UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

MASTER DEGREE IN
CONTROL SYSTEMS ENGINEERING

Rectification Strategies for a Binary Coded Structured Light 3D Scanner

---

Author:
Mattia BARO

Supervisor:
Prof. Alberto PRETTO
Co-Supervisor:
Ing. Nicola CARLON

ACADEMIC YEAR: 2022/2023
Oct 18, 2022

Abstract of thesis entitled

Rectification Strategies for a Binary Coded Structured Light 3D Scanner

Submitted by

Mattia BARO

for the degree of Doctor in Control System Engineering

at The University of Padua

in October, 2022

Making a computer able to see exactly as a human being does was for many years one of the most interesting and challenging tasks involving lots of experts and pioneers in fields such as Computer Science and Artificial Intelligence. As a result, a whole field called Computer Vision has emerged becoming very soon a part of our daily life. The successful methodologies of this discipline have been applied in countless areas of application and their use is still in continuous expansion. On the other hand, in an increasing number of applications extracting information from simple 2D images is not enough and what is more requested instead is to use three-dimensional imaging techniques in order to reconstruct the 3D shape of the imaged objects and scene. The techniques developed in this context include both active systems, where some form of illumination is projected onto the scene, and passive systems, where the natural illumination of the scene is used.

Among the active systems, one of the most reliable approaches for recovering the surface of objects is the use of structured light. This technique is based on projecting a light pattern and viewing the illuminated scene from one or more points of view. Since the pattern is coded, correspondences between image points and points of the projected pattern can be easily found. In particular, the performances of this kind of 3D scanner are determined by two key aspects, the accuracy and the acquisition time.

This thesis aims to design and experiment some rectification strategies for a prototype of binary coded structured light 3D scanner. The rectification is a commonly used technique for stereo vision systems which, in case of structured light, facilitates the establishment of correspondences across a projected pattern and an acquired image and reduces the number of pattern images to be projected, resulting finally in a speeding-up of the acquisition times.

Rectification Strategies for a Binary Coded Structured Light 3D Scanner

by

Mattia BARO

October, 2022

Contents

Chapter 1

Introduction

The physical world around us is a three-dimensional world. Despite this fact, commonly used technologies such as traditional cameras and imaging sensors are able to acquire only two-dimensional images losing the depth information. This fundamental restriction greatly limits the perception of real-world environment.

Figure 1.1: Depth perception in real-world environment

The past several decades have marked tremendous advances in research, development, and commercialization of 3D surface imaging technologies, stimulated by application demands in a variety of market segments, advances in high-resolution and high-speed electronic imaging sensors, and

ever-increasing computational power. Nowadays different 3D surface imaging techniques are available in order to measure the (x, y, z) coordinates of points on the surface of an object, returning as output a point cloud where each surface point is associated with some kind of scalar value. Likewise, a colored point cloud is represented by $\{P_i = (x_i, y_i, z_i, r_i, g_i, b_i),\ i = 1, 2, ..., N\}$, where the vector (r_i, g_i, b_i) represents the red, green, and blue color components associated with the i-th surface point.

Figure 1.2: Example of point cloud returned by a 3D reconstruction

One principal method of 3D surface imaging is based on the use of structured light. A structured light scanning system projects different light patterns, or structures, and captures the light as it falls onto the scene. It then uses the information about how the patterns appear after being distorted by the scene to eventually recover the 3D geometry. The potential speed of data acquisition, non-contact nature, the availability of necessary hardware, and the high precision of measurement offered by modern 3D structured light scanning technologies are what make them highly adoptable in industries such as medicine, biology, manufacturing, security, communications, remote environment reconstruction, and consumer electronics.

As the number of applications in which structured light techniques are employed increases, more interesting and challenging problems arise. It should be noted that there is not one 3D sensing technology that solves each issue and works as a general solution. Structured light in particular is still

Figure 1.3: Use of structured light for 3D surface imaging

nowadays one of the most reliable approaches, able to provide good performances both from the accuracy and acquisition time point of view.

1.1 Goal of the thesis

This master thesis aims to improve the performances presented by a prototype of structured light 3D scanner, realized with low-cost hardware, by experimenting some rectification strategies.

As known from basic computer vision, rectification is the process of transforming a pair of camera frames as they have been acquired by two perfectly aligned cameras with the same focal length. The two rectifying transformations are homographies and are calculated from a combination of the intrinsic parameters of each optical device, and the extrinsic parameters linking each device's frame. By using epipolar geometry terminology, one can equivalently say that, by rectification, corresponding epipolar lines between the two camera frames will align, by rows or by columns according to the cameras' configuration. It is easy to deduce that rectification is a commonly used approach for stereo vision systems. This kind of device usually employs two cameras in order to see the same object from two different points of view and produces as output a 3D point cloud simply by analyzing the differences between the two images captured simultaneously by the cameras. Essentially, one needs to accurately identify the pixels that

represent the projection of the same 3D point in both images, known as the problem of correspondence between the two cameras, and once a so called correspondence is found the associated 3D point comes from a triangulation process. If the two cameras are placed parallel one to the other, given a point in one image, its correspondence in the other image is on the same epipolar line. As result, rectification can transform the correspondence problem from 2D to 1D search resulting in a speeding up of the entire 3D stereo vision system.

What about a structured light system? Being the principle of work of this kind of sensor very similar to stereo vision, rectification could be applied also in this slightly different scenario by keeping in mind that one camera is substituted with a projector. In particular in this master thesis two different approaches are proposed for rectifying a structured light 3D scanner: one acting only on the camera and the other involving both camera and projector. Differently from the first approach, that has been formulated only from the theoretical point of view, the second one has been effectively implemented on the available prototype of 3D scanner. This prototype consists basically in a binary coded structured light sensor which works by projecting a sequence of patterns composed by vertical and horizontal binary stripes. In particular, binary coded technology offers very robust results given the simple patterns to be projected but at the same time presents quite long acquisition times since for each acquisition multiple patterns are required.

In particular, the practical part of this thesis project can be summarized in three steps:

1. Mounting the sensor in a new support. This implies that the projector-camera system must be accurately re-calibrated.

2. Implementing the second rectification strategy which aims to horizontally align camera and projector. As result, only vertical patterns need to be projected in order to find horizontal correspondences, since vertical correspondences come from the alignment.

3. Compare the performances obtained by the sensor before and after the rectification in order to see if the rectification could be effectively considered as an improvement for the structured light sensor.

It is worth mentioning that the rectification of a projector-camera system is not a simple task if compared with a common stereo system. Many issues come from the presence of the projector in place of a camera. In addition, the eventually different resolution between the camera and projector may complicate the alignment between these two devices. Moreover, the rectification of the camera image implies a pixel interpolation and, as consequence, a loss of information. As result, it is expected as final solution a speeding-up of the acquisition time but also a slightly less accurate 3D reconstruction.

1.2 Related works

The already mentioned prototype used for this thesis project has been realized from scratch by student Mattia Piccoli in his master thesis [21]. In particular, Fig. 1.4 shows how the sensor appears in his original support. In a nutshell, besides the calibration of the device, this thesis presents also

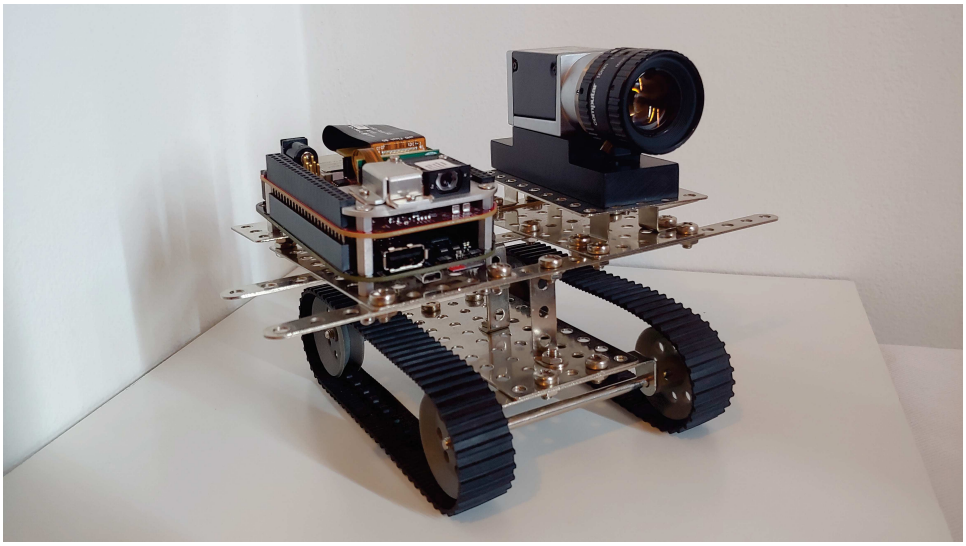


Figure 1.4: Mattia Piccoli master thesis project

the comparison between different coding and decoding strategies applied directly on the sensor. More in detail, it has been tested the binary and Gray code as coding strategies, while for the binarization of the acquired camera images under patterns illumination, it has been experimented the comparison of the camera image with a uniform image and with a camera image taken under complementary pattern illumination. This last binarization strategy together with the Gray code, are the choice that leads to the

best performances. More information about the working principle of this binary coded structured light sensor will be provided later on. As well as Mattia Piccoli's thesis, there exists many other projects dealing with the realization of structured light sensors. One example is SLStudio [32], see Fig. 1.5, a modular open-source software that allows everyone to develop a custom structured light 3D scanner controlled by a dedicated GUI.

Figure 1.5: SLStudio framework

Even if structured light systems are used in 3D reconstruction for many years now, many researchers are still involved in this field trying to further improve the performances of these kinds of sensors. This is mainly due to the fact that realizing a 3D scanner based on this technology is relatively simple and cheap and at the same time provides interesting performances. In particular, in the rectification context, one of the most recent results comes from [22]. A method, called *inverse rectification*, is proposed, which facilitates the establishment of correspondences across a projected pattern and an acquired image. In this case a pattern of features comprising vertical dashes is warped by the inverse of the rectifying homography of the projector-camera pair, prior to projection. This warping imparts upon the system the property that projected features will fall on distinct conjugate epipolar lines of the rectified projector and acquired camera images. This reduces the correspondence search to a trivial constant-time table lookup once a feature is found in the camera image, and leads to robust, accurate, and extremely efficient disparity calculations. A projector-camera range sensor is developed based on this method, and is shown experimentally to be effective, with bandwidth exceeding some existing consumer-level range sensors. Another paper dealing with rectification of structured light systems is [24]. In this case, rectification is only a small part of the entire proposed work, but it can be deduced

that the solution adopted consists in rectifying both camera and projector, exactly as a stereo pair. This means that common rectification approaches for stereo systems can be used, as the one proposed in [15].

1.3 About IT+Robotics

This master thesis project has been developed thanks to an internship at IT+Robotics srl, a well-known industry leader in advanced 3D vision for robotics and automation.

Figure 1.6: EyeT+ Pick, one of the standard products at IT+Robotics

The company has been founded in 2005 by the collaboration of professors at the University of Padua and a group of researchers in the field of Robotics and nowadays it has a deep understanding and expertise in artificial vision systems, visual inspection and vision-guided robotic systems. In Fig. 1.6 is depicted one of the most successful standard products for the company, the *EyeT+ Pick*. This is basically a vision system developed specifically for random bin-picking, so it allows to recognize, precisely locate and grasp objects randomly placed inside a bin. What makes this product so interesting, is its flexibility: it can be easily integrated in existing working processes and it presents a very compact structure with respect to other devices. Among the other standard products of the company, it is worth mentioning also the *EyeT+ Inspect*, that is instead a visual system for quality inspection.

From these examples it is easy to understand how the field of 3D vision, of which also this thesis is part, is fundamental in an Industry 4.0 scenario.

1.4 Thesis organization

The following chapters of this thesis are structured so that the reader is progressively driven to the core of the project. The next chapter presents basic notions of camera and multi-camera geometry, while the third chapter starts with an overview of structured light sensors for moving then to a description of the particular prototype used for this project. These two chapters are important in order to motivate all the choices taken on the experimental part of the thesis. Chapter 4 illustrates the two proposed rectification strategies and how the second one can be applied to the specific prototype of a structured light 3D scanner, while Chapter 5 shows the experimental results comparing them with ones obtained by the non-rectified 3D scanner. Finally, Chapter 6 provides final considerations about the strategies implemented in this thesis evaluating also the possibility to further improve the performances of the scanner.

Chapter 2

Basis of camera and multi-camera geometry

This chapter aims to provide an overview of camera and multi-camera geometry. The mathematical notions presented in this chapter allow to deeply understand the main strategies that will be adopted in this thesis in order to improve the performance of the sensor.

2.1 Single-view geometry

This section describes the frontal pinhole camera model depicted in Fig. 2.1, and how a point Q in the 3D space is mapped through this model into a point q on the image plane.

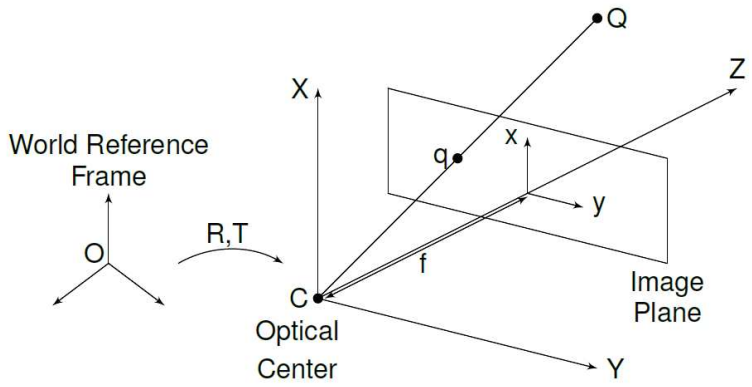


Figure 2.1: Framework for the single camera view problem

Given the world reference frame, consider the camera frame (X, Y, Z) attached to the camera, whose Z-axis, known as optical axis, is directed towards the scene. The relation in the space between these two frames is given

by a rototranslation transformation defined by the matrices $(R, T) \in \mathbb{SE}(3)$. The distance between the optical center C and the image plane is the focal length f. The point Q can be rewritten in terms of world reference frame coordinates as Q_O and in terms of camera frame coordinates as Q_C

$$Q \quad \longrightarrow \quad Q_O = (X_O, Y_O, Z_O), \quad Q_C = (X, Y, Z) \tag{2.1}$$

while the point q in the image plane can be represented as:

$$q \quad \longrightarrow \quad \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.2}$$

2.1.1 Derivation of the camera matrix

After having introduced the setup for the single camera view problem, the aim now is to derive how the parameters of the pinhole camera model can be summarized into one matrix P, called *camera matrix*. This matrix can be obtained by considering the following composition of geometric transformations:

1. World 3D coordinates $\rightarrow$ Camera 3D coordinates

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R \begin{bmatrix} X_O \\ Y_O \\ Z_O \end{bmatrix} + T \quad \Longrightarrow \quad Q_C = RQ_O + T \tag{2.3}$$

2. Camera 3D coordinates $\rightarrow$ Image plane 2D coordinates Consider to project the point Q into the image plane as illustrated in Fig. 2.2. This geometric operation results in two triangles along the yz and xz directions and, from similar triangles principle, the transformation from a 3D point in the camera frame to the corresponding 2D point in the image plane can be written as follows:

$$\begin{cases} x = f\frac{X}{Z} \\ y = f\frac{Y}{Z} \end{cases} \tag{2.4}$$

meaning that to every point q are associated two equations called *perspective geometry equations*.

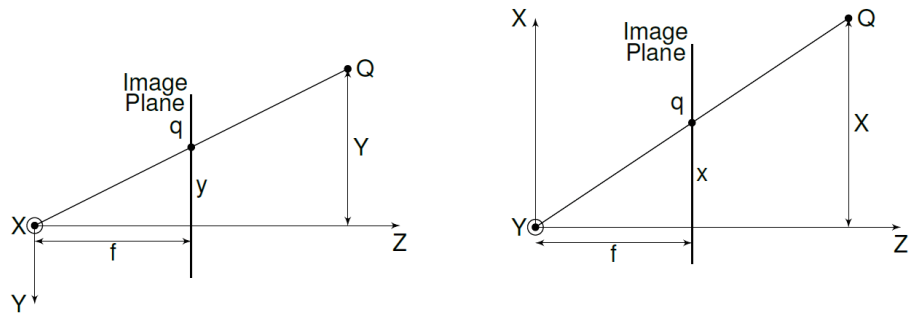


Figure 2.2: *yz* and *xz* directions view of the Q point projection q on the image plane

3. Image plane 2D coordinates $\rightarrow$ 2D homogeneous coordinates The transformation from canonical coordinates to homogeneous coordinates and its inverse relation is defined as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow \lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} \longrightarrow \lambda \begin{bmatrix} \frac{\alpha}{\gamma} \\ \frac{\beta}{\gamma} \end{bmatrix}, \qquad \lambda \neq 0 \tag{2.5}$$

Notice that homogeneous coordinates are defined up to a scaling factor λ that is not known in general, and this implies that in homogeneous notation the two following forms represent the same point:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} \tag{2.6}$$

4. Applying homogeneous coordinates to perspective geometry equations

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f\frac{X}{Z} \\ f\frac{Y}{Z} \\ 1 \end{bmatrix} \sim \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \left[\begin{array}{ccc|c} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \left[I_{3x3} \mid 0 \right] \left[\begin{array}{c|c} R & T \\ \hline 0 & 1 \end{array}\right] \begin{bmatrix} X_O \\ Y_O \\ Z_O \\ 1 \end{bmatrix} = K_f \cdot \Pi_0 \cdot g \cdot \begin{bmatrix} X_O \\ Y_O \\ Z_O \\ 1 \end{bmatrix} \tag{2.7}$$

where K_f is called *intrinsic parameters matrix* and Π_0 is called *standard projection matrix*.

5. Millimeters to pixels transformation In the real case of digital cameras, the image plane is discretized in a finite number of pixels, therefore on the image plane two reference frames must be considered, one in pixels and the other in millimeters. In the following, $[\tilde{x}\ \tilde{y}]^T$ represents the coordinates in pixels, while $[x\ y]^T$ is the respective representation in millimeters and, using this notation, the relation between the two can be described as follows:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = f \begin{bmatrix} S_x & S_\theta \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} O_x \\ O_y \end{bmatrix} \tag{2.8}$$

where f is the focal length that plays the role of a scaling factor, S_x and S_y are the pixel/mm transformation coefficients and $[O_x\ O_y]^T$ is an offset vector that keep into account the fact that usually the pixel coordinate frame is centered in the top left corner of the image plane. The skew component S_θ instead is related to the fact that the pixels may not be rectangular, as illustrated in Fig. 2.3. Real cameras are also affected by radial and tangential

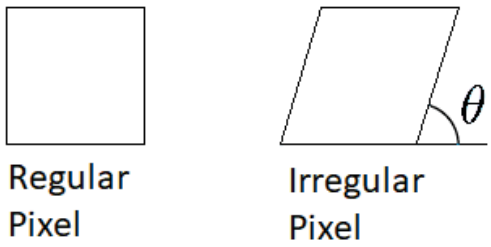


Figure 2.3: Difference between a regular pixel and an irregular one

distortion. For simplicity, these effects are not included in this camera model but in practical applications they must be considered and removed, by calibrating the camera for instance. More details about the modelling of camera distortion can be found in [9].

6. Putting it all together The complete model for the intrinsic parameters takes into account the fact that pixels may not be rectangular through the skew component S_θ, related to the angle θ that affects the x coordinate. Usually it is assumed $\theta \simeq \pi/2$ so that $S_\theta = 0$, but in a more accurate model the

final equation 2.7 must be rewritten as:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & S_\theta & O_x \\ 0 & S_y & O_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \Pi_0 \, g \begin{bmatrix} X_O \\ Y_O \\ Z_O \\ 1 \end{bmatrix} = K_s \cdot K_f \cdot \Pi_0 \cdot g \cdot \begin{bmatrix} X_O \\ Y_O \\ Z_O \\ 1 \end{bmatrix} \quad (2.9)$$

Therefore the camera matrix is defined as $P = K \cdot \Pi_0 \cdot g$ in which the camera parameters are divided in extrinsic parameters contained in g and intrinsic parameters contained in $K = K_s \cdot K_f$; in particular matrix K_s takes into account the pixel/mm mapping.

2.2 Two-view geometry

This section covers the geometry of two perspective views, as illustrated in Fig. 2.4. Even if the focus of this thesis is more oriented to the case of two views acquired simultaneously by a stereo rig, it is worth mentioning that the reasoning is the same for a camera moving relative to the scene.

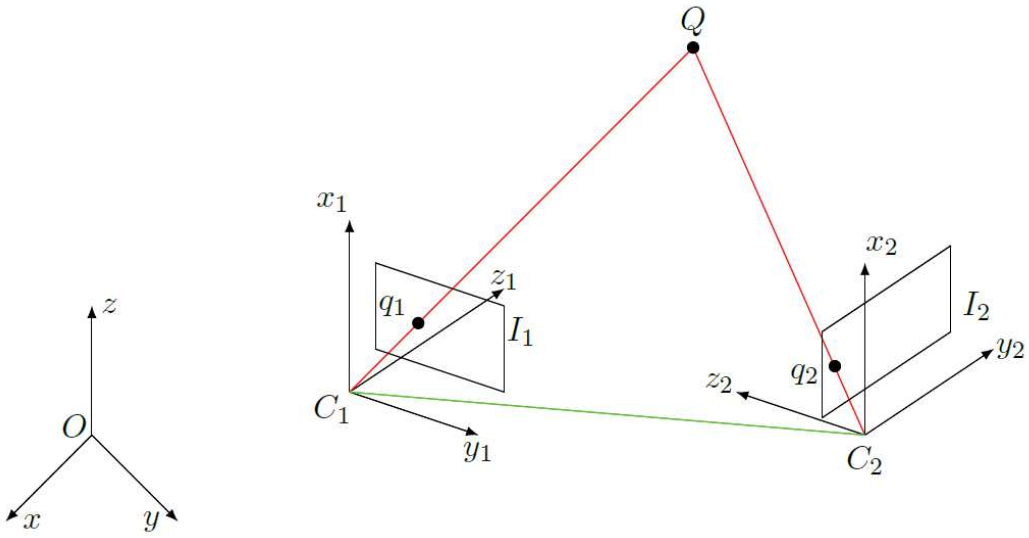


Figure 2.4: Two-view geometry

Given a point Q in the world reference frame system (x, y, z), let q_1 and q_2 be the projections of this point onto two different image planes, one with reference to the first camera system C_1 and the other to the second camera

system C_2. The goal of the study performed in the following sections is to find a relation between Q, q_1 and q_2.

2.2.1 Essential Matrix

The importance of the *essential matrix* is linked to the fact that it encodes the relative pose between the two cameras C_1 and C_2 with $(R, T) \in \mathbb{SE}(3)$. The derivation of this matrix comes from the following steps:

1. Camera correspondences Let Q_i be the point Q seen from frame $i = 0, 1, 2$. Then

$$\begin{cases} Q_1 = R_1 Q_0 + T_1 \\ Q_2 = R_2 Q_0 + T_2 \end{cases} \longrightarrow \begin{cases} Q_0 = R_1^T Q_1 - R_1^T T_1 \\ Q_2 = R_2(R_1^T Q_1 - R_1^T T_1) + T_2 \\ \quad = R_2 R_1^T Q_1 - R_2 R_1^T T_1 + T_2 \end{cases} \tag{2.10}$$

and by defining $\hat{R} = R_2 R_1^T$, $\hat{T} = -R_2 R_1^T T_1 + T_2$, the relation between Q_1 and Q_2 can be rewritten as

$$Q_2 = \hat{R} Q_1 + \hat{T} \tag{2.11}$$

2. Camera relations For each camera i, the camera relation is $\lambda_i q_i = K_i \Pi_0 Q_i$ with $i = 1, 2$ where q_i and Q_i are expressed in homogeneous coordinates and the camera matrix is $P_i = K_i \Pi_0$.

3. Putting it all together Under the assumption that $K_i = K_{f,i} = I_{3x3}$, consider

$$\begin{cases} \lambda_1 q_1 = Q_1 \\ \lambda_2 q_2 = Q_2 \\ Q_2 = \hat{R} Q_1 + \hat{T} \end{cases} \implies \lambda_2 q_2 = \hat{R} \lambda_1 q_1 + \hat{T} \tag{2.12}$$

4. Algebraic manipulation of the above equation Consider the skew-symmetric operator $[\hat{T}]_\times$ related to $\hat{T}$ and multiply both members by such quantity, this operation is equivalent to an outer product:

$$\hat{T} \times (\lambda_2 q_2) = \hat{T} \times (\hat{R} \lambda_1 q_1 + \hat{T}) \implies \lambda_2 [\hat{T}]_\times q_2 = \lambda_1 [\hat{T}]_\times \hat{R} q_1 \tag{2.13}$$

Then consider the inner scalar product of the equation by q_2:

$$q_2^T [\hat{T}]_\times \hat{R} q_1 \;=\; 0 \tag{2.14}$$

This result is the formulation of the *Longuet-Higgins Equation* and represents the relation satisfied by two images q_1, q_2 of the same point Q. This relation is also known as *epipolar constraint* and the matrix $E \doteq [\hat{T}]_\times \hat{R}$ is the *essential matrix*.

What is the geometric interpretation? From the geometric point of view the epipolar constraint is equivalent to saying that the vectors C_1Q, C_2Q, C_1C_2 are coplanar. A way to show this is by recalling that the triple product $a \cdot (b \times c)$ represents the volume of the solid with sides a, b, c, as depicted in Fig. 2.5. Specifically, the cross-product between b and c stands for the area of the base, while the projection of a on the vector orthogonal to the plane generated by b and c is the height. Then $a \cdot (b \times c) = 0$ means the volume is zero so the three vectors are coplanar. Taking inspiration from

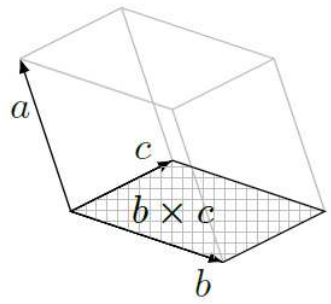


Figure 2.5: Example of the volume of a solid as $a \cdot (b \times c)$

this observation, it is possible to rewrite the Longuet-Higgins equation as a triple product

$$q_2^T [\hat{T}]_\times \hat{R} q_1 \;=\; q_2^T (\hat{T} \times \hat{R} q_1) \;=\; 0 \tag{2.15}$$

and so it follows the coplanarity of the vectors C_1Q, C_2Q, C_1C_2.

Characterization of Essential matrix Is E any $\mathbb{R}^{3x3}$ matrix? The answer is no: from its definition, it is worth noting that this matrix is given by the product of a skew-symmetric matrix and a rotation matrix. More into detail, it can be proved that a general matrix $E \in \mathbb{R}^{3x3}$ is an essential matrix if and

only if it admits a SVD decomposition of this kind:

$$E = U\Sigma V, \quad U, V \in \mathbb{SE}(3) \quad \Sigma = \begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix}, \sigma \in \mathbb{R}_+ \tag{2.16}$$

a complete proof of this fact can be found in [3].

2.2.2 Fundamental matrix and epipolar constraint

The essential matrix described before has been computed assuming a camera with no intrinsic parameters, hence $\lambda_i q_1 = \Pi_0 Q_i$ with $K_i = I$. In the general case of non normalized coordinates the relation becomes $\lambda_i \hat{q}_i = K_i \Pi_0 Q_i$, where K_i is the matrix of intrinsic parameters. By comparing these relations it results $\hat{q}_i = K_i q_i$, and hence $q_i = K_i^{-1} \hat{q}_i$.

By keeping in mind this fact, the epipolar constraint can be rewritten in this more general form:

$$q_2^T E q_1 = \hat{q_2}^T K_2^{-T} E K_1^{-1} \hat{q_1} = 0 \quad \Leftrightarrow \quad \hat{q_2}^T \underbrace{(K_2^{-T} E K_1^{-1})}_{F} \hat{q_1} = 0 \tag{2.17}$$

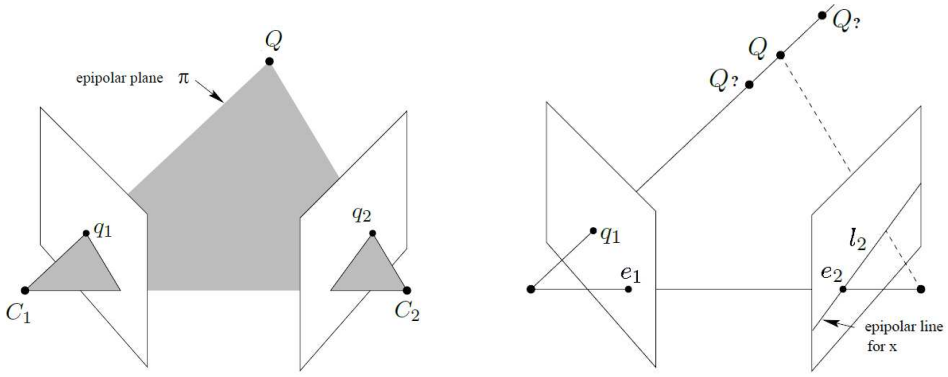
where F is known as *fundamental matrix*, and through this matrix it is possible to express the epipolar constraint for non-normalized coordinates.

2.2.3 Epipolar geometry

Indicating with the term baseline the line joining the two camera centers C_1 and C_2, the epipolar geometry could be define as the geometry of the intersection of the image planes with the pencil of planes having the baseline as axis. It is independent of scene structure, and only depends on the cameras' internal parameters and relative pose. This geometry is usually motivated by considering the search for corresponding points in stereo matching.

Considering again the projection of a 3D point Q in the two views, at q_1 in the first, and q_2 in the second. As previously proved the image points q_1 and q_2, space point Q and camera centres are coplanar, and, denoting

this plane as π, it is possible to deduce from Fig.2.6(a) that the rays back-projected from q_1 and q_2 intersect Q and the rays are coplanar, lying in π. This latter property is of most significance in searching for a correspondence.



(a) *Coplanarity of the image points q_1 and q_2, space point Q*

(b) *The corresponding point q_2, given the point q_1, lies on line l_2*

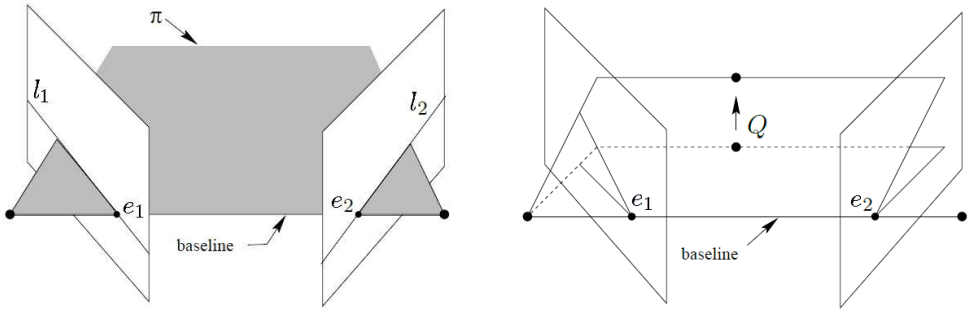
Figure 2.6: Epipolar geometry basic results

Supposing now that only the point q_1 is known, how the corresponding point q_2 is constrained? The plane π is determined by the baseline and the ray is defined by q_1. From above, the ray corresponding to the unknown point q_2 lies in π, hence the point q_2 lies on the line of intersection l_2 of π with the second image plane. This reasoning is graphically depicted in Fig. 2.6(b). In terms of a stereo correspondence algorithm the benefit is that the search for the point corresponding to q_1 need not to cover the entire image plane but can be restricted to the line l_2.

The geometric entities involved in epipolar geometry are illustrated in Fig. 2.7. The terminology is:

- The *epipole* is the point of intersection of the baseline with the image plane, or equivalently it is the image in one view of the camera centre of the other view. It is also the vanishing point of the baseline (translation) direction.
- An *epipolar plane* is a plane containing the baseline. There is a one-parameter family (a pencil) of epipolar planes.
- An *epipolar line* is the intersection of a epipolar plane with the image plane. All epipolar lines intersect at the epipole. An epipolar plane

intersects the left and right image planes in epipolar lines, and defines the correspondence between the lines.



(a) *Definition of epipoles e_1 e_2 and the epipolar lines l_1 l_2*

(b) *As the position of the 3D point Q varies, the epipolar planes rotate about the baseline*

Figure 2.7: Epipolar geometry entities

While considering the epipolar geometry, the most emblematic examples are the case of converging cameras, Fig. 2.8, and the case of motion parallel with image planes, Fig. 2.9.

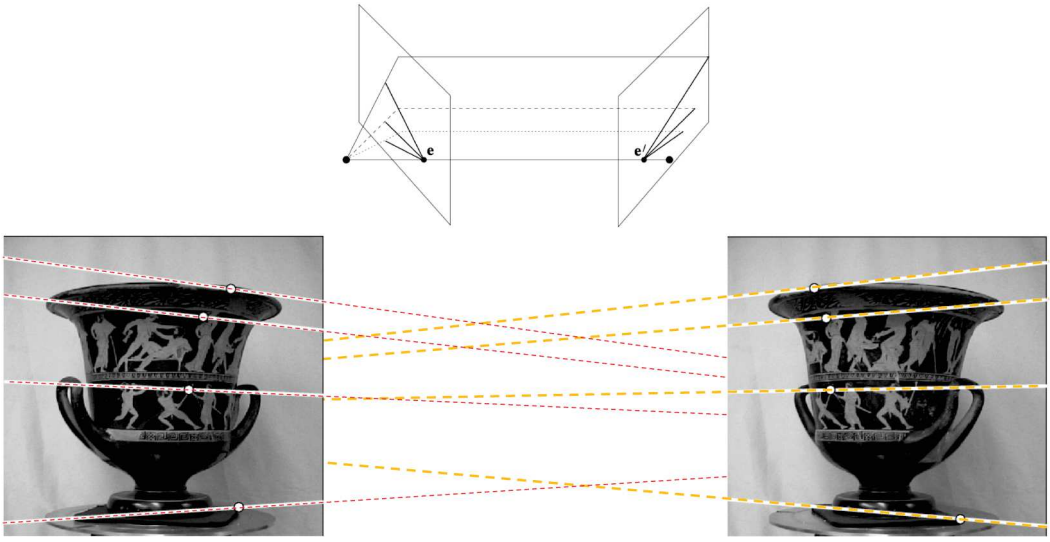


Figure 2.8: Epipolar geometry for converging cameras

From Fig. 2.8 a pair of images with superimposed corresponding points and their epipolar lines are depicted: the motion between the views is a translation and rotation. In each image, the direction of the other camera may be inferred from the intersection of the pencil of epipolar lines. In this case, both epipoles lie outside of the visible image.

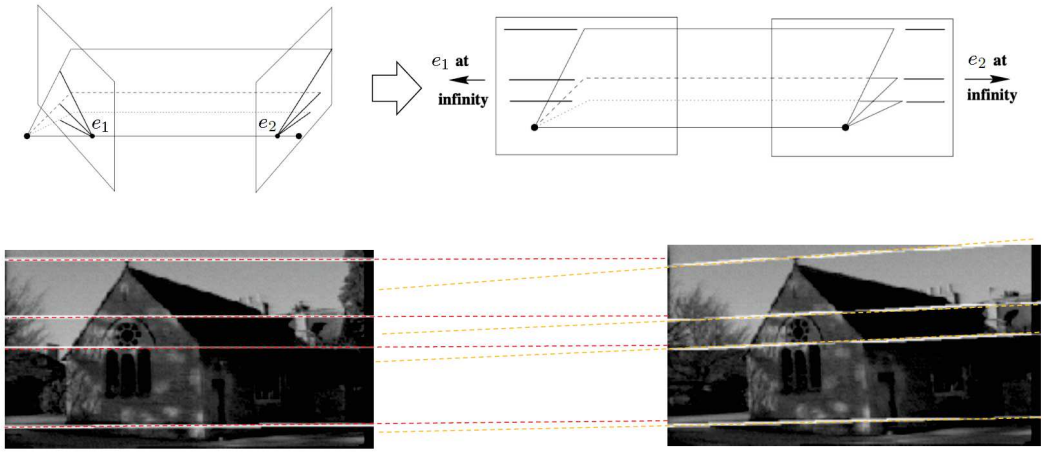


Figure 2.9: Epipolar geometry for motion parallel to the image plane

In the case of parallel cameras instead the intersection of the baseline with the image plane is at infinity. Consequently, the epipoles are at infinity and epipolar lines are parallel. In particular, what is depicted in Fig. 2.9 is a pair of images for which the motion between the views is approximately a translation parallel to the x-axis, with no rotation. Note that corresponding points lie on corresponding epipolar lines. This last example will be of fundamental importance in the next sections in order to deeply understand how rectification could improve the performance of a stereo rig, and then of a structure light 3D scanner.

2.3 Rectification

Typically in a stereo rig the cameras are horizontally displaced and rotated towards each other by an equal amount (verged) in order to overlap their fields of view. This is the camera configuration of Fig. 2.8: as already noted, epipolar lines in this case lie at a variety of angles across the two images, complicating the search for correspondences. In contrast, if these cameras had their principal axes parallel to each other (no vergence) and the two cameras had identical intrinsic parameters, corresponding epipolar lines would lie along the same horizontal scanline in each image, as observed in Fig. 2.9. This configuration is known as a standard rectilinear stereo rig. Clearly it is desirable to retain the improved stereo viewing volume associated with

verged cameras and yet have the simplicity of correspondence search associated with a rectilinear rig.

To achieve this a solution could be to warp or *rectify* the raw images associated with the verged system such that corresponding epipolar lines become collinear and lie on the same scanline. A second advantage is that the equations for 3D reconstruction become very simply related to image disparity after image rectification, since they correspond to those of a simple rectilinear stereo rig, this fact will be clear later.

Rectification can be achieved either with camera calibration information, for example in a typical stereo application, or without calibration information, for example in a typical structure from motion application. In the next subsections both the two cases will be discussed, with more focus on the calibrated case.

2.3.1 Rectification with calibration information

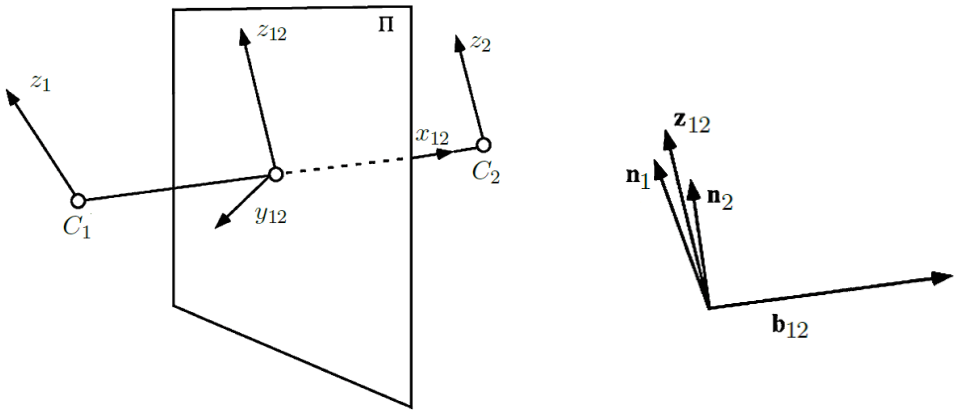
Given a calibrated stereo rig where both the intrinsic and the extrinsic parameters are known, the idea is to identify a common viewing direction for cameras 1 and 2. There exist many ways to achieve this goal. Here just a simple example is illustrated. In order to simplify the reasoning, it is assumed that the lens distortion has been calibrated before and hence does not need to be included anymore in the set of intrinsic parameters.

Common viewing direction for rectifying cameras 1 and 2 Let Π be a plane perpendicular to the baseline vector b_{12} from the projection center of camera 1 to the projection center of camera 2, as illustrated in Fig. 2.10(a). First of all project the unit vectors z_1 and z_2 of both optical axes into Π, which results in vectors n_1 and n_j, respectively. The algebraic relations are as follows:

$$n_1 \;=\; (b_{12} \times z_1) \times b_{12}, \quad n_2 \;=\; (b_{12} \times z_2) \times b_{12} \tag{2.18}$$

Aiming at a "balanced treatment" of both cameras, in order to find the unit vector of the common direction the bisector of n_1 and n_2 is considered:

$$z_{12} \;=\; \frac{n_1 + n_2}{\|n_1 + n_2\|_2} \tag{2.19}$$



(a) *Plane Π perpendicular to the baseline*

(b) *Vectors n_1 and n_2 as result of projection of vectors z_1 and z_2*

Figure 2.10: An illustration for calculating the common viewing direction for cameras 1 and 2

Consider the unit vector x_{12} in the same direction as b_{12}, and the unit vector y_{12} is finally defined by the constraint of ensuring a right-hand 3D Cartesian coordinate system. Formally, the result is the following:

$$x_{12} = \frac{b_{12}}{\|b_{12}\|_2} \quad y_{12} = x_{12} \times z_{12} \tag{2.20}$$

The two images of camera 1 and Camera 2 need to be modified as though both would have been taken in the direction $R_{12} = [x_{12}\, y_{12}\, z_{12}]^T$, instead of the actually used directions R_1 and R_2.

Getting rectification homographies The rotation matrices that rotate both cameras into their new virtual viewing direction are as follows:

$$R_i^* = R_{12}R_i^T \quad R_j^* = R_{12}R_j^T \tag{2.21}$$

In general, when rotating any camera around its projection center about the matrix R, the image is transformed by a rotation homography:

$$H = K \cdot R \cdot K^{-1} \tag{2.22}$$

where K is the 3x3 matrix of intrinsic parameters of this camera. The matrix K^{-1} transfers pixel coordinates into camera coordinates in world units, the

matrix R rotates them into the common plane, and the matrix K transfers them back into pixel coordinates.

Producing the rectified image pair A rectified image is calculated pixel by pixel using:

$$p = H^{-1}\hat{p} \tag{2.23}$$

such that the new value at pixel location $\hat{p}$ is calculated based on the original image values in a neighborhood of a point p, using for instance bilinear interpolation. This is an essential step since the rectified coordinates are in general not integers, so it is required to resample using some form of interpolation.

Note that, even with the same make and model of the camera, the focal length of the two cameras may be slightly different: if this happens it is necessary to scale one rectified image by the ratio of focal lengths in order to place them on the same focal plane.

2.3.2 Rectification without calibration information

When calibration information is not available, rectification can be achieved using an estimate of the fundamental matrix, which is computed from correspondences within the raw image data. To this aim, several methods can be found in the literature; an example is a method proposed by Hartley [10]. This method is implemented by the `cv::stereoRectifyUncalibrated()` function, that is the function that *OpenCV* uses to perform rectification without calibration data.

In addition, rectification can be performed also without relying on epipolar geometry. In this context, a prominent example is given by Nozick [19] that in his paper presents a strategy for multiple view image rectification: given some point correspondences between each view, the goal of this method is to find an homography matrix H_i for each camera such that the transformed point correspondences are horizontally aligned. Each homography is defined as $H_i = K_i' R_i K_i^{-1}$, where it is easy to deduce that K_i is the camera matrix for camera i, R_i is the rotation matrix imposing the desired orientation for the alignment, and K_i' is the new camera matrix for the rectified camera i. Given this setup, the idea is to perform a bundle adjustment on

K_i' and R_i from each view using Levenberg-Marquartd method. This means that these two matrices must be estimated trying to minimize as more as possible a sort of alignment error between rectified point correspondences, given some initial conditions on their values. From what it can be seen, this method is not related to epipolar geometry and hence can be extended for an arbitrary number of views. Of course, it is also very well suited for a two image rectification. Not only, it can also be extended in order to deal with different resolutions.

2.4 Finding correspondences in a stereo pair

Finding correspondences is an essential step for 3D reconstruction from multiple views. The correspondence problem can be viewed as a search problem, which asks, given a pixel in the left image, which is the corresponding pixel in the right image? As previously stated, the epipolar geometry constraint strongly simplify this problem by reducing the search space from a 2D search to the epipolar line only.

This fact leads to most of the methods for finding correspondences in image pairs. These assumptions hold when the distance of the world point from the cameras is much larger than the baseline: in this way most scene points are visible from both viewpoints and corresponding image regions are similar.

Two questions are involved: what is a suitable image element to match and what is a good similarity measure to adopt? There are two main classes of correspondence algorithms: correlation-based and feature-based methods, that will be briefly described in the following sections.

2.4.1 Correlation-Based methods

If the element to match is only a single image pixel, ambiguous matches exist. Therefore, windows are used for matching in correlation-based methods and the similarity criterion is a measure of the correlation between the two windows. The selected correspondence is given by the window that maximizes a similarity criterion or minimizes a dissimilarity criterion within a search range. Once a match is found, the offset between the two windows

can be computed, which is called the disparity from which the depth can be recovered. Some commonly used criteria for correlation-based methods are described next.

Based on the rectified images in Fig. 2.11, we define the window function where m, an odd integer, is the image window size so that:

$$W_m(x,y) = \{(u,v) \mid x - \frac{(m-1)}{2} \leq u \leq x + \frac{(m-1)}{2}, \\ y - \frac{(m-1)}{2} \leq v \leq y + \frac{(m-1)}{2}\} \tag{2.24}$$

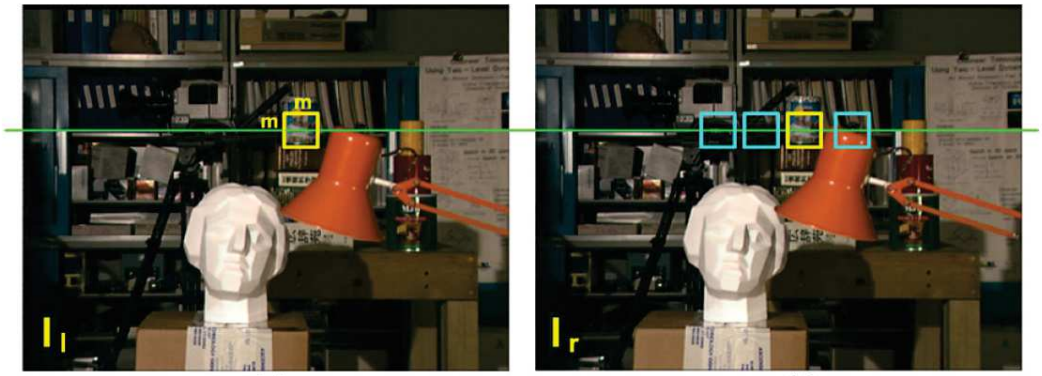


Figure 2.11: Correlation-based methods look for the matching image window between the left and right rectified images

The dissimilarity can be measured by the Sum of Squared Differences (SSD) cost for instance, which is the intensity difference as a function of disparity d:

$$SSD(x,y,d) = \sum_{(u,v)\in W_m(x,y)} \left[I_l(u,v) - I_r(u-d,v)\right]^2 \tag{2.25}$$

where I_l and I_r refer to the intensities of the left and right images respectively. If two image windows correspond to the same world object, the pixel values of the windows should be similar and hence the SSD value would be relatively small. As shown in Fig. 2.11, for each pixel in the left image, correlation-based methods would compare the SSD measure for pixels within a search range along the corresponding epipolar line in the right image. The disparity value that gives the lowest SSD value indicates the best match.

A slight variation of SSD is the Sum of Absolute Differences (SAD) where the absolute values of the differences are added instead of the squared values:

$$SAD(x,y,d) = \sum_{(u,v)\in W_m(x,y)} |I_l(u,v) - I_r(u-d,v)| \tag{2.26}$$

This cost measure is less computationally expensive as it avoids the multiplication operation required for SSD. On the other hand, the SSD cost function penalizes the large intensity difference more due to the squaring operation.

The intensities between the two image windows may vary due to illumination changes and non-Lambertian reflection. Even if the two images are captured at the same time by two cameras with identical models, non-Lambertian reflection and differences in the gain and sensitivity can cause variations in the intensity. In these cases, SSD or SAD may not give a low value even for the correct matches. For these reasons, it is a good idea to normalize the pixels in each window. A first level of normalization would be to ensure that the intensities in each window are zero-mean. A second level of normalization would be to scale the zero-mean intensities so that they either have the same range or, preferably, unit variance. This can be achieved by dividing each pixel intensity by the standard deviation of window pixel intensities, after the zero mean operation, i.e. normalized pixel intensities are given as:

$$I_n = \frac{I - \bar{I}}{\sigma_I} \tag{2.27}$$

where $\bar{I}$ is the mean intensity and σ_I is the standard deviation of window intensities.

While SSD measures the dissimilarity, Normalized Cross-Correlation (NCC) measures the similarity. Again, the pixel values in the image window are normalized first by subtracting the average intensity of the window so that only the relative variation would be correlated. The NCC measure is computed as follows:

$$NCC(x,y,d) = \frac{\sum_{(u,v)\in W_m(x,y)} (I_l(u,v) - \bar{I}_l)(I_r(u-d,v) - \bar{I}_r)}{\sqrt{\sum_{(u,v)\in W_m(x,y)} (I_l(u,v) - \bar{I}_l)^2 (I_r(u-d,v) - \bar{I}_r)^2}} \tag{2.28}$$

where

$$\bar{I}_l = \frac{1}{m^2} \sum_{(u,v)\in W_m(x,y)} I_l(u,v), \quad \bar{I}_r = \frac{1}{m^2} \sum_{(u,v)\in W_m(x,y)} I_r(u,v) \tag{2.29}$$

2.4.2 Feature-Based methods

Rather than matching each pixel, feature-based methods only search for correspondences to a sparse set of features, such as those located by a repeatable, well-localized interest point detector (e.g. a corner detector). Apart from locating the features, feature extraction algorithms also compute some sort of feature descriptors for their representation, which can be used for the similarity criterion. The correct correspondence is given by the most similar feature pair, the one with the minimum distance between the feature descriptors.

Stable features are preferred in feature-based methods to facilitate matching between images. Typical examples of image features are edge points, lines and corners. In particular, what is preferred in this kind of application are point-based features. Some examples of point-based features detectors developed in recent years are the Scale Invariant Feature Transform (SIFT) and the Speeded-Up Robust Feature (SURF). The SIFT feature is described by a local image vector with 128 elements, which is invariant to image translation, scaling, rotation and partially invariant to illumination changes and affine or 3D projections. Fig. 2.12 shows an example of matching SIFT fea-

Figure 2.12: Wide baseline matching between two images with SIFT

tures across large baseline and viewpoint variation. It can be seen that most

matches are correct, thanks to the invariance and discriminative nature of SIFT features.

2.5 3D reconstruction

Stereo vision refers to the ability to infer information on the 3D structure and distance of a scene from two or more images taken from different viewpoints. The disparities of all the image points form the disparity map, which can be displayed as an image. If the stereo system is calibrated, the disparity map can be converted to a 3D point cloud representing the scene.

2.5.1 Triangulation

When the corresponding left and right image points are known, two rays from the camera center through the left and right image points can be back-projected. The two rays and the stereo baseline lie on a plane (the epipolar plane) and form a triangle, hence the reconstruction is termed *triangulation*. Here triangulation for two rectified views is described.

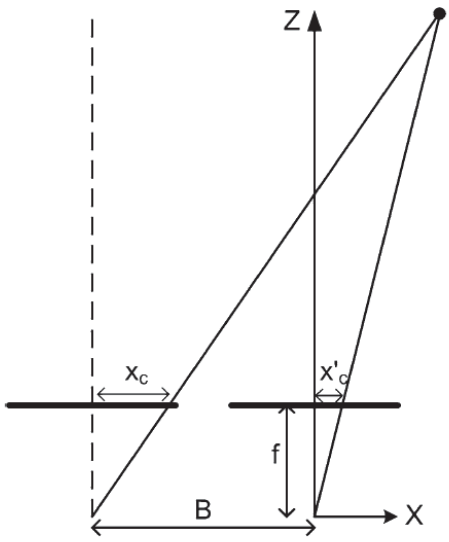


Figure 2.13: Stereo geometry after rectification

After image rectification, the stereo geometry becomes quite simple as shown in Fig. 2.13, which shows the top-down view of a stereo system composed of two pinhole cameras. The necessary parameters, such as baseline

and focal length, are obtained from the original stereo calibration. The following two equations can be obtained based on the geometry:

$$x'_c = f\frac{X}{Z}, \quad x_c = f\frac{X+B}{Z} \tag{2.30}$$

where x'_c and x_c are the corresponding horizontal image coordinates in metric units in the right and left images respectively, f is the focal length and B is the baseline distance.

Disparity d is defined as the difference in horizontal image coordinates between the corresponding left and right image points, given by:

$$d = x_c - x'_c = \frac{fB}{Z} \tag{2.31}$$

Therefore,

$$Z = \frac{fB}{d}, \quad X = \frac{Zx'_c}{f}, \quad Y = \frac{Zy'_c}{f} \tag{2.32}$$

where y'_c is the vertical image coordinates in the right image.

This shows that the 3D world point can be computed once the disparity is available: $(x'_c, y'_c, d) \mapsto (X, Y, Z)$. Disparity maps can be converted into depth maps using these equations to generate a 3D point cloud.

2.5.2 Uncertainty in 3D reconstruction

Stereo matches are found by seeking the minimum of some cost functions across the disparity search range. This computes a set of disparity estimates in some discretized space, typically integer disparities, which may not be accurate enough for 3D recovery. 3D reconstruction using such quantized disparity maps leads to many thin layers of the scene. Interpolation can be applied to obtain sub-pixel disparity accuracy, such as fitting a curve to the SSD values for the neighboring pixels to find the peak of the curve, which provides more accurate 3D world coordinates.

By taking the derivatives of Eq. 2.33, the standard deviation of depth is given by:

$$\Delta Z = \frac{Z^2}{Bf}\Delta d \tag{2.33}$$

where Δd is the standard deviation of the disparity. This equation shows that the depth uncertainty increases quadratically with depth. Therefore, stereo systems typically are operated within a limited range. If the object is far away, the depth estimation becomes more uncertain. The depth error can be reduced by increasing the baseline, focal length or image resolution. However, each of these has detrimental effects. For example, increasing the baseline makes matching harder and causes viewed objects to self-occlude, increasing the focal length reduces the depth of field, and increasing image resolution increases processing time and data bandwidth requirements. Thus, we can see that the design of stereo cameras typically involves a range of performance trade-offs, where trade-offs are selected according to the application requirements.

Chapter 3

Structured light sensors: principles and calibration

This chapter introduces the operating principles of several structured light sensors and how they are classified according to different coding strategies. Calibration strategies are also discussed since they play critical roles in achieving the required accuracy. Part of the chapter is also dedicated to describing the particular prototype used in this project both from the hardware and the software point of view.

3.1 Structured light sensors: an overview

The concept of *structured light imaging* is quite simple: a known pattern is projected onto a surface. When the camera views the pattern from one (or more) different points of view, the surface shape of the target distorts the pattern, as shown in Fig. 3.1. The direction and size of the pattern distortions are used to reconstruct the surface topography of the target object.

From this preliminary information, it can be deduced that structured light cameras employ the same operating principle as stereo cameras with a clear difference: one of the two cameras is replaced with a light projector that illuminates the scene with a textured visual pattern. The light projector can be seen as a virtual camera that always "sees" the same, fixed image: its projected pattern. The pattern is seen also by the camera but in this case, due to the baseline between the projector and the camera, it is projected in different 2D points of the imaging sensor, depending on the 3D structure of the

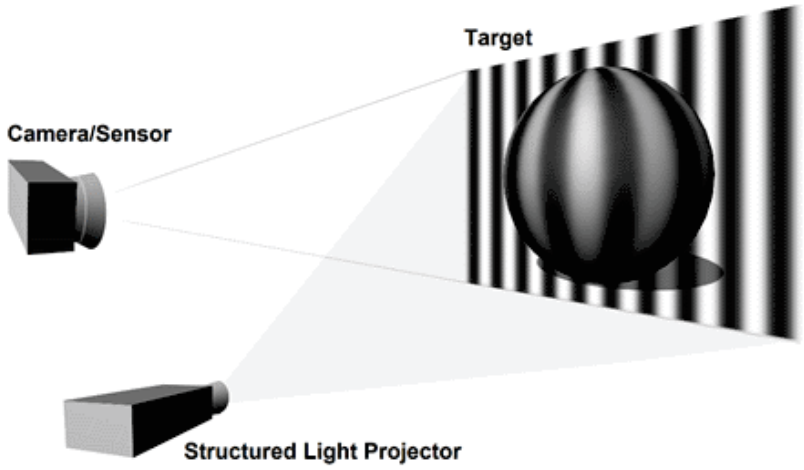


Figure 3.1: Example of structured light imaging: a regular striped pattern is projected onto the ball

framed scene. It is obvious that many principles of stereo vision, introduced in Chapter 2, can be directly applied also in this slightly different scenario.

In general, it is possible to distinguish two different projection methods:

- *projection of 2D images* by using video-projectors. These devices project a light pattern on the measuring scene and since the pattern is coded, correspondences between image points and points of the projected pattern can be easily found.
- *projection of narrow lines* using laser technology. The main advantage of laser light is its limitless depth of field that allows to project a very narrow line on a certain surface at every distance. On the other hand, illuminating only a single stripe, it is necessary to mechanically move the laser beam along the entire surface to be reconstructed. This implies long acquisition times and also the reconstructing object to be stable during the operation.

Structured light sensors based on the projections of 2D images could be implemented using numerous techniques. In the general case, the idea is to project onto the measuring scene a pattern or a set of patterns designed in such a way that unique identifiers (or codes) can be assigned to a set of pixels. Every coded pixel has its own identifier, so there is a direct mapping

from the identifiers to the corresponding coordinates of the pixel in the pattern. The identifiers are simply numbers, which are mapped in the pattern by using grey levels, color or geometrical representations. The greater the number of points that need to be coded, the greater the number of codes and, therefore, the mapping of those codes to a pattern is more difficult.

The available strategies used to represent such codes could be firstly classified into sequential (multiple-shot) or single-shot categories. In the following sections, an overview of both these categories is provided focusing more on the sequential coding, being the strategy that has been adopted for the project. A more detailed description of these coding methods can be found in [8].

3.1.1 Sequential projection techniques

These techniques consist in projecting a set of patterns over time on the measuring surface and, by sequentially capturing them, it is possible to compute the code for each pixel. The coding strategies that are described below allows to encode only one axis, and so it is not possible to identify a unique pixel coordinate, but only group of pixels belonging to the same row or column. In order to identify single pixels it would be necessary to consider two sequence of patterns: one for the vertical code, one for the horizontal code. Actually, it will be clear in Chapter 4 how, given for instance the vertical code, it is possible with rectification to retrieve the horizontal one without projecting another sequence of patterns but simply by using epipolar geometry. Therefore, the following paragraphs will refer only to vertical codes for simplicity, an analog reason could be done for horizontal ones.

Binary patterns The binary coding technique exploits black and white stripes to form a sequence of projection patterns, such that each point on the surface of the object possesses a unique binary code that differs from any other codes of different points. In general, N patterns can code 2^N stripes. A simplified example for $N = 5$ is depicted in Fig. 3.2(a). This technique is very reliable and less sensitive to surface characteristics since only binary values exist in all pixels. However, to achieve high spatial resolution, a large number of

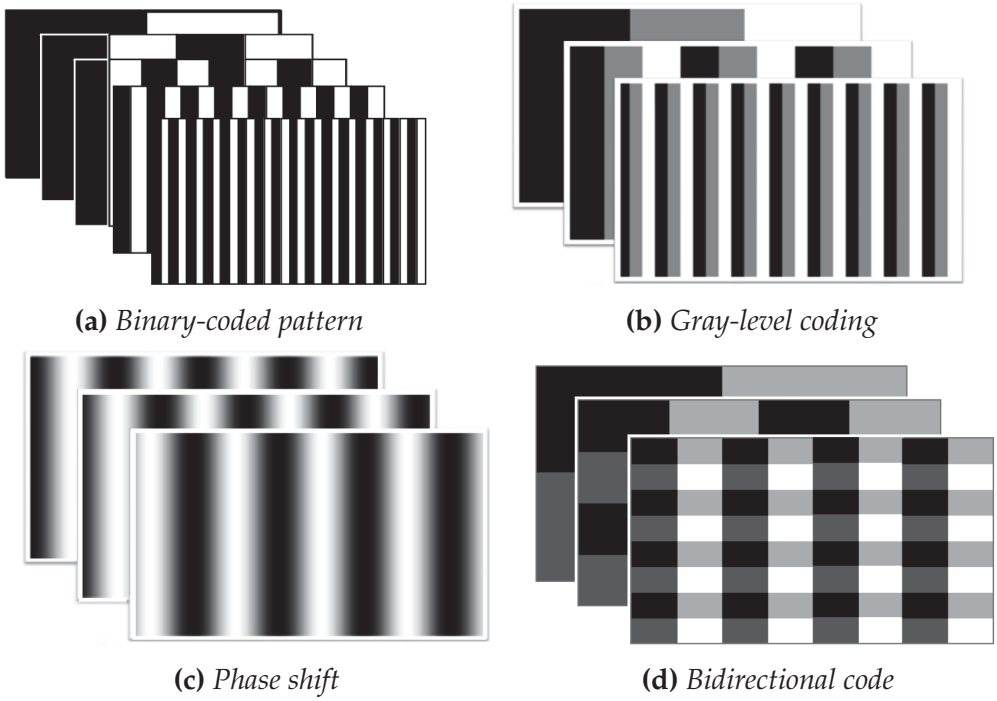
sequential patterns need to be projected. All objects in the scene have to remain static. The entire duration of 3D image acquisition may be longer than a practical 3D application allows for.

Gray-level patterns To effectively reduce the number of patterns needed to obtain a high-resolution 3D image, gray-level patterns are developed. For example, one can use M distinct levels of intensity (instead of only two in the binary code) to produce unique coding of the projection patterns. In this case, N patterns can code M^N stripes. Each stripe code can be visualized as a point in an N-based space, and each dimension has M distinct values. In Fig. 3.2(b) an example is shown for $N = M = 3$.

Phase shift Phase shift is a well-known fringe projection method for 3D surface imaging. A set of sinusoidal patterns is projected onto the object surface, where each sinusoidal pattern is shifted in phase with respect to the preceding one by a constant value which depends on the number of sequential patterns, see Fig. 3.2(c) for a simple example with three projection patterns. Phase unwrapping is the name of the decoding process: for each pixel coordinate one can extract the intensity values from all the acquired images, and from these data, it is possible to retrieve the absolute phase of that pixel coordinate and then also its own column.

The coding strategies based on binary and gray level patterns are also known as *time-multiplexing* strategies because the bits of codes are multiplexed in time. As already seen, these strategies allow to identify the columns of pixels, but if more than two color intensities are available, fractal structures can be used, as the one in Fig. 3.2(d).

If the target 3D object is static and the application does not impose strong constraints on the acquisition time, sequential projection techniques can be used and may often result in more reliable and accurate results. However, if the target is moving, single-shot techniques have to be used to acquire a snapshot 3D surface image of the 3D object at a particular time instant.



(a) *Binary-coded pattern* **(b)** *Gray-level coding*

(c) *Phase shift* **(d)** *Bidirectional code*

Figure 3.2: Examples of different sequential techniques

3.1.2 Stripe indexing (single shot)

Stripe indexing is necessary to achieve robust 3D surface reconstruction because the order in which the stripes are observed is not necessarily the same as the order in which the stripes are projected. This is due to the inherent parallax existing in triangulation-based 3D surface imaging systems and the possibility of stripes missing from the acquired image because of occlusion of the object's 3D surface features. A few representative stripe indexing techniques are presented here.

Stripe indexing using colors Color image sensors usually have three independent acquisition channels, each corresponding to a spectrum band. The linear combination of the values of these color components can produce an infinite number of colors. Three 8-bit channels can represent 2^{24} different colors. Such rich color information can be used to enhance 3D imaging accuracy and to reduce acquisition time. This type of color-coded system can achieve real-time 3D surface imaging capability. It is also possible to encode multiple patterns into a single color projection image, each pattern possessing a unique color value in the color space. To reduce the decoding error

rate, one can select a color set in which each color has a maximum distance from any other color in the set. The maximal number of colors in the set is limited to the distance between colors that generate minimal cross talk in the acquired images.

Stripe indexing using segment pattern To distinguish one stripe from others, one can add some unique segment patterns to each stripe, as in Fig. 3.3(a), such that, when performing 3D reconstruction, the algorithm can use the unique segment pattern of each stripe to distinguish them. This indexing method is intriguing and clever, but it only applies to a 3D object with a smooth and continuous surface when the pattern distortion due to surface shape is not severe. Otherwise, it may be very difficult to recover the unique segment pattern, owing to deformation of the pattern and/or discontinuity of the object surface.

Stripe indexing using repeated gray-scale pattern If more than two intensity levels are used, it is possible to arrange the intensity levels of stripes such that any group of stripes (a sliding window of N stripes) has unique intensity pattern within a period of length. For example, if three gray levels are used (black **B**, gray **G** and white **W**), a pattern can be designed as **BWGWBGWGBGWBGBWBGW** which is depicted in Fig. 3.3(b). The pattern matching process starts with a correlation of acquired image intensity with projected intensity pattern. Once a match is located, a further search is perform on a sub-gray-level-sequence match, such as three-letter sequences **WGB**, **GWB**, etc.

Stripe indexing based on De Bruijn sequence A De Bruijn sequence of rank n on an alphabet of size k is a cyclic word in which each of the k^n words of length n appears exactly once as we travel around the cycle. A simple example of a De Bruijn circle with $n = 3$ and $k = 2$ (the alphabet is $\{0, 1\}$) is shown in Fig. 3.4(a). Travelling around the cycle (either clockwise or counterclockwise), each of the $2^3 = 8$ three-digit patterns 000, 001, 010, 011,100, 101, 110, 111 is encountered exactly once. There is no repeated three-digit pattern in the sequence. In other word, no subsequence is correlated to any other in the De Bruijn sequence. This unique feature of the De Bruijn sequence can be used in constructing a stripe pattern sequence

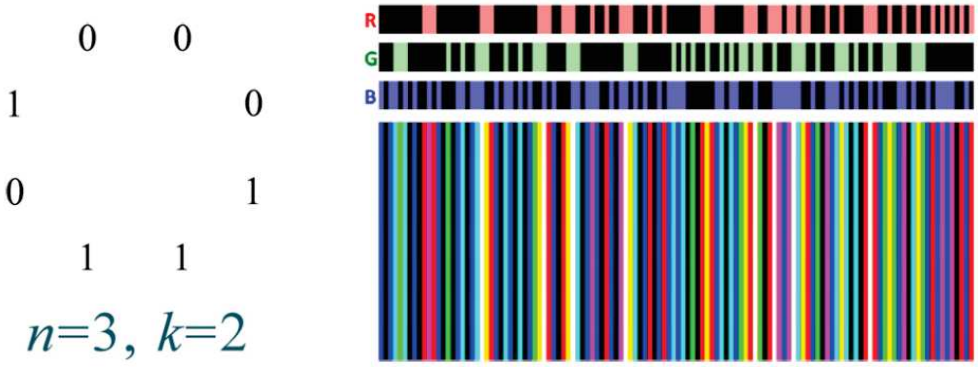
(a) *Stripe indexing using segment pattern*

(b) *Stripe indexing using repeated gray-scale pattern*

Figure 3.3: Examples of different stripe indexing techniques

that has unique local variation patterns that do not repeat themselves. Such uniqueness makes the pattern decoding an easier task. In Fig. 3.4(b) it is possible to see an example of using binary combinations of (R, G, B) colors to produce a color-indexed stripe based on De Bruijn sequence. The maximum number of combinations of three colors is $2^3 = 8$. In this example some constraints are taken into account:

- the combination (0,0,0) is avoided and the total number of stripes is reduced, in particular $k = 5$ and $n = 3$.
- all neighboring stripes must have different colors. Otherwise, some stripes with double or triple width would occur, confusing the 3D reconstruction algorithms.



(a) *Simple example of De Bruijn sequence*

(b) *Example of color stripe indexing based on De Bruijn sequence* $(k = 5, n = 3)$

Figure 3.4: De Bruijn sequences

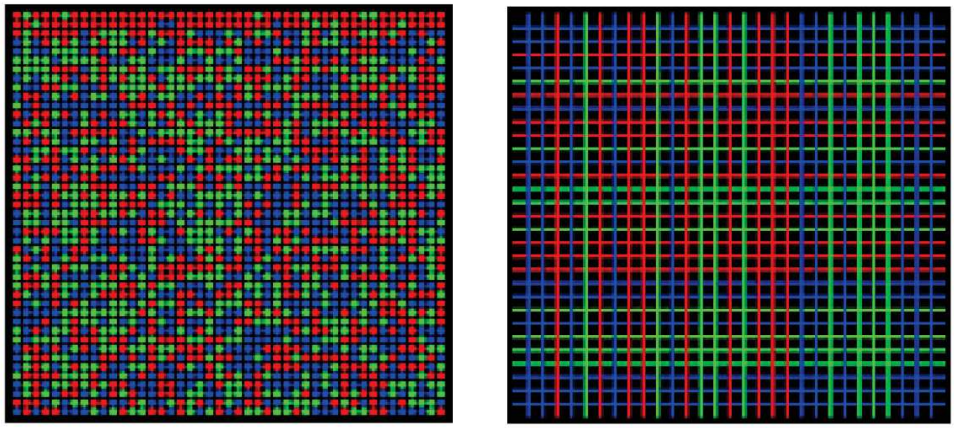
3.1.3 Grid indexing: 2D spatial grid patterns (single shot)

The basic concept of 2D grid pattern techniques is to uniquely label every subwindow in the projected 2D pattern, such that the pattern in any subwindow is unique and fully identifiable with respect to its 2D position in the pattern. Here some of the most popular patterns belonging to this category.

Pseudo-random binary array (PRBA) One grid indexing strategy is to use a pseudo-random binary array (PRBA) to produce grid locations that can be marked by dots or other patterns, such that the coded pattern of any subwindow is unique.

2D array of color-coded dots This approach extends the previous binary case allowing to use lower dimension subwindows for identifying a point position. Generating a matrix that preserves the uniqueness of subwindows could be done by brute force algorithms or dynamic programming approaches.

Other methods Such as matrices composed by geometrical sub-patterns used as key words, patterns composed by vertical and horizontal lines of different colors or matrices of periodic color dots as proposed in [25].



(a) *Pseudo-random color dots coding*

(b) *Pattern composed by vertical and horizontal lines of different colors*

Figure 3.5: Different coding techniques based on matrices

3.1.4 Performance evaluation

Aiming to accurately analyze the experimental results that has been obtained in this thesis it is essential having some background knowledge on the performance evaluation for a general 3D vision system. There are many factors that characterize technical performance of a 3D surface imaging system. From application point of view, the following three aspects are often used as the primary performance indexes to be used to evaluate 3D imaging systems:

Accuracy Measurement accuracy denotes the maximum deviation of the measurement value obtained by a 3D surface imaging system from the ground truth of the actual dimension of the 3D object. Quite often, different manufacturers may use different ways to characterize accuracy such as average (mean) error, uncertainty, $\pm$error, RMS, or other statistical values.

Resolution In the literature, 3D image resolution denotes the smallest portion of the object surface that a 3D imaging system can resolve. However, in the 3D imaging community, the term "image resolution" sometimes also denotes the maximum number of measurement points a system is able to obtain in single frame.

Speed Acquisition speed is important for imaging moving objects (such as the human body). For single-shot 3D imaging systems, the frame rate represents their ability to repeat the full-frame acquisition in a short time interval. For sequential 3D imaging systems (e.g., laser scanning systems), in addition to the frame rate, there is another issue that needs to be considered: the object is moving while sequential acquisition is performed; therefore, the obtained full-frame 3D image may not represent a snapshot of the 3D object at a single location. Instead, it becomes an integration of measurement points acquired in different time instances; therefore the 3D shape may be distorted from the original shape of the 3D object. There is another distinction, between acquisition speed and the computation speed. For example, some systems are able to acquire 3D images at 30 frames/s, but these acquired images need to be postprocessed at a much slower frame rate to generate 3D data.

The above-mentioned three key performance indexes can be used to compare 3D imaging systems. As already say in the introductory part, it will be shown in this master thesis project how both the acquisition and computation times can be reduced thanks to an accurate rectification process. Of course, this will result in a less accurate 3D reconstruction.

3.2 The structured light sensor used for this thesis

This section aims to describe the specific prototype of structured light 3D scanner that has been used during this master thesis project. Knowing the hardware components characteristics of this prototype is of fundamental importance in order to understand and analyze the final results that has been obtained from a critical point of view.

3.2.1 Hardware components

The functionalities of this prototype are based on an industrial camera for acquiring images, a DLP display module for projecting light patterns on the scene and an open-source development platform based on ARM processor, used in order to communicate with the projector. Here a more detailed description of these components is provided.

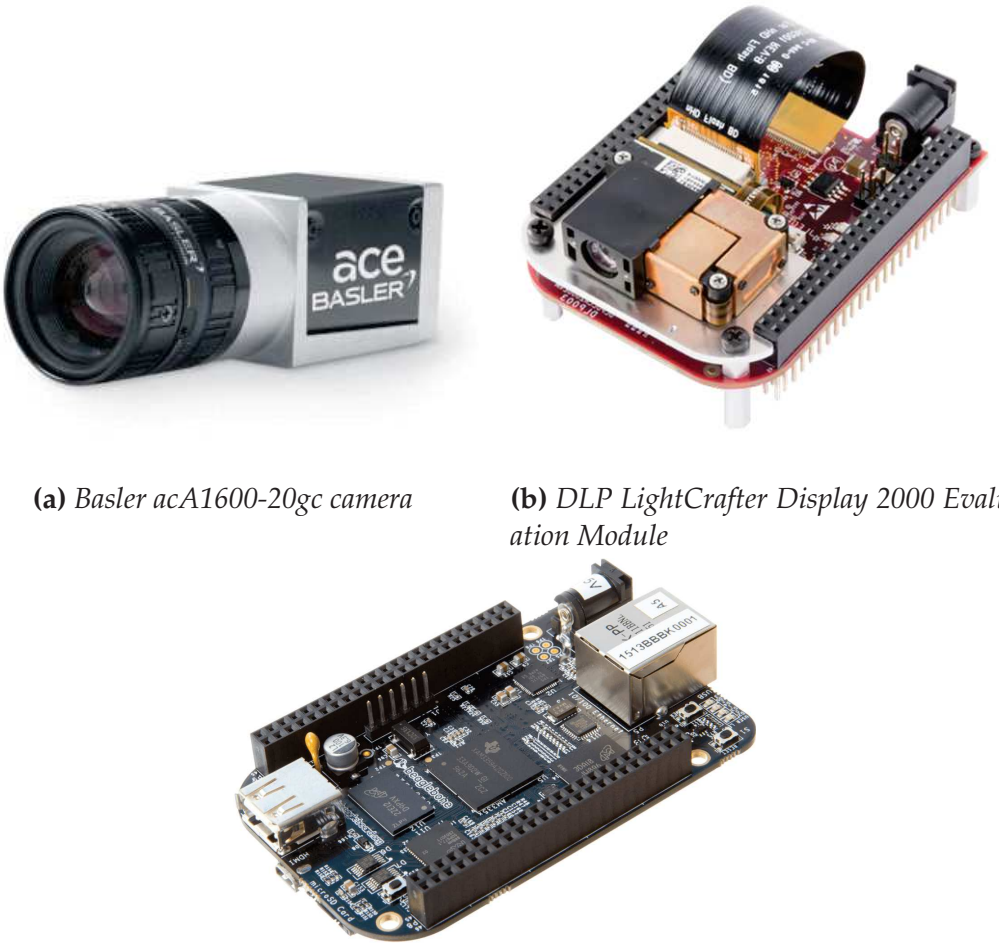
Basler acA1600-20gc camera The acA1600-20gc is built on a Sony progressive scan color CCD having 1628 x 1236 pixel resolution. It delivers up to 20 frames per second at full resolution. Pixel data can be output in 8 or 12 bit depth. Because this camera uses the same 29x29 mm footprint that has been standard on analog cameras for many years, replacement of analog cameras is easy. Moreover, by using a Power over Ethernet ("PoE") configuration, a single cable may be used to apply camera power and to transfer data between this camera and a PC, keeping cable runs to a minimum. This kind of camera provides a full set of features to address a wide range of applications such as the possibility to adjust the camera's black level, gain, area of interest, input debounce, and trigger delay. It features automatic exposure control, pixel binning, horizontal image mirroring, event reporting, sending of

test images, and a programmable lookup table. Regarding the optics, Computar lens M1614-MP2 C-mount has been used, with a fixed focal length of 16 mm, an aperture range from F1.4 - F16 and a manual aperture.

DLP LightCrafter Display 2000 EVM The e DLP LightCrafter Display 2000 EVM is a compact, plug-and-play, low cost platform enabling the use of DLP technology with embedded host processors, such as Raspberry Pi and BeagleBone Black. This evaluation module has a production-ready optical engine and processor interface supporting an 8/16/24-bit RGB parallel video interface in a small form factor. The evaluation module features the DLP2000 chipset produced by Texas Instruments, comprised of the DLP2000 (0.2 nHD) digital micromirror device (DMD), the DLPC2607 display controller, and the DLPA1000 PMIC/LED driver. This EVM comes equipped with a production-ready optical engine and processor interface supporting 8/16/24-bit RGB parallel video interface in a small form-factor. Despite the reduced cost, performances offered by this device are really good: 640x360 pixel resolution (nHD), high-contrast images and an optical engine that supports up to 30 lumen. This evaluation module covers a wide array of ultra-mobile and ultra-portable display applications in consumer, wearables, industrial, medical, and Internet of Things (IoT) markets.

BeagleBone Black BeagleBone Black is a low-cost, community-supported development platform for developers and hobbyists. This board is based on an economic processor such as the Sitara AM335x Cortex-A8 from by Texas Instruments, and is equipped with a 512MB DDR3L RAM memory and a 4GB eMMC flash memory. The BeagleBone Black is also populated with a single microSD connector to act as a secondary boot source for the board and, if selected as such, can be the primary boot source. Regarding the connectivity, one can find an ethernet and USB interfaces, the last one can be used both to provide power supply and to communicate with an external PC. The BeagleBone offers also the possibility to add cape plug-in boards in order to extend the functionalities; an example the DLP2000 projector described above, that can be connected on the board in a very easy way. From what it can be seen, the BeagleBone black is indeed a good solution for simple embedded applications: it offers a simple interface to lots of robotic motors,

sensors and both 2D and 3D cameras, providing also the possibility to execute PCL, OpenCV, OpenNI and many other software for image processing. Regarding in particular this prototype of 3D scanner, this board represents a convenient solution for its rendering 3D data processing capabilities.



(a) *Basler acA1600-20gc camera*

(b) *DLP LightCrafter Display 2000 Evaluation Module*

(c) *BeagleBone Black*

Figure 3.6: Hardware components

3.2.2 Building the prototype

The prototype of structured light 3D scanner, built starting from the hardware components previously described, is depicted in Fig. 3.7 from different point of views. The camera and the projector have been rigidly fixed in a 3D printed support and oriented in such a way that the area illuminated by the projector can be completely framed by the camera. By using the same terminology seen in Chapter 2 for stereo vision, this could be define

Figure 3.7: Prototype of structured light 3D scanner

as a slightly verged configuration. In particular, the vergence between camera and projector can be effectively appreciated by the bottom-right image, framing the prototype from top-view. The overall structure presents a footprint of roughly 105x105 mm, with the projector-camera system placed at a height of approximately 120 mm from the base. A more detailed description about how the camera is placed with respect to the projector can be obtained only after having calibrate the system.

Regarding the performances, this prototype is able to produce as output an accurate 3D reconstruction of objects placed at a distance from 200 mm to 800 mm from the sensor. The lower bound comes from different reasons:

this camera is not able to focus too short distances; the chosen orientation between camera and projector does not allow to the camera to completely frame the projector image; the pattern projected on these too close surfaces appears very small leading as consequence to loose many details in the reconstruction. The upper bound instead comes from the fact that placing the sensor too far from the acquired object makes the accuracy of the reconstruction really small, and this is due to the low resolution of the projector. The focus distance of the scanner can be manually adjusted acting on the lens system of the camera and the projector, however it must be kept in mind that, after any adjustment, it is necessary to re-calibrate the overall system.

At this point, it is worth spending some words discussing how the hardware components are interconnected and communicates among them. Basically, the functionalities of the scanner are the result of two processes running in parallel: one on the beaglebone, managing all the tasks related to the patterns generation and their projection; and the other on an external computer, by which it is possible to communicate with the camera, calibrate the projector-camera system, reconstruct and visualize the target objects on the scene. It is obvious that in order to make this solution correctly works it is essential the communication and synchronization between beaglebone and computer. The board is connected to the computer by USB interface, creating a virtual LAN network, and the two processes running in parallel interacts thanks to a socket-based communication, according to a client-server infrastructure:

- the *server*, running on the beaglebone, starts automatically once the device is turned on, listening for possible requests from the client
- the *client*, running on the computer, sends a requests to the server each time the projector is needed

The two processes communicates by exchanging some default commands as strings of bytes. The synchronization between computer and beaglebone is achieved by using blocking requests, meaning that the client, once sending the request, wait for a response from the server and in the meantime it does not perform any operation. The server, after having received and analyze the string of bytes from the client, performs the required task and then, if the operation succeed, the server communicates this fact to the client, otherwise

it still sends a response to the client, in particular a sort of error message. Only after receiving a response from the server, the client can come back to its execution.

A note about hardware interconnection Reconsidering again the three hardware components of the scanner, and in particular their connectivity, one may wonder why not implementing an on-board solution in which all the processes are executed by the beaglebone. This might be possible since both camera and beaglebone presents an ethernet interface, and surely this would be a simpler solution. However, the problem arises from a bug in the ethernet interface of the beaglebone that, many times, is not able to recognize the camera, making not possible a communication with that device. Nonetheless, the socket-based solution still presents some advantages with respect to this simpler solution, first of all the fact that all the algorithms related to calibration and triangulation are executed by the computer and this makes the elaboration part faster given the higher computational power. Otherwise, in the beaglebone such kind of operations would have required some optimization strategies in the memory access, due to the quite limited memory of the board.

3.3 Calibration of a projector-camera system

As introduced in Chapter 2, knowing the geometric characteristics of the cameras and the transformation that relates them is an essential requirement for a stereovision system in order to correctly perform the triangulation and hence the 3D reconstruction. Intrinsic and stereo calibration are the two processes that allow to find out all these geometric parameters. These calibrations are essential also for projector-camera pairs, being their principle of work very similar to stereo systems, with the advantage that a properly chosen projected pattern simplifies the task of finding point correspondences. In such systems, projectors are modeled as inverse cameras and all considerations known for passive stereo systems may be applied with almost no change. However, the calibration procedure must be adapted to the fact that projectors cannot directly measure the pixel coordinates of 3D points projected onto the projector image plane as cameras do. This means that in general, the projector calibration requires the use of an external camera

for acquiring the illuminated scene, an information that the projector is not able to perceive. Then, once the correspondences between 3D points in the world frame and 2D points in the projector plane are known, the intrinsic parameters of the projectors can be easily found as cameras.

This section assumes that the reader is familiar with the basic calibration procedure for cameras, such as the one presented by Zhang in [35], and the basic calibration procedure for stereo pairs. Moreover, a pair of novel methods for the calibration of projector-camera systems will be presented together with the particular calibration procedure adopted with the prototype.

3.3.1 Calibration by patterns projection

The calibration method proposed by Daniel Moreno and Gabriel Taubin [18] simply tries to estimate the coordinates of the calibration points in the projector image plane using local homographies. First, a dense set of correspondences between the projector and camera pixels is found by projecting onto the calibration object a pattern sequence identical to the one used to perform 3D reconstructions. This allows for reusing most of the software components written for the scanning application. Second, the set of correspondences is used to compute a group of local homographies that allow to find the projection of any of the points in the calibration object onto the projector image plane with sub-pixel precision. In the end, the data projector is calibrated as a normal camera. As a result, any camera model can be used to describe the projector, including the extended pinhole model with radial and tangential distortion coefficients, or even those with more complex lens distortion models.

The first step in this calibration procedure involves collecting images of a planar checkerboard: for each plane orientation, instead of capturing only one image, the user must project and capture a complete structured-light pattern sequence, theoretically by using any preferable coding strategies. After this operation, the intrinsic camera parameters can be obtained by using any camera calibration method. The procedure to compute checkerboard corner coordinates in the projector coordinate system can be decomposed into three steps: first, the structured-light sequence is decoded and

every camera pixel is associated with a projector row and column; second, a local homography is estimated for each checkerboard corner in the camera image; third and final, each of the corners is converted, as illustrated in Fig. 3.8 from camera coordinates to projector coordinates applying the local homography just found. Once camera-projector correspondences are known

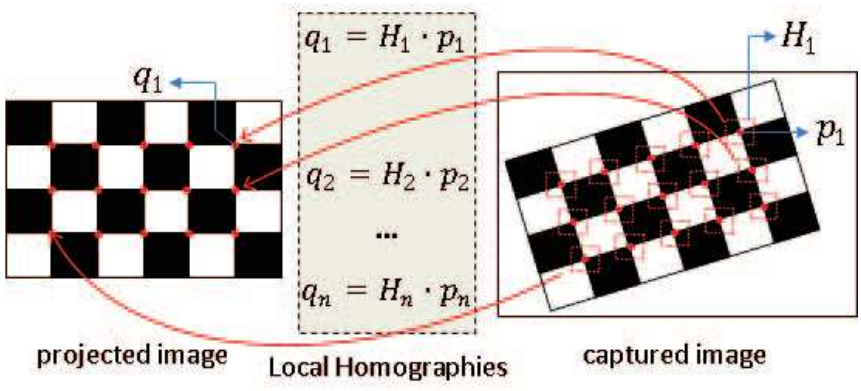


Figure 3.8: Projector corner locations are estimated with sub-pixel precision using local homographies to each corner in the camera image

any calibration technique available for passive stereo can be applied directly to the structured-light system. This method does not rely on the camera calibration parameters to find the set of correspondences. As a result, the projector calibration is not affected in any way by the accuracy of the camera calibration. Another advantage of this proposed method is the fact that it can be implemented in such a way that no user intervention is necessary after data acquisition, making the procedure effective even for unexperienced users.

3.3.2 Calibration by checkerboard projection

Tuotuo Li and Hongyan Zhang [14] proposed a calibration method based on the idea of acquiring images of the chessboard with its pattern originally printed on the board and then under projector illumination, in particular, what is projected is another chessboard in different colors. Based on those acquired images, geometric calibrations for both the camera and the projector can be performed.

In this type of approach, the choice of color is of fundamental importance to make the two different checkerboard structures distinguishable from

each other. In particular, three color schemes must be defined: a printed chessboard, a uniform pattern and a chessboard to be projected onto the first one. These color schemes must be defined in such a way that the printed chessboard is very clear under uniform pattern illumination and at the same time "invisible" under chessboard pattern illumination. The choice of colors is not unique and may depend on the kind of printer and projector used. Li and Zhang presented also an optimization in selecting the three colors for the best performance calibration. One example of colors selection is the one depicted in Fig. 3.9: the printed chessboard is composed of yellow and white cells; for making this pattern clear a blue uniform pattern is projected; the printed chessboard instead is composed by red and black cells, colors that allow to completely mask the printed pattern.

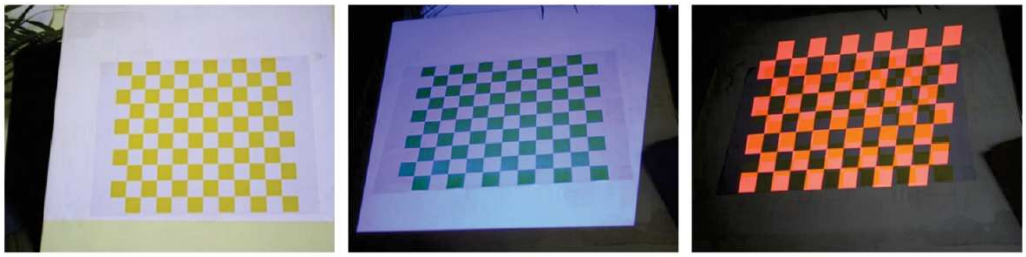


Figure 3.9: Acquired images of the chessboard under different illuminations: white uniform pattern; blue uniform pattern; black-red chessboard pattern

The calibration procedure requires capturing the printed chessboard placed in different positions, under a uniform pattern and then under the projected chessboard pattern. The set of acquired images under a uniform pattern is used for camera calibration, while the other images allow to find out the intrinsic parameters of the projector. As previously seen, in the calibration context the projector can be considered as an inverse camera meaning that it can be calibrated by Zhang method, once known how 3D points in the scene are related to their corresponding 2D points in the projector plane. In this particular case, the pixel coordinates of chessboard corners are known, being known the chessboard pattern to be projected. What must be found are the 3D corners of the projected chessboard. These quantities can be computed by using the calibrated camera: from the acquired images is it easy to extract the 2D corners of the projected chessboard, then, by considering the lines passing through these points and the center of projection, the corresponding 3D coordinates are found by imposing zero Z coordinate,

being the printed chessboard on the XY plane of the world reference frame. Once 2D-3D couples are found, the intrinsic projector parameters can be determined and the projector-camera system can be calibrated using standard methods for stereo systems.

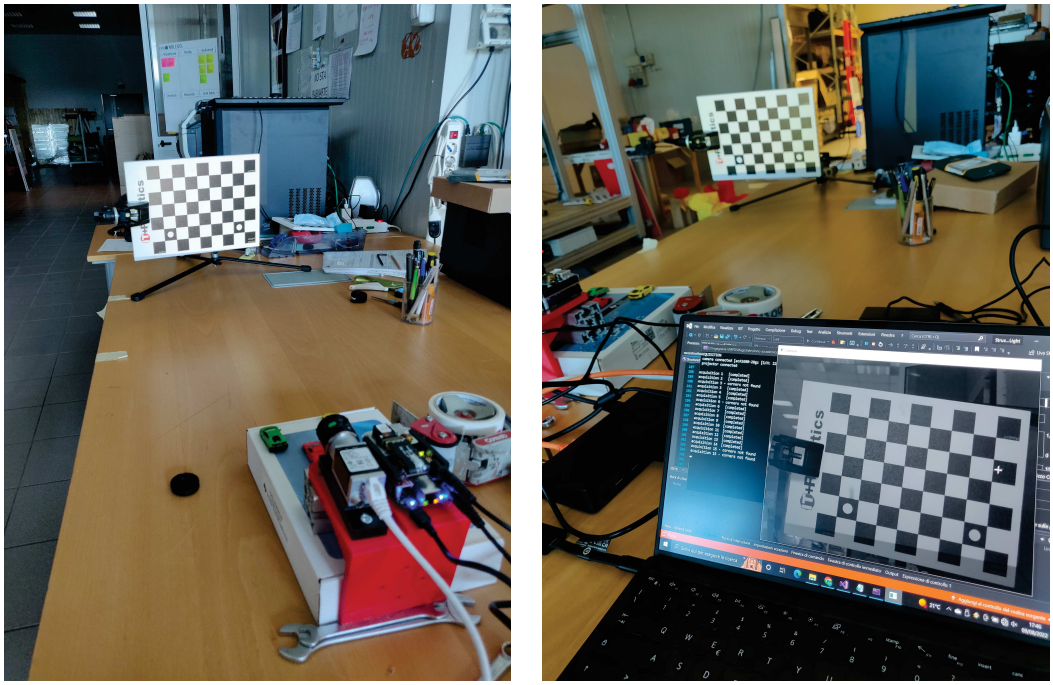
3.3.3 Prototype calibration

In general structured-light systems represents a really simple and effective way to acquire 3D models, but such high precision is only possible if the camera and projector are both accurately calibrated. As result, calibration is an essential step for this project. The prototype used in this master thesis has been calibrated following the approach proposed by Daniel Moreno and Gabriel Taubin, presented in Sec. 3.3.1. As already said, the main advantage of this method is the fact that the camera and projector are calibrated independently, meaning that the projector calibration is not affected in any way by the accuracy of the camera calibration. Furthermore, since this procedure is implemented by using the same software components dedicated to the scanning application, describing how this prototype is calibrated allows also highlighting some fundamental aspects related to the 3D reconstruction procedure. The complete calibration procedure can be summarized in these steps:

1. for each checkerboard orientation, find out corners coordinates from the image acquired with uniform pattern, use structured light projections for decoding each pixel of the checkerboard image.
2. for each corner consider a square window centered in the pixel coordinate of the corner itself and use the decoded pixels inside that window in order to compute a local homography.
3. find the corresponding corners coordinates in the projector plane by using the homographies computed in the previous step.
4. once a 3D world reference frame, fixed with the checkerboard plane, has been defined, use the Zhang method for determining the intrinsic parameters of the camera and projector, using as 2D points the pixel coordinates computed in step 1 for camera and in step 3 with local homographies for the projector.

5. Once the geometrical characteristics of the two devices have been found, use 3D points and corresponding 2D points in the camera and projector plane in order to estimate extrinsic parameters (i.e., the stereo parameters).

The images depicted in Fig. 3.10 give an idea of the experimental setup built for the calibration of the prototype, that has been performed in the laboratories of IT+Robotics. The checkerboard is placed at different orientations



(a) *Acquiring the chessboard images* **(b)** *Checking the acquired images*

Figure 3.10: Experimental setup for the prototype calibration at IT+Robotics

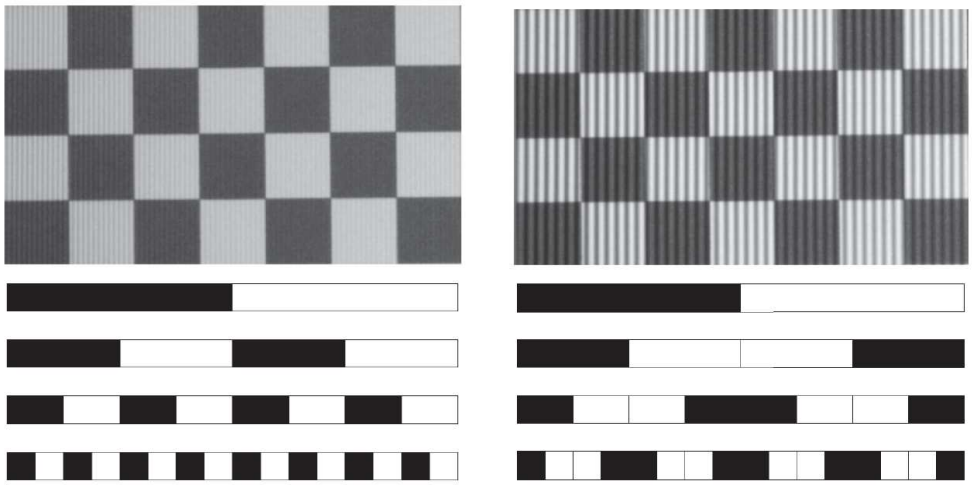
with respect to the sensor by using a specific mechanical support, see Fig. 3.10(a). In doing that it is important that the inner corners of the checkerboard, at any orientation, are inside the projected pattern of the projector and at the same time inside the field of view of the camera. While the first requirement is relatively easy to fulfill (just check if the corners are within the illuminated area), the second might be a little more difficult to manage. This is the reason why the calibration procedure has been monitored through the camera viewer software, to check what the camera is actually seeing, as shown in Fig. 3.10(b).

This calibration method requires the acquisition of structured light patterns that are able to encode each pixel of the projector: in this way each pixel of the camera can be related to its corresponding pixel in the projector. As already seen in Sec. 3.1, there are many coding strategies in order to do that. In this particular case, a binary code is adopted, by projecting both vertical and horizontal patterns. This choice is justified by mainly two reasons:

- it is very simple to generate and then decode this kind of patterns
- being part of the sequential projection techniques, this kind of coding strategy is able to guarantee a good accuracy

On the other hand, projecting for each checkerboard orientation a sequence of patterns means that acquisition times are very long but this is not a problem since the calibration is an offline operation that has to be executed once. As result, accurate calibration methods are preferable, neglecting timing aspects.

Instead of the classical binary code, the reflected binary code has been used, also known as *Gray code*, which is an ordering of the binary numerical system such that two successive values differ in only one bit. From Fig.



(a) *Binary code* **(b)** *Gray code*

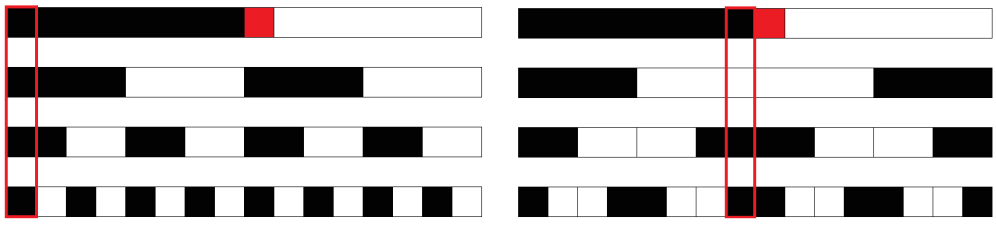
Figure 3.11: Acquisition of the least significant bit for the horizontal pattern

3.11 it is relative simple to deduce what are the advantages related to this choice. First of all, it can be seen that the transitions from black to white and vice versa are reduced by half. Furthermore, comparing bits at the same

position, the striped projected in the Gray code are thicker than the ones projected in the binary code: this fact helps in reducing the effects due to the light diffusion and hence in reinforcing the contrast between black and white stripes.

Once acquired, the entire sequence of patterns must be decoded in order to obtain a map associating each pixel of the camera to the corresponding pixel of the projector. In order to achieve this, the first step is the binarization of all the acquired images related to a sequence, meaning that to each pixel of the camera image is associated to a binary value 0 or 1 according to the fact it is illuminated or not. After having repeat this operation for all the pattern images, to each pixel of the camera image is associated a binary string that, if decoded, indicates the horizontal or vertical coordinate inside the projector image.

In the decoding part, the binarization of an acquired image represents the most challenging task since the light diffusion tends to make less clear the boundary between black and white stripes making the second ones thicker than the first ones. The proposed solution for this prototype consists projecting not only the original patterns but also the complementary patterns and then making a comparison: to each pixel is associated the 1 value if and only if its intensity in the image to be decoded is bigger than the intensity of the reference image, the one with the complementary pattern. The advantage of this approach with respect to others is that both original and complementary patterns are affected in the same way by the light diffusion phenomena making the binarization process more robust. In this context, an additional advantage of using Gray code is related to where a transition occurs. For



(a) *Binary code* **(b)** *Gray code*

Figure 3.12: Consequence of binarization error in the two coding strategies

the classical binary code, each transition between two colors occurs in correspondence with a transition of the least significant bit and this means that

in a single pixel there could be an error of more than one bit; not only, even a one-bit error could bring to a wrong code very far from the correct one. In the Gray code instead, the transition from one color to another in a pattern image occurs always in uniform areas of the other pattern images of the sequence. Assuming that only in color transitions there could be a decoding error, this last observation ensures that in one pixel there could be at most a one-bit error and, if this happens, the resulting stripe assigned to the pixel is in any case adjacent to the correct one. To make things more clear, Fig. 3.12 shows the consequence of a binarization error for the most significant bit in the black-to-white transition. From what can be seen, detecting zero in place of one, generate the highlighted stripe as result: a very far stripe for the binary code or an adjacent stripe for the Gray code.

To sum up, all these considerations allow explaining not only how the prototype has been calibrated but also how it basically works from the software point of view in order to perform a 3D acquisition.

Chapter 4

Rectification pipeline for structured light sensors

Up to now rectification has been presented as a prominent approach in stereo vision systems, but what about structured light sensors? This chapter will present two possible approaches for rectifying a projector-camera system, one involving only the camera and the other both the camera and projector. In particular, the second approach will be also tested in the available prototype of 3D scanner. Before going into detail, we present the starting point for rectifying a structured lighting system.

4.1 Sensors setup analysis

First of all, in order to understand rectification also from the geometrical point of view, it could be interesting to analyze the calibration results obtained for the available prototype.

Camera calibration The calibration results for the camera can be summarized by estimated camera matrix K_{cam} and distortion parameters D_{cam}:

$$K_{cam} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3673.59 & 0 & 799.50 \\ 0 & 3673.02 & 599.50 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

$$D_{cam} = \begin{bmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{bmatrix} = \begin{bmatrix} -0.12 & 0.68 & 0 & 0 & -2.47 \end{bmatrix} \tag{4.2}$$

where, for the camera matrix, f_x, f_y represents the camera focal length in pixels along x and y direction respectively, and c_x, c_y are the coordinates of the principal point in pixels; while in the distortion parameters, k_1, k_2, k_3 describe the radial distortion and p_1, p_2 the tangential distortion.

Projector calibration In a similar way, since the projector is seen mathematically as an inverse camera, the calibration results are represented by K_{prj} and D_{prj}:

$$K_{prj} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1417.98 & 0 & 319.50 \\ 0 & 1417.20 & 179.50 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

$$D_{prj} = \begin{bmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{bmatrix} = \begin{bmatrix} 0.04 & -5.73 & -0.03 & 0 & 83.71 \end{bmatrix} \tag{4.4}$$

Stereo calibration The calibration results are basically the translation vector T and the rotation matrix R describing the mutual position and orientation of camera and projector frames:

$$T = \begin{bmatrix} -46.13 \\ -2.47 \\ 10.91 \end{bmatrix} [mm] \qquad R = \begin{bmatrix} 1.00 & 0.01 & 0.06 \\ -0.01 & 1.00 & 0.01 \\ -0.06 & -0.01 & 1.00 \end{bmatrix} \tag{4.5}$$

How to interpret these results?

4.1.1 Analysis of stereo parameters

In Fig. 4.1 is depicted how camera and projector are placed in space relative to each other, based on the results reported in (4.5). The matrix R describes how the camera frame is rotated with respect to the projector frame; more in detail, its columns represent the x, y, z coordinates of the camera frame axis expressed in the projector frame, imaging that the two reference frames have hypothetically the same origin. The vector T instead describes in millimeters how the origin of the camera frame is translated with respect to the origin of the projector frame. Therefore, the couple (R, T) defines a so called rototranslation that combines translation and rotation in order to describe the pose of the camera in space, once the pose of the projector is known.

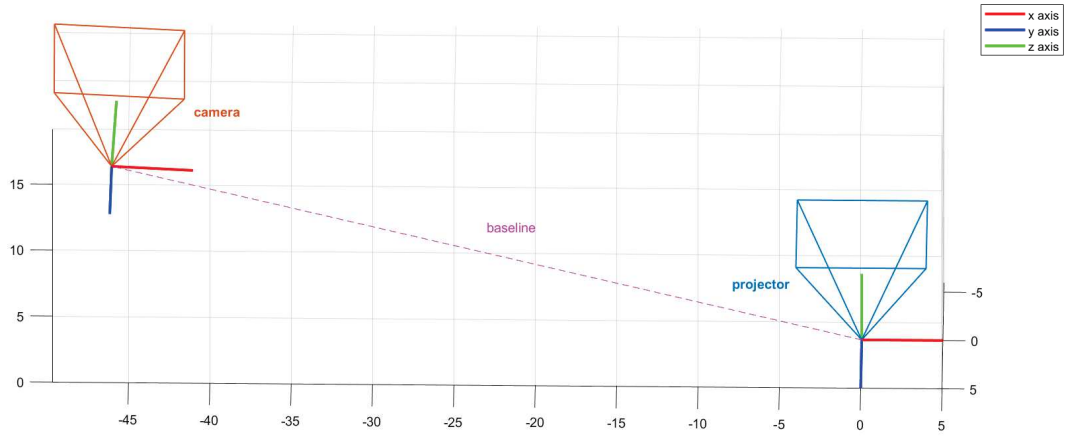
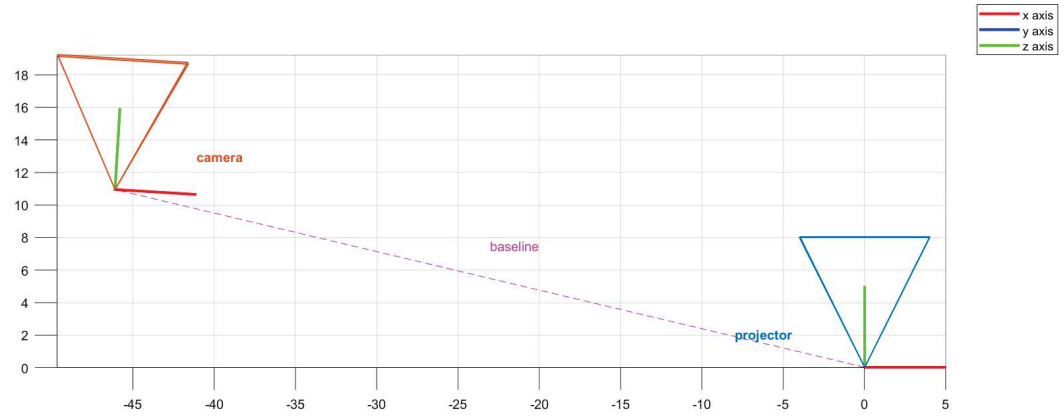
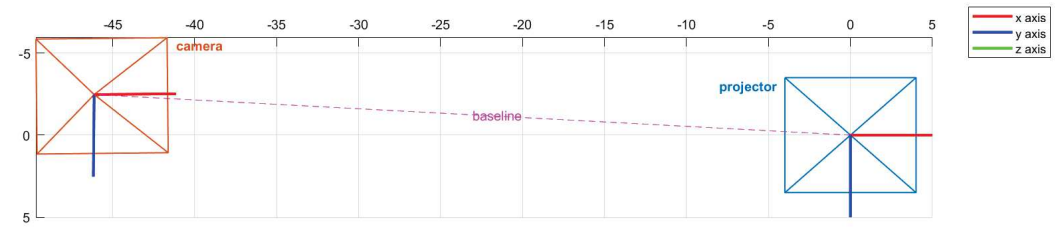


Figure 4.1: Mutual position between camera and projector

From what can be seen, the camera frame seems to be almost aligned with the projector frame; this is an expected result since matrix R is very close to an identity matrix. Of course, this is not a perfect alignment, as it can be better appreciated in Fig. 4.2, presenting the camera-projector couple from a different point of view. In particular, Fig. 4.2(a) confirms what has



(a) *Top view*



(b) *Front view*

Figure 4.2: Camera and projector pose from a different points of view

been already said about the projector and camera configuration, namely the little vergence existing between these two devices.

Starting from the knowledge of stereo parameters it is possible to compute the essential matrix E and the fundamental matrix F for the prototype of 3D scanner:

$$\begin{aligned} E &= [T]_{\times} R \simeq \begin{bmatrix} 0.2573 & -10.8853 & -2.5791 \\ 8.1422 & -0.3522 & 46.7846 \\ 2.9313 & -46.1053 & -0.3131 \end{bmatrix} \\ F &= K_{cam}^{-T} E K_{prj}^{-1} \simeq \begin{bmatrix} 0 & 0 & -0.0006 \\ 0 & 0 & 0.0318 \\ 0.0005 & -0.0119 & 1.0591 \end{bmatrix} \end{aligned} \tag{4.6}$$

These two matrices are the main entities describing the epipolar geometry and will be useful later in describing both the two rectification strategies. As one can imagine stereo parameters are important not only because they describe the mutual position between camera and projector, but also because they give an intuition about how rectification rotates, and hence modifies, camera and projector reference frames. In addition, they are also essential information when applying rectification strategies based on calibration data, as already seen in Sec. 2.3.1.

How do stereo parameters change after rectification? From the stereo parameters point of view, the action performed by rectification can be equivalently formulated in imposing that the couple (R, T) related to the projector-camera pair assumes the following form:

$$\bar{T} = \begin{bmatrix} \bar{t}_x \\ 0 \\ 0 \end{bmatrix} \qquad \bar{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

meaning that the two frames must be perfectly aligned one respect to the other and translated only along x axis, as depicted in Fig. 4.3 where, in most of the cases, the translation along x axis is exactly as the norm of vector T, namely $\bar{t}_x = \|T\|$.

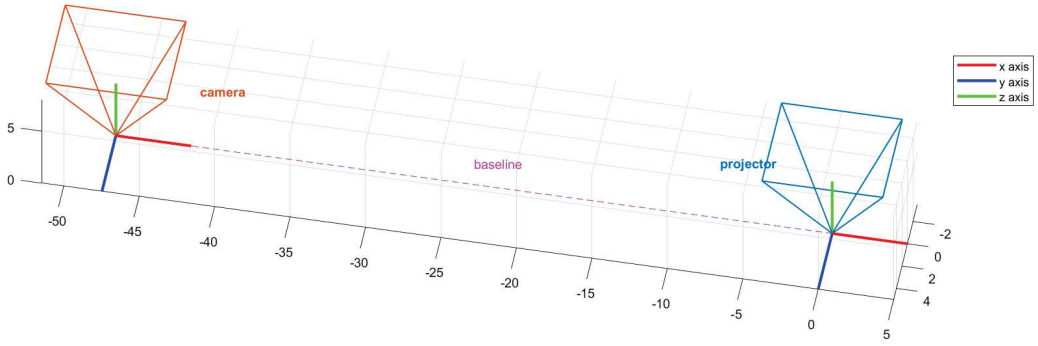


Figure 4.3: Camera and projector pose after rectification

4.2 Only camera rectification

A first rectification strategy that could be applied for the available prototype of structured light 3D scanner consists in trying to act only on the camera in order to align it with the projector, that on the contrary must remain fixed. This section illustrates what are the main issues that must be considered in order to apply this strategy and the related advantages and drawbacks.

4.2.1 Preliminary observations

As already seen, stereo rectification is in general applied by two homography matrices, and the effect of an homography transformation is to move from an image to the same image as it would be taken by the camera in the desired orientation different from the original one. In other words, the result after rectification for a camera is an image containing what the camera itself should see in this new orientation. In the particular case of the available prototype, the first two steps are the following:

- Remove distortion from both camera and projector in order to improve the quality of 3D reconstruction
- Virtually rotate the camera by applying the homography $H = K_{cam}^{-1} R K_{cam}$, with R defined as in (4.5) so that it has the same orientation of the projector

Since the camera in its original orientation was almost aligned with the projector, it is expected that this pure rotation does not produce a strong distortion in the camera image, as it can be visualized in Fig. 4.4.

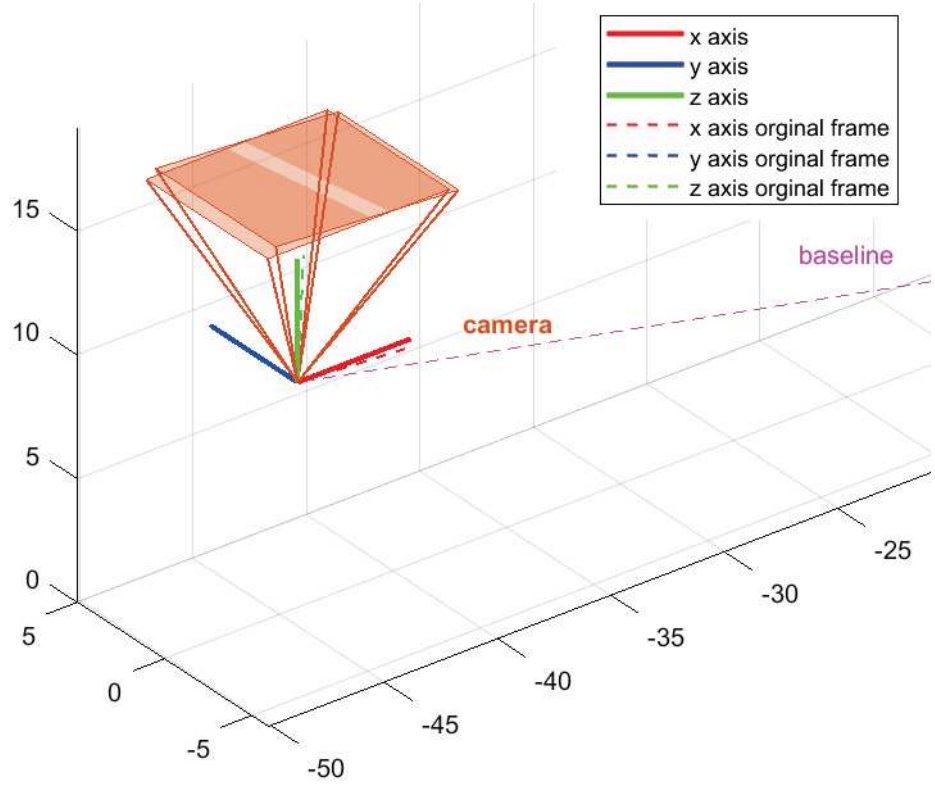


Figure 4.4: Pure rotation of camera frame

By considering again the camera and projector setup in Fig. 4.1 it should be clear that only a pure rotation is not enough in order to align the camera with the projector; also by observing that the y and z components of the T vector are different from zero. This section provides a proof of this fact both from an algebraic and a geometric point of view and then indicates some ideas in order to overcome this problem.

4.2.2 A single rotation is not enough

Consider a rectilinear stereo rig, namely with stereo parameters as in (4.7). It could be proved that in this case, the essential matrix $\bar{E}$ relating the two cameras has the following form:

$$\bar{E} = [\bar{T}]_{\times}\bar{R} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\bar{t}_x \\ 0 & \bar{t}_x & 0 \end{bmatrix} \tag{4.8}$$

This fact also holds for a structured light 3D scanner, where the projector is seen as an inverse camera. The fundamental matrix $\bar{F}$ instead has a more complicated form, depending also on the intrinsic parameters of the optical devices. In particular, imagining to have rectified the available prototype, numerically it results:

$$\bar{F} = K_{cam}^{-T}\bar{E}K_{prj}^{-1} \simeq \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \bar{f}_{23} \\ 0 & \bar{f}_{32} & \bar{f}_{33} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -0.0335 \\ 0 & 0.0129 & -1.7353 \end{bmatrix} \tag{4.9}$$

Given a generic pixel in camera plane p_c and in projector plane p_p, the goal is to find a unique homography matrix H mapping p_c in the new virtual image $p'_c = Hp_c$ such that this new system behaves as a rectilinear stereo rig: if this H matrix exists the rectification problem is solved. The epipolar constraint becomes $p'_c\bar{F}p_p = 0$ and this implies:

$$p_c^T H^T \bar{F} p_p = p_c^T F p_p = 0 \quad \Longrightarrow \quad H^T\bar{F} = F \tag{4.10}$$

$$F = K_c^{-T}EK_p^{-1} = K_c^{-T}[T]_\times RK_p^{-1} = H^T\bar{F} \tag{4.11}$$

The unknown homography matrix is a 3x3 matrix meaning that 9 parameters need to be estimated. However, it is known that the homography is in general defined up to a scaling factor so nothing forbids setting $h_{33} = 1$, reducing to 8 the number of parameters to be estimated:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{21} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \tag{4.12}$$

In addition, matrix H is defined from a rotation matrix R^* according to

$$H = sK_cR^*K_c^{-1}, \qquad s \in \mathbb{R},\ R^* \in \mathbb{SE}(3) \tag{4.13}$$

where s is the scaling factor. This last consideration should further reduce the degrees of freedom of matrix H to 4, hence simplifying the problem. From now on the matrix system $H^T\bar{F} = F$ to be solved could lead to different situations in the general case:

- the system admits only one solution H
- the system admits more than one solution or an infinite number of solutions
- there are no matrices H able to satisfy that condition

In addition, if there exists a solution, it is not guaranteed the possibility to get it in close form.

By considering again the results obtained from stereo calibration it is immediate to see that for the available prototype the system admits no solution, indeed:

$$H^T \bar{F} = \begin{bmatrix} h_{11} & h_{12} & h_{21} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \bar{f}_{23} \\ 0 & \bar{f}_{32} & \bar{f}_{33} \end{bmatrix} = \begin{bmatrix} 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \tag{4.14}$$

which cannot be equal to the actual fundamental matrix previously reported. This is not a surprising result; it is simply the proof of what already anticipated: it is not possible to align the camera and projector simply by applying a pure rotation on the camera.

In the same way it is possible to end up at the same result also by reasoning from a geometric point of view.

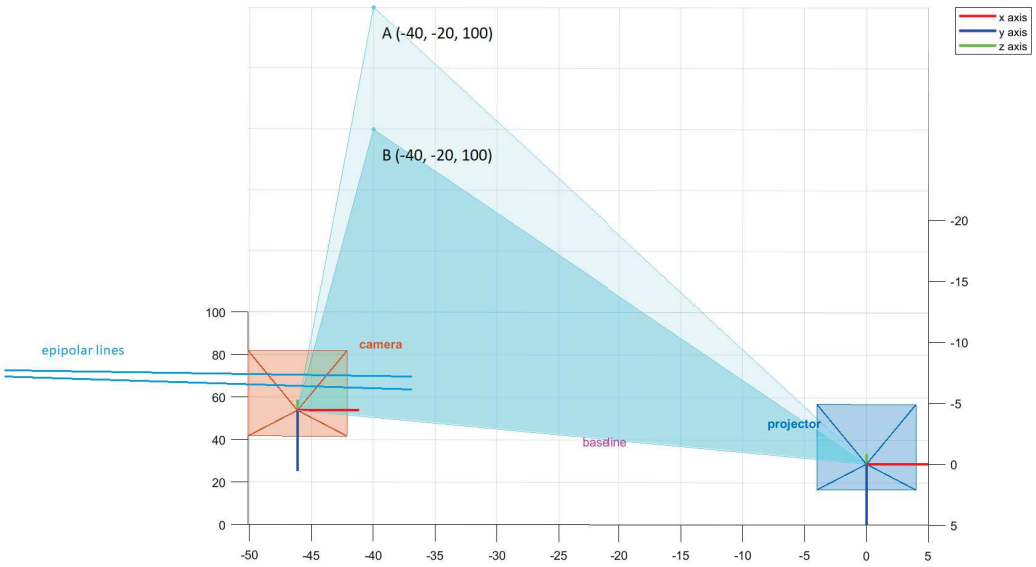


Figure 4.5: Epipolar geometry for rotated camera frame

In Fig. 4.5 is visualized the projection of two 3D points A, B in the camera and projector planes and two epipolar planes formed from these projections. In this case it is assumed that camera and projector has the same focal length just for simplicity, but the reasoning is the same even in the more general case in which the two focal lengths differs. Points A, B have coordinates $[-40, -20, 100]^T$ and $[-40, -10, 100]^T$ $[mm]$ respectively, so they differs only on a displacement of 10 $[mm]$ along y axis, and the two epipolar lines in the camera plane are defined exactly by moving these two points along the projection line passing through the center of projector frame and seeing where they are projected in the camera plane. It is immediate to see these epipolar lines are neither parallel among them nor parallel respect to the x axis of the camera, concluding as before that having camera and projector with the same orientation is a necessary but not sufficient condition for rectification.

4.2.3 Virtual translation of the camera

Up to now what is missing in order to achieve the goal of rectification is a virtual translation of the camera in order to make the camera plane exactly parallel to the projector one.

In other words, the aim is to change the actual translation vector T into its desired form $\bar{T}$ by zeroing the y and z components. The virtual camera translation in order to do that can be decompose into a sequence of more consecutive sub-translations:

- a first translation along z axis in order to set to zero the third component of T vector
- a second translation along y axis in order to set to zero the second component of T vector
- optionally, a translation along x axis making the baseline length exactly equals to the norm of T vector

of course the rectification works in principle also for different lengths of the baseline. At the end, the result should be equal with the one previously depicted in Fig. 4.3.

The main advantage of this first rectification strategy is its simple application. The projector frame remains fixed in this procedure, meaning that,

after having obtained the new virtual camera image, the search of correspondences is pretty straightforward: for each pixel of the camera the corresponding pixel of the projector is on the same horizontal epipolar line, and in particular its column coincide with the vertical decoded value at the camera pixel location, exactly as in the 3D scanner before doing rectification. Therefore, this strategy does not require strong changes in the overall scanner pipeline.

On the other hand, acting only on the camera may produce a consistent loss of information in the new transformed image, especially when the required virtual translations are quite large with respect to the image size of the camera. This is exactly the case of the prototype: the overall virtual translation causes all valid pixels of the image exit the image size and the result is a black image. In other to overcome this problem a possible solution is to change the focal length in order to capture the valid image and eventually change manually the coordinates of the principal point in order to properly center the valid image itself inside the image size. Of course, this implies changing principal point coordinates and focal length also for the projector as consequence. In general, all these changes causes a significant perspective distortion in the camera image and hence a worst accuracy of the resulting point cloud. On the contrary, by properly rectifying both camera and projector it is possible to minimize this distortion on the camera image.

4.3 Projector-camera rectification

As already anticipated, this second strategy involves also the projector and this means that all the procedure related to the search of correspondences must be accurately adapted. This section describes in detail the entire pipeline, that has been also implemented on the prototype of 3D scanner.

4.3.1 Getting rectification homographies

The basic idea for this second rectification strategy is to consider the projector-camera system as a common stereo rig. As result, any pair of rectification homographies suitable for a stereo pair can be used in principle also for the

prototype of 3D scanner. In this case, being the intrinsic and stereo parameters available after calibration, a possible choice consists in relying on the *OpenCV* function `cv::stereoRectify()`.

As stated in the documentation, this function computes the rectification transforms for each head of a calibrated stereo camera. From Fig. 4.6 it is possible to appreciate a graphical overview of the involved parameters.

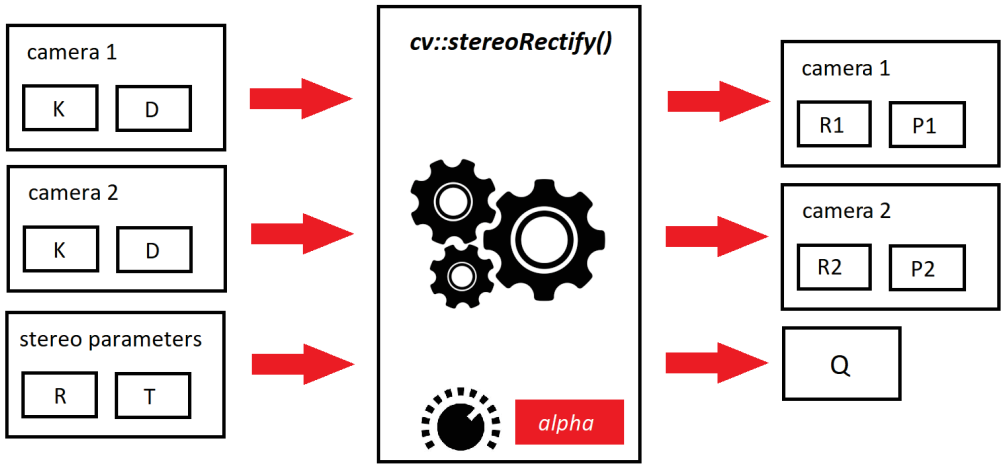


Figure 4.6: The *OpenCV* function `cv::stereoRectify()`

This function takes as input intrinsic matrices and distortion parameters for both cameras and the related stereo parameters, and returns as output:

- **R1, R2** These are the two rotation matrices respectively for the first and second camera that virtually make both camera image planes the same plane. In particular, each of these matrices brings points given in the unrectified camera's coordinate system to points in the rectified camera's coordinate system, or equivalently, performs a change of basis from the unrectified camera's coordinate system to the rectified camera's coordinate system.
- **P1, P2** These are the 3x4 projection matrices in the new rectified coordinate system respectively for first and second camera, so each of these matrices projects points given in the rectified camera coordinate system into the rectified camera's image.
- **Q** This is 4x4 disparity-to-depth mapping matrix. It will be clear later the role of this matrix.

The *alpha* parameter is a free scaling parameter ranging from 0 to 1. Setting *alpha* to 0 means that the rectified images are zoomed and shifted so that only valid pixels are visible; otherwise if *alpha* is set to 1 the rectified image is decimated and shifted so that all the pixels from the original images from the cameras are retained in the rectified images, and hence some black areas could appear after rectification. Any intermediate value yields an intermediate result between the two extreme cases, while if *alpha* is -1 or absent, the function performs the default scaling.

More details about these and other parameters related to this *OpenCV* function can be found in the documentation [6].

In the particular case of the 3D scanner, this rectification function has been applied by choosing the camera as camera 1 and the projector as camera 2, in order to be consistent with the function `cv::stereoCalibrate()`. Notice from Fig. 4.7 how camera and projector reference frames are now shifted relative to each other along the x axis as result of the returned $R1$, $R2$ rotation matrices.

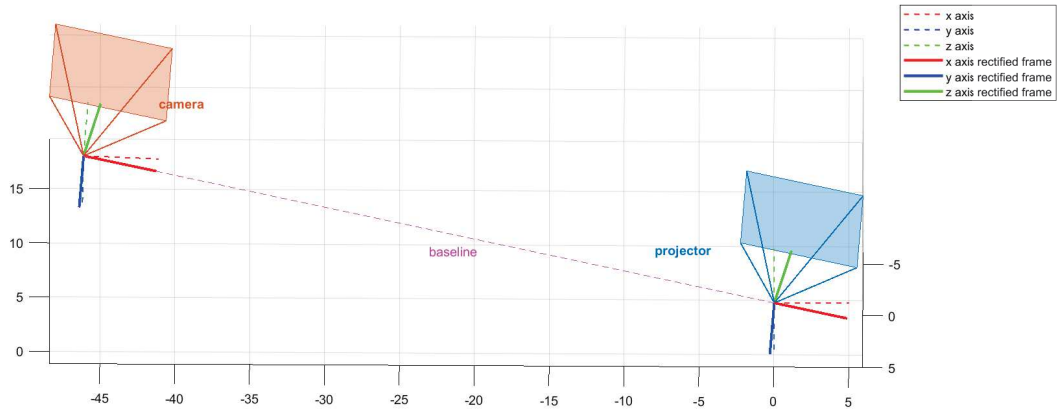


Figure 4.7: Projector-camera alignment using `cv::stereoRectify()`

Being this an horizontal stereo configuration, in the camera and projector images the corresponding epipolar lines must be horizontal with the same y coordinate. On the other hand, this prototype of 3D scanner is characterized by a huge difference in resolution between camera and projector: in particular, the size of the camera image is quite bigger respect to the one of the projector. In order to deal with this issue, it has been chosen to scale up the the projector image imposing that the two rectified images has the

same size of the camera. This solution in principle should avoid a reduction in the resolution of the rectified camera image. In this context referring to a "projector image" could seem unusual, but it will be clear later the exact meaning of this term, for now just see the projector as another camera.

A key fact influencing the aspect of the final 3D reconstruction is the scaling of the rectified camera image. As already seen, this image shows what the camera should see in this new orientation, but while rotating the camera, some valid pixels could exit the image size and at the same time some new pixels could enter, that are unknown since this is only a virtual and not real camera rotation. This fact justifies the common presence of black areas after any rectification process. However, trying to reduce as much as possible these black areas implies losing lots of valid pixels and vice-versa, trying to keep all valid pixels inside the image increases black areas. As consequence, a trade off between these two actions is needed. For the prototype of 3D scanner, this issue is managed both automatically, by setting the *alpha* parameter, and manually, by specifically acting on the new projection matrices $P1$, $P2$. In particular, this is the general form of the new projection matrices returned by the rectification function:

$$P1 = \begin{bmatrix} f & 0 & cx_1 & 0 \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad P2 = \begin{bmatrix} f & 0 & cx_2 & \|T\| \cdot f \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.15}$$

It has been proved empirically that the *alpha* parameter only acts in the focal length of these matrices producing a zoomed-in or zoomed-out version of the rectified camera image. However, if the rectified image is not properly centered inside the image size, as this is the case for the prototype, a possible zoom-in in order to reduce black areas implies a consistent lost of valid pixels, so that only a small part of the original image is still visible and could be reconstructed. This is the reason why the two projection matrices has been also manually modified by acting on the coordinates of the principal points: the idea is to center the interested image by slightly modifying cx_1 and cy and then choose a proper value for *alpha* so that the rectified image has approximately the same dimension of the original one. Of course, the same changes in cx_1, cy of $P1$ must also be applied in cx_2, cy of $P2$ in order to preserve the alignment and the disparities between the two views.

As result, the final rectification homographies for the 3D scanner can be computed as:

$$H_{cam} = P1[:,1:3] \cdot R1 \cdot K_{cam}^{-1}, \quad H_{prj} = P2[:,1:3] \cdot R2 \cdot K_{prj}^{-1} \tag{4.16}$$

where $P1[:,1:3]$ and $P2[:,1:3]$ are the 3x3 projection matrices extracted from $P1$ and $P2$ respectively by selecting only the first three columns. Just to give an idea in Fig. 4.8 is reported an example of camera image rectified according to H_{cam}.

Figure 4.8: Example of camera image rectification

A note about the centering operation The area of the image illuminated by the projector coincides with the portion of the image that will be effectively reconstructed. Therefore the centering operation previously described mainly focus in center and zoom as much as possible this illuminated area without losing too much valid pixels. In this centering operation only the camera image matters: given a point in the camera image, if the corresponding point in the "projector image" has negative coordinates no problem arises from the triangulation process, indeed the projector does not contain an image to be visualized.

4.3.2 Test the rectification quality

Before proceeding with the description of the pipeline, it could be interesting to evaluate the accuracy of the rectification homographies previously computed in order to see if camera and projector images are effectively aligned.

As already said, if rectification is performed in a proper way, a couple of correspondent points should share the same y coordinate. However this property holds only in the ideal case, while in real applications a small vertical shift is always present due to the non perfect homographies or the presence of distortion in the camera and/or projector. Aiming to quantify this shift error, a possible method consists in take some already known corresponding points between camera and projector and check how big is the error in the alignment. This method is quite simple to be applied for the available prototype, indeed a possible way to find correspondences between camera and projector is to take inspiration from the calibration procedure. Basically, the idea is to consider among the calibration dataset an image in which the checkerboard takes most of the image area, detect the inner corners and then find the correspondences corners in the projector plane. These corners can be computed simply by using the same procedure adopted for the calibration, namely by decoding all the camera points thanks to the pattern images and then by the use of local homographies in order to get a more accurate results. As it can be seen, this method is pretty simple and do not require in this case the implementation of new code.

In Fig. 4.10 is reported a comparison between a checkerboard image used for calibration and an image containing only the inner corners of the same checkerboard seen from the projector point of view. Consider that both

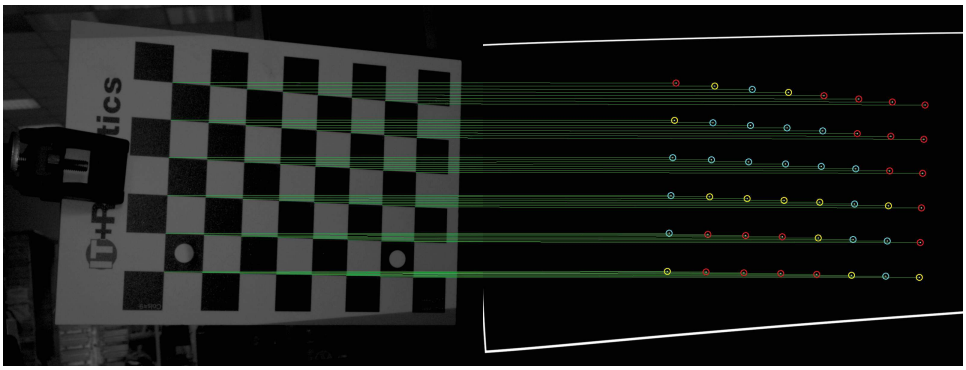


Figure 4.9: Rectification quality for undistorted images

in camera and projector distortion has been removed. For the i-th corner the shift error err_i has been computed as:

$$err_i = y_i - \hat{y}_i \tag{4.17}$$

where y_i is the y coordinate of the i-th camera corner and $\hat{y}_i$ is the corresponding corner seen by the projector. From this image it is difficult to appreciate how big is this error and this is the reason why the projector corners has been circled of different colors just to give an idea of the measured shift error:

- **cyan circles** shift error less than 1 pixel
- **yellow circles** shift error greater than 1 pixel and less than 1.5 pixel
- **red circles** shift error grater than 1.5 pixels

Moreover, to provide a final value it has been computed the mean of shift errors for all the corners resulting in:

$$e_{mean} = \frac{1}{N}\sum_{i=1}^{N} err_i = 1.24 \tag{4.18}$$

where N is the total number of inner corners. This results indicates that in the mean case there is always an error of roughly one pixel between two correspondence points. This is due to many reasons, principally the non perfect calibration results that influence the rectification, which in turn from the *OpenCV* documentation is not guaranteed to be perfect.

Just for completeness, is reported in Fig. 4.10 the same results obtained without removing the camera and projector distortion. In this case the measured mean error is about 1.78 confirming how the distortion influences the quality of rectification: this can be also noted from the greater number of red circles in the image.

Another possible method that can be used for evaluating the quality of rectification consists in estimating from camera and projector correspondences the fundamental matrix, for instance by using RANSAC algorithm, and then visualizing the epipolar lines to see if they are parallel or not.

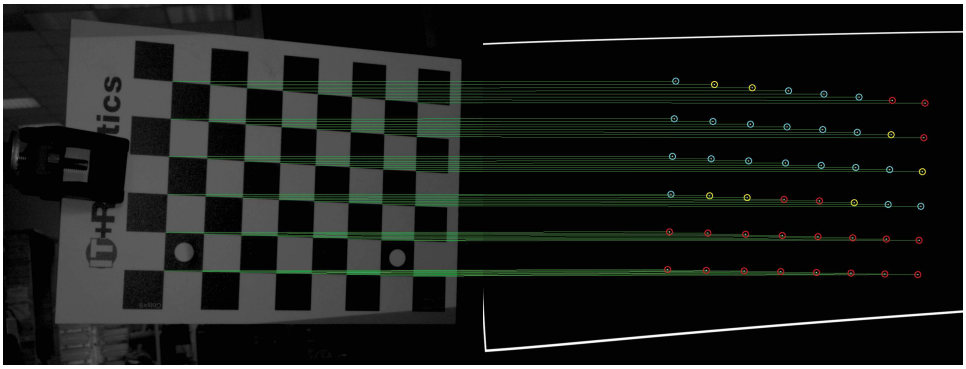


Figure 4.10: Rectification quality for original images

From what it can be seen, given the immediateness in obtaining correspondence points between camera and projector, another rectification strategy that could be use is the *OpenCV* function `cv::stereoRectifyUncalibrated()` that, as the name suggests, tries to compute the rectification homographies without using calibration results. In particular this function implements the algorithm [10]. On the other hand, this function has been tested on the same correspondence points used for the evaluation above but the obtained results seemed to be worst respect to the `cv::stereoRectify()` function. Indeed, this is a general fact: if good calibration data are available it is better to rely on them in order to perform rectification.

4.3.3 Rectification in coding and decoding

Up to now a detailed description about the chosen rectification homographies has been provided, but how is it possible to take into account rectification in the classical pipeline of a structured light 3D scanner?

The most convenient solution for the available prototype can be summarized in the following steps:

- project the original sequence of patterns related to the coding of the columns according to Gray code
- for each projected pattern acquire the corresponding camera image and then consider its rectified version according to H_{cam}
- based on the sequence of rectified camera images, perform the decoding using the strategy with complementary patterns

- for the search of correspondences between camera and projector consider the rectified camera image and the rectified projector image

As it can be seen, the proposed solution tries to replicate the same steps followed for a common stereo rig and it basically works for a general structured light 3D scanner composed by a camera and projector pair. In a nutshell, after having acquired the sequence of camera images, consider the projector image as the image seen from the projector point of view, as it would be a second camera. Of course this image is not directly accessible since the projector is not a camera but it can be retrieved by considering for each camera point the corresponding projector point, that can be obtained after the rows and columns decoding process. This reasoning should clarify the meaning of the term "projector image". Therefore, imagine to get a camera image and a projector image, then the matching is performed between the rectified versions of these two images, exactly as a stereo rig does.

The main advantage of this solution is the possibility to project the original vertical pattern, the same used for the structured light 3D scanner without rectification. Because of the very low resolution of the projector, considering to hypothetically project a rectified pattern would be a serious problem for the accuracy of the 3D reconstruction; in fact, especially for the pattern related to the least significant bit, the pixel discretization would distort a lot the ideal form of the pattern. Just to give an idea, Fig. 4.11 shows a comparison between the original pattern and its rectified version according to H_{prj}. As consequence, any solution involving pattern rectification should be avoided for this prototype of 3D scanner.

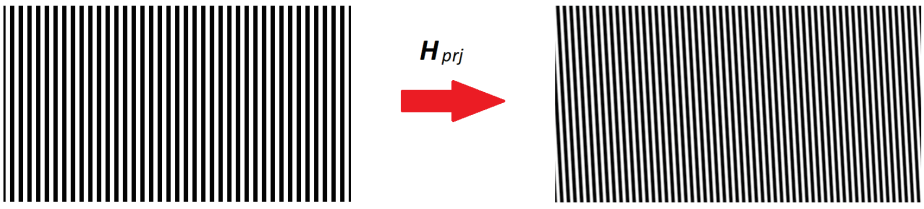


Figure 4.11: Avoid to project the rectified pattern

Nothing new regarding the decoding part, except for the fact that the decoding strategy must be applied not in the original sequence of acquired camera images but in its rectified version. Of course, being the projector-camera system in an horizontal configuration, only the vertical code is needed.

In Fig. 4.12 is reported as example the decoding result related to the image of Fig. 4.8.

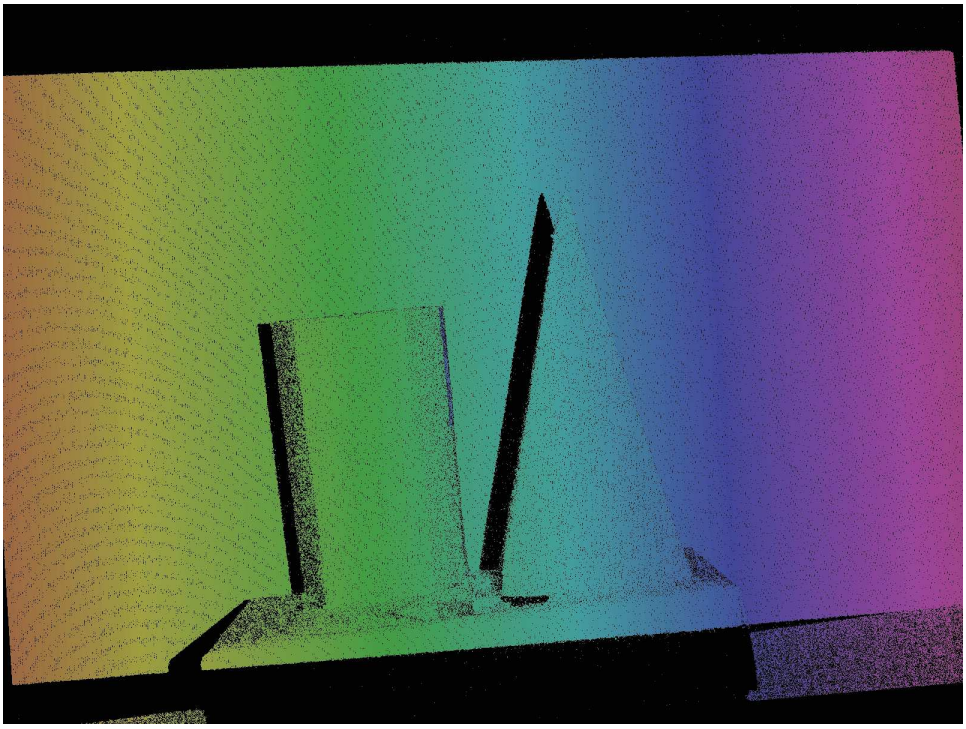


Figure 4.12: Vertical code

4.3.4 Correspondences problem

In general, for a structured light system the presence of a projector in place of a camera makes the correspondences problem quite different respect to a stereo rig. As one can imagine, strategies as the ones described in Sec. 2.4 cannot be applied in this context. In particular, in this subsection it is described in detail what solution has been developed in order to solve the correspondences problem for the rectified prototype of 3D scanner.

Given a pair of rectified camera and projector images, the goal is to find for each pixel in the camera image the corresponding one, in terms of rows and columns, in the projector image. To better visualize this fact, consider Fig. 4.13 which presents an example of camera and projector rectified images; in particular the camera image is represented as its decoded version, meaning that for each pixel the corresponding vertical code is available, while the projector image contains only the rectified vertical code relative

to a certain camera pixel. This gives an idea about how the corresponding projector pixel can be retrieve and more in general, how the projector image can be constructed.

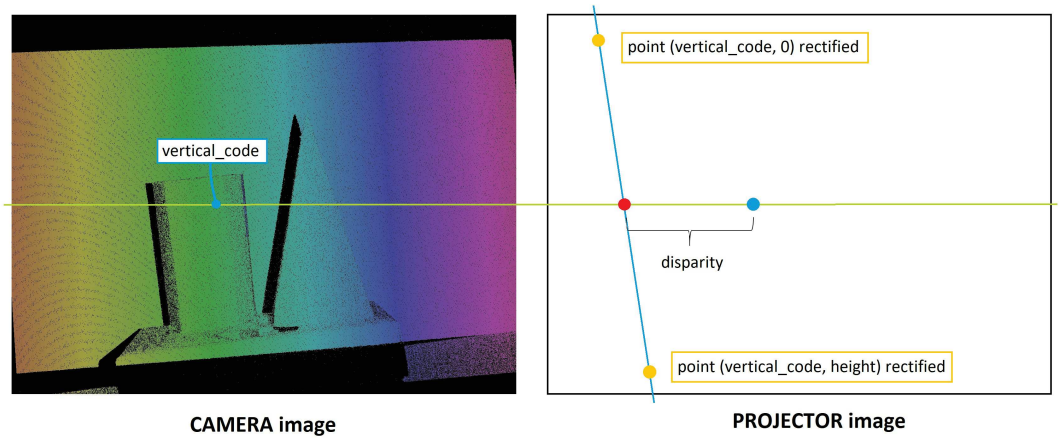


Figure 4.13: Finding correspondences

By keeping in mind Fig. 4.13, consider the following procedure described step by step for solving the correspondences problem:

1. For each camera pixel $p_c = (u_c, v_c)$ consider the related scanline, namely the horizontal line passing through it.
2. The corresponding projector pixel $p_p = (u_p, v_p)$ lies in the same scanline, meaning that $v_c = v_p$, notice that in this way the y coordinate of the projector pixel is easily retrievable, at this point the only unknown is the x coordinate u_p.
3. Read the binary code *vertical_code* related to p_c.
4. Consider in the projector image the two points $A = (vertical\_code, 0)$ and $B = (vertical\_code, height)$, where *height* is the height of the projector image.
5. Compute from A and B the points $A' = H_{prj}A$, $B' = H_{prj}B$, these are the two yellow points in the projector image of Fig. 4.13.
6. In order to get the coordinate u_p, consider the line passing through A' and B': this line corresponds to the rectified version of the original vertical line indexed by *vertical_code*. Then compute the intersection

between this line and the scanline previously defined, and this is the target projector point p_p.

Notice that from the mathematical point of view this procedure can be summarized by simply considering the following relation that is consequence of basic geometric observations:

$$\begin{cases} v_p &= v_c \\ u_p &= A'.x + \frac{u_c - A'.y}{B'.y - A'.y} \cdot (B'.x - A'.x) \end{cases} \tag{4.19}$$

where $A'.x$, $A'.y$ and $B'.x$, $B'.y$ are respectively the x and y coordinate of points A' and B', rectified according to H_{prj}. Once the projector point is known, the disparity d related to the couple p_c, p_p can be easily computed as $d = u_c - u_p$.

A note about the rectified projector Consider the 3D scanner without rectification, in the projector image the vertical pattern should be intended as a composition of vertical stripes infinitely extended along the vertical direction, while in the horizontal direction the pattern is limited by the width of the projector image. The opposite reasoning holds for the horizontal pattern, in this case the stripes are infinitely extended in the horizontal direction, while limited by the height of the projector image along the vertical direction. This reasoning still holds with some adaptations for the rectified version of the 3D scanner: in this case the lines described in the procedure above which represent the rectified vertical pattern are infinitely extended along their directions and this means that in this kind of application it is very common to find a projector point that has a negative x coordinate, namely the intersection between the rectified vertical pattern and the scanline is outside the rectified projector image.

4.3.5 Look-up table

Consider to iterate the procedure described in the previous subsection for all the pixels of the camera image, it is easy to realized that the formula described in (4.19) many times is computed with the same vertical code as input value and hence with the same $A'.x$, $A'.y$ and $B'.x$, $B'.y$ quantities. Moreover, also in the same scanline it could happen that more camera points

share the same vertical code and in this case the corresponding $u_p$value will be the same. This is not an unusual event but it is something that happens very often due to the difference in resolution between camera and projector.

As result, in order to avoid to recompute always the same quantities, a look-up table has been created. The main idea is to have a sort of structure from that the algorithm can access during the online execution to directly read the corresponding u_p value once the vertical code and the u_c coordinate are known. This results in a speeding-up of the 3D reconstruction process.

More in detail, Fig. 4.14 explains how this look-up table has been designed. Basically, it contains as many rows as the number of scanline, which coincides with the number of rows of the rectified camera and projector images. While the number of columns coincides with the number of indexable vertical coordinates: for the prototype of 3D scanner, a sequence of 10 vertical patterns are projected so 1024 are the columns of the look-up table. The element (i, j) of the look-up table contains the vertical coordinate u_p associated with the camera point (j, i). In other words given the camera point (j, i) and the look-up table *tab*, the corresponding projector point is given by $(tab[i, j], i)$. All the entries of this look-up table are filled only once by using formulas (4.19) and this should be intended as an offline process, like calibration. Then during the online process the search for correspondences becomes a simple look-up table access.

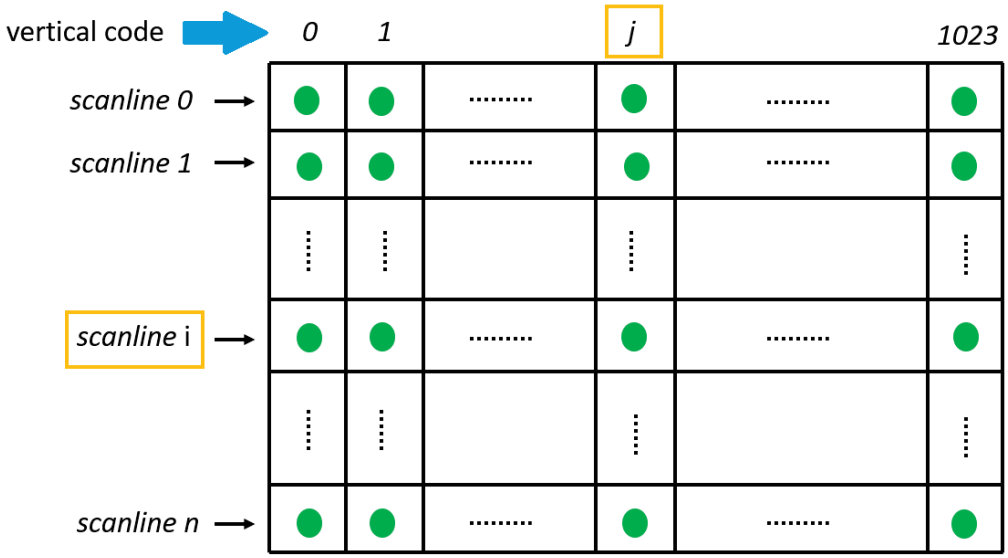


Figure 4.14: Structure of the look-up table

4.3.6 From disparities to depth

Once disparities for all camera pixels has been computed, the next steps in order to retrieve the 3D points of the scene are the same illustrated for a rectilinear stereo rig in Chapter 2.

What is new in this case is the possibility to rely on the disparity-to-depth mapping matrix Q returned by the `cv::stereoRectify()` function. As the name suggests, for each camera pixel (x, y) and the corresponding disparity $d = disparity(x, y)$, this 4x4 matrix allows to directly compute the corresponding 3D point:

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = Q \cdot \begin{bmatrix} x \\ y \\ disparity(x,y) \\ z \end{bmatrix} \tag{4.20}$$

Notice that in this case homogeneous coordinates are considered. As result matrix Q encapsulates the triangulation formulas seen for a rectilinear stereo rig.

4.3.7 Final considerations

The implementation of this second rectification strategy on the prototype of 3D scanner shows how it is possible to use a pair of rectification homographies, suitable to rectify a stereo rig, also in this slightly different scenario. In this case the homography transformations have been computed through the `cv::stereoRectify()` function, but there is also the possibility to get the homographies by other approaches and to apply them in this rectification strategy without no changes in the pipeline. This last observation makes this rectification strategy easily adaptable to further improvements. In addition this chapter has strongly highlighted the effective similarity between stereo systems and structured light systems.

Chapter 5

Experimental results

Implementing the rectification on the prototype of 3D scanner for sure implies a speeding up in the acquisition times, but up to now, nothing has been said about how the accuracy changes. For this reason, this chapter will present a qualitative and quantitative comparison between some 3D reconstructions taken with and without rectification. Moreover, this comparison will be also enriched by some interesting observations related to the point clouds obtained by the rectified 3D scanner, and by some improvements which derive from these observations.

5.1 Experimental setup

Before presenting the experiments it is worth spending some words explaining how the projector-camera system has been configured in order to perform the experimental acquisitions. First of all, the focus distance has been fixed to roughly 450 mm. Then the overall system has been calibrated acquiring 32 different checkerboard orientations. The calibration data have been already illustrated in the previous chapter, while the obtained reprojection errors are reported here: 0.21 pixels for the camera, 0.47 pixels for the projector, and 0.67 pixels for the stereo pair. Once the calibration has been completed, the prototype has been tested by acquiring first of all two models of geometric solids, and then some Lego bricks. The main advantage of Lego bricks is the availability of very accurate CAD models that can be used as reference when analyzing the point clouds returned by the prototype. In order to get the best results all the acquisitions have been performed using Gray code and the decoding method with complementary patterns.

5.2 Qualitative evaluation

5.2.1 Rectified 3D scanner

Fig. 5.1 presents the main results obtained from the prototype after having implemented the second rectification strategy, that can be summarized in:

- **disparity map** This map visualizes for each camera pixel the disparity value as a gray scale intensity. As known form the theory, disparity is inversely proportional to the distance, meaning that a point with large disparity is a very close point, while a small disparity indicates a far point. In this case, large disparity are visualized by a high intensity value of the pixel while small disparities by a lower intensity value. Therefore, higher intensity points are the closer ones.
- **depth map** This map visualizes for each camera pixel the associated depth, namely the Z coordinate of the 3D point. Even in this case, as in the disparity map, the depth information is associated to a gray scale intensity: higher intensity points are the closer ones. In this case, the range of depths to be visualized has been manually chosen so that only the reconstructed object appears in the depth map, without the background.
- **point cloud** By putting together the information coming from disparity and depth maps, namely the 3D points coordinates, and information coming from the camera image, namely the RGB components for each pixel, it is possible to obtain a point cloud which represents the 3D reconstruction of the object. In the image it is possible to appreciate the 3D reconstruction from two different point of views.

The map visualizing the vertical code associated with each camera pixel, obtained after the binarization process has been already reported in Fig. 4.12 so for this reason, it is not reported here. An important observation about the obtained results is related to the fact that, during the triangulation process, it is considered as the world reference frame not the original camera frame but the rectified one. Therefore, all the 3D points coordinates are measured with respect to the rectified camera frame and, as consequence, the 3D reconstructions appear a little bit rotated with respect to the original images acquired from the camera.

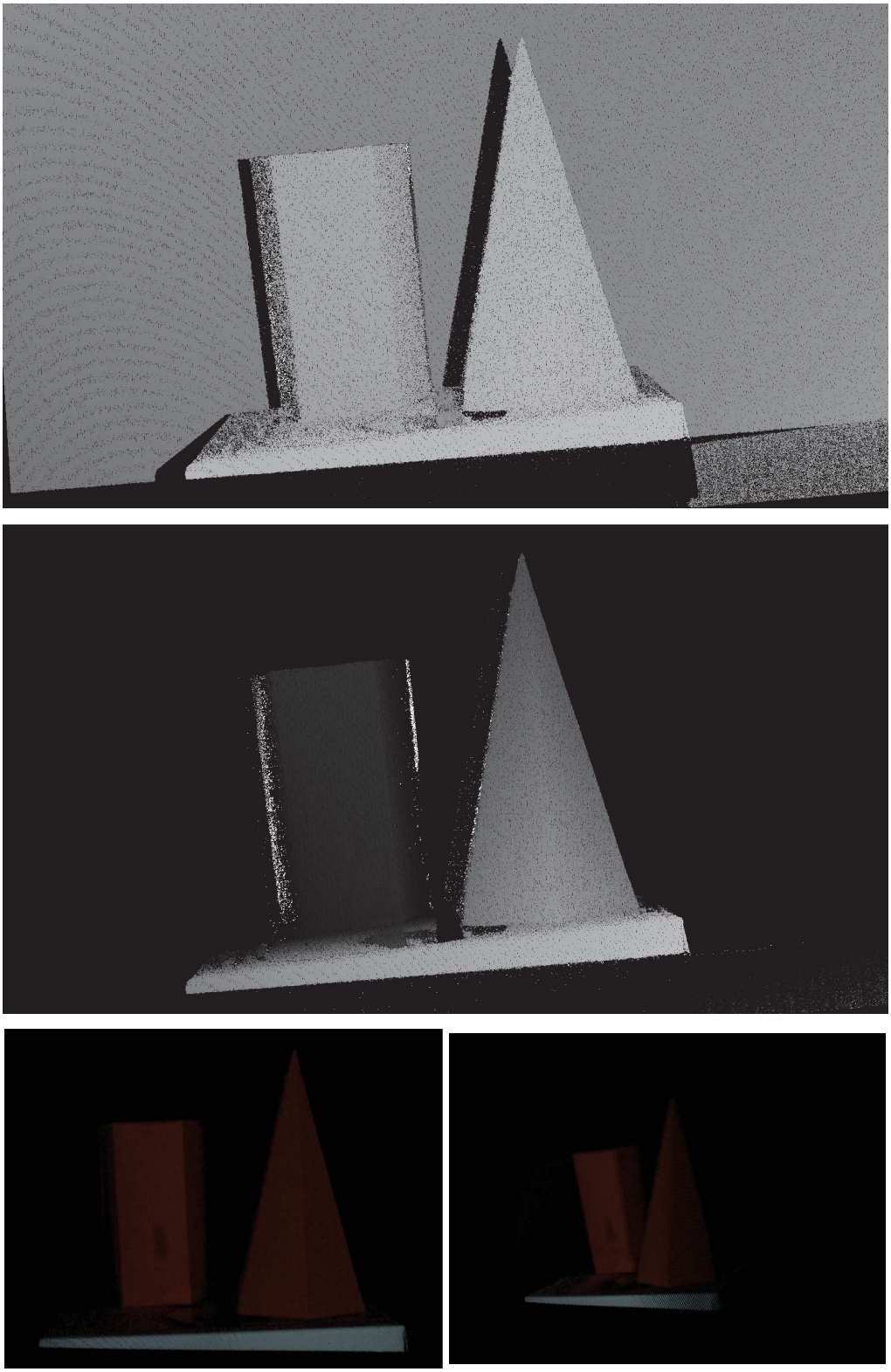


Figure 5.1: Rectified structured light 3D scanner results

From what can be seen, the results obtained from this acquisition are quite accurate from a qualitative point of view. What could seem a little bit unusual is the presence of a sort of quantization in the point cloud. In other words, the points composing the point cloud seem to be distributed according to quantized levels and not in a continuous way along the surface of the object. This fact is more evident when seeing the point cloud sideways. A useful tool in other to better investigate this artifact is the use of the *CloudCompare* software. This open source software provides a set of basic tools for manually editing and rendering 3D point clouds and triangular meshes. It also offers various advanced processing algorithms among which methods for performing projections, registration, distance computation, statistics computation, segmentation, geometric features estimation, ecc. More details about these and other functionalities offered by this software can be found in the tutorials provided by the official website [4]. In this case, Fig. 5.2 shows how the point cloud returned by the 3D scanner appears in *CloudCompare*. By zooming into the point cloud, as shown in Fig. 5.3, it is possible

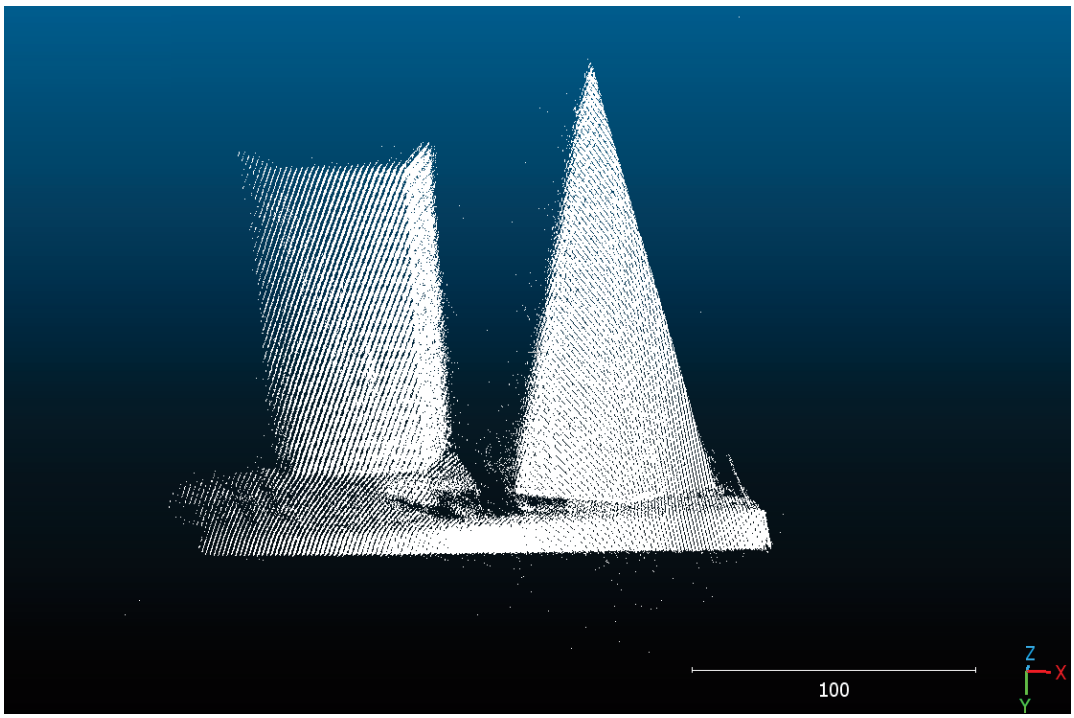


Figure 5.2: Returned point cloud view form *CloudCompare*

to perceive better this discretization effect. We have observed that this effect is mainly due to the low resolution of the projector; actually in this second

rectification strategy horizontal correspondences are identified by projecting vertical stripes while vertical correspondences are identified by rectification and so by using only calibration data. This explains why there is a different resolution between the green and yellow lines. These lines of course coincide with the x and y axis of the rectified projector frame.

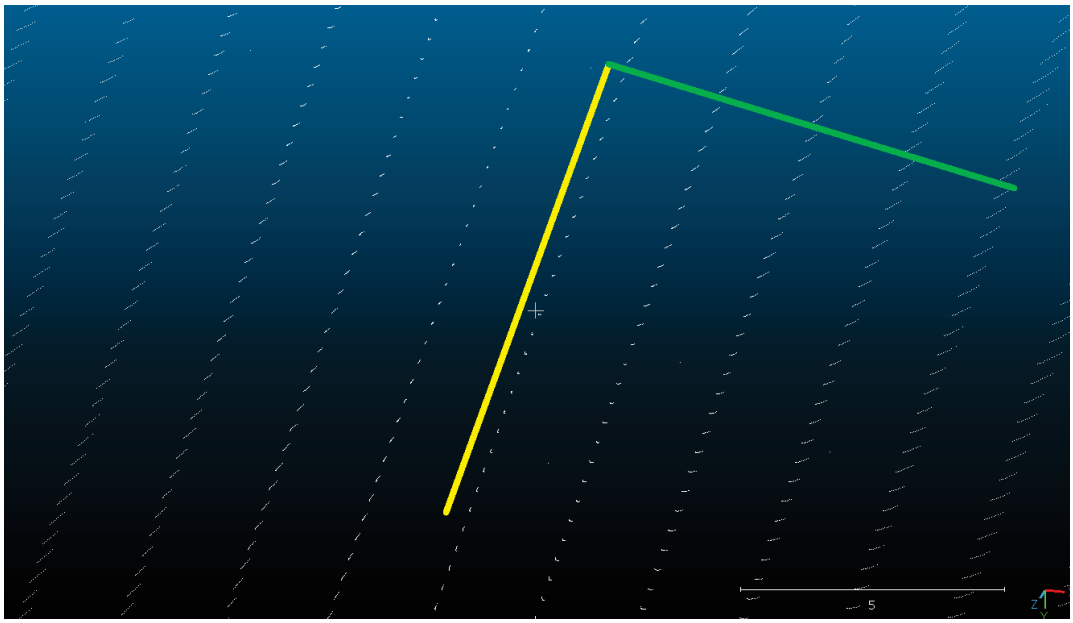


Figure 5.3: Discretization effect in the point clouds

Another effect that comes out from the analysis of the point cloud is the fact that the point cloud in turn is composed by roughly three surfaces one close to the other, when in the ideal case there should be only one surface, and this can be appreciated by Fig. 5.4. This effect can be motivated by basic principles of epipolar geometry: since the projector size is smaller than the camera size, it is expected that one projector point is associated to more camera points, and in particular these points lies on the same projection line passing through the centre of the projector, as illustrated in Fig. 5.5 where camera and projector are seen as a common stereo pair. These points, lying on the same projection line, creates these multiple surfaces. In order to understand how many camera points are associated to one projector point it is sufficient to check the number of points on the same scanline that have been decoded with the same value: it results that one projector point is associated to 3 camera points on average, and this is consistent with Fig. 5.4 where 3D points are grouped as clusters of 2-3 elements. By analyzing the camera

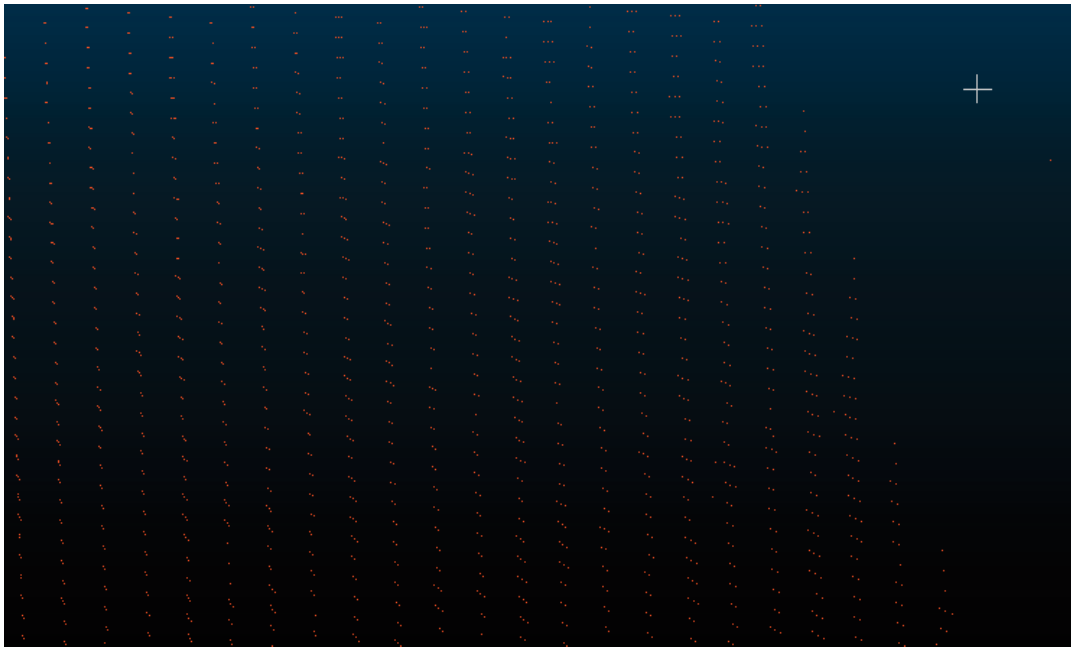


Figure 5.4: More camera points associated to the same projector point

image related to the projection of the pattern associated with the least significant bit it is possible to end up to the same result: each vertical stripe in the camera image has a width of about 5-6 pixels, but since this is a Gray code, this width has to be divided by two. As consequence, each code value is associated to roughly 2-3 pixels.

A possible solution in order to overcome this problem consists in computing the centroid for each group of camera pixels that are associate to the same projector pixel and then compute the point cloud considering only the centroids. In Fig. 5.6 are depicted the obtained results after having applied this solution. What can be immediately observed from these results is a strong reduction in the number of 3D points of the point cloud, in fact in this case the triangulation process involves a number of camera points that coincides with the number of projector points. As result the obtained point cloud contains about one third of the points present in the original result. This effect can be seen also from the disparity and depth maps where the triangulated camera points are spread in all the full resolution camera image creating a sort of salt and pepper effect. An alternative method that could be applied for solving the problem of different resolution between camera and projector is to initially apply a linear transformation in the camera image in

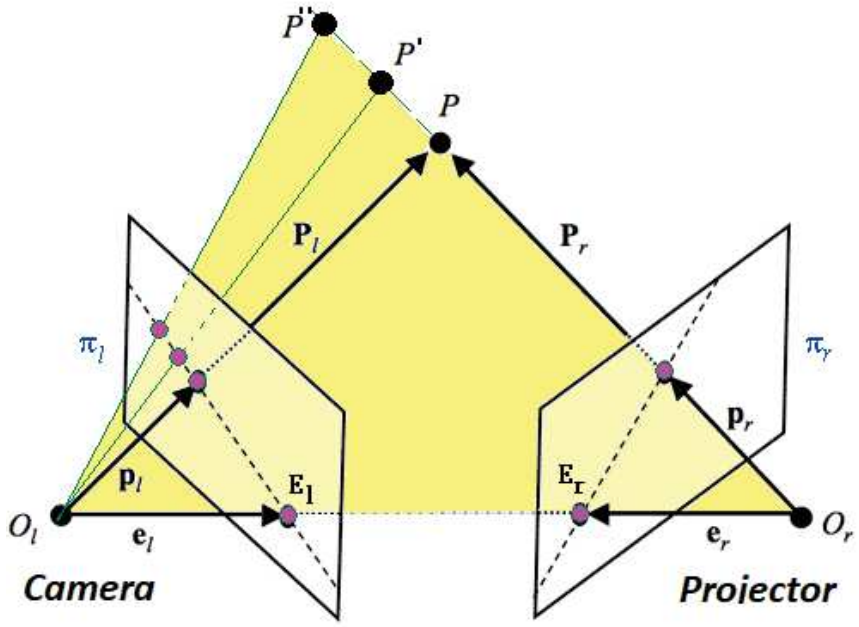


Figure 5.5: Epipolar geometry: more camera points associated to the same projector point

order to make it of the same size of the projector image and then implement the second rectification strategy: in this way there will be a point-to-point correspondence between camera and projector and in particular there will be no salt and pepper effect, being the camera image of smaller resolution. Of course the results in term of point cloud will be the same.

5.2.2 A comparison with non-rectified 3D scanner

For completeness in Fig. 5.7 are reported the same results obtained for the non-rectified 3D scanner. In this case, being the camera and projector image not aligned it is impossible to get a disparity map and this is the reason why the provided results contains the camera image with the decoded vertical codes in place of the disparity map. Obviously, for the non-rectified 3D scanner, also the horizontal code is necessary. Note that the triangulation in this case has been performed by considering as world frame the original camera frame meaning that all the 3D points coordinates are measured respect to the camera frame.

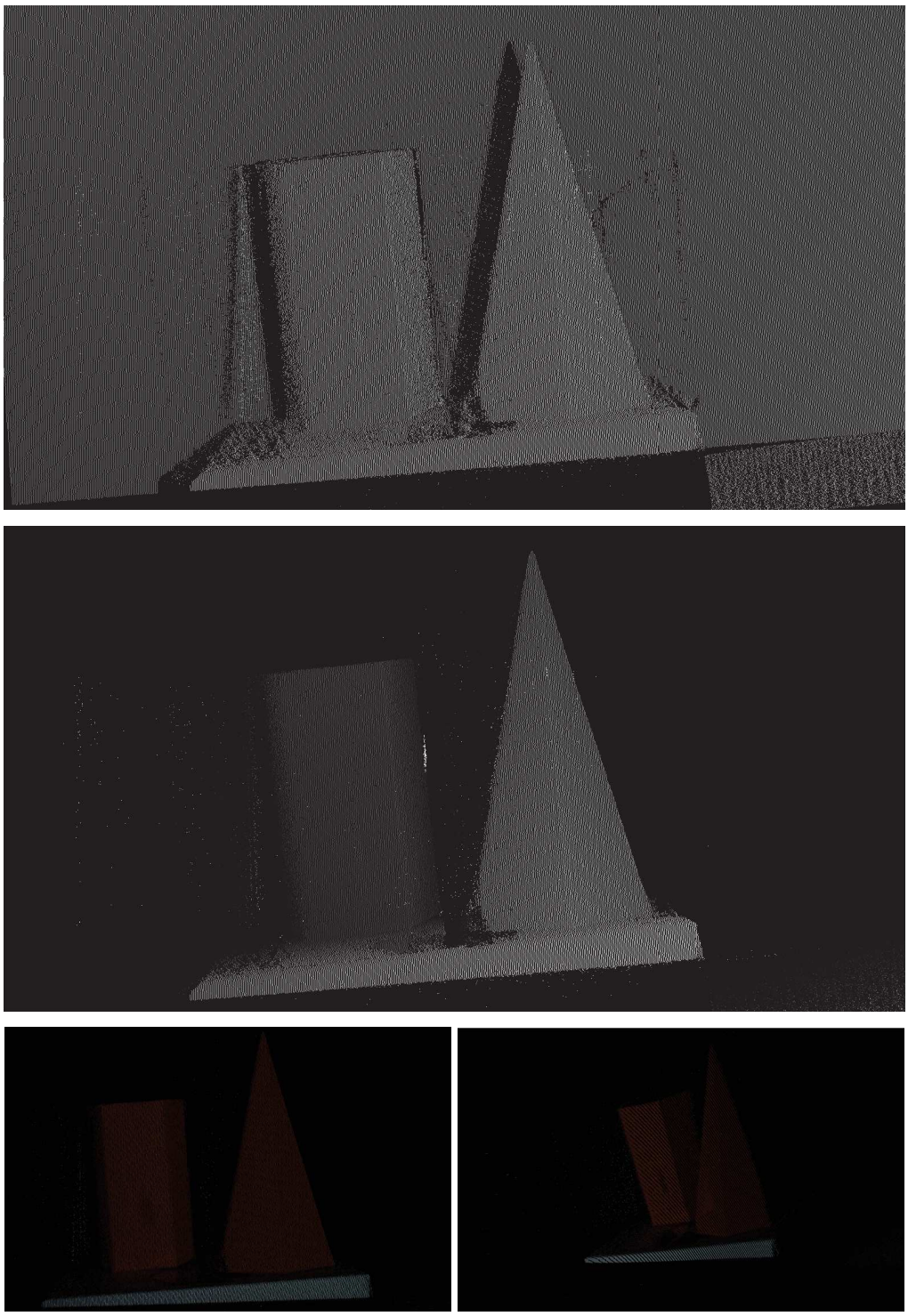


Figure 5.6: Results with the rectified structured light 3D scanner after clustering

Differently from the rectified 3D scanner, in this case horizontal correspondences are identified by projecting vertical stripes and vertical correspondences are identified by projecting horizontal stripes. This means that the returned point cloud in this case has no discretization effect as the there is no different resolution between vertical and horizontal coordinates. On the other end, also the non-rectified 3D scanner suffers of the same problem previously described about the different resolution between the camera and projector: even in this case more camera pixels are associated with the same projector pixel, and so even in this case the 3D points are grouped in clusters. Of course, the presence of the discretization effect makes this artifact more evident in the rectified case with respect to the non-rectified one.

Leaving out all these observations, there is not a strong difference in accuracy between the point clouds obtained with and without rectification. Of course, in the original 3D scanner without rectification the returned point clouds are slightly more accurate. However, the acquisition times are almost duplicated for the necessity of projecting another sequence of patterns composed by horizontal binary stripes.

5.3 Quantitative evaluation

In the previous section, a qualitative comparison has been presented between the rectified and non-rectified 3D scanner. From the analysis, it results that, even if the non-rectified 3D scanner provides the best results, also in the rectified case the results are convincing. In addition, some methods in order to improve the correctness of the point clouds returned by the rectified 3D scanner have been proposed. Aiming to a quantitative evaluation of the experimental results, it is necessary to keep into account the repeatability and accuracy of the 3D reconstructions. Checking the repeatability is quite simple, it requires only acquiring more times the same scene and to compare the returned point clouds: the less the distance among the results, the bigger the repeatability of the system. Regarding the accuracy, it is necessary to acquire objects with known 3D shapes and check how close is the returned point cloud to the real object. This section in particular describes the accuracy evaluation of the 3D reconstructions and the improvement in the acquisition times as a consequence of the rectification.

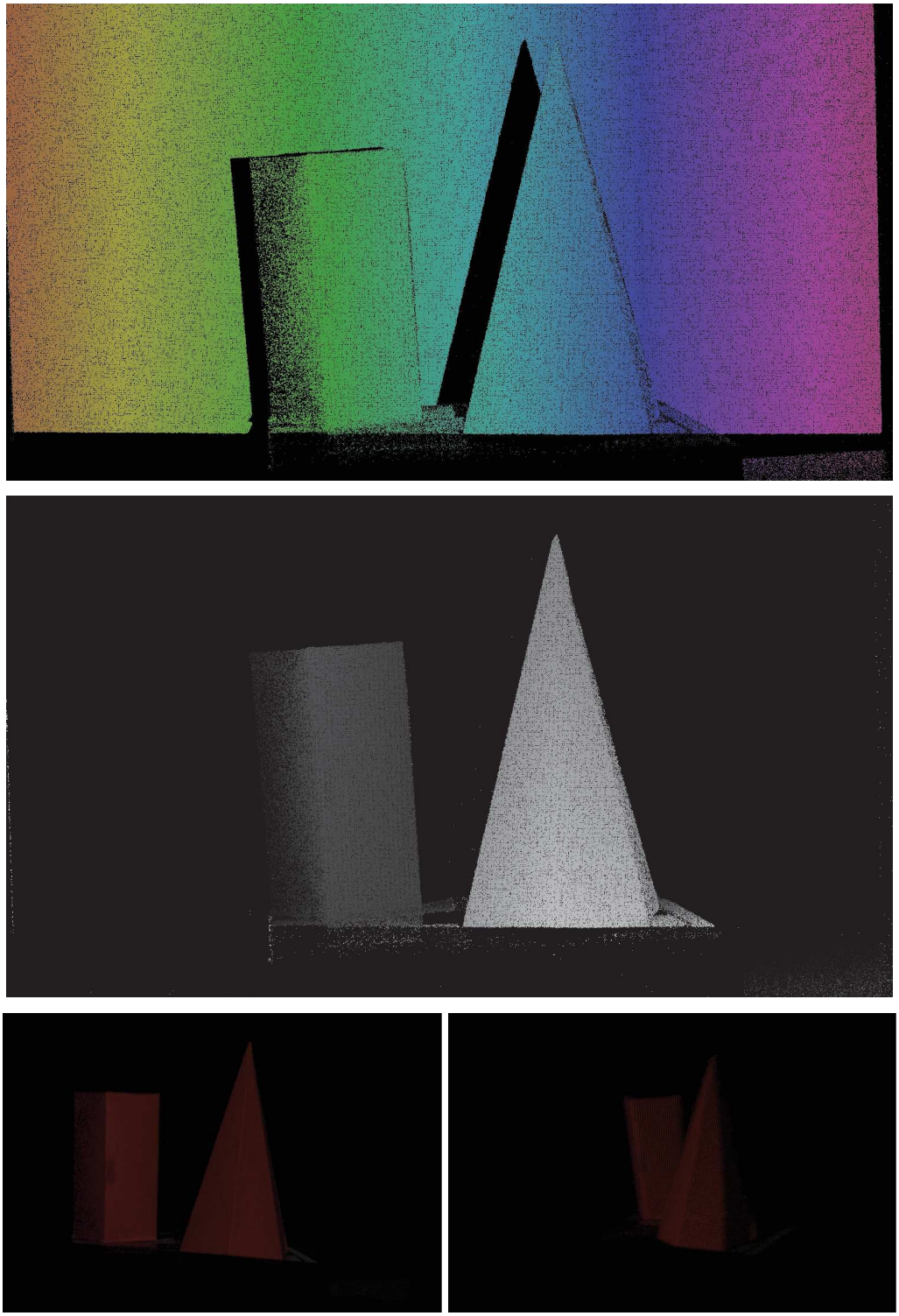


Figure 5.7: Non-rectified structured light 3D scanner results

5.3.1 Accuracy

There are many ways in order to evaluate the accuracy of the available prototype. A possible approach consists in acquiring some three-dimensional objects and comparing their CAD model with the returned point cloud. To this aim, some Lego bricks have been acquired for this evaluation. In general, Lego bricks are realized with very high precision and so they are a good candidate for this kind of approach.

In particular, the idea is to acquire four different Lego bricks of different colors, each one from three different orientations. Once they have been reconstructed, the related point clouds have been modified in order to remove the support of the Lego bricks and other spurious elements of the scene. By using again *CloudCompare* each point cloud has been aligned in an almost automatic way to the related CAD model and then distances between points of the two three-dimensional structures have been computed. The main evaluation index, in this case, is the *Root Mean Square Error* (RMSE), given by the square root of the arithmetic mean of the square of the distances. In this case, the distances play the role of an error between the obtained point cloud and the CAD model. It is worth mentioning that having cut each point cloud in order to select only the Lego brick, some imperfections may have been eliminated; especially when in the 3D reconstruction some points are projected at very large distances from the surface of the object. This means that the RMSE measured for each acquisition must be intended as an underestimate of the real error. The tables 5.1 and 5.2 show respectively the measured RMSE values and the dimensions for the point clouds for the acquired Lego bricks both for the rectified and non-rectified 3D scanner. The obtained results are consistent with the qualitative analysis discussed in the previous section. The use of rectification simplifies the correspondences problem from a 2D to a 1D search but at the same time introduces additional error sources: first of all the error in the alignment between camera and projector images, which is small in general but not negligible; but also the homography transformation for the camera image that is applied by mean of interpolation methods, and this results in a distortion of the original camera image. The non-rectified 3D scanner instead relies only on the vertical and horizontal patterns for solving the correspondence problem and as consequence, the point clouds

will have higher accuracy. What is interesting to observe from the two tables is the fact that the actual difference between rectified and non-rectified 3D scanners is rather small, especially from the accuracy point of view. Notice that the difference in the mean RMSE is only about 0.04 mm. Therefore, this loss in accuracy for the rectified prototype is acceptable, considering the remarkable improvement in the acquisition times.

3D scanner	**Object**	**RMSE**		
		1^{st} *view*	2^{nd} *view*	3^{rd} *view*
Rectified	lego 2x3 white	0.34 mm	0.49 mm	0.41 mm
	lego 2x4 blue	0.46 mm	0.53 mm	0.49 mm
	lego 2x4 red	0.47 mm	0.57 mm	0.47 mm
	lego 2x4 yellow	0.46 mm	0.55 mm	0.39 mm
		mean: 0.47 mm		
Non-rectified	lego 2x3 white	0.32 mm	0.47 mm	0.39 mm
	lego 2x4 blue	0.38 mm	0.51 mm	0.33 mm
	lego 2x4 red	0.44 mm	0.54 mm	0.44 mm
	lego 2x4 yellow	0.43 mm	0.53 mm	0.37 mm
		mean: 0.43 mm		
Removing multiple surfaces				
Rectified	lego 2x3 white	0.18 mm	0.26 mm	0.20 mm
	lego 2x4 blue	0.24 mm	0.25 mm	0.19 mm
	lego 2x4 red	0.30 mm	0.28 mm	0.29 mm
	lego 2x4 yellow	0.27 mm	0.27 mm	0.21 mm
		mean: 0.25 mm		

Table 5.1: Quantitative evaluation of the reconstructed Lego bricks

The two tables present also a third section dedicated to the rectified 3D scanner in which has been experimented the solution previously described for removing the multiple surfaces effect. As expected, the obtained point

3D scanner	Object	Number of points		
		1^{st} *view*	2^{nd} *view*	3^{rd} *view*
Rectified	lego 2x3 white	28327	24316	21921
	lego 2x4 blue	32671	29470	31806
	lego 2x4 red	30412	26116	22822
	lego 2x4 yellow	28367	25584	28617
			mean: 27535	
Non-rectified	lego 2x3 white	27690	25278	22112
	lego 2x4 blue	26983	29888	31049
	lego 2x4 red	22620	26998	18627
	lego 2x4 yellow	27564	27880	33064
			mean: 26646	
Removing multiple surfaces				
Rectified	lego 2x3 white	10111	8592	7577
	lego 2x4 blue	11317	10760	12105
	lego 2x4 red	11107	9763	9354
	lego 2x4 yellow	10363	9789	12101
			mean: 10245	

Table 5.2: Point clouds dimension for the reconstructed Lego bricks

clouds, in this case, contain a lower number of points with respect to the other two cases, but the results from the accuracy point of view are quite interesting: in this case, the average RMSE is smaller not only respect to the original rectified scanner but also respect to the non-rectified one. This is not a surprising fact since also the non-rectified scanner suffers from the multiple surfaces effect, and this is something that in some way tends to deteriorate the accuracy of 3D reconstructions. In addition, it is important to clarify that the RMSE values obtained for this third case are a very large underestimate of the real error: from the qualitative analysis it is possible

to observe how this third solution tends to produce lots of outliers in the final 3D reconstructions and all these points during this quantitative analysis have been removed together with the support of Lego bricks.

Aiming to provide a more intuitive idea of what all the numbers reported in these tables means, consider Fig. 5.8 in which are reported the point clouds associated to the first view of the white Lego brick. The color map in this case visualizes the distance from the real CAD model: the color scale starts from blue, which corresponds to points with a lower distance from the reference, moving gradually to green, yellow, and finally red, indicating points with greater error. Even from this figure, it is possible to observe how moving from the rectified to the non-rectified scanner, the accuracy tends to improve. In particular, in the rectified case there are not so many yellow and red points with respect to the non-rectified case, meaning that the accuracy between the two is not so different. The third row in this figure is related to the rectified 3D scanner after having removed the multiple surfaces effect. From what it can be seen, the point cloud is very poor in terms of the number of points but at the same time more accurate with respect to the other two cases, confirming what was already observed from the numerical results.

5.3.2 Acquisition times

Many times in describing this master thesis project it has been mentioned the speeding-up of the acquisition times as a consequence of rectification. This subsection presents a comparison between non-rectified and rectified 3D scanner in order to approximately quantify this improvement for the available prototype.

The DLP2000 module mounted in the prototype takes about 250 ms for projecting a full-resolution image. Given the resolution of 640x360 pixels of the projector, for a pattern sequence able to encode every single row/column, it is necessary to consider 9/10 bit and so as many images. For the non-rectified 3D scanner using the complementary pattern approach for decoding, it is necessary to project 38 images for decoding both rows and columns and this leads to an acquisition time of roughly 10 seconds, considering also the processing after the projection of the pattern. In the rectified 3D scanner

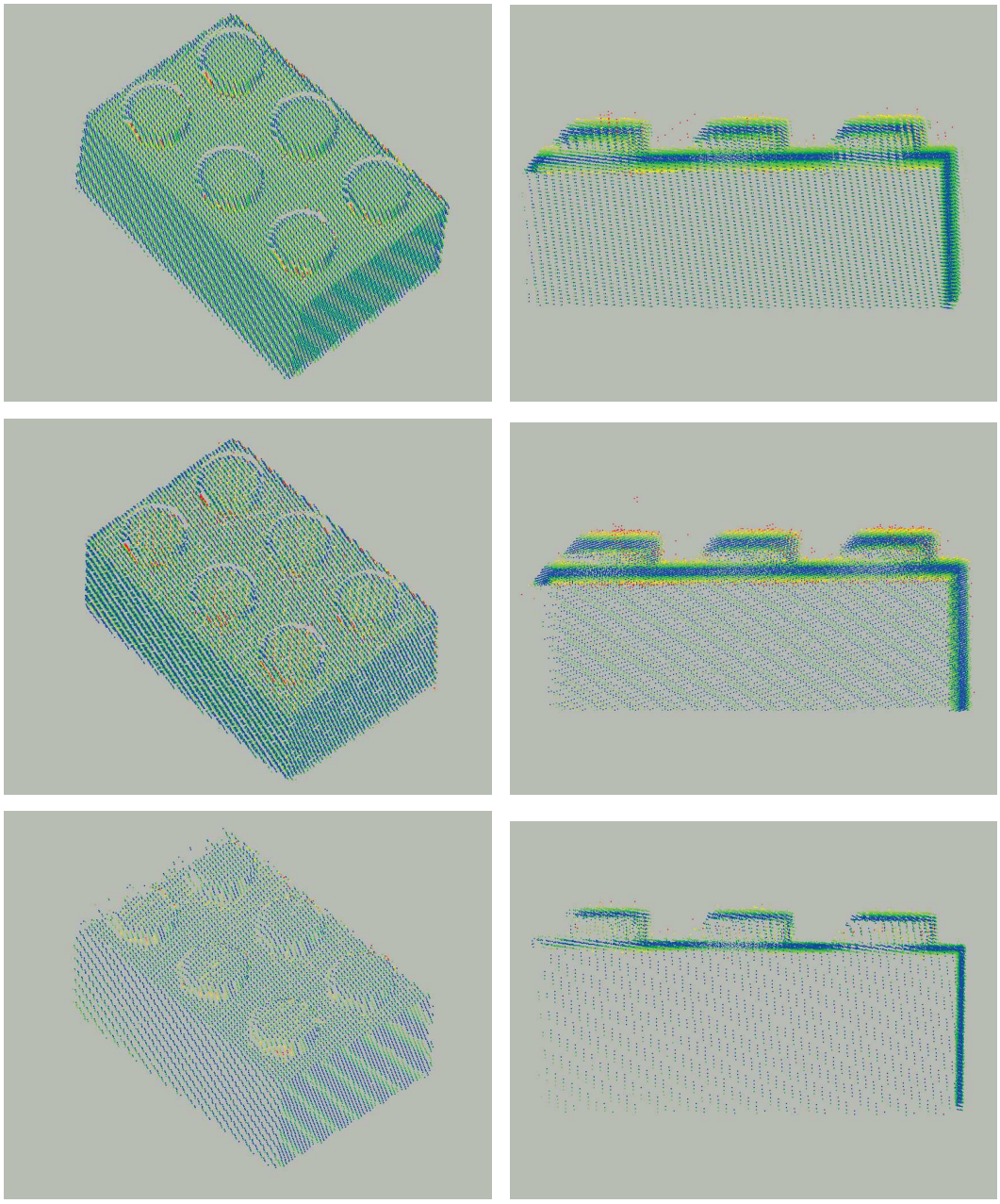


Figure 5.8: 3D reconstruction of a Lego brick: in the first raw by the rectified 3D scanner, in the second raw by the non-rectified 3D scanner, in the third row by the rectified 3D scanner after removing the multiple surfaces effect

instead, it is sufficient to project only half of the patterns for encoding only the columns. Theoretically, this should divide by two the acquisition time; in practice, by taking into account also the processing part, the acquisition time is about 6 seconds. In any case, this represents a remarkable improvement, considering also that the accuracy obtained in both cases is very similar.

Chapter 6

Conclusion

The core of this master thesis is the rectification of a structured light sensor, in order to improve its performance in the same way as a typical stereo vision system. Related to this topic two strategies have been proposed: the first one tries to achieve the desired horizontal alignment between camera and projector planes just by acting on the camera, while the second one involves also the projector considering it, for some aspects, as another camera. The first approach is not so difficult to be implemented since it does not require a strong re-adaptation of the already implemented pipeline of the structured light sensor. On the other hand, in several cases the alignment between the camera and the projector could imply a strong distortion of camera images resulting finally in a very poor accuracy. The second approach instead by acting also on the projector gives the opportunity to highlight an interesting fact: the possibility to apply a rectification technique typical of a stereo vision system also to a structured light scanner. In other words, a pair of homographies suitable to rectify two stereo cameras could be used also for a projector-camera pair, seeing the projector as a camera. Of course, the projector actually is not a camera and this must be taken into account in the pipeline of 3D reconstruction. To make things more concrete, this second rectification strategy has been implemented on a prototype of binary coded structured light 3D scanner. This allows showing how the rectification pipeline must be adapted in order to cope with practical problems related to the particular hardware components mounted on the scanner. For instance, the main problem related to the prototype is the great difference in resolution between the camera and the projector which affects not only the rectification pipeline but also the accuracy of the final 3D reconstruction. The obtained results after having rectified the prototype of the structured

light 3D scanner have been compared with the results obtained by the same prototype without rectification. Since the beginning, it has been expected that the rectification reduces the accuracy of the obtained point clouds. But the interesting fact is that, from an analysis of the obtained results, there are no big differences between the rectified and the unrectified prototype, even if the second one remains the most accurate sensor between the two. Besides this fact, the rectification leads to a remarkable speeding-up of the acquisition times, not only in the patterns projection but also in the processing. In summary, the obtained results show the great improvements given by the rectification of a structured light sensor.

6.1 Future developments

The performances obtained for the rectified prototype are still far from the state of art. Just to make a quick comparison, the 3D industrial camera *Zivid 2* [36] is able to produce very accurate 3D reconstructions with an acquisition time from 100 ms to almost 1 s. More details about this kind of sensor can be found in the manufacturer's official website. However, there are many additional improvements that can be achieved on this prototype.

From the rectification point of view, a possible improvement could be the use of new pairs of rectification homographies: in this thesis the rectification homographies used for the prototype comes from the *OpenCV* function `cv::stereoRectify()`, but there exists many other approaches for obtaining them. In this context, a prominent example is the method proposed in [13] which takes into account also the perspective distortion. Rectifying transformations, in general, introduce perspective distortion on the obtained images, which shall be minimised to improve the accuracy of the following algorithm dealing with the correspondence problem. The search for the optimal transformations is usually carried out relying on numerical optimisation. This work proposes instead a closed-form solution for the rectifying homographies that minimise perspective distortion.

The accuracy of the rectification for the prototype depends also on the calibration process. This motivates the implementation of new calibration methods that hopefully are able to provide more precise results. For the available prototype, it could be an interesting tentative implementing the

calibration method proposed in the SLStudio project [33] that is as valid as the one adopted in this thesis.

Even if this thesis deals principally with rectification, the performance of the prototype of 3D scanner can be further improved in many other ways. First of all, a significant improvement both from the accuracy and speed point of view could be obtained by changing the hardware components: as seen many times in this thesis the light projector is the weaker element in the device; as consequence, having a quicker projector with a resolution as close as possible to the one of the camera would have the double effect of drastically reduce the acquisition times and increase the quality of the 3D reconstruction, avoiding the multiple surfaces effect.

From the software point of view, it could be interesting to experiment with other coding strategies and to study eventually new binarization techniques that do not require the projection of additional images. Focusing only on the accuracy, a considerable improvement can be obtained by performing some filtering actions on the returned point cloud. Regarding this topic, a first example could be to implement a *noise filter* that keeps only the points with a sufficiently high SNR value, where the SNR is computed considering as signal the light emitted by the projector and as noise, all the light sources coming from the surrounding environment. A second example could be an *outlier filter* that considers for each 3D point a spherical neighborhood centered on it and if in this region there are at least N points, with N parameter of the filter that must be set, the point is kept otherwise it is removed meaning that it is probably an outlier. Besides these two examples, there are many other filters that could be applied in order to enhance the quality of a 3D reconstruction.

Bibliography

[1] Henrik Andreasson et al. "Sensors for Mobile Robots". In: *Encyclopedia of Robotics* (preprint).

[2] Tyler Bell, Beiwen Li, and Song Zhang. "Structured Light Techniques and Applications". In: *Wiley Encyclopedia of Electrical and Electronics Engineering* (2016).

[3] Angelo Cenedese. *Robotics Control 2 Lecture Notes*. 2022.

[4] *CloudCompare Tutorial videos*. URL: https://www.danielgm.net/cc/.

[5] Peter Corke. *Robotics, Vision and Control*. Vol. 2. Springer, 2017.

[6] *cv::stereoRectify()*. URL: https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga617b1685d4059c6040827800e72ad2b6.

[7] W. Forstner and B. P. Wrobel. *Photogrammetric Computer Vision*. Vol. 2. Springer, 2016.

[8] Jason Geng. "Structured-light 3D surface imaging: a tutorial". In: *Advances in Optics and Photonics* 3.2 (2011).

[9] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Vol. 2. Cambridge Unversity Press, 2004.

[10] Richard I. Hartley. "Theory and Practice of Projective Rectification". In: *International Journal of Computer Vision* (1999).

[11] Chih-Hung Huang. "An improved method for the binarization in structured light 3D scanning systems". In: (2007).

[12] Changsoo Je, Sangwook Lee, and Rae-Hong Park. "High-Contrast Color-Stripe Pattern for Rapid Structured-Light Range Imaging". In: *Lecture Notes in Computer Science* 3021 (2004).

[13] Pasquale Lafiosca and Marta Ceccaroni. "Rectifying Homographies for Stereo Vision: Analytical Solution for Minimal Distortion". In: *Intelligent Computing*. Vol. 507. Springer International Publishing, 2022, pp. 484–503. ISBN: 978-3-031-10464-0. DOI: 10.1007/978-3-031-10464-0_33. URL: https://doi.org/10.1007/978-3-031-10464-0_33.

[14] Tuotuo Li, Hongyan Zhang, and Jason Geng. "Geometric calibration of a camera-projector 3D imaging system". In: *2010 25th International Conference of Image and Vision Computing New Zealand* (2010).

[15] Charls Loop and Zhengyou Zhang. "Computing Rectifying Rectifying Homographies for Stereo Vision". In: *Computer Society Conference on Computer Vision and Pattern Recognition* (1999).

[16] Y. Ma et al. *An invitation to 3D Vision*. Springer, 2004.

[17] M. Mihelj et al. *Robotics*. Vol. 2. Springer, 2019.

[18] Daniel Moreno and Gabriel Taubin. "Simple, Accurate, and Robust Projector-Camera Calibration". In: *Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission* (2012).

[19] Vincent Nozick. "Multiple View Image Rectification". In: *Fist International Symposium on Access Spaces (ISAS)* (2011).

[20] N. Pears, Y. Liu, and P. Bunting. *3D Imaging, Analysis and Applications*. Vol. 2. Springer, 2020.

[21] Mattia Piccoli. "Ricostruzione 3D tramite proiezione di luce strutturata". 2021.

[22] Yubo Qiu et al. "Inverse Rectification for Efficient Procam Pattern Correspondence". In: *Computer Vision Foundation* (2020).

[23] Joaquim Salvi, Jordi Pagès, and Joan Batlle. "Pattern codification strategies in structured light systems". In: *Pattern Recognition* 37.4 (2004).

[24] Daniel Scharstein and Richard Szeliski. "High-Accuracy Stereo Depth Maps Using Structured Light". In: *Computer Society Conference on Computer Vision and Pattern Recognition* (2003).

[25] Yongcan Shuang and Zhenzhou Wang. "Active stereo vision three-dimensional reconstruction by RGB dot pattern projection and ray intersection". In: (2020).

[26] Yongcan Shuang and Zhenzhou Wang. "Active stereo vision three-dimensional reconstruction by RGB dot pattern projection and ray intersection". In: (2020).

[27] Henrik Stewénius, Christopher Engels, and David Nistér. "Recent developments on direct relative orientation". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 60 (2006).

[28] Vignesh Suresh. "Calibration of structured light system using unidirectional fringe patterns". PhD thesis. 2019.

[29] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Vol. 1. Springer, 2010.

[30] Gabriel Taubin, Daniel Moreno, and Douglas Lanman. "3D scanning for personal 3D printing: Build your own desktop 3D scanner". In: *ACM SIGGRAPH 2014 Studio* (2014).

[31] Jakob Wilm, Oline Olesen, and Rasmus. Larsen. "Accurate and Simple Calibration of DLP Projector Systems". In: *Proceedings of SPIE - The International Society for Optical Engineering* 8979 (2014).

[32] Jakob Wilm, Oline V. Olesen, and Rasmus Larsen. "SLStudio: Open-source framework for real-time structured light". In: *4th International Conference on Image Processing Theory, Tools and Applications (IPTA)* (2014).

[33] Jakob Wilm, Oline Vinter Olesen, and Rasmus Larsen. "Accurate and Simple Calibration of DLP Projector Systems". In: *Proceedings of SPIE, the International Society for Optical Engineering* (2014).

[34] Song Zhang and Peisen Huang. "Novel method for structured light system calibration". PhD thesis. 2006.

[35] Z. Zhang. "A flexible new technique for camera calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (2000).

[36] *Zivid Two Specifications*. URL: https://www.zivid.com/zivid-two-specs.