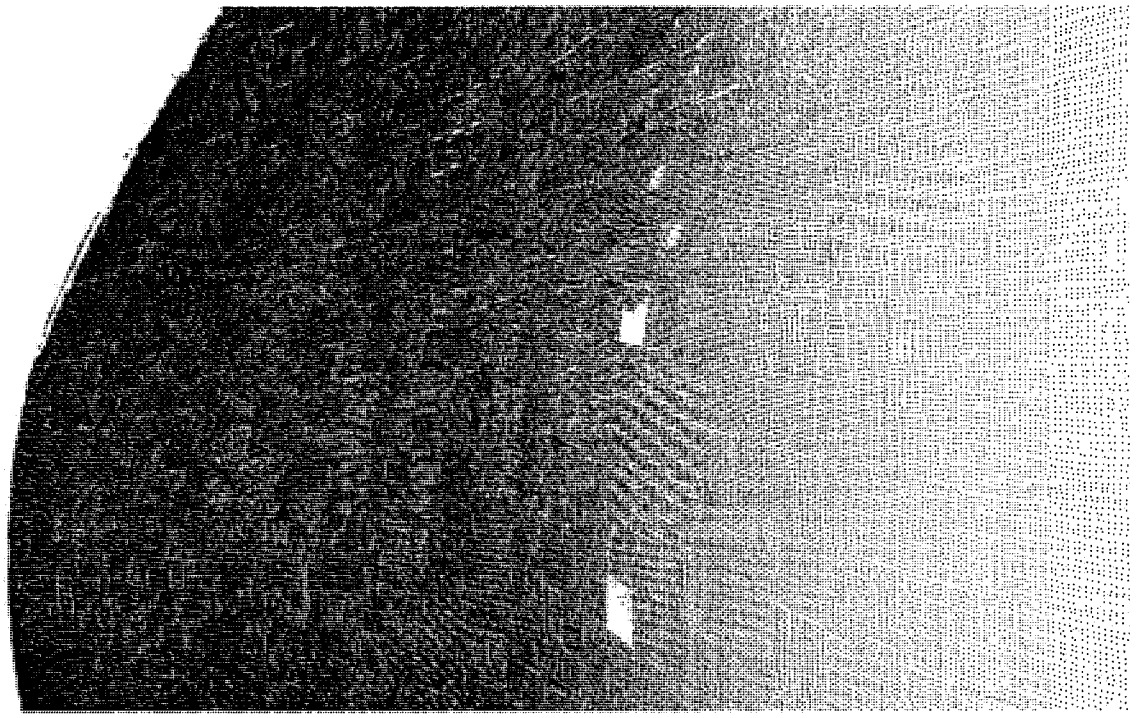


Figure 4.36 – Close up view of the high resolution reconstruction of the chair back rest with 324 marching pseudo-random patterns.

The reconstructions of the chair back rest are very precise even with a subset of the marching patterns (only 25 out of 324 possibilities). The curves of the chair back rest can clearly be seen in Figure 4.34 where the top, front and side of the chair back rest are illustrated. This reconstruction contains 35142 points. Furthermore, the black patches present on the chair do not excessively influence the reconstruction process and hence the results are uniformly dense with very few holes originating from the darkest color features present on the object. The exposure time is not long enough to capture the projected pattern reliably on all these dark features. When the full resolution reconstruction is performed, 456540 points are reconstructed. Figure 4.35 is so dense that the reconstructed points appear as a black surface. Figure 4.36 is a close up view of Figure 4.35 to demonstrate how dense the reconstructed points are and the very limited number of missing points. The missing points can be explained by the fact that the object does not exhibit uniform reflective properties on the borders where the normal to the surface changes its average orientation. As a result, codes are more difficult to extract.

Figure 4.37 and Figure 4.38 demonstrate the color reconstruction of the chair back rest. The textures are clearly shown and are very close to that of the original object. The subtle details of the textures are crisp, clear and clearly visible. Comparing the details of the colored reconstruction with the original image it can be observed that a dense reconstruction of objects is achieved and yields a colored representation that is very close to the original object textures, in spit of its inherent complexity. Moreover, this quality can be achieved without any particular selection of colors in the pseudo-random pattern that is projected.

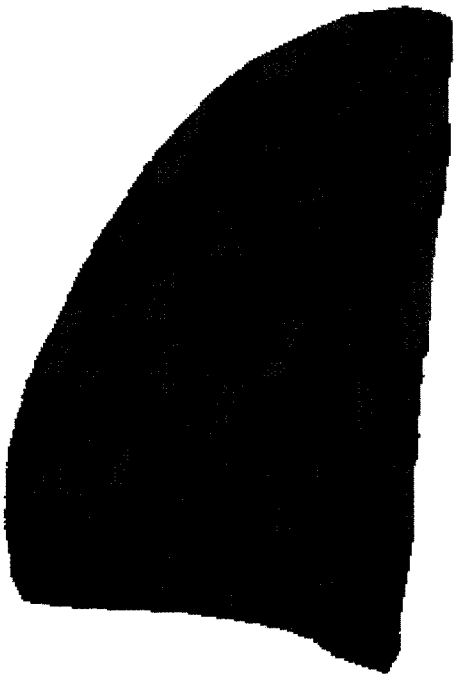


Figure 4.37 – Side view of the high resolution colored reconstruction of the chair back rest.

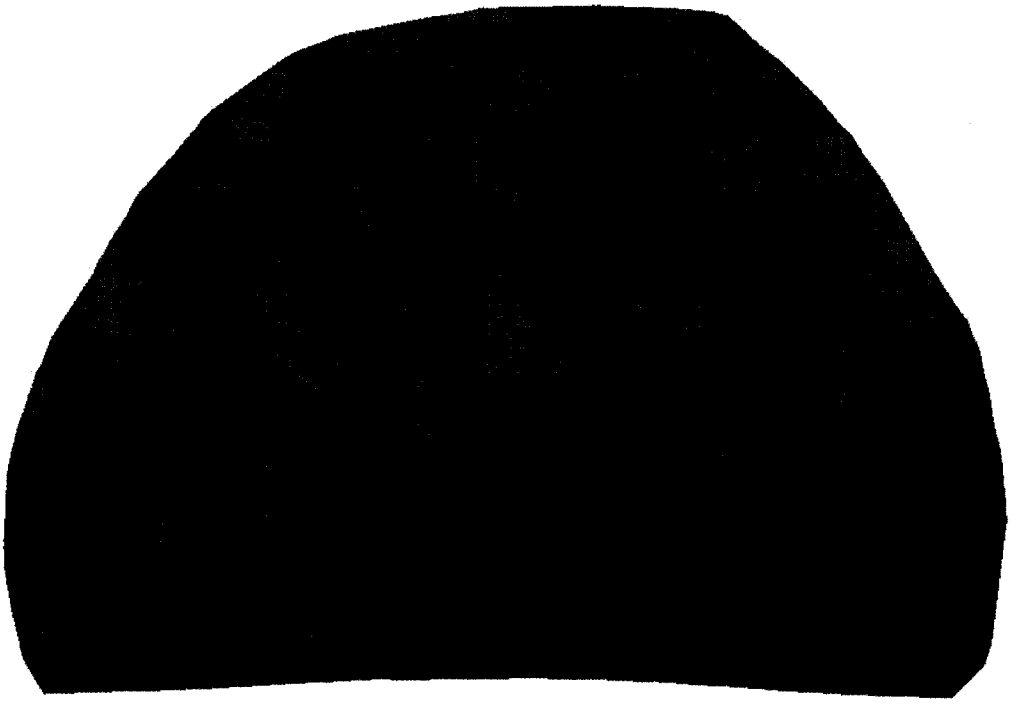


Figure 4.38 – Front view of the high resolution colored reconstruction of the chair back rest.

4.4 Natural Lighting Evaluation

The experiments presented in section 4.3 were performed inside a laboratory environment where ambient light can be controlled. However, the system was designed in such a way to minimize the effect of different lighting conditions. This is achieved with a robust image processing module that does not overly depend on the characteristics of ambient lighting. In order to evaluate this aspect of the proposed imaging system, the experimental setup was taken out of the laboratory into an open area with large windows introducing uncontrolled and non-uniformly distributed light in the environment. The simple chair object is used for testing purposes in this environment as shown in Figure 4.39.

Figure 4.39 – Simple chair exposed to ambient lighting.

A high resolution 3D reconstruction is performed on the simple chair and the results are shown in Figure 4.40.

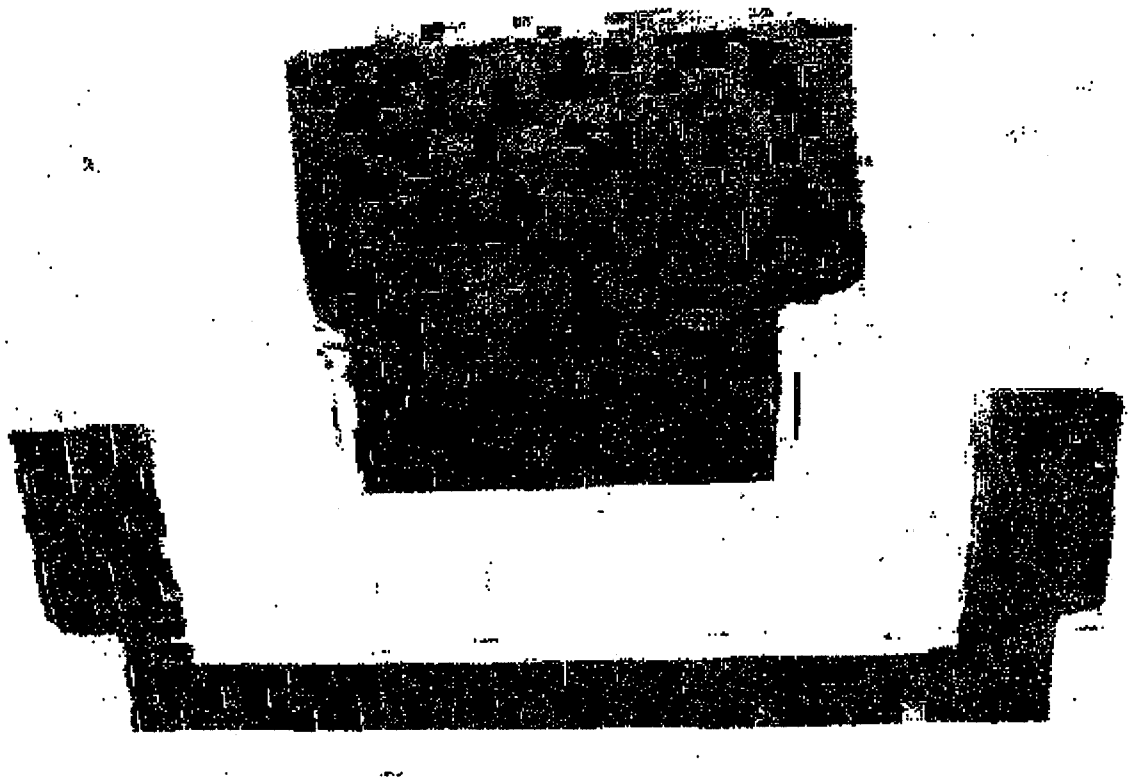


Figure 4.40 – 3D reconstruction of the simple chair in bright ambient light.

The reconstruction density achieved is quite similar to the reconstruction performed in the lab environment (see Figure 4.7). However, slightly more outliers are present in this reconstruction. These can be explained by the longer exposure time required to capture the patterns. Also, some colors, such as the red component of the projected pattern, are

not captured as well as other colors and create inconsistent pseudo-random codes. Nonetheless, the chair can still be reconstructed adequately enough for most exploratory vision tasks.

4.5 Performance Analysis

The integrated range sensor system can process large stereoscopic images adequately in a reasonable period of time. Table 4.3 presents the time required to reconstruct a scene from one pair of images of resolution 1392 x 1040 with a pseudo-random pattern containing 55x40 color codes when different numbers of points are actually reconstructed.

Points	237	618	1651
Acquisition time (seconds)	1.180	1.180	1.180
Image processing time (seconds)	1.231	1.336	10.98
Code recognition time (seconds)	0.178	0.272	1.47
3D reconstruction time (seconds)	0.047	0.127	0.343
Total time (seconds)	2.636	2.915	13.973

Table 4.3 – Processing times.

The total time represents the total time spent from image acquisition until the 3D colored reconstruction results are written to disk.

The timing measurements are performed on an Intel Core 2 Q9450 at 2.66 GHz with 4 gigabyte of RAM running Microsoft Windows Vista Business 64bit edition. Most of the processing time, when modeling a scene, is spent on image processing. The optimal reconstruction algorithm takes only 3% of the total time of processing and demonstrates that its added complexity compared to the middle point algorithm does not affect the performance significantly. The processing time required in the image processing module is mostly spent in the group labeling algorithm which is highly dependant on the resolution of the images. To improve performance, smaller images can be used.

The timing results demonstrate well the need for the introduction of marching patterns. Because of the amount of time required to completely process images, it is important to be able to select the desired resolution depending on the needs of the user. A high resolution scan of an object composed of 324 pairs of images given the pseudo-random pattern defined in chapter 3 that is able to reconstruct 237 points per image would take about 14 min to process all the data. However, a quick scan of the environment would only take about 3 seconds, which can then be selectively increased by extra scans with the marching patterns, depending on the available time for processing and the required resolution.

Chapter 5 Conclusion

5.1 Summary

The research performed and presented in this thesis consists of the development and integration of a complete system that performs acquisition of 3D points using structured lighting stereoscopic vision with a calibrated setup and compiles this information into a 3D colored reconstruction of the scene.

First, a literature review is proposed in Chapter 2 that covers techniques relevant to the development of the proposed imaging system. Active and passive 3D vision technologies are reviewed and the focus is placed on active stereoscopic approaches combined with structured lighting. Here several types of projected patterns are analyzed and compared. Classical image processing algorithms of use for the implementation of our prototype are examined and, finally, the potential of two 3D reconstruction methods is evaluated. This analysis resulted in the selection of the proper methodologies for the development of the proposed structured lighting stereoscopic system with marching pseudo-random patterns.

Chapter 3 elaborates on the structural design of the proposed sensor and details all aspects ranging from the construction of the pseudo-random patterns, their projection and the acquisition of pairs of images, the image processing stage that includes some particularities related to the nature of the acquisition that is performed, and up to the pseudo-random code recovery, validation and extraction of matched feature points that serve for the 3D colored reconstruction. In this chapter, the emphasis is placed on implementation issues and original aspects that were introduced throughout this work.

Finally, Chapter 4 proposes an extensive set of results demonstrating that the proposed approach performs very well under numerous circumstances and over objects exhibiting a wide variety of characteristics either on their size, shape, or surface reflectance. As well,

several performance metrics are used to validate the quality of the result in time of execution, accuracy of reconstruction and robustness to external factors.

5.2 Contributions

This thesis provides some contributions to the field of 3D imaging with structured lighting, including:

1) The combination of spatial-neighboring and time-multiplexing patterns for structured lighting into marching pseudo-random patterns.

The proposed approach combines the idea and the strengths of time-multiplexing patterns and that of spatial neighboring patterns. Spatial neighboring patterns can be used to completely capture a scene in a single scan. However, they usually do not achieve a high density compared to time-multiplexing patterns. Also, this work introduces the concept of marching pseudo-random patterns. A pseudo-random pattern is projected and shifted onto a scene multiple times on demand. The number of projections can be modified depending on the desired resolution and on certain time constrains. This provides the user with the necessary flexibility to achieve dense results when required with no modification to the acquisition system. This feature can be eventually automated.

2) A robust pseudo-random code recovery and validation technique.

A second contribution originates from how pseudo-random codes are recovered and how outliers are determined and removed using statistical code validation. Due to some errors in segmentation and acquisition some duplicate and non existent codes might be present after the initial recovery of codes. A first pass is performed on all codes and removes any non existing codes. The second pass identifies any duplicate codes found. The determination of invalid codes is performed by calculating the confidence of a code. The code with the highest confidence is kept and all other duplicate codes are removed. This

varies from the traditional approach of calculating the confidence of all codes and dropping codes that fall below a certain threshold.

3) The simultaneous reconstruction of 3D shape and colored models.

The 3D reconstruction performed on objects also simultaneously gathers color information. This information is used to achieve a full colored reconstruction of an object which enables a more accurate representation of models that makes their interpretation easier and compensates where some data might be absent in the model. It results in a fully integrated and affordable sensor that runs from off-the-shelf equipment, and readily provides a rich 3D shape and colored model of a scene. Such a feature is not provided by most 3D sensing technologies currently available.

4) Image processing methods for color pseudo-random code extraction.

Finally, this thesis proposes new image processing methods specially designed to segment and recognize color pseudo-random codes. The algorithm is efficient, does not require prior knowledge of the scenes and is fairly robust with respect to different lighting conditions and objects reflective properties and textures.

This research work was initially published as "Dense Stereo Range Imaging with Marching Pseudo-Random Patterns" at the 2007 Canadian Conference on Computer and Robot Vision [46].

5.3 Future Work

As demonstrated in Chapter 4, the proposed system performs well but still has some limitations. The results achieved highly depend on the relative location of the objects of interest to the focal point of the projector and the cameras. This limits the depth that can be scanned by the system. A module could be implemented to mechanically change the focal points, based on some parameters, to achieve optimal results. A calibration procedure could be designed to determine what focal points are best suited for a given

scene so that focal points can be changed dynamically. Auto-zoom and auto-focus functions could be investigated and added for both the cameras and the projector to provide perception capabilities on scenes exhibiting several depths.

More sophisticated 3D reconstruction and calibration techniques may be explored further to refine the current performance. A possibility would be to integrate a bundle adjustment [47] for calibration and reconstruction phases. This approach excels in the calibration of multiple cameras and also performs non-scaled reconstruction.

It has been demonstrated that the group labeling algorithm is the most demanding module in terms of processing time for the whole system. The algorithm could be revisited to improve its efficiency or another strategy might be adopted to extract color blobs in a more efficient way without compromising the accuracy of the codes recovered.

A further step will also be to implement the system on a mobile platform. In this way more information about scenes could be captured and merged through registration between multiple points of view. Finally, it was observed that under some circumstances some colors cannot be captured properly on certain scenes. A supplementary module could be added to be able to deal with uncooperative colored surfaces, as proposed in A. Kazantsev's work [29].

References

[1] Z. Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Corfu, Greece, Vol. 22, no 11, pp. 1330-1334, 2000.

[2] S. Parthasarathy, J. Birk and J. Dessimoz, "Laser Rangefinder for Robot Control and Inspection", *In Proc. SPIE Robot Vision*, Vol. 336, pp. 2-11, 1982.

[3] M. Rioux, "Laser Range Finder Based Upon Synchronous Scanners", *Applied Optics*, Vol. 23, no 21, pp. 3837-3844, 1984.

[4] R.J. Popplestone, C.M. Brown, A.P. Ambler and G.F. Crawford, "Forming Models of Plane-and-cylinder Faceted Bodies from Light Stripes", *In Proc. Int. Joint Conf. On Artificial Intelligence*, Tbilisi, Georgia, USSR, pp. 664-668, 1975.

[5] G. Porter and J. Mundy, "Noncontact Profile Sensing System for Visual Inspection", *In Proc. SPIE Robot Vision*, Vol. 336, pp. 67-76, 1982.

[6] E. Trucco and A. Verri, *Introductory Techniques from 3-D Computer Vision*, Prentice Hall, 1998.

[7] P. Albrecht and B. Michaelis, "Stereo Photogrammetry with Improved Spatial Resolution", *In Proc. 14th International Conference on Pattern Recognition*, Brisbane, Australia, pp. 845-849, 1998.

[8] B.K.P. Horn, *Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View*, Ph.D. Thesis, Massachusetts Institute of Technology, 1970.

[9] S. Ullman, *The Interpretation of Visual Motion*, MIT Press, Cambridge, 1979.

[10] J. Marszalec and R. Myllylä, "Shape Measurements Using Time-Of-Flight-Based Imaging Lidar", *In Proc. SPIE Conference on Three-Dimensional Imaging and Laser-based Systems for Metrology and Inspection III*, Pittsburgh, USA, Vol. 3204, pp. 14-15, October 1997.

[11] F.D. Nisi, F. Comper, L. Gonzo, M. Gottardi, D. Stoppa, A. Simoni and J.A. Beraldin, "A CMOS Sensor Optimized for Laser Spot-Position Detection", *IEEE Sensors Journal*, Vol. 5, no 6, pp. 1296–1304, Dec. 2005.

[12] J. Salvi, J. Pagès and J. Batlle, "Pattern Codification Strategies in Structured Light Systems", *Pattern Recognition*, Vol. 37, no 4, pp. 827-849, 2004.

[13] D. Scharstein and R. Szeliski, "High-Accuracy Stereo Depth Maps Using Structured Light", *In Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, Vol. 1, pp. 195-202, Madison, WI, June 2003.

[14] G. Sansoni, S. Corini, S. Lazzari, R. Rodella, and F. Docchio, "Three-Dimensional Imaging Based on Gray-Code Light Projection: Characterization of The Measuring Algorithm and Development of A Measuring System for Industrial Applications", *Applied Optics*, Vol. 36, no. 19, pp. 4463-4472, July 1997.

[15] G. Häusler and D. Ritter, "Parallel Three-Dimensional Sensing by Color-Coded Triangulation", *Applied Optics*, Vol. 32, no 35, pp. 7164–7169, 1993.

[16] E. Schubert, "Fast 3D Object Recognition Using Multiple Color Coded Illumination", *In Proc. IEEE Conf. Acoustics, Speech, and Signal Processing*, Munich, Vol. 4, pp. 3057–3060, 1997.

[17] S. Zhang and P.S. Huang, "High-Resolution, Real-Time 3D Shape Acquisition", *In Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Washington DC, MA, pp. 28-38, 2004.

[18] Y. Hu, J. Xi, Z. Yang, E. Li and J. Chicharo, "Study on Generalized Analysis Model for Fringe Pattern Profilometry", *IEEE Transactions on Instrumentation and Measurement*, pp. 160-167, January 2008.

[19] 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*, Vol. 20, no 3, pp. 322-327, March 1998.

[20] C. Albitar, P. Graebling and C. Doignon, "Robust Structured Light Coding for 3D Reconstruction", *In Proc. IEEE 11th International Conference on Computer Vision*, pp. 1-6, October 2007.

[21] H. Hügli and G. Maitre, "Generation and Use of Color Pseudo Random Sequences for Coding Structured Light in Active Ranging", *Industrial Inspection*, Vol. 1010, pp. 75–82, 1989.

[22] P. Vuylsteke and A. Oosterlinck, "Range Image Acquisition with a Single Binary-Encoded Light Pattern", *Pattern Analysis and Machine Intelligence*, pp. 148–163, 1990.

[23] C.J. Mitchell, T. Etzion and K.G. Paterson, "A Method for Constructing Decodable de Bruijn Sequences", *IEEE Transactions on Information Theory*, IT-42, pp. 1472-1478, 1996.

[24] P. Lavoie, D. Ionescu and E. Petriu, "3-D Object Model Recovery from 2-D Images Using Structured Light", *In Proc. IEEE Instrumentation and Measurement Technology Conference*, Brussels, Belgium, pp. 377-382, 1996.

[25] W.H. Press, W.T. Vetterling, S.A. Teukolsky and B.P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, pp. 298 and pp. 375, 1992.

[26] A. Moica, *Coprocessor for Decoding Multi-Valued Pseudo-Random Sequence Patterns*, M.A.Sc. Thesis, University of Ottawa, 2000.

[27] F.J. MacWilliams and N.J.A. Sloane, "Pseudo-Random Sequences and Arrays", *In Proc. of the IEEE*, Vol. 64, no 12, pp. 1715–1729, Dec. 1976.

[28] P. Lavoie, D. Ionescu and E. Petriu, "A High Precision 3D Object Reconstruction Method Using a Color Coded Grid and Nurbs", *In Proc. International Conference on Image Analysis and Processing*, Venice, Italy, pp. 370–375, 1999.

[29] A. Kazantsev, *Recovery of Pseudo-Random Coded Coloured Structured Light Information for 3D Object Recognition from Uncooperative Coloured Surfaces*, Master's Thesis, University of Ottawa, December 2007.

[30] B. Carrihill and R. Hummel, "Experiments with the Intensity Ratio Depth Sensor", *In Proc. Computer Vision, Graphics and Image Processing*, Vol. 32, Academic Press, pp. 337–358, 1985.

[31] J. Tajima and M. Iwakawa, "3-D Data Acquisition by Rainbow Range Finder", *In Proc. International Conference on Pattern Recognition*, pp. 309-313, 1990.

[32] E. Mouaddib, J. Batlle and J. Salvi, "Recent Progress in Structured Light in order to Solve the Correspondence Problem in Stereovision", *In Proc. IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 130-136, Albuquerque, NM, USA, April 1997.

[33] R. Gonzalez and R. Woods, *Digital Image Processing*, Prentice Hall, pp. 567-624, 2002.

[34] T. N. Pappas, "An Adaptive Clustering Algorithm for Image Segmentation", *IEEE Transactions on Signal Processing*, Vol. 40, no 4, pp. 901-914, Apr. 1992.

[35] R. Hartley and P. Sturm, "Triangulation", *Computer Vision and Image Understanding*, Vol. 68, no 2, pp. 146-157, 1997.

[36] R. Hartley and A. Zisserman, *Multiple View Geometry*, Cambridge University Press, June 2000.

[37] R. Legarda-Saenz, T. Bothe and W.P. Juptner, "Accurate Procedure for the Calibration of a Structured Light System", *Optical Engineering*, Vol. 43, no 2, pp. 464-471, February 2004.

[38] *Intel, Open Source Computer Vision Library*, Release 1.0, October 2006, *http://www.intel.com/technology/computing/opencv/*.

[39] Z. Tang and Z. Miao, "Fast Background Subtraction and Shadow Elimination Using Improved Gaussian Mixture Model", *In Proc. of the IEEE International Workshop on Haptic Audio Visual Environments and Their Applications*, pp. 38-41, Ottawa, Oct. 2007.

[40] R. Jain, R. Kasturi and B. Schunck, *Machine Vision*. McGraw-Hill, pp. 78-80, 1995.

[41] M. Fischler and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Images Analysis and Automated Cartography", *Communications of the ACM*, Vol. 24, no 6, June 1981.

[42] S. Priester, *Delaunay Triangles*, July 2005, *http://www.codeguru.com/cpp/data/mfc_database/misc/article.php/c8901*.

[43] P. Curtis and P. Payeur, "A Frequency Domain Approach to Registration Estimation in Three-Dimensional Space", *IEEE Transactions on Instrumentation and Measurement*, Vol. 57, no 1, pp. 110-120, Jan. 2008.

[44] Lumenera, *Lu135 SPEC SHEET*, 2007, *http://www.lumenera.com/industrial/lu135.php.*

[45] Engineering Fundamentals, Multiple Regression, 2007, http://www.efunda.com/math/leastsquares/lstsqrzrwtxy1d.cfm.

[46] D. Desjardins and P. Payeur, "Dense Stereo Range Sensing with Marching Pseudo-Random Patterns", *In Proc. of the Canadian Conference on Computer and Robot Vision*, Montreal, Qc., Canada, pp. 216-226, 2007.

[47] B. Triggs, P. McLauchlan, R. Hartley and A. Fitzgibbon, "Bundle Adjustment: a Modern Synthesis", *Vision Algorithms: Theory and Practice*, LNCS, Vol. 1883, pp. 298-372, Springer-Verlag, 2000.