Enhanced detection of point correspondences in single-shot structured light

by

Kasra Sadatsharifi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2021

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The majority content of Chapters 4 and 5 are from two papers that I have authored in collaboration with Prof. Paul Fieguth and Dr. Mohamed A. Naiel. The equipment for experiments were provided by Christie Digital Systems Inc. with the help of Mr. Mark Lamm. I am the first author of both papers and have contributed to the design, implementation, experimentation, and majority of the writing of the papers. Problem formulation and further comments and advice were provided by Prof. Paul Fieguth and Dr. Mohamed A. Naiel. Content of Chapter 4 is from [103] and content of Chapter 5 is from a future publication [104].

Abstract

The crucial role of point correspondences in the process of stereo vision and camera projector calibration is to determine the relationship between the camera view(s) and the projector view(s). Consequently, acquiring accurate and robust point correspondences can result in a very accurate 3D point cloud of a scene. Designing a method that can detect pixel correspondences quickly and accurately and be robust to factors such as object motions and color is an important subject of study. The information that lies in the point correspondences determines the geometry of the scene in which depth plays a very important role, if not the most important. However, point correspondences will include some outliers. Outlier removal is another important aspect of obtaining correspondences that can have substantial impact on the reconstructed point cloud of an object. During the Single-Shot Structured Light (SSSL) calibration process, a pattern consisting of tags with differently shaped symbols inside and separated by grids are projected onto the object. The intersections of these grid lines are considered to be potential pixel correspondences between a camera image and the projector pattern.

The purpose of this thesis is to study the robustness and accuracy of pixel correspondences and to enhance their quality. In this thesis we propose a detection method that uses the model of the pattern, specifically the grid lines, which are the largest and brightest feature of the pattern. The input image is partitioned into smaller patches and then the optimization process is executed on each patch. Eventually, the grid lines will be detected and fitted to the grid, and the intersections of those lines are taken as potential corresponding pixels between the views. In order to remove incorrect pixel correspondences, or in other words, outliers, Connected Component Analysis is used to find the closest detected point to the top left corner of each tag. The points remaining after this step are the correct pixel correspondences.

Experimental results show the improvement of using a locally adaptive thresholding method against the baseline in detecting tags. The proposed thresholding method showed a maintained accuracy compared to the baseline method while automatically tune all the parameters whereas in the baseline method some parameters need fine tuning. Introduced model-based grid intersection detection yields an approximately 50 times improvement in speed. Inaccuracy in point correspondences are compared with state-of-the-art method based on the generated final reconstructed point clouds using both methods against the CAD model as ground truth. Results show an average of 3 pixels higher error in distance, between the reconstructed point clouds and the CAD model, in the proposed method compared to the baseline.

Acknowledgements

I owe my deepest gratitude and appreciation to my supervisor Dr. Paul Fieguth. I appreciate all the efforts and support he had for me during the pandemic and my work. I learned a lot, working with him and his team.

I would like to thank Dr. John Zelek and Dr. Zhou Wang for serving as readers for my thesis.

I would like to thank Mohamed A. Naiel for his support and help during my study. I appreciate all the input and suggestions that helped me become a better researcher.

I want to thank Christie Digital Systems and Mark Lamm for facilitating the necessary equipment for my study.

I want to dedicate my work to my parents for supporting me in every way they could during my study. They were there for me even from thousand miles away. I appreciate you and your continuous sacrifices for me.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

3D reconstruction is a well studied field over the past decades [99, 22, 40, 138, 48]. 3D reconstruction is a core technology of many applications such as medical imaging, projection mapping, computational science, virtual reality, and digital media. The 3D reconstructed model of an object contains information about its physical appearance.

There are different ways to create a 3D model from an object, which will be discussed in detail in Chapter 2. The fundamental equipment for creating a 3D model of a scene is a light source, which can be either ambient light or light emitted from a projector, and camera(s) which capture images from the scene, as shown in Fig. 1.1. However, capturing 2D images will not be of much help without knowing the relative position between the views. This leads to the key principle in 3D reconstruction, which is the establishment of pixel correspondences. Pixel correspondences relate different pixels of an image to the corresponding pixels in another image. Finally, after pixel correspondences between the camera image(s) and projector view(s) are established, they are triangulated to find their location in 3D space.

This thesis aims to improve the speed and robustness of a recent method [27] used in single-shot structured light (SSSL) systems to create a faster 3D reconstruction of a scene, maintaining the relative accuracy of the final reconstruction. In this SSSL system a hand-designed structured pattern is projected on an object. As the name suggests, only one image is captured by a camera. The captured image is then processed to extract the necessary information including point correspondences from it and reconstruct a 3D point cloud from the object. One of the effective factors is the accuracy and acquisition speed of pixel correspondences. The accuracy of the pixel correspondences affects the accuracy of the final point cloud. On the other hand, acquiring pixel correspondences in real-time will

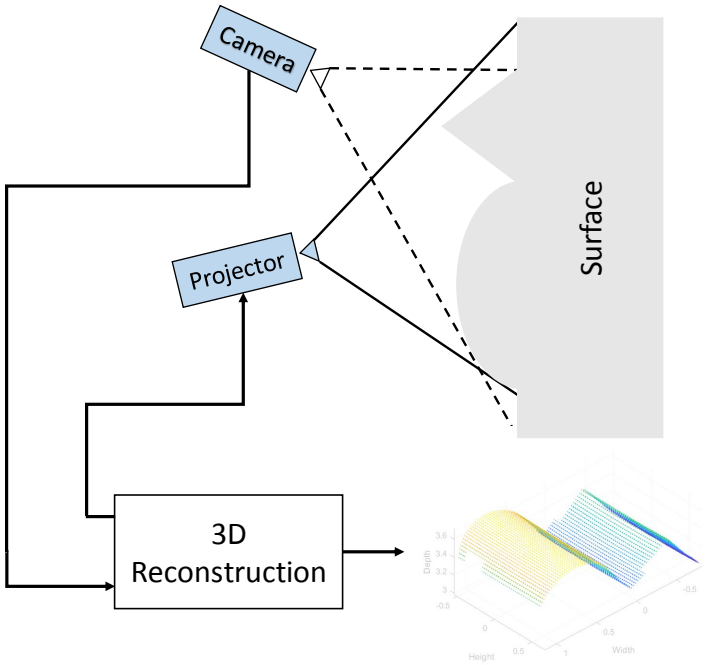

Figure 1.1: Schematics of a camera projector system

enable the process to be used in different conditions and applications.

1.1 Motivation and Overview

The equipment provided for this project is one camera and one projector. The projector is used to project a structured light pattern on an object in front of it, and an image is captured by the camera.

Pixel correspondences, or the correspondence problem in general, help associate different parts of an image to corresponding parts in another image. In a structured light pattern, such as in Fig. 1.2, the pattern is designed with a certain structure such as colored blobs or vertical and horizontal stripes that contains a variety of features that can be used as correspondences between the camera and projector views. For instance, the features in Fig. 1.2 (a) are the centroid of the colored blobs as the position of the correspondences. Their color is a factor that associates the blob in the camera image with the correct one

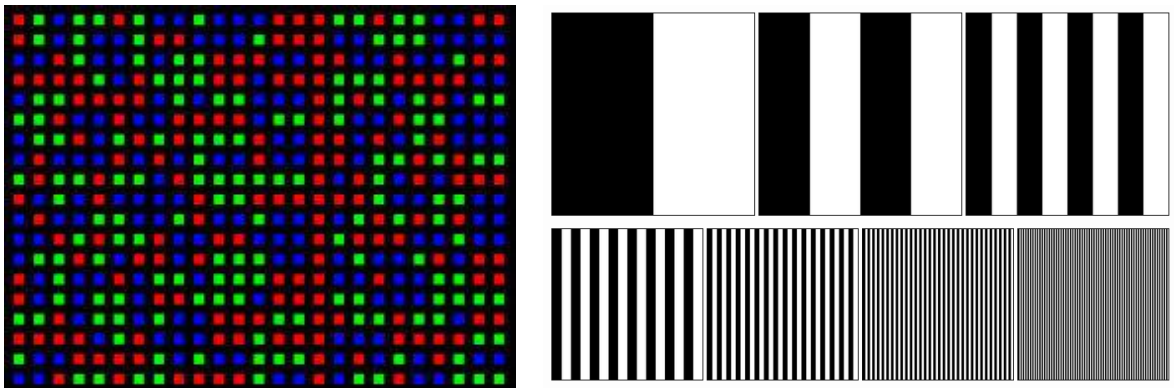

Figure 1.2: An example of a structured light pattern. Notice the structure of the patterns, color blobs separated by black grid in (a) and the number and width of the stripes in (b)

in the projector pattern. Fig. 1.2 (b) is a multi-shot pattern meaning that several patterns with specific properties are projected on an object, sequentially, and in the end each pixel will have a unique ID that associates with a pixel with the same ID in the projector pattern.

Detecting features and establishing pixel correspondences are essential parts of 3D reconstruction [22, 155]. Furthermore, establishing an accurate set of pixel correspondences can result in an accurate 3D reconstruction. However, establishing an accurate set of correspondences can be time-consuming. Some of the applications that involve moving objects, such as reconstructing the surrounding environment of a moving car, the reconstruction should be carried out quickly. Hence, designing a method that can detect and find pixel correspondences quickly will ultimately help the whole reconstruction process significantly. Different approaches have been developed to acquire accurate correspondences [94, 17, 159, 138]. However, there are some limitations to these approaches. For instance, the multi-shot approaches in [94, 17] projected sequential patterns onto an object to get a dense and accurate set of pixel correspondences. However, they projected multiple patterns onto the object which requires the object to be still at all times. The advantage of single-shot approaches in [159, 138] over the ones in [94, 17], is utilizing one image for pattern projection which enables the object to move. The single-shot approaches in [159, 138] used colored speckle and colored dot patterns which are susceptible to noise, and using color can be troublesome on colorful objects. Hence, designing a method that can detect pixel correspondences quickly and accurately and be robust to factors such as

object motions and color is an important subject of study. In this thesis, we have proposed a detection method to tackle the computation speed of the previous method in [27], which we use as baseline to compare our results. The baseline method aimed to obtain the optimal four corners of a quadrilateral bounding box that can best fit each tag. This is a time consuming process that we have aimed to improve in this thesis.

1.2 Research Contributions

We have different goals when we use 3D reconstruction and depending on what goal is in mind, different techniques are used throughout the process of reconstruction including the process of establishing pixel correspondences. For instance, the density of the point cloud might be of interest or the computational complexity of the acquired point cloud could be of importance. The types of features, susceptibility of a detection technique to noise and distortion, aspect ratio between projector and camera views, etc. are amongst the factors that determine the computational complexity of the process or the density and accuracy of the final 3D model.

The purpose of this thesis is to reconstruct a 3D model of an object relatively fast with less concern for the density of the final point cloud. Hence, the establishment of pixel correspondences needs to be done in real-time, where there is a short amount of time to capture the image and reconstruct the model, in addition to being distributed over the whole shape so that enough depth information can be recovered. We propose a technique for single-shot structured light (SSSL) where the projected pattern consists of black squares, which we call tags, containing hand-designed symbols separated by grid lines. This pattern has multiple features that allow us to use them for different purposes. The grid lines and the tags are good for detecting the correspondences while the symbols can help associating the right pixel in the camera image to its corresponding pixel in the projector pattern. This binary pattern does not have any colored pixel which can be useful when projecting on a colorful object. This pattern is also used in [26] and [27]. The correspondence technique used in [27] tries to find four points per each tag using a time-consuming optimization process that is carried on each tag individually.

To tackle the speed complexity, the proposed method finds only one point per grid intersection. This reduces the number of correspondences roughly by four times. However, the proposed method finds multiple intersections in one patch simultaneously using the Radon transform, which improves the time complexity by approximately 50 times. To facilitate the process of detection, the image is binarized using the adaptive thresholding method introduced in Section 3.1. The purpose of thresholding the image is to help ease

the detection process. In this manner, the thresholding method is improved accordingly and is tested to show the efficiency of the method. However, thresholding is not the main contribution of the thesis. The grid intersection detection method is the problem that this thesis tries to address. The method is designed and tested under different synthetic situations with various distortions added to the image to test the performance of the method. Additionally, the performance of the method is tested on real objects to prove the applicability of the method in practical situations.

1.3 Thesis Outline

This thesis consists of six chapters. Chapter 2 will provide the necessary background material for the methods introduced in Chapter 4 and 5. The problems are formulated in Chapter 3. Finally, the work is concluded and future directions are discussed in Chapter 6.

Chapter 2

Background

This chapter reviews the background material needed for understanding the contents of future chapters. In this chapter we will introduce the concept of 3D reconstruction and different approaches that are used to do 3D reconstruction in Section 2.1. Further background is provided on camera calibration systems and different thresholding methods in Section 2.1.1. The main background concepts to understand the binarization and the feature detection method proposed in this thesis are reviewed at the end of this chapter in Sections 2.2 and 2.3.

2.1 3D Reconstruction

The 3D reconstruction of objects has been widely studied for years [99, 42, 60, 108, 66, 155, 48]. Different approaches and methods have been developed to extract rich and accurate depth information from a 3D scene. Generally, 3D reconstruction can be divided into two major categories, as shown in Fig. 2.1, of active [108, 40, 155, 48] and passive [22, 60, 155] methods. Each of these fields was developed to extract information from a scene under different circumstances.

Passive methods [22, 60, 155] are developed to work only with the ambient lighting. Stereo vision is one of the most important and frequently used methods in this category. In stereo vision systems, cameras capture images from at least two different views. Different algorithms have been designed to find corresponding points between views in such systems, such as *sum of absolute intensity difference (SAD)* [78, 38], *squared intensity difference (SD)* [107, 39] and *normalized cross correlation* [39, 145]. Every algorithm makes use

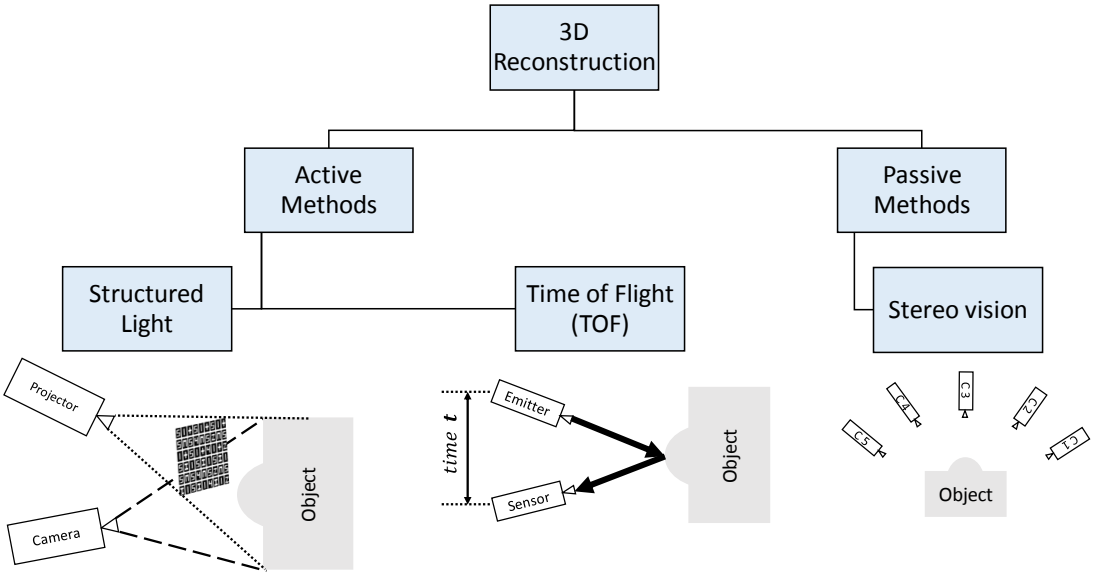

Figure 2.1: Illustration of different 3D reconstruction methods

of a matching cost function to calculate the level of similarity between different views to establish correspondences between two pixels. Passive reconstruction methods are generally fast. However, the performance of these methods is highly dependant on the richness and clarity of surface texture.

Active methods [108, 40, 155], on the other hand, actively interact with the scene. Each active method illuminates the scene and uses a technique to recover the depth from it. One of the techniques is *time of flight* (TOF) [40, 58] during which a light source illuminates a scene and a sensor collects the scattered light from the scene and calculates the depth based on the time difference from when the light leaves the device until the light reaches the sensor, as shown in Fig. 2.1. This technique does not require triangulation since the depth information is readily available and there is no need for corresponding pixels, hence, it is suitable for mobile applications. However, this method lacks sufficient accuracy in cases of short-range emission since the speed of light is high.

Structured light [108, 32] is another subcategory of active reconstruction methods which is used to recover depth information from a 3D scene. Structured light techniques are similar to stereo vision, where one of the cameras is replaced by a projector. The projector in these systems plays the role of the light source, illuminating the scene with a particular structured light which encodes information that will be used to establish corresponding

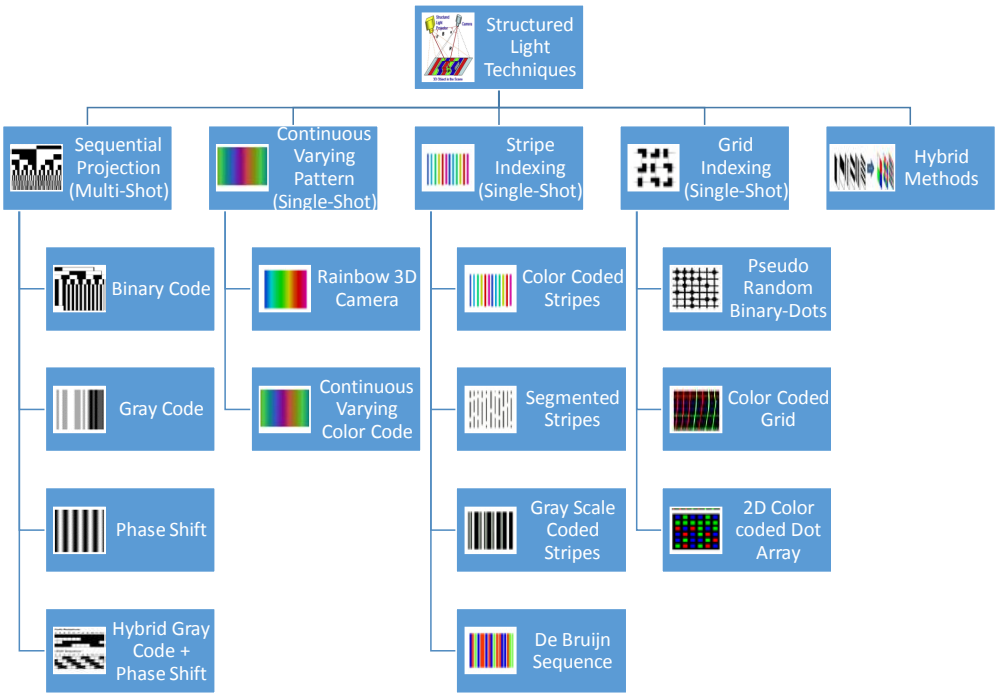

Figure 2.2: Illustration of different structured light techniques. Thumbprint images copied from [32].

pixels between camera images. Structured light comes in different models with different properties suitable for various scenarios [32]. Sequential projections or multi-shot structured light and single-shot structured light are the two major categories that structured lights can be divided into.

Multi-shot structured light (MSSL) [136, 49, 66], as shown in Fig. 2.2, can be divided into three major categories: binary patterns and Gray coding [113, 49, 66], gray level patterns [46, 32, 17] and phase shifts [156, 68, 94]. *Binary patterns* are made up of a sequence of different black and white striped patterns projected sequentially onto the surface over time. Each pattern projected on the surface is captured by a camera. After projection, each camera pixel has a binary code assigned to it which is then used to find the corresponding pixel in the projector view with the same binary code. N patterns can code 2^N stripes, which can be a high number of images compared to a technique like gray-level patterns. *Gray level patterns* are used to reduce the number of patterns needed to encode the image frame compared to binary patterns. Let us say each gray level pattern can consist of M different levels of intensity. N patterns can encode M^N stripes which compared to binary

code is a lesser number of patterns for an object depending on M. *Phase shift patterns* are another MSSL techniques during which every pixel along a line can be represented by a specific phase. Any deformation on a surface will cause the phase of the points to differ, which is recorded as a phase deviation. These deviations provide us with information about the depth of the surface.

MSSL techniques create a dense set of correspondences from a scene which in turn helps to create an accurate 3D point cloud of the scene. However, the scene needs to be completely still during the projection time, which makes these techniques unsuitable for moving objects or in cases where fast image acquisition is needed.

To solve this problem, single-shot structured light (SSSL) [79, 147, 159, 48, 138] is introduced. SSSL can be divided into three major categories of **continuous varying patterns, stripe indexing patterns**, and **grid indexing patterns**.

Continuous varying patterns [32] consist of main color channels with fixed wavelengths mixed, creating a rainbow color pattern. The fixed geometry of the rainbow light projector helps establish a one to one corresponding pixels between the projection angle and a specific wavelength of the rainbow light. Ultimately, triangulation from the known correspondences with a known baseline and known viewing angle will be straightforward. **Stripe indexing patterns** [32] try to index each stripe in the pattern which consequently makes it easy to track the stripes even though the order of the stripes in the projected pattern is not the same as the one captured by the camera. This displacement happens due to the inherent parallax caused by the angle difference between the projector and camera position. Stripe indexing patterns include *colored stripes* [10, 28], *segmented stripes patterns* [73], and *grayscale patterns* [24].

The idea behind **grid indexing patterns** [32] is to label each subwindow of the pattern so that each subwindow would be unique and identifiable with respect to its position in the pattern. This method can be divided into Pseudo-Random Arrays (PRA), Color-coded grids, and Dot arrays.

Pseudo-Random Arrays (PRAs) [61, 79, 147] are constructed by folding pseudo-random sequences [72, 71] into two dimensional arrays. These pseudo-random sequences are obtained from a primitive polynomial. The pseudo-random sequence is then used to fill an array, called a pseudo-random array. One of the constraints of pseudo-random arrays is the uniqueness of a sliding subwindow, of a certain size, in the array. By searching through the pattern and leveraging the uniqueness of subwindows in the pattern, we can locate the corresponding subwindow in the projected pattern resulting in corresponding pixels between the views. *Color coded grids* [90, 87] benefit from coloring the horizontal and vertical lines of the generated grid lines. The uniqueness of the encoded subwindows in the pattern is

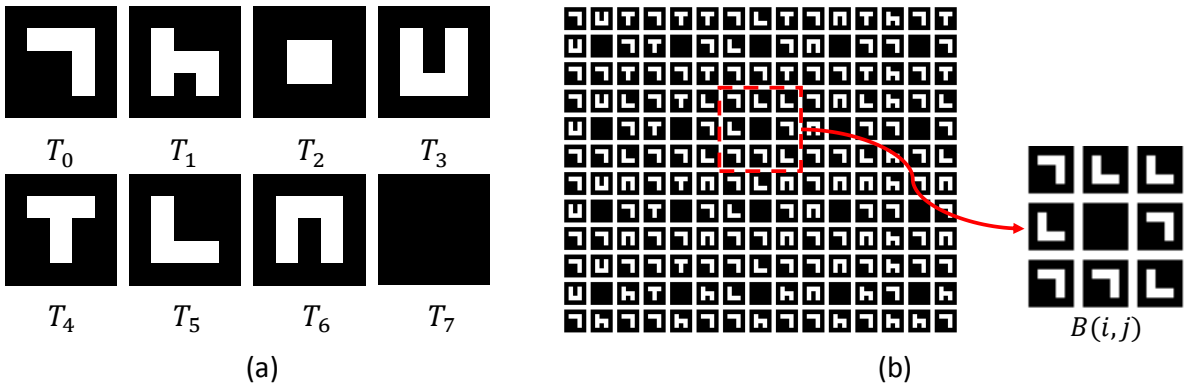

Figure 2.3: Sample tags and blocks in a block address encoding produced structured light pattern. The BAE pattern, (b), is generated using the hand designed symbols of (a). Each distinct block, $B(i, j)$, in the pattern is unique.

not guaranteed. Additionally, color encoding sequences for vertical and horizontal lines are not mandatory to differ depending on the application. Using dots and squares along with color-coded grid lines will help produce more reliable results.

All of the encoding schemes mentioned above can also be combined to broaden the applications of structured light. Combining structured light to create a sort of hybrid structure light pattern may help in some applications and produce more accurate results [32]. Figure 2.2 illustrates different types of structured light reviewed earlier along with examples of how each structured light appears.

A new approach introduced in [26] creates a grid structured binary pattern, similar to PRA, which will be used in this thesis. The methods introduced in Chapters 4 and 5 are employed and tested using this encoding scheme. Block Address Encoding (BAE) [26] generates a binary pattern consisting of unique non-overlapping blocks, where each block consists of K distinguishable tags. Each tag is identified by an integer $i = 0, 1, 2, ..., K-1$. Every distinct block has a tag inside it, one of which is a unique marker tag that is located in the center of each block and is used as the origin. The paired indices of each block, (i, j), denote the location of a block, $B(i, j)$, as shown in Fig. 2.3. Similar to PRA, uniqueness is important in this encoding scheme. However, uniqueness in BAE patterns is defined block-wise, meaning every distinct block is unique in the projected pattern, as opposed to PRA, where a sliding subwindow has to be unique. Assigning addresses to distinct blocks enables the algorithm to avoid searching through the whole pattern and directly corresponds blocks in both views using their addresses. After assigning the center marker tag, k^*, the remaining $\hat{n}^2 - 1$ tag locations and $K - 1$ symbols encode the rest of the block,

$B(i,j)$

$$B(i,j) = \begin{bmatrix} k_i^1 & k_i^0 & k_j^0 \\ k_i^0 & k^* & k_j^1 \\ k_i^1 & k_j^1 & k_j^1 \end{bmatrix} \tag{2.1}$$

Tags surrounding k^* in (2.1) are associated with specific error detection and correction digits. The purpose of error detection and correction digits is to detect any misclassified or unclassified tag caused by bad illumination or blurriness in the camera image. The block indices are converted to base $K-1$ and $[k_i^0, k_i^1]$ and $[k_j^0, k_j^1]$ from (2.1) encode the digits of the converted indices.

2.1.1 Calibration and Triangulation

One of the crucial steps in 3D reconstruction is camera calibration. Recent camera calibration studies have been done in the computer vision community [134, 140, 158, 155]. In order for camera calibration to be done, first the relative pose of the views needs to be defined. This is carried out by finding corresponding pixels between the views. Since the system used in this project utilizes structured light to calibrate the system, at least one projector must be present in the system. However, establishing pixel correspondences gives us the ability to project the camera image to the projector image and treat the projector as a camera. Hence, the camera-projector calibration becomes camera calibration, which is a well-studied field [134, 140, 158, 156, 80, 66, 48]. Henceforth, we refer to camera-projector calibration as camera calibration since we will treat the projector as a camera view after establishing pixel correspondences.

Pixel correspondences are the main tool to define the geometric transformation from one view to another. The structured light system calibration uses this tool to infer the transformation from the world to the camera and projector coordinate systems, as shown in Fig. 2.4. The camera is usually described as a pinhole model [158], which mathematically describes the transformation from the world coordinate system $[X^w, Y^w, Z^w, 1]^T$ to the 2D projected image in camera image plane coordinate system $[x^c, y^c, 1]^T$. The relationship between the 3D point and its 2D image projected point is

$$s \cdot [x^c, y^c, 1]^T = A[R|t][X^w, Y^w, Z^w, 1]^T \tag{2.2}$$

with matrix A representing the intrinsic parameters of the camera

$$A = \begin{bmatrix} f_x & \gamma & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

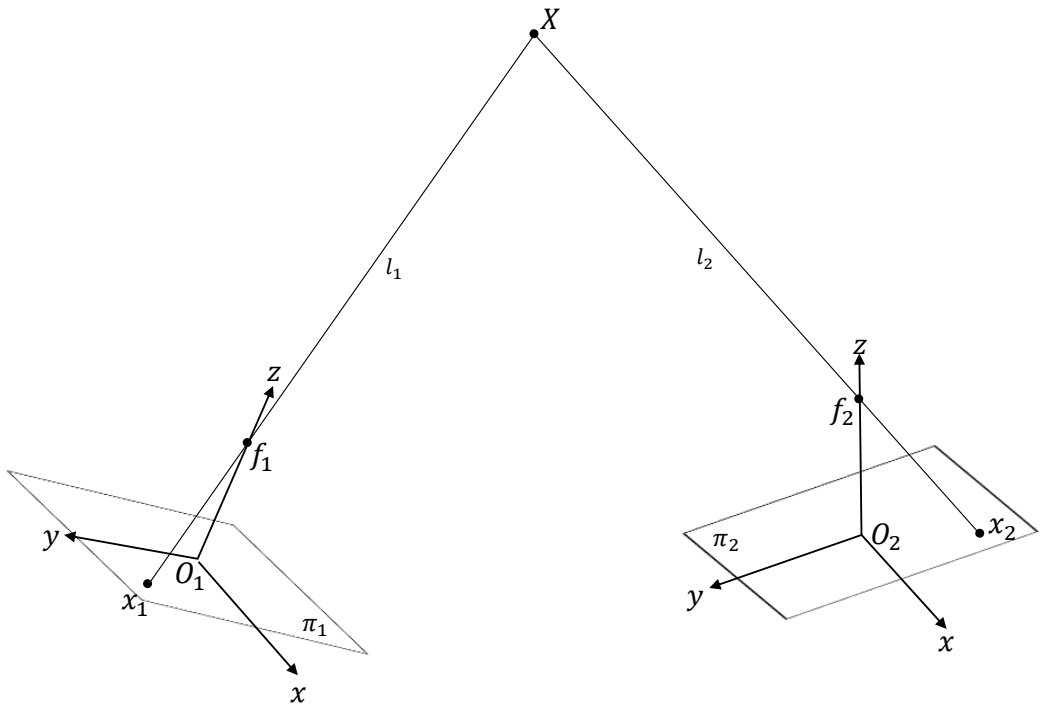

Figure 2.4: A general stereoscopic system showing the relationship between two optical sensor coordinate systems, O_1 and O_2 with focal lengths, f_1 and f_2. Two points, x_1 and x_2, represent the 2D locations corresponding to the 3D point, X. The corresponding lines, l_1 and l_2, intersect in 3D space at X. Inferring the projection from X in 3D space to x_1 and x_2 in 2D image plane is the calibration process.

The intrinsic parameters in (2.3), matrix $A_{3\times3}$, represent the inherent properties of a camera or a projector such as their focal length, f_x and f_y, measured in pixels, in the x and y direction, the principle points (p_x, p_y) and the skewness γ between the x and y axes. Without loss of generality, we can assume that the model plane is on $Z = 0$, hence, $Z_w = 0$. This will result in

$$s \cdot [x^c, y^c, 1]^T = A[R|t][X^w, Y^w, 0, 1]^T \tag{2.4}$$

which means that there exists a homography between the model plane and its image plane coordinate systems,

$$s \cdot [x^c, y^c, 1]^T = H[X^w, Y^w, 0, 1]^T \quad \text{with} \quad H = [R|t] \tag{2.5}$$

Using the homography in (2.5), one can find the transformation from the 3D lens coordi-

nate system to the 2D image coordinate system of the camera up to a scale. $[R|t]$ denotes the rotation, $R_{3\times3}$ and translation, $t_{3\times1}$ matrices from which the world coordinate system is related to the camera coordinate system. Using the extrinsic parameters, the transformation from the 3D world coordinate system to the 3D camera lens coordinate system is achievable. Additionally, the formula in (2.2) can be written as

$$s \cdot [x^c, y^c, 1]^T = P[X^w, Y^w, Z^w, 1]^T \tag{2.6}$$

where $P_{3\times4}$ denotes the transformation matrix from the 3D world coordinate system to the 2D image coordinate system, up to a scale, which is called the camera matrix [42]. It should be noted that the coordinate systems are all represented in homogeneous coordinates in (2.6), the additional 1 for the last element of coordinates is added so that the extrinsic parameters can be represented in a single matrix [5]. The camera matrix, P, is in principle a homography which maps the 3D world coordinates, $\tilde{X}^w$, to the 2D image coordinates, $\tilde{x}^c$, similar to that in (2.5). Camera calibration will then boil down to minimizing a cost function, as in (2.7), over the intrinsic and extrinsic parameters or, in other words, the homography from the world to the camera image plane.

Having N correspondences between at least two views, $x_1 \leftrightarrow x_2$, as shown in Fig. 2.4, the goal is to minimize the distance between the correspondences and their projection from the 3D model,

$$\arg\min_{A,R,t} \sum_{i=1}^{M} \sum_{j=1}^{N} \|x_{ij} - \hat{x}(A, R_i, t_i, X_j)\|^2 \tag{2.7}$$

where M denotes the number of views and N denotes the total number of correspondences in the j_{th} view.

As mentioned in Section 2.1, MSSL techniques establish a dense set of correspondences which in turn creates a richer distribution of samples to pick from and calibrate the system. However, the inability of MSSL techniques to perform on moving scenes leads to using SSSL techniques that do not result in a dense set of correspondences, hence there are not as many samples to calibrate the system, which magnifies the importance of correspondence accuracy.

The camera calibration process is followed by triangulation which is trivial in principle [44]. Each point in a plane corresponds to a line in space, as illustrated in Fig. 2.4, thus corresponding pixels in two views, $x_1 \leftrightarrow x_2$ have corresponding lines in space, $l_1 \leftrightarrow l_2$. Considering that the views are angled towards the 3D shape in front of them, l_1 and l_2 would intersect at a point in 3D space, X, that lies on the 3D shape. In order to find the corresponding lines, camera matrices P and P' are needed which are computed by the

calibration process described in (2.7) where P is divided into intrinsic, A_i and extrinsic, R_i and t_i, components in the equation. Similar to camera calibration, triangulation has to deal with the misplacement in the correspondences, hence the corresponding lines $l_1 \leftrightarrow l_2$ do not necessarily intersect in the 3D world coordinate system. However, part of the misplacement may be already taken care of since during calibration one can also formulate a constraint for correspondences, which is another optimization trying to minimize the distance between the established correspondences in each view, x'_{1i} and x'_{2i}, and the correct ones, x_{1i} and x_{2i}, as they did in [66].

$$\min \sum_{i=1}^{N} |x_{1i} - x'_{1i}|^2 + |x_{2i} - x'_{2i}|^2 \tag{2.8}$$

Minimizing (2.8) we will obtain the closest estimate of the true correspondences, which can be used to estimate the fundamental matrix by:

$$x_{2i}^T F x_{1i} = 0 \tag{2.9}$$

Numerous triangulation techniques have been proposed over the years [43, 44, 42, 93, 45, 148] that used different techniques in order to optimize the estimated location of the points in 3D space.

2.2 Image Binarization

Image binarization is the simplest segmentation method [118] that has been used for various applications such as target detection [9, 128], printed character extraction [54, 1], quality inspection of materials [119, 117], cell images [91], segmentation of various image modalities for nondestructive testing [35, 52], and many other applications. In principle, every thresholding method, regardless of performing locally or globally, assigns a threshold to each pixel in the image to **binarize** the pixel values. One of the major categories of thresholding methods is **Global thresholding** [85, 63, 15] where a single threshold value, T, is assigned to compare every single pixel in the whole image, I, to and binarize the pixel accordingly:

$$I'(x,y) = \begin{cases} 1, & \text{if} \quad I(x,y) \geq T \\ 0, & \text{if} \quad I(x,y) < T \end{cases} \tag{2.10}$$

The other major category is **Local thresholding** [112, 11, 103] where different areas of the image are treated differently in terms of assigning a threshold value, meaning that if there are variations in intensity over the whole image, we cannot assign a single value as

a threshold. Therefore, a value should be assigned to a certain area according to the pixel intensity distribution of that area.

$$I'(x,y) = \begin{cases} 1, & \text{if} \quad I(x,y) \geq T(x,y) \\ 0, & \text{if} \quad I(x,y) < T(x,y) \end{cases} \tag{2.11}$$

Global Thresholding

Image thresholding is often used to preprocess an image for subsequent operations, depending on the application. Global thresholding methods use different techniques to assign one value as a threshold to every image pixel. These techniques can be classified into different categories such as:

Clustering-based thresholding methods [64, 150] cluster gray level pixels into two sets corresponding to two lobes of the histogram. There is a disadvantage of these methods since the intensity distribution may not be two lobes. Some of these methods simply pick the midpoint of the peaks of each cluster.

$$T = (p_1 + p_2)/2 \tag{2.12}$$

where p_1 and p_2 are the peaks of each cluster. Another group of clustering based methods try to minimize the weighted sum of the within-class variances of the foreground and background pixels (two lobes of the histogram). One of the most common global thresholding methods introduced by Otsu [85] falls under this subcategory, which defines three criteria based on which an optimization is done and a threshold value is obtained. These criteria are

$$\lambda = \sigma_B^2/\sigma_w^2 \qquad \gamma = \sigma_t^2/\sigma_w^2 \qquad \eta = \sigma_B^2/\sigma_t^2 \tag{2.13}$$

The optimization tries to maximize any one of the criteria in (2.13) where σ_w, σ_B^2 and σ_t^2 denotes the within-class variance, the between-class variance and the total variance in pixel intensity levels, respectively [30]. Since these three criteria are dependent on one another,

$$\sigma_t^2 = \sigma_w^2 + \sigma_B^2 \Longleftrightarrow \gamma = \lambda + 1, \eta = \lambda/(\lambda + 1) \tag{2.14}$$

therefore, by maximizing one the optimum value of others is automatically achieved. It is noted that σ_w^2 and σ_B^2 are functions of threshold level T, but σ_t^2 is independent of T. The chosen criterion in [85] is η due to its simplicity with respect to the threshold value T since σ_B^2 is a function of the threshold value whereas σ_t^2 is not.

$$\eta(T) = \sigma_B^2(T)/\sigma_t^2 \tag{2.15}$$

$$\sigma_B(T^*)^2 = \max_{1<T<G} \sigma_B(T)^2 \tag{2.16}$$

where G is the maximum gray level in the image. Maximizing η, or equivalently maximizing σ_B^2 in (2.15), will result in the optimum threshold value T^*. Other subcategories of Clustering-based thresholding methods are minimum error thresholding [57, 53] and fuzzy clustering thresholding [50, 3].

the second global thresholding technique is **Entropy-based methods**, which try to maximize the entropy between the foreground and background resulting in maximum information being preserved during thresholding [88, 153]. **Histogram shape-based** techniques [101, 37, 142] find the threshold value based on the shape properties of the image histogram. **Thresholding based on attribute similarity** select the threshold value based on the quality or similarity of attributes in gray level and binarized image. These attributes can be edge matching [110, 122], gray level moments [135, 122], and connectivity [86, 63].

Locally Adaptive Thresholding

All of the methods discussed in Section 2.2, output a single value, T, as the threshold value for thresholding an image, which can be useful in stabilized conditions in terms of illumination where the image has a relatively smooth distribution of pixel intensities. Binarizing a document image in the stabilized lighting condition of a lab for text detection can be one of the cases where global thresholding performs reasonably well [118]. However, these methods would not perform well under conditions of high illumination variation. In other words, if the distribution of the pixel intensities in different areas in the image have different foreground and background intensity distributions or the distribution histogram does not contain two clear classes, where they have high overlap or more than two clusters are present, these methods might not perform well. Therefore, to handle these shortcomings, local thresholding methods are used.

Local thresholding methods, assign a threshold value, $T(x, y)$ to each pixel, (x, y) in the image based on **local statistical properties** of their neighborhood such as variance [84, 112, 14] and contrast [143, 8, 126].

The local thresholding method in [84] uses the mean, $m(x, y)$, and standard deviation, $\sigma(x, y)$, of a local region to adapt threshold values to the pixel in the region.

$$T(x, y) = m(x, y) + k\sigma(x, y) \tag{2.17}$$

where k is a bias parameter to adjust the threshold value above or below the mean, $m(x, y)$. Another local thresholding method that uses local variance is introduced in [112] where the effect of standard deviation is adjusted based on the dynamic range, R in the image.

$$T(x, y) = m(x, y)\left(1 - k\left(1 - \frac{\sigma(x, y)}{R}\right)\right) \tag{2.18}$$

The similarity between (2.18) and (2.17) suggests that the method in [112] is an improved version of [84] specifically designed for stained and badly illuminated images [118]. Different experiments suggest different values for k (including our own where $k = -0.1$) were satisfactory in (2.18). In [133] the value of $k = 0.2$ is chosen and in [126] a range of $k = [0.2, 0.5]$ has achieved optimum results. As suggested the value of k is highly case dependent. In (2.18) k can have positive values, where the threshold is set below the mean or negative values meaning the threshold is set above the mean. Although the value of k in (2.17) is still case dependent, the effect of negative and positive values of k is the opposite way around than (2.18). This means that parameter, k needs to be fine-tuned by the user to result in an optimum thresholded image. In Chapter 4 we introduce a new adaptive method where this value is obtained automatically. These local thresholding methods fall under *Local variance methods*.

Another subcategory of the local thresholding methods is *Local contrast* methods. Bernsen's method [8] is one of the most used methods under this subcategory. It is a local thresholding method that compares the contrast of each pixel with its neighboring pixels in a window. A threshold value is set at the midrange of the minimum, $I_{min}(x, y)$ and maximum, $I_{max}(x, y)$ intensities in the neighborhood.

$$T(x, y) = \frac{I_{min}(x, y) + I_{max}(x, y)}{2} \tag{2.19}$$

However, if the difference in contrast is lower than a certain value, γ then the whole neighborhood, $N(x, y)$ is considered to be consisting of only one class of pixels depending on $T(x, y)$.

$$\begin{aligned} &I_{max}(x, y) - I_{min}(x, y) < \gamma \\ &N(x, y) = \begin{cases} 1, & \text{if} \quad T(x, y) \geq \frac{V_{max}}{2} \\ 0, & \text{if} \quad T(x, y) < \frac{V_{max}}{2} \end{cases} \end{aligned} \tag{2.20}$$

where V_{max} is the maximum intensity value in the image. Locally adaptive thresholding methods have the advantage of doing statistical operations in local neighborhoods thus they are more robust to noise and intensity variations than global thresholding methods. However, since the operations are being done locally the distribution of the pixel intensities

plays a crucial role in the quality of their performance. Hence, choosing the right window size can affect the performance of the method.

The reviewed locally adaptive thresholding methods in this section need to be fine tuned by the user. For instance, the bias parameter, k, in both Niblack [84] and Sauvola-Pietikäinen [112] needs to be fine-tuned by the user before applying to image. For this purpose, we have proposed our method in Chapter 4 to automatically pick the optimum values for the method in each local region.

2.3 Low Level Image Feature Detection

The transformation between the views in a camera calibration system boils down to mapping certain pixels, which are also called key points, in one view to the corresponding pixels in the other view. To find the corresponding pixels, corresponding features need to be detected. In the field of computer vision, a feature is defined as a piece of information that indicates a certain region in an image with certain properties. In the context of an image, low-level features can be described as pixels, edges, lines/curves [67]. SIFT [83, 56, 131], SURF [7, 23, 131], and ORB [102, 81, 131] are more recent feature detectors and consist of two parts; one is a key point detector, which detects the location of the pixel and the second is a feature descriptor, which describes the surrounding neighborhood of the pixel. Traditional feature detectors only detect and locate the key points in an image and they don't have a descriptor. These detectors include the Harris corner detector [41, 21] and Shi-Tomasi detector [123, 12]. In this chapter we only review the key point detection part of the feature detectors since the descriptor used in our project is defined based on classifying tags and is done differently from the algorithms in SIFT, SURF and ORB.

One of the most common feature detectors in computer vision is the SIFT detector [6, 149, 55, 131], the scale-invariant feature transform, that tries to convert an image, I, to a set of key points, (x, y), and describe each key point with a descriptor vector. For a given image, I, the key point, (x, y) is achieved by blurring the image with a Gaussian filter with different scale factors, σ, and computing the difference of intensity between them to locate the points that stand out, which are the points along the edges in the image. These points are the points of interest:

$$\begin{aligned} L(x, y, \sigma^2) &= G(x, y, \sigma^2) * I(x, y) \\ D(x, y, \sigma^2) &= L(x, y, k\sigma^2) - L(x, y, \sigma^2) = (G(x, y, k\sigma^2) - G(x, y, \sigma^2)) * I(x, y) \end{aligned} \tag{2.21}$$

where $G(x, y, \sigma^2)$ is the Gaussian filter, with a certain variance, σ^2, which is convolved with the image $I(x, y)$, and $L(x, y, \sigma^2)$ is the scale space associated with the image. The

difference of scale spaces, $D(x, y, \sigma^2)$, highlights the points of interest in the image. The same operation is done on different scales of the image as well to account for variation in distance between the camera and an object. These operations direct the focus on the changes in the pixel intensities and away from their absolute value, which makes this algorithm robust to different illuminations and scales.

Older feature detectors that have been used in many studies to compare against newer feature detectors [114, 75, 76, 129] are the Harris [41] and Shi-Tomasi [123] corner detectors.

The Harris and Shi-Tomasi feature detectors have similar basic principles to detect key points in the image. Both methods leverage the gradients in a local region of an image to detect the intensity changes in the region. The difference between these methods is the way they determine a point in an image to be a key point. As mentioned before, the principle is to detect intensity changes in a local region in an image. These changes can be described as

$$f(x, y) = \sum_{(u,v)\in N(x,y)} (I(u, v) - I(u + \delta_u, v + \delta_v))^2 \tag{2.22}$$

where $f(x, y)$ is the intensity change in the surrounding neighborhood of the key point location, (x, y) and δ_u and δ_v are the shifts in horizontal and vertical directions. The squared difference between the image, $I(u, v)$ and its shifted version, $I(u + \delta_u, v + \delta_v)$, yields the information about the gradients of the image, I in the local region. Using a Taylor expansion, we can approximate the shifts in both direction as

$$I(u + \delta_u, v + \delta_v) \approx I(u, v) + \begin{bmatrix} J_x & J_y \end{bmatrix} \begin{bmatrix} \delta_u \\ \delta_v \end{bmatrix} \tag{2.23}$$

where $[J_x J_y]$ is the Jacobin matrix, which contains the first-order partial derivatives of the image, I, in both directions. Substituting (2.23) into (2.22) results in

$$f(x, y) = \sum_{(u,v)\in N_{x,y}} (\begin{bmatrix} J_x & J_y \end{bmatrix} \begin{bmatrix} \delta_u \\ \delta_v \end{bmatrix})^2 \tag{2.24}$$

Now the intensity changes, $f(x, y)$ are (2.24) is only dependent on the neighboring pixels, $N(x, y)$. Since the shifts, δ_u and δ_v, are not dependent on (x, y) the summation can be moved into the Jacobian matrix:

$$f(x, y) = \begin{bmatrix} \delta_u \\ \delta_v \end{bmatrix}^T \underbrace{\begin{bmatrix} \sum_N J_x^2 & \sum_N J_x J_y \\ \sum_N J_x J_y & \sum_N J_y^2 \end{bmatrix}}_{\mathrm{M}} \begin{bmatrix} \delta_u \\ \delta_v \end{bmatrix} \tag{2.25}$$

where M is the structure matrix containing the gradient information of the region surrounding the key point, (x, y). Matrix M essentially contains all the information about the gradients in the local region in both directions. In other words, the partial derivatives in M can give us the necessary information about the magnitude of gradients and their directions in a local patch to detect and locate a corner.

The Harris corner detector uses a criterion, R to determine whether a point should be counted as a key point. It uses the determinant and trace of matrix M, which are related to its eigenvalues.

$$R = det(m) - k(\text{trace}(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{2.26}$$

where k is a weighting factor. Essentially, if (x, y) is a corner then the intensity change or gradient in both direction has a large value hence the eigenvalues of M would be large for both directions and J_x and J_y will be large, whereas if (x, y) is on an edge or a flat region these value would not both be large.

$$(x, y) = \begin{cases} \text{flat region} & \text{if} \quad \lambda_1 \approx \lambda_2 \approx 0 \\ \text{edge} & \text{if} \quad \lambda_1 \gg \lambda_2 \quad or \quad \lambda_1 \ll \lambda_2 \\ \text{corner} & \text{if} \quad \lambda_1 \approx \lambda_2 \gg 0 \end{cases} \tag{2.27}$$

Shi-Tomasi [123] uses the same parameters to calculate the criterion. However, it finds the minimum eigenvalue, which we denote here as λ_{min}, of the structure matrix, M, and defines the property of a point based on one value. Based on the value of λ_{min}, a pixel would be defined as a corner if minimum eigenvalue λ_{min} is bigger than a threshold T.

These feature detectors leverage image gradients or difference of Gaussians in order to find the candidate corner detectors.

The recent feature detectors such as SIFT [83], SURF [7] and ORB [102] have powerful properties that makes them suitable for feature detection. However, the BAE pattern that we have used in this project has defined structure which makes it easier to detect features. Additionally, the keypoint detector in those feature detectors detect keypoints based on the intensity changes in a local region, which will in turn detect feature points along the edges of the tags and the edges of the symbols. This introduces many unwanted outliers, which are difficult to remove. Therefore, we propose a method that can detect the feature points according to the defined structure of the BAE pattern and is fast enough to process an image in real-time The biggest and brightest feature in the BAE patterns is the grid. Therefore, detecting the grid and trying to find features in it makes more sense. Since grid lines are essentially horizontal and vertical lines separating the tags, fitting lines to the grid

seems a logical approach which is studied in this thesis. One of the common algorithms in line detection is the Hough transform, which was introduced by Paul VC. Hough [47] in 1962.

2.3.1 Hough Transform

The Hough transform has been used in different applications such as object detection [125, 132], medical imaging [25, 157], robot navigation [51, 34].

The simplest utilization of the Hough transform is for line detection, where a line is defined in polar coordinates as

$$\rho = x \sin\theta + y \cos\theta \tag{2.28}$$

where x, y are the pixel coordinates, and ρ is the distance between the origin and the

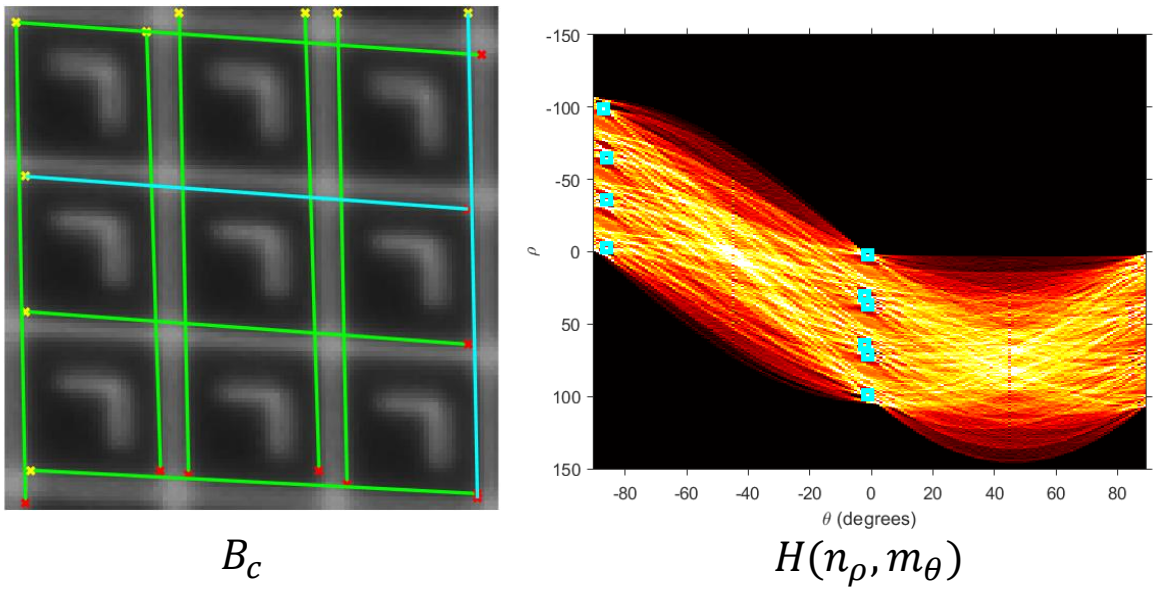

Figure 2.5: Illustration of a sample patch, B_c with grid lines separating the tags and the corresponding look-up table obtained from the Hough transform, $H(n_\rho, m_\theta)$. Notice the intersection of the curves that are bright, which we call maxima. Shown in cyan are the desired maxima in $H(n_\rho, m_\theta)$, which correspond to the lines in B_c. The longest horizontal and vertical lines are shown in cyan in B_c.

line, and $-90° \leq \theta < 90°$ is the angle between the perpendicular vector and the positive direction of the x axis.

Let H be a Hough transform matrix consisting of rows and columns corresponding to ρ and θ values, respectively. Let N_ρ and M_θ denote the number of different ρ and θ values, respectively. Let $n_\rho = 1, 2, ..., N_\rho$ and $m_\theta = 1, 2, ..., M_\theta$ be the indices of different ρ and θ, respectively. Now, for a given pixel, one can compute the value of ρ, n_ρ, at every possible angle θ, n_θ, and the corresponding element at the accumulator matrix, $H(n_\rho, m_\theta)$, can be incremented by one as:

$$H(n_\rho, m_\theta) = H(n_\rho, m_\theta) + 1 \tag{2.29}$$

Obtaining the look-up table, $H(n_\rho, m_\theta)$ simply contains the information about the summation over pixel intensities along every line. A maximum in this look-up table would be a line in the image along the pixels with the highest intensity values, as shown in Fig. 2.5. Thus the challenging process of global line detection is converted to a simple peak detection process.

2.3.2 Radon transform

The Radon transform [96] can be considered as a branch of the Hough transform in our grid detection application. The Radon transform is tool that is used for medical MR images, as well as other disciplines, including radar, geophysical and medical imaging [4] and some structured light applications [20, 74]. The basic principle of the Radon transform is to consider a ray going through an object as a line which is defined based on two parameters, ρ and θ, The Radon transform calculates the accumulated energy of this ray going through the object. In our case, the ray going through an object would be a line in an image patch and the absorbed energy is translated to summing up the pixel intensities along the line. Given an arbitrary line L with slope s and intercept t, the Radon transform is defined by:

$$\mathcal{R}(\theta, t) = \begin{cases} Radon(y = \tan(\theta)x + t, B), & \theta \in \left[\frac{-\pi}{4}, \frac{\pi}{4}\right], \\ Radon(x = \cot(\theta)y + t, B), & \theta \in \left[\frac{\pi}{4}, \frac{3\pi}{4}\right], \end{cases} \tag{2.30}$$

Two look-up tables are constructed in (2.30), which separately captures the horizontal and vertical lines with the defined range of the angle, θ. The constructed look-up tables similar to that created in the Hough transform are constructed, which simply contain the information about the summation over pixel intensities of every line in each direction. A maximum in each look-up table corresponds to a line in the image along which lies the

pixels with the highest intensities. In Chapter 5, we use these look-up tables achieved from the Radon transform to find the peaks which give us the equation of the desired lines in the image. Therefore, the challenging process of global curve detection is converted to a simple peak detection process.

2.4 Conclusion

In this chapter we reviewed the basic concepts of 3D reconstruction and camera calibration. Additionally, we reviewed different techniques of global and local thresholding in Section 2.2, which need user interference and fine tuning parameters. Due to this disadvantage, we propose a new method in Chapter 4 that does not need any of the parameters to be fine tuned. In Section 2.3, we reviewed some of the recent non-network object detectors such as Harris corner detectors and SIFT feature detectors. Although these feature detectors are used commonly in various projects, they are hard to restrict to find specific structures such as the grid intersections in the BAE pattern. Therefore, we have used the Radon transform in our detector to address this challenge.

Chapter 3

Problem Formulation

As mentioned in Chapter 1, one of the main principles to reconstruct the 3D model of a scene is establishing pixel correspondences between projector-camera views. The purpose of pixel correspondences is to define the relationship between the light source, projector, and camera.

In this thesis, the Block Address Encoding (BAE) pattern defined in Section 2.1 is used, which is basically a set of symbols that are separated by horizontal and vertical lines. This pattern has multiple features that allow us to use them for different purposes. The grid lines and the tags are good for detecting the feature points while the symbols can help associating the feature point in the camera image to its corresponding pixel in the projector pattern after the feature points have been detected. This binary pattern does not have any colored pixel which can be useful when projecting on a colorful object. Ultimately, different features can be detected for establishing pixel correspondences between the views,

Table 3.1: Formulation notations

(x, y) & (i, j)	pixel coordinates
T	threshold value
I_c	camera image
I'_c	binarized camera image
$\mathcal{M}$	grid line map
I_p	projector image
θ	angle of a line
ρ	distance between a line and center of an image

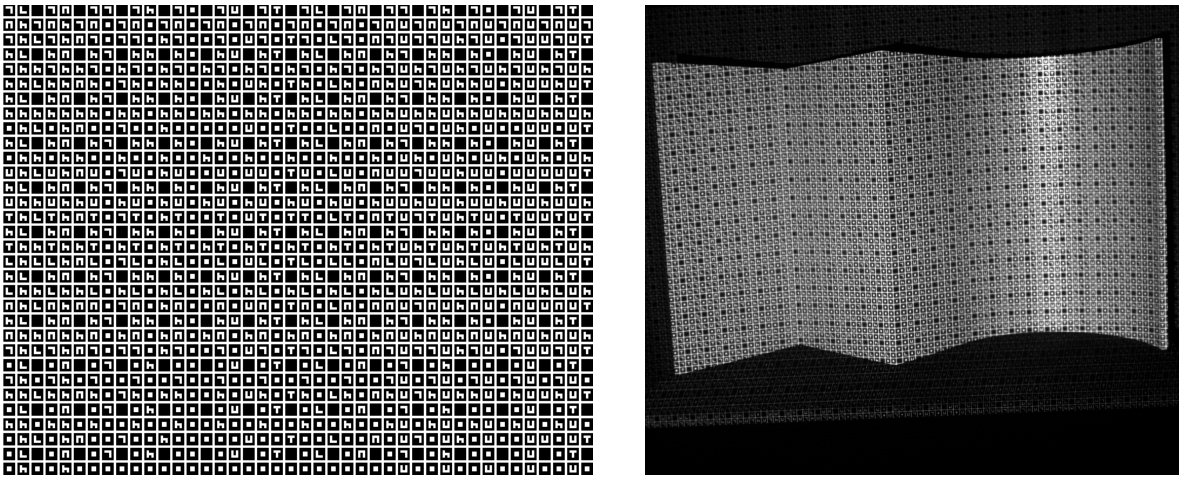

Figure 3.1: Illustration of a projected pattern, I_p using the BAE method [26] and corresponding camera output image, I_c.

such as the top left corner of each tag, four corners of the tags, and the intersection of the grid lines. The proposed detection method detects grid intersections which are then used to generate pixel correspondences.

In this chapter, we formulate the methods including a preprocessing binarization method and grid intersection detection method.

3.1 Thresholding

As mentioned at the beginning of this chapter, the goal of this thesis is to detect grid intersections of the camera image(s), which would eventually be used as pixel correspondences in inferring 3D structure. The rationale behind binarization is that the projected pattern is a binary image projected on a scene from which camera images are taken, which are gray level images as shown in Fig. 3.1. In order to reduce the effect of the noise in the gray-scale image, normalizing the pixel intensities to some degree and also, have the camera image and the projector image in the same class would enable us to compare them and assess the performance. Thus, binarizing the image would enable us to cast these images into the same class. By utilizing a good thresholding method the end image should have the tags completely separated with the symbols being completely contained in the boundaries of

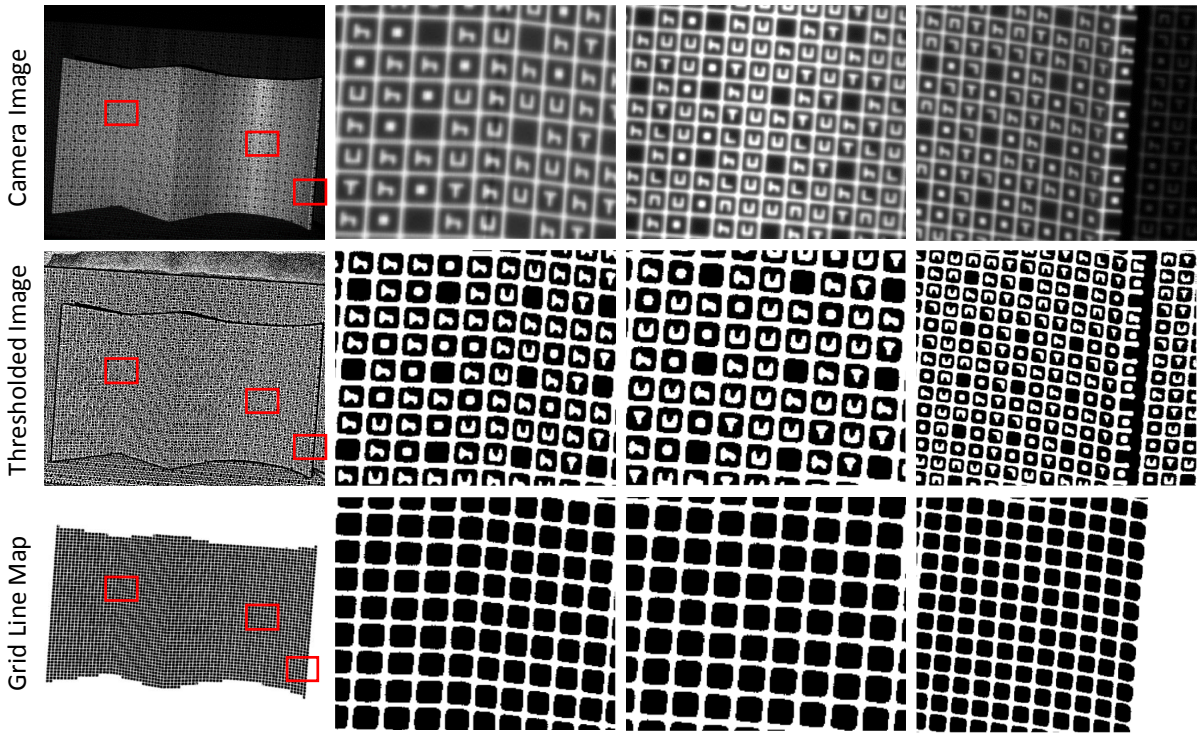

Figure 3.2: Example showing the preprocessing done on the input image, I_c. Symbol removal is carried out using connected component analysis and the grid line map, $\mathcal{M}_c$ is constructed. The tags on the background in $\mathcal{M}_c$ are removed using a mask created during the process of connected component analysis.

the tag, I'_c in Fig. 3.2. The reason for that is for the connected component analysis to be able to distinguish symbols and tags from the grid and remove the symbols from the tags which result in an image containing only the black squares separated by the grid, $\mathcal{M}_c$, as shown in Fig. 3.2.

Thresholding the camera-captured image can be done in a global or local manner, as discussed in Section 2.2. Binarizing an image globally is definitely advantageous in some aspects, such as low computational complexity. However, these methods do not always perform the way that is desired under different circumstances. For instance, these methods would not perform well when there is high illumination variation in the image, meaning that some areas of the image have low pixel intensity differentiation between foreground and background, as opposed to other areas which have high differentiation, as shown in

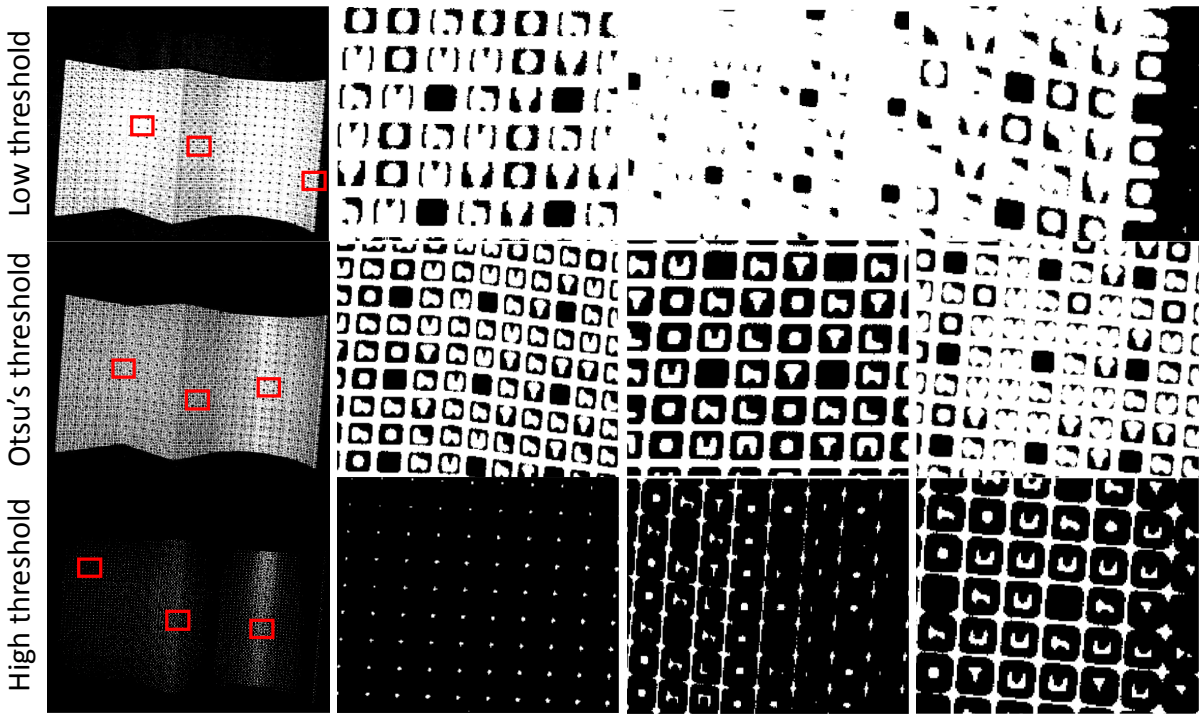

Figure 3.3: Global thresholding performance with three different thresholds along with three zoomed-in areas for better illustration.

Fig. 3.3. In Fig. 3.3, a sample camera image is thresholded with three different thresholds,

$$I'_c = \begin{cases} 1, & \text{if} \quad I(i,j) > T \\ 0, & \text{if} \quad I(i,j) \leq T \end{cases} \tag{3.1}$$

In the first row of Fig. 3.3, labeled as low threshold, the threshold is set in a lower part of the dynamic range of the image. The second row chooses the threshold level based on Otsu's automatic global thresholding method [85], reviewed in Section 2.2. The third row indicates the result of setting a threshold in the higher part of the dynamic range.

In all three cases in Fig. 3.3, it can be observed that the thresholded image does not have the quality that we are looking for. The best performance out of these three cases is Otsu's thresholding method, but the image is poorly binarized in some areas. In the other two cases, which show a higher and lower threshold value, the quality of the binarized image is worse.

The other approach to thresholding is local thresholding, reviewed in Section 2.2, which assigns a threshold value to each pixel depending on a local property of the image:

$$I'_c(i,j) = \begin{cases} 1, & \text{if} \quad I(i,j) > T(i,j) \\ 0, & \text{if} \quad I(i,j) \leq T(i,j) \end{cases} \tag{3.2}$$

where $T(i,j)$ is computed based on some local property of the neighborhood of $I_c(i,j)$. The produced grid line map, $\mathcal{M}_c$, should have several properties:

1. All of the symbols are removed from the binarized image.
2. Grid lines are preserved.
3. The shape of the tags is retained.

Constructing a grid line map, $\mathcal{M}_c$ is specifically implemented to improve the performance of the proposed detection method as explained in Chapter 5. In order to have a better detection performance, the image is cleared from every feature except the grid lines.

A locally adaptive thresholding method accounts for varying intensity differentiation in different areas of a camera image I_c. Additionally, a good quality binarized image will result in a good quality grid line map as shown in Fig. 3.2. However, since the thresholding method is implemented as a preprocessing step in the pipeline it is important that the method be completely automatic. In other words, the method should not need any input from the user and should only rely on the information extracted from the image itself and its local properties. A new locally adaptive thresholding method is proposed in Chapter 4.

3.2 Grid Intersection Detection

The initial step to obtaining pixel correspondences is to have at least one camera image from the scene to correspond to features from the projector pattern, I_p.

In SSSL systems, there is only one image captured per camera from the scene. The pattern projected onto the scene has certain features which can be used to establish corresponding pixels between the views. The information from the scene lies in the acquired image by the camera(s). Hence, the captured image(s) must undergo a detection process to detect points to use as potential pixel correspondences.

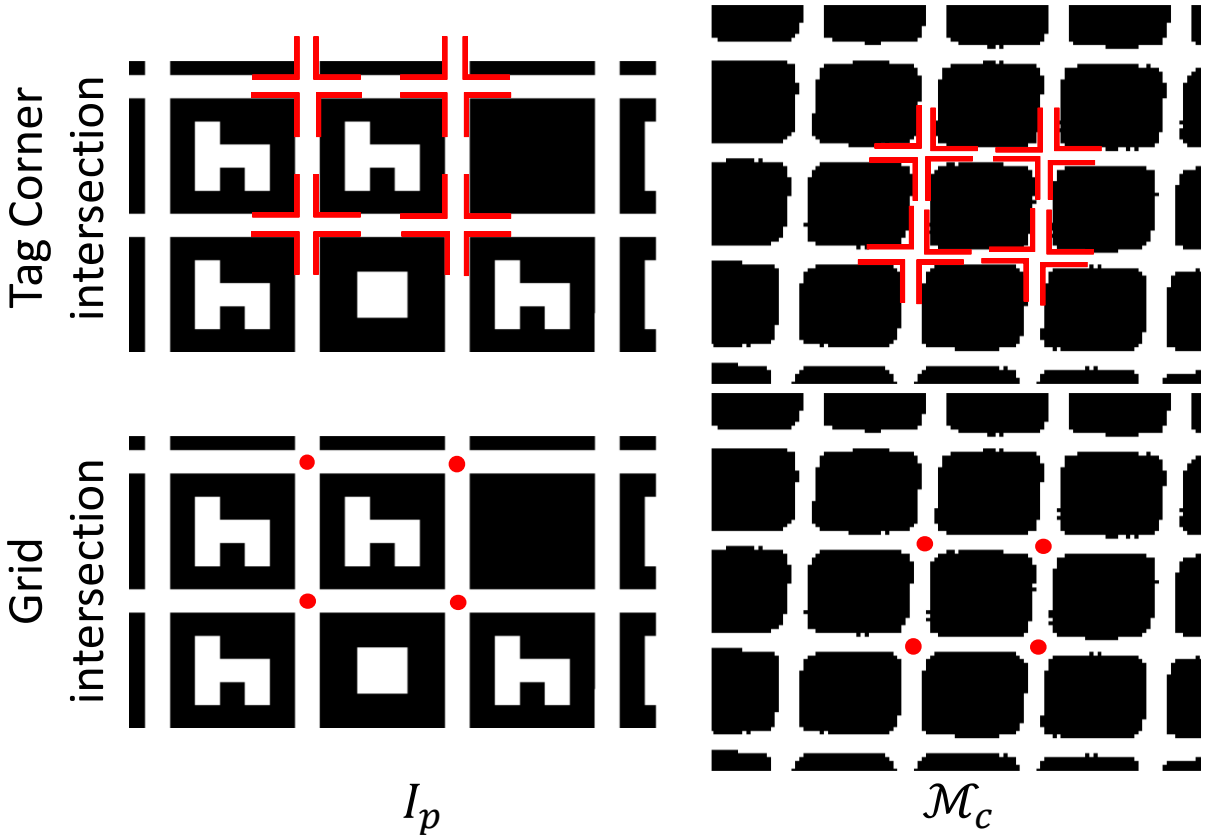

Figure 3.4: Illustration of two different techniques for grid intersection detection: Tag Corner intersection detection and Grid intersection detection. The left column shows the grid intersections using the two Grid-Tag and grid intersection approach on the projector pattern, I_p and the right column shows the two approaches on a sample grid line map, $\mathcal{M}_c$.

As mentioned in Section 2.1, there are several features to detect in a BAE binary pattern, one of which is grid intersections. As shown in Fig. 3.1, the grid is the biggest feature present in the image and the grid intersections are effective features to detect. For the sake of argument, let us assume that a good quality binarized image has been achieved and a final grid line map, $\mathcal{M}_c$, is produced, as shown in Fig. 3.2.

Grid intersection detection can be looked at from multiple aspects. There are two locations that one might call grid intersection as indicated in Fig. 3.4. One is the intersection of the horizontal and vertical edges of the tags which results in four intersections for each tag, or in other words, corners of each tag. Two is the intersection of the intersecting grid

lines, which is located in the center of the intersection. The former approach of finding tag corners is not easy and this is due to the curved corners and jaggy edges of the tags, as shown in Fig. 3.4. The jaggedness in the thresholded image is the result of binarizing the gray level image. Additionally, a tag is basically a squared shape, which can be deformed by projective transformations or discontinuities in the image and in turn, adds more challenge to detecting the corners of the square. However, Since an intersection is a point rather than a square, it is robust to any projective transformation. it is also robust to discontinuities since it is a single point and it is more difficult to break a point or change its location by discontinuity.

In a grid line map, $\mathcal{M}_c$, such as the one in Fig. 3.2, there are many intersection locations. This simply means that in order to find all these pixel locations at once, we would need to optimize the distance between all of the variable pixel locations and the grid intersections. Let us define an objective function of the input image, which is the grid line map $\mathcal{M}_c$, as $f(\mathcal{M}_c, \{x_i, y_i\})$, that describes the goodness of fit of the variable detected points, $\{x_i, y_i\}$, to the grid intersections in the image. In order to optimize the distance between the detected coordinates and the correct grid intersection locations, we need to optimize over all the parameters, $\{x_i, y_i\}$:

$$\min_{x_i, y_i} f(\mathcal{M}_c, \{x_i, y_i\}) \tag{3.3}$$

The number of parameters, $\{x_i, y_i\}$, in (3.3) is equal to the number of the intersections in the image thus it will be challenging for the optimizer to optimize all these variables all together. The search space for this optimization increases as the input image gets larger in size.

Let's consider the BAE pattern to find any properties in the pattern that would result in a simpler solution. Since the BAE pattern in the projector view, I_p, consists of straight parallel and perpendicular grid lines, as shown in Fig. 3.4, the problem can be formulated in terms of lines instead of pixel locations and hence, can be formulated with three different variables per direction globally over the whole image regardless of its size. Lines can be described by three different parameters, θ, ρ, and $\mathcal{S}$ in each direction. θ is the angle of a line and ρ is its distance from the origin. The third parameter, $\mathcal{S}$ is the spacing between the lines since all the lines in the projector pattern, I_p, are spaced equally. With these assumptions we can formulate the objective function, f, optimized over the line parameters. These parameters can further be described in the horizontal and vertical directions and ultimately the objective function f from (3.3) can be redefined as a function of line parameters fitting the grid lines and optimized over six parameters in total.

$$\min_{\theta_h, \rho_h, \mathcal{S}_h, \theta_v, \rho_v, \mathcal{S}_v} f(\mathcal{M}_c, \theta_h, \rho_h, \mathcal{S}_h, \theta_v, \rho_v, \mathcal{S}_v) \tag{3.4}$$

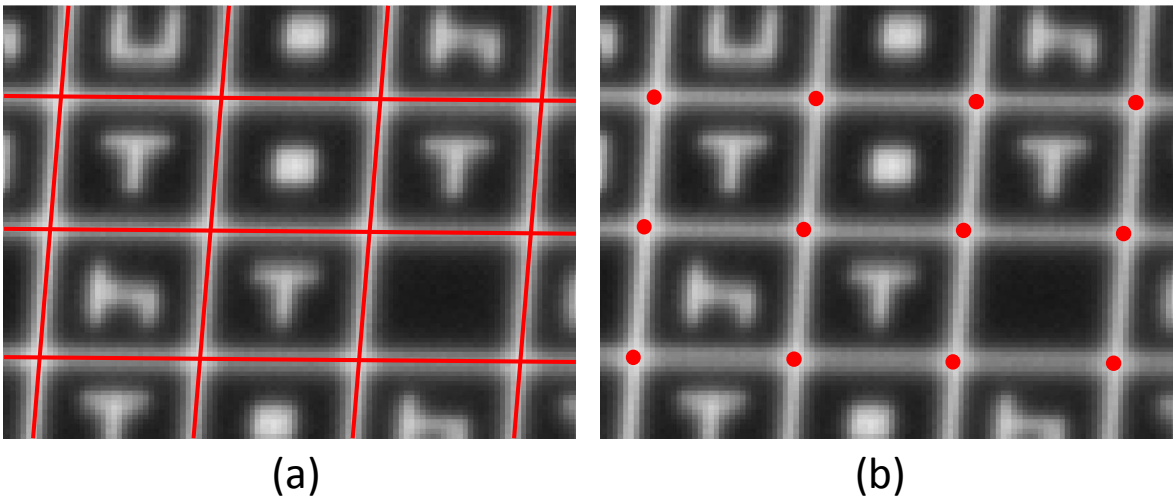

Figure 3.5: Grid intersection detection by finding the parameters of the lines in a sample patch. Lines are parameterized (a) and their intersections are obtained (b).

By switching to lines instead of point locations, the variables of our optimization have changed from thousands of coordinates in (3.3) to only six parameters in (3.4). However, The assumptions made based on the I_p do not hold in the camera image, I_c, and the grid line map, $\mathcal{M}_c$, constructed from I_c. Deformations such as surface discontinuities and curvature changes the properties of the grid lines in I_c compared to I_p such as perpendicularity and shape of the lines. Therefore, in order to be able to implement the same optimization with the same variables, we need to localize the process. Partitioning the camera image into smaller blocks, will localize the process and enable us to approximate the curves caused by the surface variations of the object with lines. Comparing the image shown in Fig. 3.1 and the sample patch in Fig. 3.5 shows the effect of localization on the surface deformations and line detection. The localization enables the optimization defined for the projector pattern in (3.4) to be used here as well. However, in the projector pattern the grid lines have equal spacing between them, which allows us to find a set of lines instead of individual lines, whereas in the camera image lines may not be equally spaced. Therefore, each line in both directions is represented by its own parameters.

$$\min_{\theta_h^i,\rho_h^i,\theta_v^j,\rho_v^j} f(\mathcal{M}_c, \theta_h^i, \rho_h^i, \theta_v^j, \rho_v^j) \tag{3.5}$$

The horizontal and vertical lines in (3.5) are optimized on their angles, θ_i and θ_j, and their distance from the origin, ρ_i and ρ_j, respectively. By optimizing f in (3.5), the optimum

parameters and therefore the equation of each line is obtained and, finally, the problem boils down to a mathematical process of finding the intersections of the lines. The computational complexity is increased compared to (3.4) since each line should be detected individually with two parameters in (3.5) whereas two sets of lines are detected with three parameters using (3.4), but there are far fewer variables compared to (3.3). The method is explained in detail in Chapter 5.

Chapter 4

Locally Adaptive Thresholding

Image thresholding usually serves as a preprocessing operation to prepare an image for future steps in a process. The reconstruction scheme used in this thesis is no different. As shown in Fig. 4.1, the proposed scheme starts with binarizing a given camera image, I_c to preprocess it for future operations. To this end, we propose a locally adaptive thresholding method in this chapter, which we have formulated in Section 3.1. According to Fig. 4.1, the input image is first binarized and used to create a grid line map, which is the input of the following detection process.

In Sauvola and Pietikäinen's (SP) method [112], as reviewed in Section 2.2, given an input camera image I_c, this method uses the local mean $m(x, y)$ and the local standard

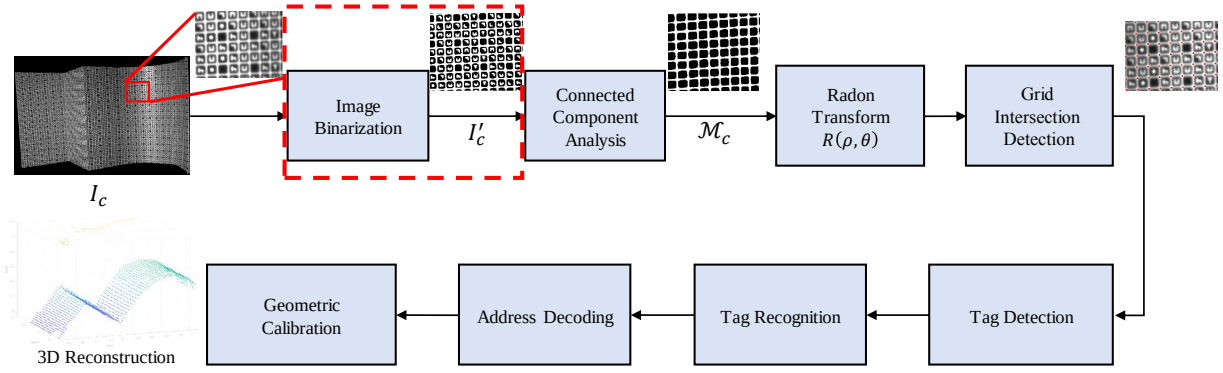

Figure 4.1: Block diagram of the proposed single-shot 3D reconstruction scheme. The first block highlighted in this process uses the proposed locally adaptive thresholding method explained in this chapter.

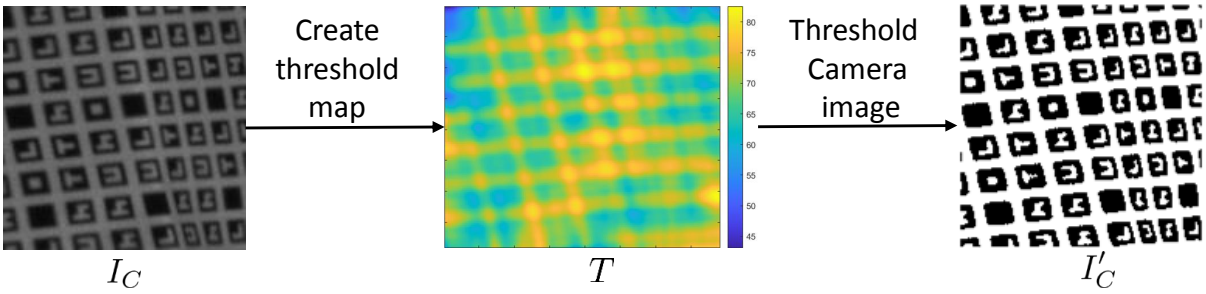

Figure 4.2: Block diagram of the local thresholding method introduced in [112]

deviation $\sigma(x, y)$ computed around the (x, y)th pixel location to set a threshold value, $T(x, y)$ as

$$T(x, y) = m(x, y)\left(1 - k\left(1 - \frac{\sigma(x, y)}{R}\right)\right) \tag{4.1}$$

where R is the dynamic range of the image pixels and k is the bias parameter. The threshold value, $T(x, y)$, is then used to binarize the input image, $I_c(x, y)$.

$$I'_c(x, y) = \begin{cases} 1, & \text{if} \quad I_c(x, y) \geq T(x, y) \\ 0, & \text{if} \quad I_c(x, y) < T(x, y) \end{cases} \tag{4.2}$$

An illustration of SP's thresholding method on a sample image patch is shown in Fig. 4.2. Unlike SP's method that requires fine-tuning for the parameter k in (4.1) for every image, our proposed method aims to obtain automatically an optimal k to adaptively threshold each image pixel within a local region. Fig. 4.3 shows the overall architecture of the proposed method. Different areas of the image have different pixel intensity distributions, thus assigning threshold values adaptively to each area is of importance. This will be carried out in a block-wise process during which adaptive threshold values are assigned to each pixel in a block. In the proposed method, an integral image [11] of the input image, I_c is obtained. Essentially, an integral image is a tool that can be used to compute the sum of multiple overlapping blocks with only a single pass through the image. To compute an integral image, terms to the left and above the pixel (x, y) are stored at each location, $I_c(x, y)$.

$$I_c(x, y) = I_c(x, y) + I_c(x - 1, y) + I_c(x, y - 1) - I_c(x - 1, y - 1). \tag{4.3}$$

The camera image is partitioned into a number of distinct blocks to enable the method to analyse the pixel intensity distribution locally. Therefore, the effect of lighting conditions

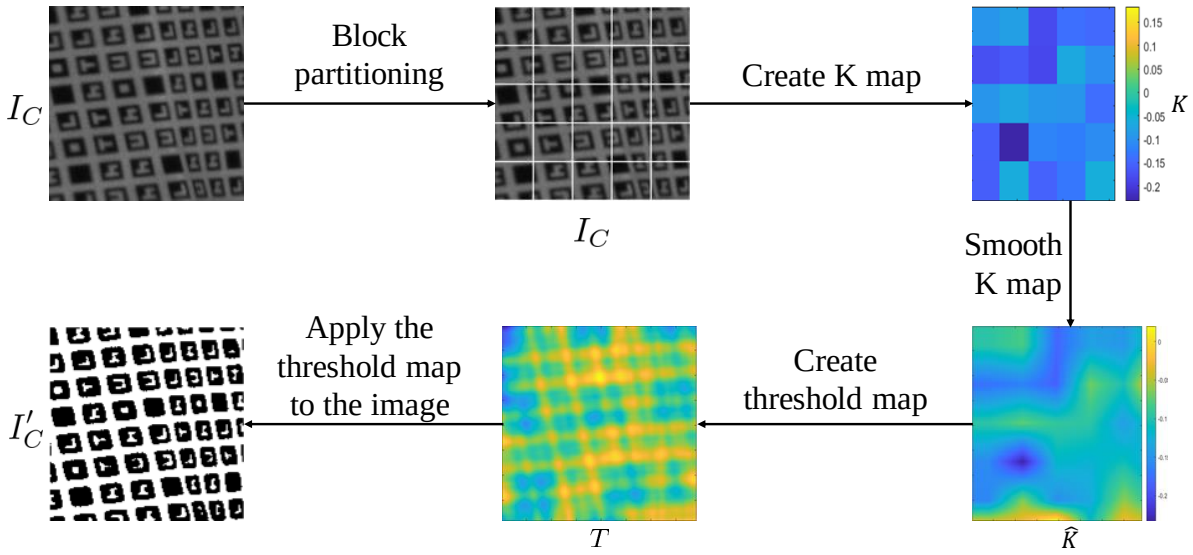

Figure 4.3: Block diagram of our proposed method, where the $\hat{K}$ map is produced, from which, T, is computed. The smoothed $\hat{K}$ map consists of the value of the bias parameter k for each pixel in the image. The $\hat{K}$ map is constructed by smoothing the K map, denoted by K in the figure, with an average filter.

are minimized. A criterion in each block is needed that not only thresholds the darkest and brightest pixels correctly, but also binarizes the mid-range gray pixels as accurately as possible. Additionally, this criterion needs to be robust to imbalance pixel intensity distributions and the noise added by lighting conditions or the camera. Since the structure of the pattern is known, we can have a large enough window, which we call blocks, to look through the image to include a mixed number of black and white pixels in every block and avoid any blocks with only one class of pixel intensities. In the gray-scale camera image, in addition to white and black pixels, mid-range gray pixels, which can be caused by noise or interpolation on the edges of the tags and symbols, exist. Therefore, in some blocks of the gray-scale image the distribution can be imbalanced, meaning that the majority of the block might be black or white, and in these cases criteria such as mean and median would fail to set a good threshold. Percentiles on the other hand, would have a reasonable estimation, if set correctly, of the very bright and dark pixel distributions. Again, based on the structure of the pattern we know that in every block we have at least 10% very dark and very bright pixels hence, taking the 90_{th},$p_{90^{th}}$ and 10_{th},$p_{10^{th}}$ percentiles would be a good categorization of these groups of pixels. The threshold set at the average of the

90_{th} and 10_{th} percentiles would ensure the correct binarization of the very dark and very bright pixels:

$$\Gamma(u,v) = \frac{p_{10^{th}} + p_{90^{th}}}{2} \tag{4.4}$$

however, (4.4) only gives us a basic threshold with regards to these pixels, but the mid-range gray pixels need to be thresholded as accurately as possible as well. Therefore, the threshold value needs to be updated. Initially, based on the difference between $\Gamma(u,v)$ and the average intensity value of the block, $M(u,v)$, the algorithm assigns a specific k to each block as follows:

$$k(u,v) = \frac{\Gamma(u,v) - M(u,v)}{R} \tag{4.5}$$

where R is the image dynamic range. Updating the primary threshold is then carried out by the bias parameter k, which we automatically compute using (4.5). $k(u,v)$ is the parameter that will be used to update the threshold value with regards to the majority of white and black pixels, above or below $M(u,v)$.

After defining k for each block, a K map that defines the k value per block is created as shown in Fig. 4.3. In order to avoid blocking artifacts, a smooth version of the K map is constructed, which we denote by $\hat{K}$, using an average (mean) filter. The average filter replaces each pixel intensity of the K map, which in this case are the values of k, by the average values of an arbitrary neighborhood of the pixel, which smooths the sharp edges of the K map shown in Fig. 4.3. Finally, the smoothed K map, i.e., $\hat{K}$, which has the values of k corresponding to each pixel in the image, is then used to create the threshold map, T:

$$T(x,y) = m(x,y)\left(1 - \hat{K}(x,y)\left(1 - \frac{\sigma(x,y)}{R}\right)\right) \tag{4.6}$$

where $m(x,y)$ and $\sigma(x,y)$ are the local average and local standard deviation of each pixel, and $\hat{K}(x,y)$ is the bias parameter by which we decide how far below or above $m(x,y)$ we need the threshold to be set. The usual values for the parameter k suggested by [112, 118] would not necessarily be the optimum value for the SSSL application and are case dependent, as we will show in Section 4.1.2, whereas our method can compute $\hat{K}(x,y)$ automatically, which eliminates the need to get parameter k from the user.

Finally, the thresholded image, I'_c, is achieved by applying the acquired threshold map, T, on the camera image, I_c, using (4.2).

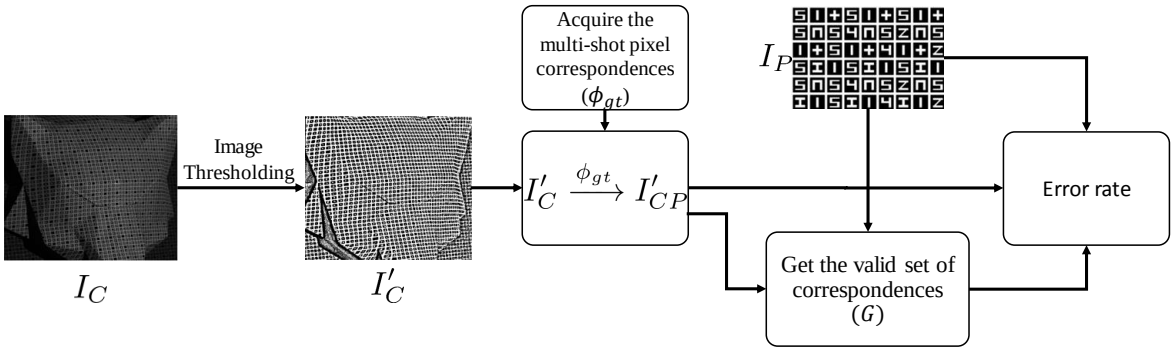

Figure 4.4: Block diagram of the evaluation process for assessing the proposed method, where I_c, is binarized using the proposed thresholding method, or any other thresholding method, and the ground truth correspondences, ϕ_{gt}, are used to map the binarized image, I'_c, to the projector view. The values of valid corresponding pixels, G, in I'_{cp} are compared to the projector pattern, I_p.

4.1 Experimental Results

For the experimental setup, we have used a Flea 3 camera with a resolution of 2448x2048 and a Christie Digital projector with a resolution of 1920x1200. We tested the proposed method using SSSL patterns projected on three different 3D-printed objects, named *Wolf Head*, *Zigzag* [66] and *Building Facade* [66].

4.1.1 Evaluation Methodology

Fig. 4.4 shows a diagram for our proposed evaluation methodology for assessing the performance accuracy of the proposed method. To evaluate the local thresholding techniques, we first generate dense pixel-correspondences between the camera and projector images, ϕ_{gt}, using the multi-shot gray code structured light patterns in [66]. Our assessment strategy is to map a thresholded pixel in a given thresholded camera image to the projector space leveraging the dense correspondences, ϕ_{gt}, and then to compare the intensity value of the mapped pixel to the corresponding pixel in the original binary projector pattern, I_p, as its ground-truth. The original SSSL binary pattern, I_p, that is projected on the surface is considered as the ground-truth image due to it not being affected by the surface geometry deformation. Having a dense set of correspondences will help us to have a rich set of pixels to compare with the ground truth.

To apply this strategy, given an input SSSL camera image, I_c, we obtain the binarized image, I'_c, by applying our proposed method, or any other thresholding method in comparison. Based on the acquired multi-shot pixel-correspondences between the camera and projector images, ϕ_{gt}, we map the binarized camera image, I'_c, into the projector space, I'_{cp}. Next, for every pixel location, (x, y), in I'_c we find the corresponding pixel, (x', y') in the projector pattern, I_p, and collect the neighboring pixels, $\aleph_{x,y}$, in a neighborhood of size $\eta \times \eta$:

$$\aleph_{x,y} = \Big\{ I_p(i,j) \quad | \quad |x' - i| < \frac{\eta}{2}, |y' - j| < \frac{\eta}{2} \Big\}. \tag{4.7}$$

When all the neighboring pixels, $\aleph_{x',y'}$, have the same value as $I_p(x', y')$, then we assume the corresponding pixels $(x, y) \leftrightarrow (x', y')$ is a valid correspondence to be used to compare against the ground-truth. This process can be expressed as follows:

$$G(x,y) = \begin{cases} 1, & \text{if} \quad I_p(x',y') = \aleph_{x',y'} \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

where $G(x, y)$ is the valid pixel-correspondences that can be used to assess the performance of a thresholding method. It should be noted that since G contains the corresponding pixels, $G(x, y)$ and $G(x', y')$ are both in G thus their notation can be used interchangeably.

Having the valid set of corresponding pixels, G, helps us to compare the mapped camera image, $I'_{cp}(x', y')$ against the ground-truth, $I_p(x', y')$. Therefore, we can assess whether a

Surface	Method	ξ	k
Wolf Head	SP [112]	**0.15%**	0.2
	SP [112]	0.30%	0.0
	Proposed	0.17%	—
Zigzag	SP [112]	3.50%	0.2
	SP [112]	**2.60%**	0.1
	Proposed	2.60%	—
Building Facade	SP [112]	8.30%	0.2
	SP [112]	**3.10%**	0.0
	Proposed	3.80%	—

Table 4.1: Error rate, ξ, for the proposed method and the method in [112]. Two values for k are shown for SP's method [112], where one is the optimum value, shown in bold, and a non optimal value. The optimum k value is automatically assigned to each pixel in the proposed method.

given pixel has been binarized correctly or not. Let the sets of True, Right, and False, Wrong, pixel-correspondences be defined as

$$\mathscr{R} = \langle (x', y') \quad | \quad I'_{cp}(x', y') = I_p(x', y') \quad , \quad G(x', y') = 1 \rangle \tag{4.9}$$

$$\mathscr{W} = \langle (x', y') \quad | \quad I'_{cp}(x', y') \neq I_p(x', y') \quad , \quad G(x', y') = 1 \rangle \tag{4.10}$$

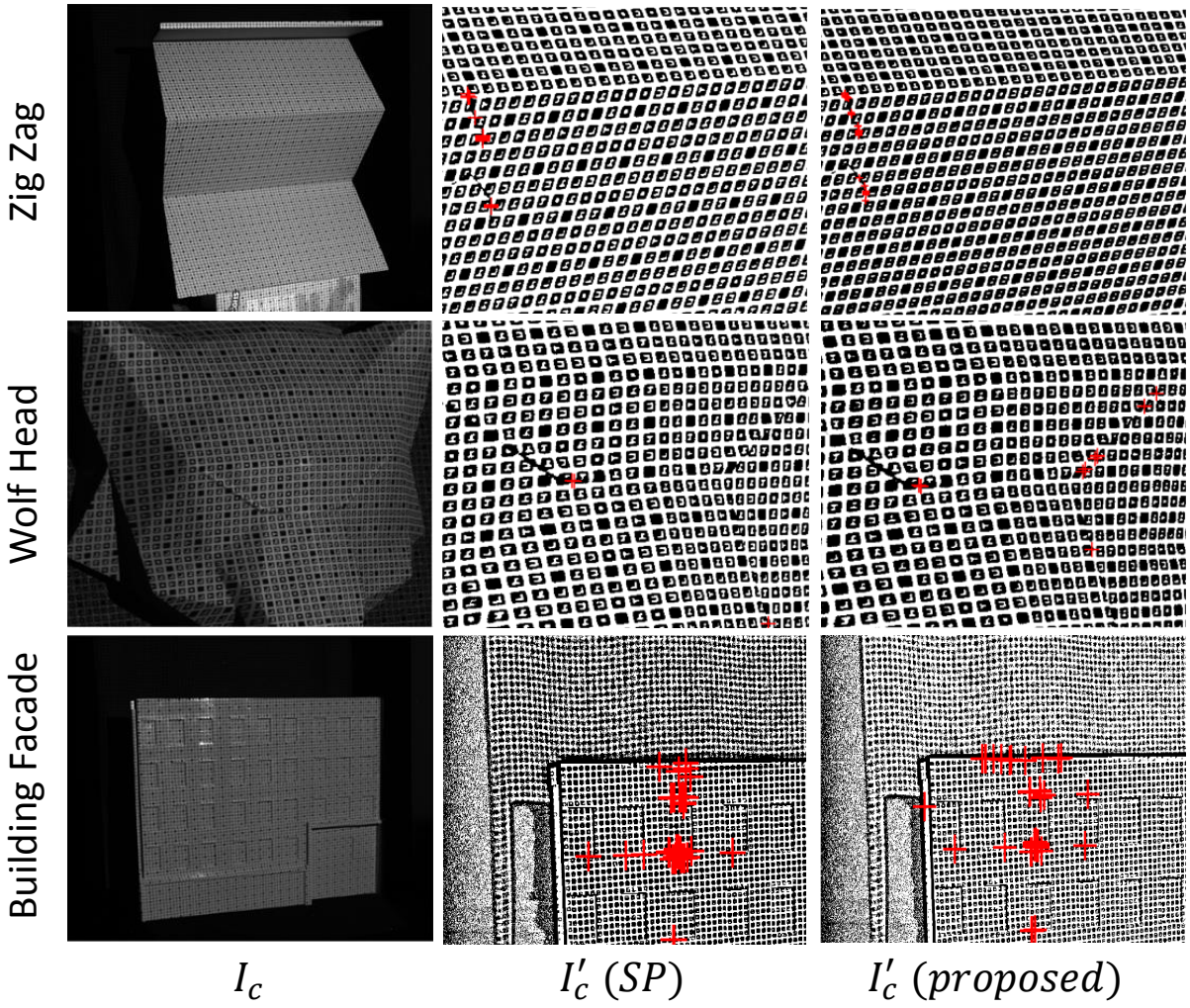

Figure 4.5: Sample qualitative results for SP's method vs our proposed method on three different surfaces, where a marker with red color denotes incorrect thresholded pixels. Note that SP's method requires adjusting k to 0.2, 0.1 and 0.0 for the *Wolf Head*, *ZigZag* and *Building Facade*, respectively, in order to generate the best results.

where $\mathscr{R}$ and $\mathscr{W}$ are the sets of correctly and incorrectly thresholded pixels, respectively. Obtaining the correct and incorrect set of thresholded pixels from (4.9) we can compute the error rate of the thresholding method. The error rate (ξ) can be expressed as

$$\xi = \frac{|\mathscr{W}|}{|\mathscr{R}| + |\mathscr{W}|} \times 100\% \tag{4.11}$$

where $|\cdot|$ indicates the cardinality of the set. The metric ξ counts the percentage of incorrectly thresholded pixels from the total number of valid correspondences. In the rest of this section, we utilize the proposed evaluation methodology to compare the thresholding error rate of the proposed scheme with that of SP's method [112].

4.1.2 Quantitative and Qualitative Results

Fig. 4.5 shows the different qualitative results of SP's method [112] and the proposed method. Both methods seem to have errors on edges and discontinuities in the image which is expected since the significant intensity variation impacts the statistics of a patch.

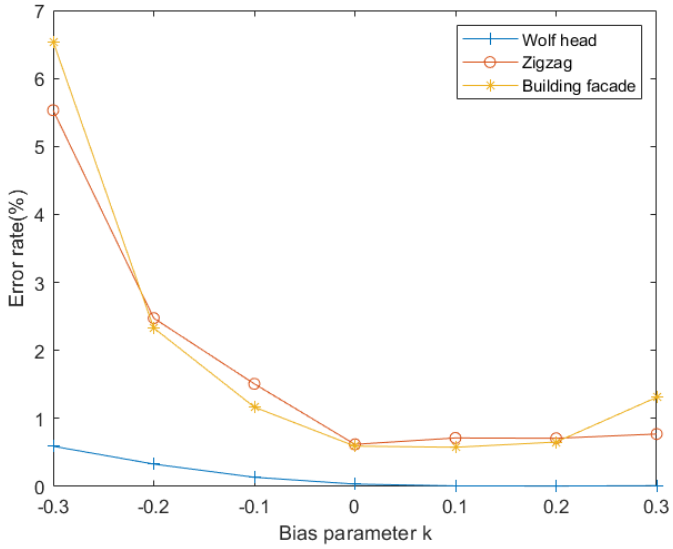

Figure 4.6: Effect of changing $k = [-0.3, -0.2, ..., 0.3]$ on the error rate, ξ, for the SP's method. Note that the optimum value of k is dependent on the image.

As shown in Fig. 4.6, SP's method performance is dependent on the value of k. This indicates that this method is case dependent and needs the k parameter to be fine tuned by the user, whereas in the proposed method this parameter is automatically obtained for every pixel.

Tab. 4.1 shows the error rate comparison between the proposed method and the SP method. As shown in the table, the error rate of the proposed method is close to that of the Sauvola and Pietikäinen's (SP) method with optimum value for k. As can be observed, the SP's method error rate becomes larger than the error rate of the proposed method as the value of k deviates from the optimum value.

Chapter 5

Grid Intersection Detection

In order to establish pixel correspondences, features in one view need to be detected and represented in a way that can be related to other views. Therefore, detecting corresponding features will result in establishing pixel correspondences, which are used in the 3D reconstruction scheme used in this thesis. We have formulated the feature detection problem in Section 3.2 and now we propose a detailed solution in this chapter.

In the SSSL pattern, as explained in Section 2.1, tags are divided by parallel grid lines which we wish to detect. Before executing the detection, the input camera image needs to be preprocessed to prepare the image for detection. The proposed method in Chapter 4 is used to binarize I_c, resulting in I'_c, and the symbols are removed using connected

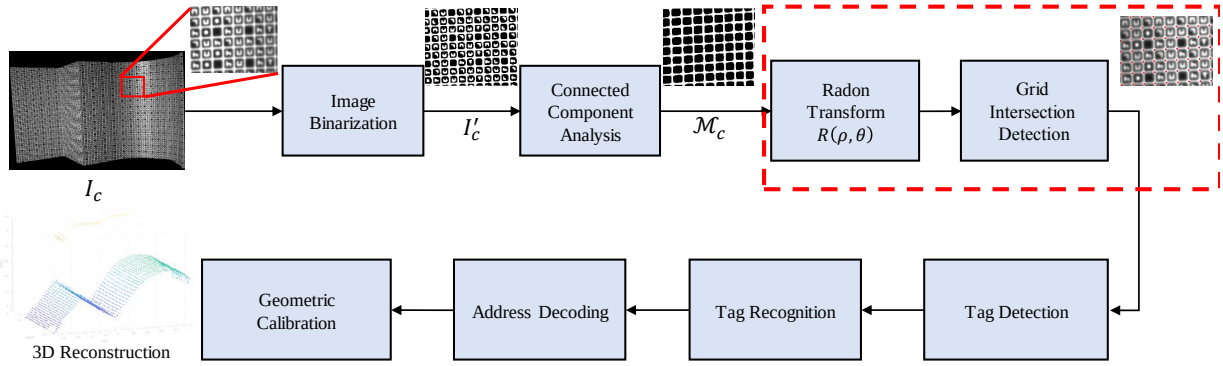

Figure 5.1: Block diagram of the proposed single-shot 3D reconstruction scheme. The two blocks, Radon transform and grid detection, highlighted in this process use the proposed grid detection method explained in this chapter.

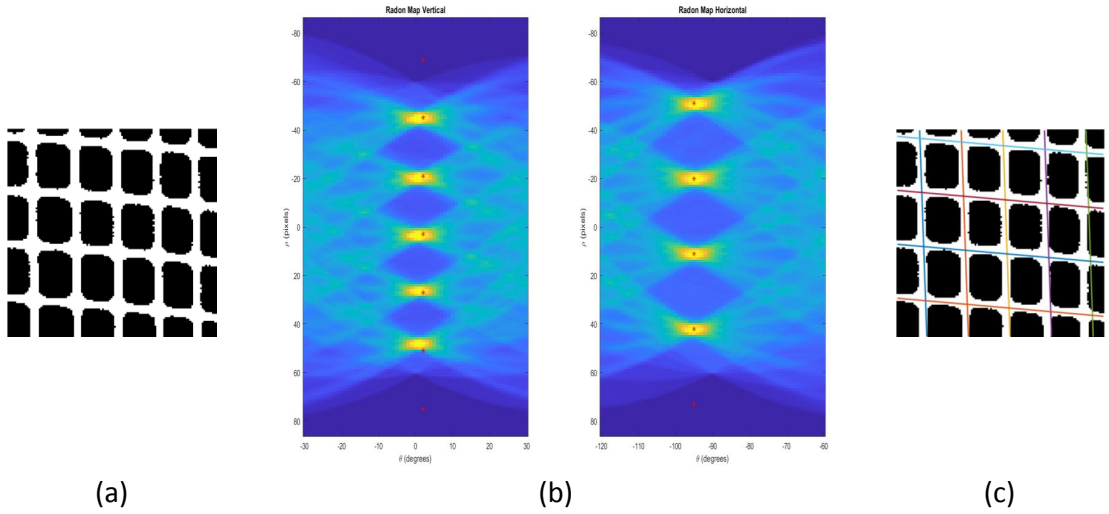

Figure 5.2: Illustration of the detection method on a sample patch. Sample patch (a) undergoes the Radon transform and the two constructed Radon maps in horizontal and vertical direction are searched to find their peaks, marked red (b), which are used as initial estimates of line parameters. The lines are then fitted based on the optimized parameters computed from the initial estimates (c).

component analysis (CCA) [121, 29] to create a grid line map, $\mathcal{M}_c$, as shown in Fig. 4.1.

The details of CCA are outside the scope of this thesis, so we give only a brief description here. CCA is an algorithm in graph theory which determines how a subset of components should be labeled based on a certain heuristic. Based on the heuristic, pixels that are connected to each other will be considered as a blob. The blobs can be used for various applications such as detection, classification, and morphological operations.

Using the result of CCA, we can remove the blobs labeled as symbols and end up with black squares, which represent the tags, as illustrated in Fig. 5.1. In order to make our model adapt to surface variations, or in other words, to overcome the effects of the curvature and discontinuities on the grid lines; as explained in Section 3.1, the binarized image I'_c is divided into overlapping blocks.

Now the Radon transform is employed on each block resulting in a radon map, $\mathcal{R}$, such as in Fig. 5.2, which is then explored to find the peaks. The peaks are the initial estimates for the parameters θ, ρ and the number of the peaks represent the number of lines in each

direction, $\mathcal{N}_h$ and $\mathcal{N}_v$. Arbitrary ranges for the angle of horizontal and vertical lines are defined based on the skewness of the lines and deformation of the surfaces.

$$\begin{aligned}(\rho_h^0, \theta_h^0) &= \mathcal{R}(\rho_h, \theta_h) \quad 60^\circ \leq \theta_h \leq 120^\circ, \\ (\rho_v^0, \theta_v^0) &= \mathcal{R}(\rho_v, \theta_v) \quad -30^\circ \leq \theta_v \leq 30^\circ\end{aligned} \tag{5.1}$$

We then formulate the problem as a model based search to find simultaneously the optimum parameters for the set of horizontal and vertical lines that maximize the average pixel intensities in each set using the following formula:

$$\max_{\rho_h, \theta_h, \rho_v, \theta_v} \sum_{i=0}^{\mathcal{N}_h - 1} \frac{\mathcal{R}(\rho_h^i, \theta_h^i)}{\mathcal{N}_h} + \sum_{j=0}^{\mathcal{N}_v - 1} \frac{\mathcal{R}(\rho_v^j, \theta_v^j)}{\mathcal{N}_v} \tag{5.2}$$

where θ_v, ρ_v, θ_h, ρ_h are the vertical and horizontal line angles and distances, respectively. The constructed Radon maps enable us to limit the search space since the initial estimates for the variables obtained from the Radon maps, θ_v, ρ_v, θ_h, and ρ_h, are reasonably good estimation of the correct peaks, as observed in Fig. 5.2. Therefore, the search space is limited to an arbitrary value for each parameter, ρ_l and θ_l°, below and above the initial estimates to cover the neighboring area of the peak in the Radon map.

$$\begin{aligned}\theta_h^0 - \theta_l \leq \theta_h^i \leq \theta_h^0 + \theta_l, \\ \theta_v^0 - \theta_l \leq \theta_v^i \leq \theta_v^0 + \theta_l, \\ \rho_h^0 - \rho_l \leq \rho_h^i \leq \rho_h^0 + \rho_l, \\ \rho_v^0 - \rho_l \leq \rho_v^i \leq \rho_v^0 + \rho_l,\end{aligned} \tag{5.3}$$

5.1 Experimental Results

In order to evaluate the detection performance of the proposed method, we captured a dataset by using two shapes, namely, a smoothly curved shape and piece-wise planar shape, we will refer to them as *Curved* and *Zigzag*, respectively. The *Curved* shape is chosen to represent smooth surfaces with occasional folds, whereas the *Zigzag* shape contains sharper discontinuities in planar surfaces. The dataset includes projector patterns with two different tag sizes, three different lighting conditions, and three different views with 18 different cases in total for each shape. To have a more complete evaluation another test is done on the projector pattern, I_p, which is the ideal image. This experiment is done to see the effect of different levels of synthetic noise and distortions solely on the detection performance without the effects of the camera and projector.

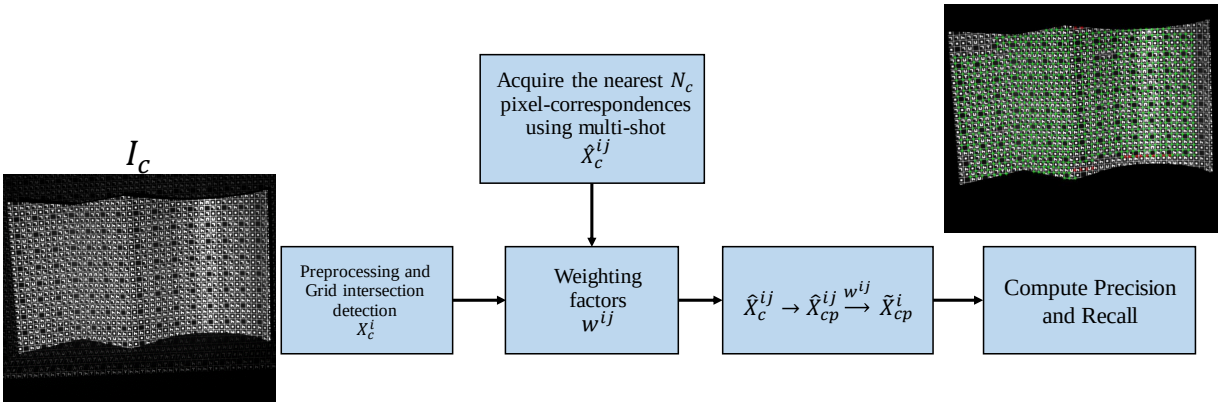

Figure 5.3: Block diagram of the evaluation methodology for the proposed grid intersection detection method. After acquiring intersections, X_c^i, nearest multi-shot correspondences, $\hat{X}_c^{ij}$, are found. These correspondences are used to find the corresponding pixels in the projector domain, $\hat{X}_{cp}^{ij}$ and the weighted average of the mapped coordinates will ultimately result in the closest correspondence, $\tilde{X}_{cp}^{i}$, to the ground truth, X_p^i, in the projector domain.

5.1.1 Evaluation Methodology

We leverage the multi-shot structured light, as shown in Fig. 5.3, to estimate the position of the detected points in the projector view and find the distance between the estimated positions and the corresponding grid intersections in the projector pattern, I_p. For this purpose, let the ordered pairs of the i^{th} detected corner in the camera image, I_c, be X_c^i, where $i = 1,2,\ldots,N_d$ and N_d is the number of detected intersections. Also let the ordered pairs of the j^{th} nearest multi-shot correspondence to a given X_c^i in the camera image be $\hat{X}_c^{ij} \leftrightarrow \hat{X}_{cp}^{ij}$, where $j = 1,2,\ldots,N_c$ and N_c is the maximum number of neighboring correspondences defined by us. The distance between the i^{th} intersection and its j^{th} neighboring correspondence in the camera space is calculated as

$$d^{ij} = \|X_c^i - \hat{X}_c^{ij}\|_2 \tag{5.4}$$

As shown in Fig. 5.3, the estimated location, $\tilde{X}^i_{cp}$, is the weighted average of the N_c multi-shot correspondences of the detected point, $\hat{X}^i_c$, in the projector domain, $\hat{X}^i_{cp}$:

$$\begin{aligned} \tilde{X}^i_{cp} &= \sum_{j=1}^{N_c} w^{ij} \cdot \hat{X}^{ij}_{cp} \\ w^{ij} &= \frac{\frac{1}{d^{ij}}}{\sum_{j=1}^{N_c} \frac{1}{d^{ij}}}, \end{aligned} \tag{5.5}$$

The error, e_i, between the estimated location, $\tilde{X}^i_{cp}$, and the nearest ground truth intersection, X^i_p, is obtained:

$$e_i = \|\tilde{X}^i_{cp} - X^i_p\|_2 \tag{5.6}$$

Now based on e_i, we can classify each corresponding coordinate, $X^i_c \leftrightarrow \tilde{X}^i_{cp}$, as either True positive, TP, if e_i is less than a threshold, ε, or False positive, FP, if e_i is greater than ε. The constant ε has an empirical value related to the thickness of the grid lines, d_{grid}, determined by the projector pattern, I_p, $\varepsilon = \sqrt{2}d_{grid}$. From the TP and FP values, Precision and Recall can be computed for a given intersection detection method.

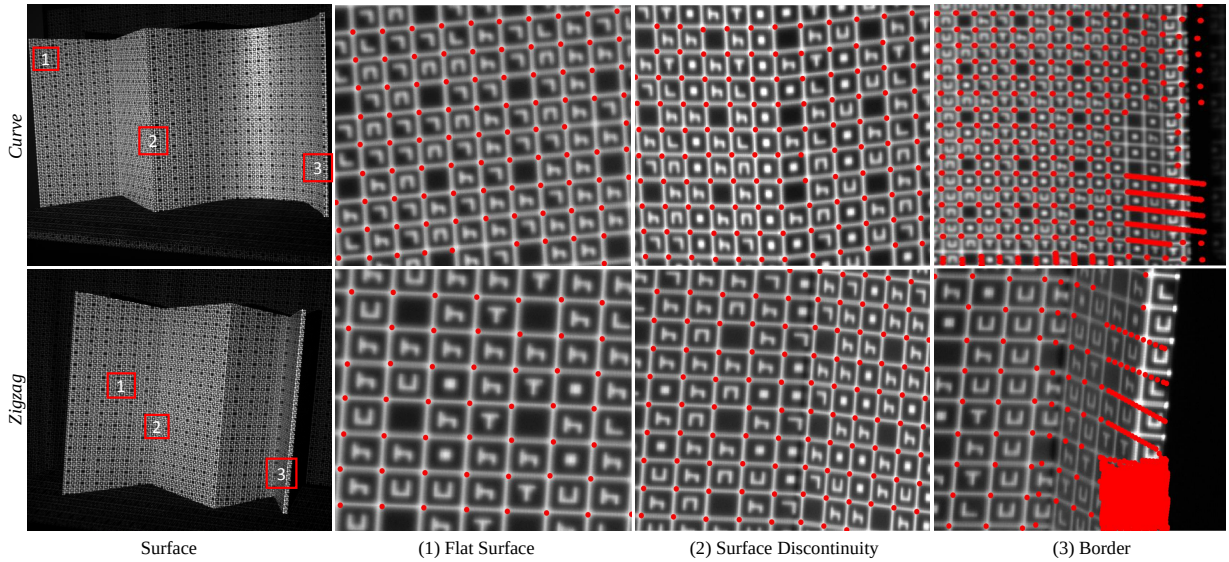

Figure 5.4: Example qualitative results for the proposed grid intersection detection method on the *Curve* and *Zigzag* shapes.

5.1.2 Qualitative Results

The proposed method is evaluated on real objects to show the effects caused by ambient lighting, and camera projector distortions as well as synthetic circumstances to measure the impact of various noise parameters on its performance. Fig. 5.4 demonstrates sample visual results when the proposed method is used to detect the grid intersections in the camera captured images, where the *Curve* and *Zigzag* shapes are used. As can be seen in this figure, the proposed method can detect the intersections on curved, smooth, and piecewise smooth surfaces, while it suffers from border areas of the shapes where sudden surface variations happen.

To show the effectiveness of the proposed scheme we utilize the pixel correspondences generated by the proposed method for 3D reconstruction, and then compare the results to the 3D CAD model as the ground-truth. The reconstructed point clouds achieved by the method in [27] is also compared to the 3D CAD model of the shapes. Fig. 5.5 shows the 3D point clouds for the proposed method and the method in [27] on the *Curve* and *Zigzag* shapes. It can be seen from this figure that the proposed method offers sparser point clouds than the method in [26], which is due to the fact that our method only detects one pixel correspondence for every grid intersection (shared by up to four tags) instead of four corners obtained per tag as in [27]. The proposed scheme is able to follow the geometry of

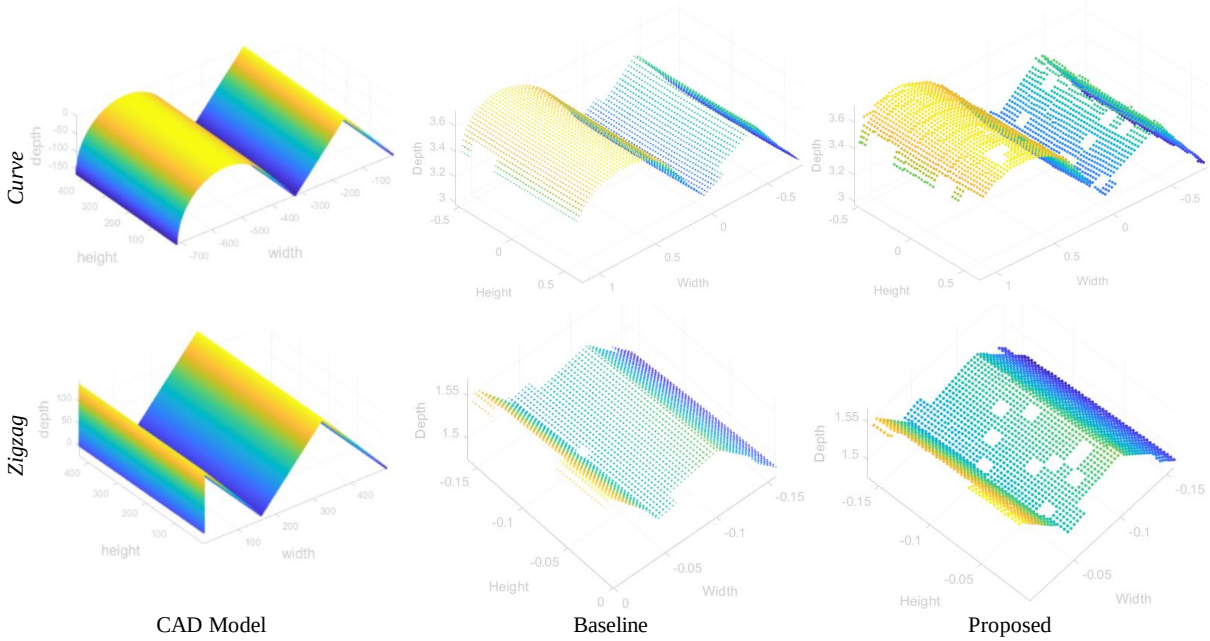

Figure 5.5: Comparison between the reconstructed point cloud from the method in [27] and the proposed method.

the surface.

5.1.3 Quantitative Results

In order to test the performance of the proposed method under different levels of synthetic noise, binary projector patterns of uint8, with white = 255 and black = 0 with two different tag sizes, 10×10 and 20×20, are used. Gaussian blur with standard deviation of ($\sigma = 1$ pixels) is applied and Zero mean Gaussian white noise (SNR = $31.7db$) is added to the image along with Random rotation. The image, $I(x, y)$, is then distorted with different level of skewness, $0 \leq SH \leq 0.6$, by shearing the image:

$$\begin{bmatrix} x_{new} \\ y_{new} \end{bmatrix} = \begin{bmatrix} 1 & SH & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.7}$$

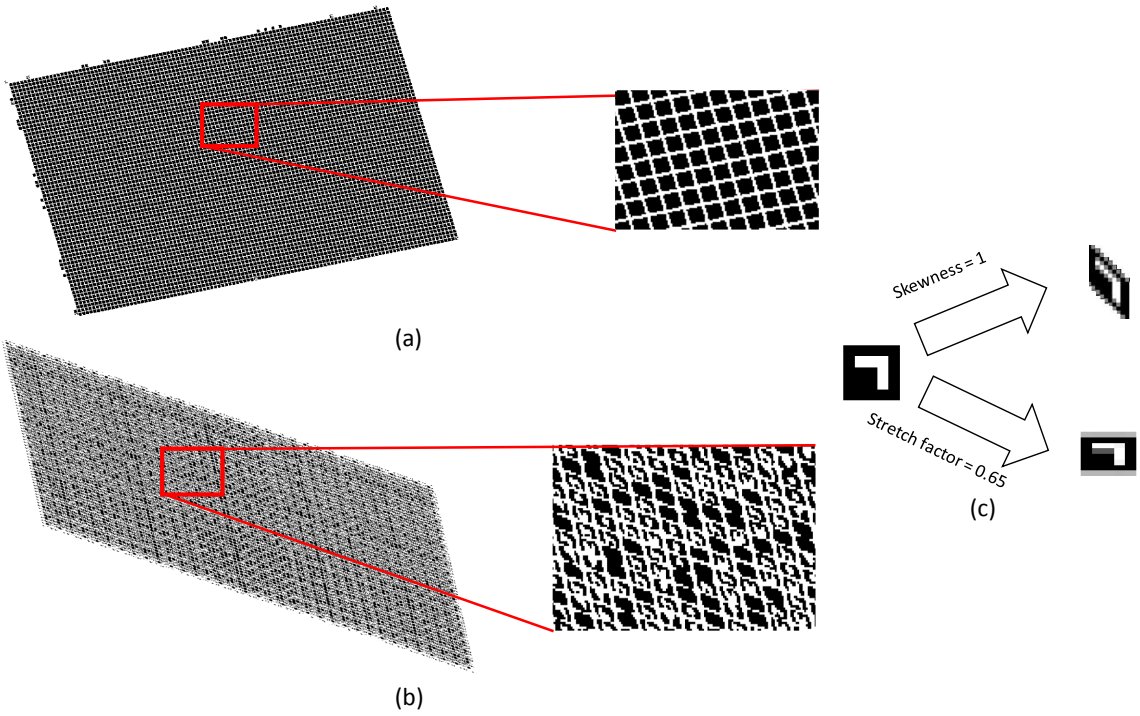

Figure 5.6: Illustration of the impact of skewness on a sample test image and the resulting grid line map. Test images are randomly rotated and different skewness levels, lowest, $SH = 0$, (a) and highest, $SH = 1$, (b) with a fixed stretch factor, $SF = 0.3$ is applied to them. Maximum level of skewness and stretch factor is shown in (c). Notice that the constructed grid line map becomes cluttered and the symbols are not removed completely from it as the image is more distorted.

and stretch factor, $0.65 \leq SF \leq 1.35$, which squishes or stretches the image.

$$\begin{bmatrix} x_{new} \\ y_{new} \end{bmatrix} = \begin{bmatrix} SF & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.8}$$

An example of the effect of shear and stretch factor on a sample grid line map, $\mathcal{M}_c$, is shown in Fig. 5.6. The neutral points in skewness and stretch factors are $SH = 0$ and $SF = 1$, meaning at $SH = 0$ and at $SF = 1$ the image remains unchanged.

The noisy image is detected for the grid intersections using the proposed detection method and the RMSE of the detected points are computed using the grid intersections of

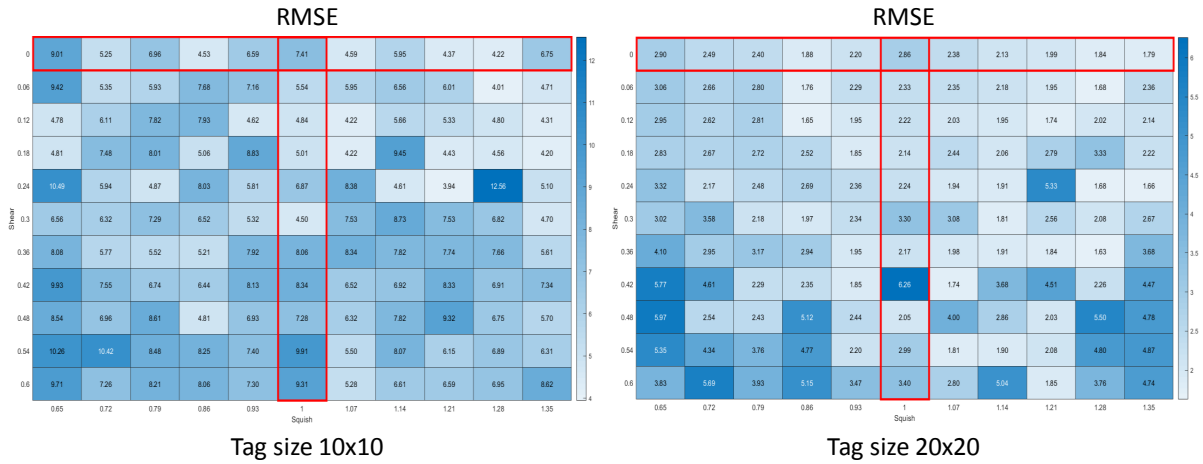

Figure 5.7: Performance accuracy of the proposed method with different levels of shear and squish added to the image with two different tag sizes, 10×10 and 20×20. The RMSE between the detected grid intersections and the pattern grid intersections as ground truth are computed. The camera image is affected by noise, blur, image resizing and rotation.

the pattern as the ground truth. As shown in Fig. 5.7, as the variables move away from the neutral points, shown in red boxes, the resulting grid line map becomes more cluttered and the accuracy starts to decrease, specifically with stretch factor. As the image is squished the performance suffers since the grid line map, $\mathcal{M}_c$, becomes noisier and the symbols are not completely removed from the tag, as shown in Fig. 5.6(b). Although squishing the image reduces the accuracy of the method, stretching the image seems to have less impact on the performance of the method according to the results shown in Fig. 5.7.

Shapes	Tag size in I_p	N_{pc}	Precision(%)	Recall(%)
Curve	10×10	3862	92.74	96.23
	20×20	1096	92.25	86.57
Zigzag	10×10	4874	89.32	93.43
	20×20	1463	88.79	87.23

Table 5.1: Evaluation results of the proposed detector on the dataset consisting of *Curved* and *Zigzag* shape, where projector pattern, I_p with two different tag sizes are projected on the objects. N_{pc} represents the number of detected pixel correspondences, which are related to the size of the tags in the projector pattern, I_p.

Shapes	Methods	N_{pc}	RMSE (pixels)	Run time (secs)
Curve	Method in [27]	4496	7.12	10.80×10^3
	Proposed	1186	11.71	29.67
ZigZag	Method in [27]	8064	5.77	9.82×10^3
	Proposed	2348	6.07	47.12

Table 5.2: Comparison between the number of pixel correspondences (N_{pc}), RMSE between the 3D reconstructed shape with the CAD model of the object and detection run time of the proposed method vs. the baseline method [27]. The run times are computed using a personal computer hardware with Windows operating system.

The quantitative evaluation of the proposed method on real objects, *Curved* and *ZigZag* surfaces, and 3D reconstruction results show the performance of the proposed method in practical situations in Tab. 5.1.

As suggested in Tab. 5.2 and based on the reconstructed point clouds in Fig. 5.5, the proposed method performs relatively well on a smooth surface, such as the *Curve* shape, while the results decrease on the *Zigzag* shape due to higher variations and sharper discontinuities.

The purpose of the proposed method is to improve the detection speed of the pipeline to be employable in practical situations. The detection run time along with the 3D reconstruction root mean square error (RMSE) for the proposed method and the method in [27] are compared, and the results are given in Tab. 5.2, where the CAD model of each shape is used as ground truth to evaluate a reconstructed point cloud. As illustrated in Tab. 5.2, the RMSE of the proposed method is slightly higher than the method in [27] because some of the detected points in the proposed method are not exactly positioned at the center of the intersection and thus the 3D points in the reconstructed point cloud are slightly misplaced. Additionally the optimization technique in [27] fits the point to the corner of the tags, which has less error in distance. Although the RMSE of the proposed method is higher than the method in [27], our proposed scheme offers ~ 50 times faster detection speed than its counterpart.

Chapter 6

Conclusions and Future Work

In Chapter 1, we introduced the problem of establishing accurate pixel correspondences and were motivated to develop a new method for detecting and obtaining them. In Chapter 2, we reviewed the background material related to the concepts used in this thesis. Detailed approaches were proposed in Chapter 4 and 5 to address the binarization and detection challenges formulated in Chapter 3. In this chapter we summarize the contributions of this thesis and present ideas for future studies.

Conclusions

In Chapter 4 of this thesis, we have proposed

- A locally adaptive thresholding method that
  - adaptively assigns a threshold value to each pixel with regard to the statistics of its local neighborhood;
  - is suitable for our SSSL system since it automatically tunes all parameters for selecting the optimum threshold value and does not require any input from the user.

In Chapter 5 we proposed

- A grid intersection detection method that

- detects grid intersections by fitting straight lines to the grid lines in an image;
- reduces the computational complexity by approximately 50 times compared to the baseline in [27];
- constructs sparser pixel correspondences since it detects grid intersections as opposed to the baseline method where four corners are found per intersection.

The experiments in Chapter 4 show a slight decrease in the accuracy of the proposed thresholding method. According to Tab. 4.1, however, the proposed method does not require any of the parameters to be tuned by the user as opposed to the baseline method [112]. Additionally, the proposed method suffers from the same condition as the baseline method such as sharp surface variations and discontinuities.

The experimental results in Chapter 5 shows an increase in distance error of 3 pixels on average on the final reconstructed point clouds compared to the baseline for the two tested objects. The accuracy of the proposed detection method, on the other hand, is expected to decrease as the surface of an object becomes more deformed and challenging; as the local areas include more severe discontinuities. Furthermore, since the assumption of the method is to find grid lines and detect the intersection from those lines, more deformation will result in curved grid lines or smaller line segments, which in turn will affect the performance of the detection method. Additionally, the final point cloud is sparser compared to the baseline method since only one point is detected per intersection.

Future Work

Although the proposed detection method is faster than the previous method, [27], it still lacks the desired accuracy in the reconstructed point clouds compared to the previous method. The resulting detected intersections are not at the center of the grid intersections as shown in Fig. 5.4. Finding these grid intersections correctly can improve the quality and prevents any deformation caused by misplacement of the detected points in the reconstructed point clouds. Since our current SSSL system tries to calibrate the camera projector system and reconstruct the 3D point cloud simultaneously, it is not possible to map the camera image back to the projector view to correct the pixel correspondences or find them because, the calibration parameters between the camera and the projector are not known. Therefore, the correspondences need to be corrected in the camera domain. Hence an idea to investigate in the future to tackle this problem is to superimpose markers at the grid intersections to correct the location of the detected grid intersections:

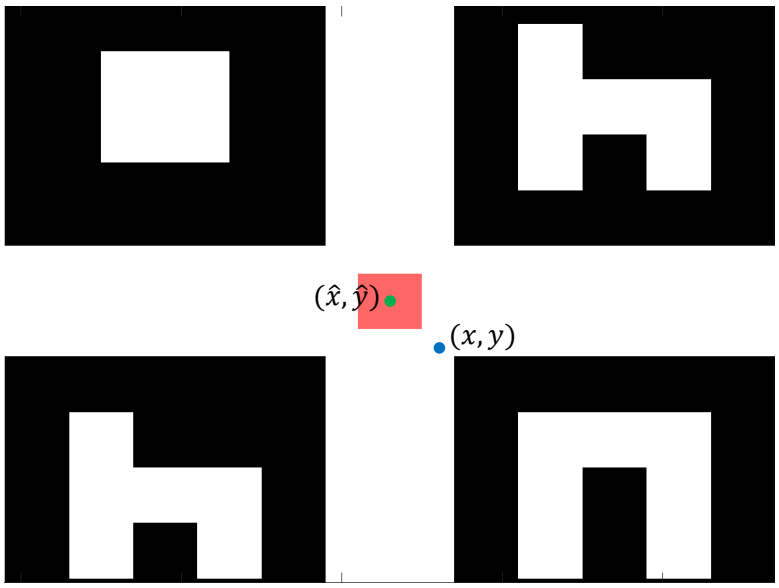

Figure 6.1: Rectangle red blob at the center of the intersection to use as a reference. Ideally, the centroid of the blob, $(\tilde{x}, \tilde{y})$, is the target point towards which the detected point, (x, y), should move.

- These reference markers can be black or colored blobs, as shown in Fig. 6.1, located at the center of intersections in the projector pattern, I_p, large enough to be detectable in the camera image after projection;
- Intensity or gradients can be used to detect the blobs and move the detected intersection to the center of the blobs.

The pattern in Fig. 6.1 shows the new design of the pattern. However, after projection, the camera captured image might affect the shape or intensity of the blobs.

References

[1] Ali T Abak, U Baris, and Bülent Sankur. The performance evaluation of thresholding algorithms for optical character recognition. In *Proceedings of the fourth international conference on document analysis and recognition*, volume 2, pages 697–700. IEEE, 1997.

[2] Yousset I Abdel-Aziz, HM Karara, and Michael Hauck. Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. *Photogrammetric engineering & remote sensing*, 81(2):103–107, 2015.

[3] Santiago Aja-Fernández, Ariel Hernán Curiale, and Gonzalo Vegas-Sánchez-Ferrero. A local fuzzy thresholding methodology for multiregion image segmentation. *Knowledge-Based Systems*, 83:1–12, 2015.

[4] Jaakko Astola and Leonid Yaroslavsky. *Advances in signal transforms: theory and applications*, volume 7. Hindawi Publishing Corporation, 2007.

[5] Joan Batlle, E Mouaddib, and Joaquim Salvi. Recent progress in coded structured light as a technique to solve the correspondence problem: a survey. *Pattern recognition*, 31(7):963–982, 1998.

[6] Johannes Bauer, Niko Sünderhauf, and Peter Protzel. Comparing several implementations of two recently published feature detectors. *IFAC Proceedings Volumes*, 40(15):143–148, 2007.

[7] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[8] John Bernsen. Dynamic thresholding of gray-level images. In *Proc. Eighth Int'l conf. Pattern Recognition, Paris*, 1986.

[9] Bir Bhanu. Automatic target recognition: State of the art survey. *IEEE transactions on aerospace and electronic systems*, (4):364–379, 1986.

[10] K. L. Boyer and A. C. Kak. Color-encoded structured light for rapid active ranging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1):14–28, 1987.

[11] Derek Bradley and Gerhard Roth. Adaptive thresholding using the integral image. *J. Graphics Tools*, 12(2):13–21, 2007.

[12] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[13] Jinhai Cai and Stan Miklavcic. Surface fitting for individual image thresholding and beyond. *IET Image Processing*, 7(6):596–605, 2013.

[14] Nabendu Chaki, Soharab Hossain Shaikh, and Khalid Saeed. A comprehensive survey on image binarization techniques. *Exploring Image Binarization Techniques*, pages 5–15, 2014.

[15] Ashutosh Kumar Chaubey. Comparison of the local and global thresholding methods in image segmentation. *World Journal of Research and Review*, 2(1):1–4, 2016.

[16] Qing Chen, Yuanzhe Chen, Dongyu Liu, Conglei Shi, Yingcai Wu, and Huamin Qu. Peakvizor: Visual analytics of peaks in video clickstreams from massive open online courses. *IEEE transactions on visualization and computer graphics*, 22(10):2315–2330, 2015.

[17] Xiangcheng Chen, Shunping Chen, Jie Luo, Mengchao Ma, Yuwei Wang, Yajun Wang, and Lei Chen. Modified gray-level coding method for absolute phase retrieval. *Sensors*, 17(10):2383, 2017.

[18] HD Cheng, Yen-Hung Chen, and Ying Sun. A novel fuzzy entropy approach to image enhancement and thresholding. *Signal Processing*, 75(3):277–301, 1999.

[19] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE computer society conference on computer vision and pattern recognition*, volume 1, pages 886–893. IEEE, 2005.

[20] Stanley R Deans. *The Radon transform and some of its applications.* Courier Corporation, 2007.

[21] Konstantinos G Derpanis. The harris corner detector. *York University*, 2, 2004.

[22] Umesh R Dhond and Jake K Aggarwal. Structure from stereo-a review. *IEEE transactions on systems, man, and cybernetics*, 19(6):1489–1510, 1989.

[23] Geng Du, Fei Su, and Anni Cai. Face recognition using surf features. In *Pattern Recognition and Computer Vision*, volume 7496, page 749628. International Society for Optics and Photonics, 2009.

[24] Nelson G Durdle, Jaishankar Thayyoor, and VJ Raso. An improved structured light technique for surface reconstruction of the human trunk. In *Conference Proceedings. IEEE Canadian Conference on Electrical and Computer Engineering (Cat. No. 98TH8341)*, volume 2, pages 874–877. IEEE, 1998.

[25] Olivier Ecabert, Jochen Peters, Hauke Schramm, Cristian Lorenz, Jens von Berg, Matthew J Walker, Mani Vembar, Mark E Olszewski, Krishna Subramanyan, Guy Lavi, et al. Automatic model-based segmentation of the heart in ct images. *IEEE transactions on medical imaging*, 27(9):1189–1201, 2008.

[26] S. Farsangi, M. A. Naiel, M. Lamm, and P. Fieguth. Efficient direct block address encoding for single-shot based 3d reconstruction. In *SID Symposium Digest of Technical Papers*, 2021. In Press.

[27] Sina Farsangi, Mohamed A Naiel, Mark Lamm, and Paul Fieguth. Rectification based single-shot structured light for accurate and dense 3d reconstruction. *Journal of Computational Vision and Imaging Systems*, 6(1):1–3, 2020.

[28] Sergio Fernandez, Joaquim Salvi, and Tomislav Pribanic. Absolute phase mapping for one-shot dense pattern projection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 64–71. IEEE, 2010.

[29] Yan Fu, Xu Chen, and Hui Gao. A new connected component analysis algorithm based on max-tree. In *IEEE International Conference on Dependable, Autonomic and Secure Computing*, pages 843–844. IEEE, 2009.

[30] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[31] Ryo Furukawa, Daisuke Miyazaki, Masashi Baba, Shinsaku Hiura, and Hiroshi Kawasaki. Robust structured light system against subsurface scattering effects achieved by CNN-based pattern detection and decoding algorithm. In *Proc. European Conf. on Computer Vision (ECCV) Workshops*, 2018.

[32] Jason Geng. Structured-light 3d surface imaging: a tutorial. *Advances in Optics and Photonics*, 3(2):128–160, 2011.

[33] Ercole Giuliao, Orazio Paita, and Luigi Stringa. Electronic character-reading system, 1977. US Patent 4,047,152.

[34] Davor Graovac, Srećko Jurić-Kavelj, and Ivan Petrović. Mobile robot pose tracking by correlation of laser range finder scans in hough domain. In *19th International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 273–278. IEEE, 2010.

[35] Xavier E Gros, Zheng Liu, Kazuhiko Tsukada, and Koichi Hanasaki. Experimenting with pixel-level ndt data fusion techniques. *IEEE Transactions on Instrumentation and Measurement*, 49(5):1083–1090, 2000.

[36] Armin Gruen and Emmanuel P Baltsavias. Close-range photogrammetry meets machine vision. In *SPIE*, volume 1395, 1990.

[37] Ladislav Halada, Gennadiy A Ososkov, and P SLAVKOVSKY. Histogram concavity analysis by quasicurvature. *Computers and artificial intelligence*, 6(6):523–533, 1987.

[38] Rostam Affendi Hamzah, Rosman Abd Rahim, and Zarina Mohd Noh. Sum of absolute differences algorithm in stereo correspondence problem for stereo matching in computer vision application. In *3rd International Conference on Computer Science and Information Technology*, volume 1, pages 652–657. IEEE, 2010.

[39] Rostam Affendi Hamzah and Haidi Ibrahim. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016, 2016.

[40] Miles Hansard, Seungkyu Lee, Ouk Choi, and Radu Patrice Horaud. *Time-of-flight cameras: principles, methods and applications*. Springer Science & Business Media, 2012.

[41] Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[42] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, USA, 2 edition, 2003.

[43] Richard I Hartley, Rajiv Gupta, and Tom Chang. Stereo from uncalibrated cameras. In *CVPR*, volume 92, pages 761–764, 1992.

[44] Richard I Hartley and Peter Sturm. Triangulation. *Computer vision and image understanding*, 68(2):146–157, 1997.

[45] Johan Hedborg, Andreas Robinson, and Michael Felsberg. Robust three-view triangulation done fast. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 152–157, 2014.

[46] Eli Horn and Nahum Kiryati. Toward optimal structured light patterns. *Image and Vision Computing*, 17(2):87–97, 1999.

[47] Paul VC Hough. Method and means for recognizing complex patterns, 1962.

[48] Bingyao Huang, Ying Tang, Samed Ozdemir, and Haibin Ling. A fast and flexible projector-camera calibration system. *IEEE Transactions on Automation Science and Engineering*, 2020.

[49] Idaku Ishii, Kenkichi Yamamoto, Kensuke Doi, and Tokuo Tsuji. High-speed 3d image acquisition using coded structured light projection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 925–930. IEEE, 2007.

[50] CV Jawahar, Prabir K Biswas, and AK Ray. Investigations on fuzzy thresholding based on fuzzy clustering. *Pattern Recognition*, 30(10):1605–1613, 1997.

[51] Guo-Quan Jiang, Cui-Jun Zhao, and Yong-Sheng Si. A machine vision based crop rows detection for agricultural robots. In *International Conference on Wavelet Analysis and Pattern Recognition*, pages 114–118. IEEE, 2010.

[52] Valérie Kaftandjian, Yue Min Zhu, Olivier Dupuis, and Daniel Babot. The combined use of the evidence theory and fuzzy logic for improving multimodal nondestructive testing systems. *IEEE Transactions on Instrumentation and Measurement*, 54(5):1968–1977, 2005.

[53] T Kalaiselvi and P Nagaraja. A rapid automatic brain tumor detection method for mri images using modified minimum error thresholding technique. *International journal of imaging systems and technology*, 1(25):77–85, 2015.

[54] Mohamed Kamel and Aiguo Zhao. Extraction of binary character/graphics images from grayscale document images. *CVGIP: Graphical Models and Image Processing*, 55(3):203–217, 1993.

[55] Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image matching using sift, surf, brief and orb: performance comparison for distorted images. *arXiv preprint arXiv:1710.02726*, 2017.

[56] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–II. IEEE, 2004.

[57] Josef Kittler and John Illingworth. Minimum error thresholding. *Pattern recognition*, 19(1):41–47, 1986.

[58] Anestis Koutsoudis, Blaž Vidmar, George Ioannakis, Fotis Arnaoutoglou, George Pavlidis, and Christodoulos Chamzas. Multi-image 3d reconstruction data evaluation. *Journal of cultural heritage*, 15(1):73–79, 2014.

[59] P. Lavoie, D. Ionescu, and E. M. Petriu. 3-d object model recovery from 2-d images using structured light. In *IEEE Instrumentation and Measurement Technology Conference and IMEKO Tec*, volume 1, pages 377–382, 1996.

[60] Nalpantidis Lazaros, Georgios Christou Sirakoulis, and Antonios Gasteratos. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics*, 2(4):435–462, 2008.

[61] JJ Le Moigne and Allen Mark Waxman. Structured light patterns for robot mobility. *IEEE Journal on Robotics and Automation*, 4(5):541–548, 1988.

[62] Chulhee Lee, Shin Huh, Terence A Ketter, and Michael Unser. Unsupervised connectivity-based thresholding segmentation of midsagittal brain mr images. *Computers in biology and medicine*, 28(3):309–338, 1998.

[63] Sang Uk Lee, Seok Yoon Chung, and Rae Hong Park. A comparative performance study of several global thresholding techniques for segmentation. *Computer Vision, Graphics, and Image Processing*, 52(2):171–190, 1990.

[64] Chi-Kin Leung and FK Lam. Performance analysis for a class of iterative image thresholding algorithms. *Pattern Recognition*, 29(9):1523–1530, 1996.

[65] Chun Hung Li and CK Lee. Minimum cross entropy thresholding. *Pattern recognition*, 26(4):617–625, 1993.

[66] F. Li, H. Sekkati, J. Deglint, C. Scharfenberger, M. Lamm, D. Clausi, J. Zelek, and A. Wong. Simultaneous projector-camera self-calibration for three-dimensional reconstruction and projection mapping. *IEEE Trans. on Computational Imaging*, 3(1):74–83, 2017.

[67] Yali Li, Shengjin Wang, Qi Tian, and Xiaoqing Ding. A survey of recent advances in visual feature detection. *Neurocomputing*, 149:736–751, 2015.

[68] Zhongwei Li, Yusheng Shi, Congjun Wang, and Yuanyuan Wang. Accurate calibration method for a structured light system. *Optical Engineering*, 47(5):053604, 2008.

[69] Hai Bo Lin and Zhan Song. A twofold symmetry based approach for the feature detection of pseudorandom color pattern. In *Proc. IEEE Int. Conf. on Information Science and Technology*, pages 623–626, 2014.

[70] DE Lloyd. Automatic target classification using moment invariant of image shapes. *IDN AW126, RAE, Farnborough, Reino Unido*, 1985.

[71] Jun Lu, Jirui Han, Elahi Ahsan, Guihua Xia, and Qi Xu. A structured light vision measurement with large size m-array for dynamic scenes. In *35th Chinese Control Conference (CCC)*, pages 3834–3839, 2016.

[72] F.J. MacWilliams and N.J.A. Sloane. Pseudo-random sequences and arrays. *Proceedings of the IEEE*, 64(12):1715–1729, 1976.

[73] Minoru Maruyama and Shigeru Abe. Range sensing by projecting multiple slits with random cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):647–651, 1993.

[74] Xavier Maurice, Pierre Graebling, and Christophe Doignon. Epipolar based structured light pattern design for 3-D reconstruction of moving surfaces. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 5301–5308, 2011.

[75] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10):1615–1630, 2005.

[76] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1):43–72, 2005.

[77] Ondrej Miksik and Krystian Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *Proceedings of the 21st International Conference on Pattern Recognition*, pages 2681–2684. IEEE, 2012.

[78] Yosuke Miyajima and Tsutomu Maruyama. A real-time stereo vision system with fpga. In *International Conference on Field Programmable Logic and Applications*, pages 448–457. Springer, 2003.

[79] Raymond A Morano, Cengizhan Ozturk, Robert Conn, Stephen Dubin, Stanley Zietz, and J Nissano. Structured light using pseudorandom codes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(3):322–327, 1998.

[80] Daniel Moreno and Gabriel Taubin. Simple, accurate, and robust projector-camera calibration. In *Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 464–471. IEEE, 2012.

[81] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[82] MSR Naidu, P Rajesh Kumar, and K Chiranjeevi. Shannon and fuzzy entropy based evolutionary image thresholding for image segmentation. *Alexandria engineering journal*, 57(3):1643–1655, 2018.

[83] Pauline C Ng and Steven Henikoff. Sift: Predicting amino acid changes that affect protein function. *Nucleic acids research*, 31(13):3812–3814, 2003.

[84] Wayne Niblack. *An introduction to digital image processing*. Prentice-Hall, Inc., 1990.

[85] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[86] Lawrence O'Gorman. Binarization and multithresholding of document images using connectivity. *CVGIP: Graphical Models and Image Processing*, 56(6):494–506, 1994.

[87] Jordi Pages, Joaquim Salvi, and Carles Matabosch. Robust segmentation and decoding of a˜ grid pattern for structured light. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 689–696. Springer, 2003.

[88] Nikhil R Pal and Sankar K Pal. Entropic thresholding. *Signal processing*, 16(2):97–108, 1989.

[89] Paul W Palumbo, Puducode Swaminathan, and Sargur N Srihari. Document image binarization: Evaluation of algorithms. In *Applications of Digital Image Processing IX*, volume 697, pages 278–285. International Society for Optics and Photonics, 1986.

[90] E.M. Petriu, Z. Sakr, H.J.W. Spoelder, and A. Moica. Object recognition using pseudo-random color encoded structured light. In *Proceedings of the 17th IEEE Instrumentation and Measurement Technology Conference [Cat. No. 00CH37066]*, volume 3, pages 1237–1241, 2000.

[91] Neerad Phansalkar, Sumit More, Ashish Sabale, and Madhuri Joshi. Adaptive local thresholding for detection of nuclei in diversity stained cytology images. In *International Conference on Communications and Signal Processing*, pages 218–220. IEEE, 2011.

[92] Gurbakash Phonsa and K Manu. A survey: Image segmentation techniques. In *Harmony Search and Nature Inspired Optimization Algorithms*, pages 1123–1140. Springer, 2019.

[93] William H Press, H William, Saul A Teukolsky, William T Vetterling, A Saul, and Brian P Flannery. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[94] Tomislav Pribanić, Saša Mrvoš, and Joaquim Salvi. Efficient multiple phase shift patterns for dense 3d acquisition in structured light scanning. *Image and Vision Computing*, 28(8):1255–1266, 2010.

[95] Thierry Pun. A new method for grey-level picture thresholding using the entropy of the histogram. *Signal Processing*, 2(3):223–237, 1980.

[96] J Radon. On the determination of functions from their integrals along certain manifolds. *Trans. of the American Mathematical Society*, 69:262–277, 1917.

[97] Christoph Resch, Peter Keitler, Christoffer Menk, and Gudrun Klinker. Semi-automatic calibration of a projector-camera system using arbitrary objects with known geometry. In *IEEE Virtual Reality (VR)*, pages 271–272. IEEE, 2015.

[98] TW Ridler, S Calvard, et al. Picture thresholding using an iterative selection method. *IEEE trans syst Man Cybern*, 8(8):630–632, 1978.

[99] Lawrence Roberts. *Machine Perception of Three-Dimensional Solids*. 01 1963.

[100] Azriel Rosenfeld. Picture processing by computer. *ACM Computing Surveys (CSUR)*, 1(3):147–176, 1969.

[101] Azriel Rosenfeld and Pilar De La Torre. Histogram concavity analysis as an aid in threshold selection. *IEEE Transactions on Systems, Man, and Cybernetics*, (2):231–235, 1983.

[102] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *International conference on computer vision*, pages 2564–2571. IEEE, 2011.

[103] Kasra Sadatsharifi, Mohamed A Naiel, Mark Lamm, and Paul Fieguth. Locally adaptive thresholding for single-shot structured light patterns. *Journal of Computational Vision and Imaging Systems*, 6(1):1–3, 2020.

[104] Kasra Sadatsharifi, Mohamed A Naiel, Mark Lamm, and Paul Fieguth. Efficient model based grid intersection detection for single-shot 3d reconstruction. 2021. manuscript under review.

[105] R. Sagawa, Y. Ota, Y. Yagi, R. Furukawa, N. Asada, and H. Kawasaki. Dense 3d reconstruction method using a single pattern for fast moving object. In *IEEE 12th International Conference on Computer Vision*, pages 1779–1786, 2009.

[106] S Sahasrabudhe and K Gupta. A valley-seeking threshold selection. *Computer vision and image processing*, 55, 1992.

[107] Ezzatollah Salari, Eddie Chou, James J Lynch, et al. Pavement distress evaluation using 3d depth information from stereo vision. Technical report, Michigan Ohio University Transportation Center, 2012.

[108] Joaquim Salvi, Sergio Fernandez, Tomislav Pribanic, and Xavier Llado. A state of the art in structured light patterns for surface profilometry. *Pattern recognition*, 43(8):2666–2680, 2010.

[109] Joaquim Salvi, Jordi Pagès, and Joan Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37:827–849, 2004.

[110] Bülent Sankur, Ali T Abak, and U Baris. Assessment of thresholding algorithms for document processing. In *International Conference on Image Processing (Cat. 99CH36348)*, volume 1, pages 580–584. IEEE, 1999.

[111] Soham Sarkar, Gyana Ranjan Patra, and Swagatam Das. A differential evolution based approach for multilevel image segmentation using minimum cross entropy thresholding. In *International Conference on Swarm, Evolutionary, and Memetic Computing*, pages 51–58. Springer, 2011.

[112] Jaakko Sauvola and Matti Pietikäinen. Adaptive document image binarization. *Pattern Recognition*, 33(2):225–236, 2000.

[113] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–I. IEEE, 2003.

[114] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of computer vision*, 37(2):151–172, 2000.

[115] Felix Scholkmann, Jens Boss, and Martin Wolf. An efficient algorithm for automatic peak detection in noisy periodic and quasi-periodic signals. *Algorithms*, 5(4):588–603, 2012.

[116] M Ibrahim Sezan. A peak detection algorithm and its application to histogram-based image data reduction. *Computer vision, graphics, and image processing*, 49(1):36–51, 1990.

[117] Mehmet Sezgin and Bülent Sankur. Selection of thresholding methods for nondestructive testing applications. In *International Conference on Image Processing (Cat. No. 01CH37205)*, volume 3, pages 764–767. IEEE, 2001.

[118] Mehmet Sezgin and Bülent Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *J. Electronic imaging*, 13(1):146–166, 2004.

[119] Mehmet Sezgin and Ramazan Taşaltın. A new dichotomization technique to multilevel thresholding devoted to inspection applications. *Pattern Recognition Letters*, 21(2):151–161, 2000.

[120] Faisal Shafait, Daniel Keysers, and Thomas M Breuel. Efficient implementation of local adaptive thresholding techniques using integral images. In *Document recognition and retrieval XV*, volume 6815, page 681510. International Society for Optics and Photonics, 2008.

[121] Linda G Shapiro. Connected component labeling and adjacency graph construction. In *Machine Intelligence and Pattern Recognition*, volume 19, pages 1–30. Elsevier, 1996.

[122] Manoj Kumar Sharma. A survey of thresholding techniques over images. *INROADS-An International Journal of Jaipur National University*, 3(2):461–478, 2014.

[123] Jianbo Shi et al. Good features to track. In *IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[124] Jichuan Shi. Adaptive local threshold with shape information and its application to oil sand image segmentation. 2010.

[125] Teresa M Silberberg, Larry Davis, and David Harwood. An iterative hough procedure for three-dimensional object recognition. *Pattern Recognition*, 17(6):621–629, 1984.

[126] T Romen Singh, Sudipta Roy, O Imocha Singh, Tejmani Sinam, Kh Singh, et al. A new local adaptive thresholding technique in binarization. *preprint arXiv:1201.5227*, 2012.

[127] Lifang Song, Suming Tang, and Zhan Song. A robust structured light pattern decoding method for single-shot 3D reconstruction. In *Proc. IEEE Int. Conf. on Real-time Computing and Robotics (RCAR)*, pages 668–672, 2017.

[128] Sun-Gu Sun. Target detection using local fuzzy thresholding and binary template matching in forward-looking infrared images. *Optical Engineering*, 46(3):036402, 2007.

[129] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[130] Suming Tang, Xu Zhang, Zhan Song, Lifang Song, and Hai Zeng. Robust pattern decoding in shape-coded structured light. *Optics and Lasers in Engineering*, 96:50–62, 2017.

[131] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *International conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–10. IEEE, 2018.

[132] Federico Tombari and Luigi Di Stefano. Object recognition in 3d scenes with occlusions and clutter by hough voting. In *Fourth Pacific-Rim Symposium on Image and Video Technology*, pages 349–355. IEEE, 2010.

[133] Oeivind Due Trier and Anil K. Jain. Goal-directed evaluation of binarization methods. *IEEE transactions on Pattern analysis and Machine Intelligence*, 17(12):1191–1201, 1995.

[134] R. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987.

[135] Wen-Hsiang Tsai. Moment-preserving thresolding: A new approach. *Computer Vision, Graphics, and Image Processing*, 29(3):377–393, 1985.

[136] Robert J Valkenburg and Alan M McIvor. Accurate 3d measurement using a structured light system. *Image and Vision Computing*, 16(2):99–110, 1998.

[137] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[138] Zhenzhou Wang, Qi Zhou, and YongCan Shuang. Three-dimensional reconstruction with single-shot structured light dot pattern and analytic solutions. *Measurement*, 151:107114, 2020.

[139] Pierre D Wellner. Adaptive thresholding for the digitaldesk. *Xerox, EPC1993-110*, pages 1–19, 1993.

[140] Juyang Weng, Paul Cohen, Marc Herniou, et al. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on pattern analysis and machine intelligence*, 14(10):965–980, 1992.

[141] John S Weszka, WESZKA JS, et al. Histogram modification for threshold selection. 1979.

[142] RJ Whatmough. Automatic threshold selection from a histogram using the "exponential hull". *CVGIP: Graphical Models and Image Processing*, 53(6):592–600, 1991.

[143] James M White and Gene D Rohrer. Image thresholding for optical character recognition and other applications requiring character image extraction. *IBM Journal of research and development*, 27(4):400–411, 1983.

[144] Choong-Wan Woo, Anjali Krishnan, and Tor D Wager. Cluster-extent based thresholding in fmri analyses: pitfalls and recommendations. *Neuroimage*, 91:412–419, 2014.

[145] Pinghua Xu, Xuemei Ding, Rongwu Wang, and Xiongying Wu. Feature-based 3d reconstruction of fabric by binocular stereo-vision. *The Journal of The Textile Institute*, 107(1):12–22, 2016.

[146] Feixiang Yan, Hong Zhang, and C Ronald Kube. A multistage adaptive thresholding method. *Pattern recognition letters*, 26(8):1183–1191, 2005.

[147] Kai Yang, Zhi Ling, Jinlong Li, Xiaorong Gao, Liming Xie, and Zijian Bai. Color m-array shape reconstruction of using grid points and center points. In *Int. Conf. Inf. Optics and Photonics (CIOP)*, volume 11209, 2019.

[148] Kui Yang, Wei Fang, Yan Zhao, and Nianmao Deng. Iteratively reweighted midpoint method for fast multiple view triangulation. *IEEE Robotics and Automation Letters*, 4(2):708–715, 2019.

[149] Yi Yang and Shawn Newsam. Comparing sift descriptors and gabor texture features for classification of remote sensed imagery. In *15th IEEE international conference on image processing*, pages 1852–1855. IEEE, 2008.

[150] MK Yanni and E Horne. A new approach to dynamic thresholding. In *EUSIPCO: 9th European Conf. Sig. Process*, volume 1, pages 34–44, 1994.

[151] Shimon D Yanowitz and Alfred M Bruckstein. A new method for image segmentation. *Computer Vision, Graphics, and Image Processing*, 46(1):82–95, 1989.

[152] Zhi-Wei Ye, Ming-Wei Wang, Wei Liu, and Shao-Bin Chen. Fuzzy entropy based optimal thresholding using bat algorithm. *Applied Soft Computing*, 31:381–395, 2015.

[153] Jui-Cheng Yen, Fu-Juay Chang, and Shyang Chang. A new criterion for automatic multilevel thresholding. *IEEE Transactions on Image Processing*, 4(3):370–378, 1995.

[154] Peng-Yeng Yin. Multilevel minimum cross entropy threshold selection based on particle swarm optimization. *Applied mathematics and computation*, 184(2):503–513, 2007.

[155] Song Zhang. High-speed 3d shape measurement with structured light methods: A review. *Optics and Lasers in Engineering*, 106:119–131, 2018.

[156] Song Zhang and Peisen S Huang. Novel method for structured light system calibration. *Optical Engineering*, 45(8):083601, 2006.

[157] Xing Zhang, Jie Tian, Kexin Deng, Yongfang Wu, and Xiuli Li. Automatic liver segmentation using a statistical shape model with optimal surface detection. *IEEE Transactions on Biomedical Engineering*, 57(10):2622–2626, 2010.

[158] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[159] Fuqiang Zhong, Ravi Kumar, and Chenggen Quan. A cost-effective single-shot structured light system for 3d shape measurement. *IEEE Sensors Journal*, 19(17):7335–7346, 2019.