POLITECNICO DI MILANO
Master in Computer Science Engineering
Dipartimento di Elettronica e Informazione

3D Reconstruction from Stereo Video Acquired from Odontoiatric Microscope

Supervisor: Prof. Marco Marcon
Tutor: Dott. Ing. Danilo Maria Martino

Master graduation thesis by:
Leonardo Spatafora

Academic Year 2017-2018

Contents

Contents **3**

List of Figures **5**

Abstract **7**

Sommario **9**

1 Introduction **11**

2 Theoretical Review **13**

2.1 Camera Models . . . 13

2.1.1 Pinhole Model . . . 13

2.2 Camera Calibration . . . 16

2.2.1 Extrinsic Parameters . . . 17

2.2.2 Intrinsic Parameters . . . 17

2.2.3 Distortion in Camera Calibration . . . 17

2.2.4 Single Camera Calibration . . . 18

2.3 Epipolar Geometry . . . 22

2.3.1 Epipole or epipolar point . . . 23

2.3.2 Epipolar line . . . 23

2.3.3 Epipolar plane . . . 23

2.3.4 The Fundamental Matrix [6] . . . 23

2.4 Image Rectification . . . 24

2.4.1 Mapping the Epipole to Infinity . . . 24

2.4.2 Matching Transformation . . . 25

2.4.3 Algorithm Outline . . . 26

2.5 Disparity Map . . . 27

3 Methodology **31**

3.1 Images Acquisition . . . 31

3.2 Compute the Calibration . . . 32

3.2.1 Camera Calibrator . . . 32

3.2.2 Stereo Camera Calibrator . . . 34

3.3 Compute the Rectification 35
3.4 Disparity Map between Stereo Images 36
3.5 3-D Reconstruction of the Scene 37

4 Results 39
4.1 Calibration 39
4.2 Rectification 41
4.3 Disparity Map 42
4.4 3D Reconstruction 44

Conclusion 44

Bibliography 53

List of Figures

1.1 System overview 12

2.1 The all model is called ***Pinhole Model*** (a)sx:3-D view (b)dx:2-D view. 14
2.2 Image *(x, y)* and camera (x_{cam}, y_{cam}) coordinate systems. . . 15
2.3 The Euclidean Transformation. 16
2.4 Negative Distortion, No Distortion and Positive Distortion. . 17
2.5 Tangential Distortion 18
2.6 Example of a calibration pattern 19
2.7 Essencial elements of a Epipolar Geometry 22
2.8 (a)sx:left image (b)dx:right image 28
2.9 A rectified setup viewed from above, showing the two camera centers, $\mathbf{C}_l$ and $\mathbf{C}_r$, and their respective image planes, $\mathcal{I}_l$ and $\mathcal{I}_r$. 28
2.10 Disparity Map 29

3.1 Size of chessboad square 32
3.2 Example of calibration frames 33
3.3 Calibration of Camera 0 33
3.4 The bar graph displays the mean reprojection error per image, along with the overall mean error. The bar labels correspond to the image IDs. The selected bar labels are the images that we delete 34
3.5 Reprojection error 35
3.6 Example of couple of images extract from left and right cameras, before the rectification 35
3.7 Example of a red-cyan anaglyph 36

4.1 Stereo Camera Calibrator tool. The Camera1 refers to the *Camera0* and the Camera2 refers to the *Camera1* 40
4.2 Example of couple of frames extract from left and right cameras, before the rectification 41
4.3 Example of a red-cyan anaglyph 41
4.4 Calculation of the minimum disparity 42

4.5 Calculation of the maximum disparity 42
4.6 Calculation of the maximum disparity after the appearance of the new tool 43
4.7 Disparity map of the frame 466 43
4.8 Disparity map of the frame 588 44
4.9 3D reconstruction of the frame 460 45
4.10 Same 3D reconstruction from different POV 46
4.11 3D reconstruction of the frame 588 47
4.12 Disparity map of different time istant 50

Abstract

The aim of the thesis is to generate a 3D model from couple of images, acquired from dental surgery operation. This is done for teaching puproses because it may help denstist student to understand better crucial part of the operation, one other possible application is to measure parts inside the mouth in case, for example, of possible carcinomas.

After the acquisition of the stereo video from a odontoiatric microscope, we modify the frame of the video in order to make their 3D reconstruction of the scene possible. In particular, after having computed the calibration parameters for both cameras (individually and stereo calibration), we rectify the images in the same time istant take from the two cameras, i.e. with different point of view, and therefore delete the rotation of the second camera with respect to the first one. Then we compute the disparity map so we can compute the depth of the image and finally reconstruct the scene. This process is done for each pair of frames.

This results in a good 3D reconstruction but with some holes in the final images. Infact the 3D is degraded because the two cameras are rotated relative to each other, moreover, the calculated depth values do not correspond to the real depth values, i.e. how much each pixels is far from the camera.

Sommario

Lo scopo della tesi è quello di generare un modello 3D da una coppia di immagini acquisite da un'operazione chirurgica dentistica. Questo è fatto per dei propositi di insegnamento in quanto potrebbe aiutare gli studenti di odontoiatria di capire al meglio le parti cruciali dell'operazione, un'altra possibile applicazione è quella di misurare le parti all'interno della bocca in caso, per esempio, di possibili carcinomi.

Dopo l'acquisizione del video in modo stereo da un microscopio odontoiatrico, modifico i frame del video così che la ricostruzione 3D della scena è possibile. In particolare, dopo aver calcolato i parametri di calibrazione per entrambe le videocamere (sia individualmente che in modo stereo), si rettificano le immagini prese dalle due videocamere nello stesso istante di tempo, cioè con un differente punto di vista, in modo da eliminare la rotazione della seconda videocamera rispetto alla prima. Quindi si computa la mappa di disparità così da poter calcolare la profondità dell'immagine e infine si ricostruisce la scena. Questo viene fatto per ogni coppia di frame.

Questo porta a una buona ricostruzione 3D anche se con qualche buco nell'immagine finale. Infatti il 3D è degradato in quanto le due videocamere sono ruotate una rispetto all'altra, inoltre i valori della profondità calcolati non corrispondono ai valori reali di profondità, cioè a quanto i singoli pixel sono distanti dalla videocamera.

Chapter 1

Introduction

With this work we want to be able to reconstruct, in 3D, the scene captured by two cameras with different point of view. In particular, we use two cameras placed on a dental microscope; in this way we can acquire a video of a dental surgery and help the dentist to observe the operation and to recognize a possible carcinoma.

The thesis is composed of four chapters. In the first one I introduce the mathematical tools, useful to understand the algorithms used in the work. First of all, I explain the principal camera model, the *Pinhole Model*, useful to map a point in the real world to a image plane.
Then we need to known the features of the two cameras, this is done with the *Calibration process*. In particular, thanks to this process, we can known the intrisic parameters of the each camera individually and the extrinsic parameters of the second camera with respect to the first one. The first type of parameters tell us the values of the internal camera parameters, like the focal length and the principal point. The second type tell us the values of the external parameters, like the rotation and the traslation. To compute the calibration we must use a chessboard patter.
After the explanation of the calibration, I explain the *Epipolar Geometry* that is useful to describe the relation and the geometric constraints between two 2D images of the same 3D scene. This is necessary because we use a stereo video, so we capture the same 3D scene with two cameras with different position and orientation. The essential points of this geometry are: the epipolar point, the epipolar line, the epipolar plane and the fundametal matrix; in particular this matrix is the algeabric representation of the epipolar geometry, it relates the corresponding points in stereo images.
As we already said, the two frame, extract at the same time istant from the two cameras, have two diffetent point of view. To solve this problem we need to explain the rectification process, i.e. the process that put two images on the same x axis, whit no difference along the vertical direction.
Finally I explain the disparity map, i.e. the map that tell us the depth of

the 2D images, this map exploit the difference of the same pixel, take from the two images, along the horizontal direction. The disparity map is an image where the darker shades represent lesser shift and the brighter shader represent more shift.

In second chapter, I explain how we build the 3D reconstruction of the scene. First of all we acquire the images from the two cameras, then, we compute the calibration (first individually, then stereo) and how to improve it. After this I explain how we compute the rectification of the two frame and put them in a red-cyan anaglyph and then, with this information, we compute the disparity map. Finally I explain how we compute the 3D reconstruction of the scene and the corrispondence between the distance from the camera in the real world and the values of the depth in the 3D representation, $points3D(:, :, 3)$.

In the next chapter I show the result of the computation for each couple of frame extract from stereo video. In particular I show how the parameters of the MATLAB function, *disparity*, are fundamental for a successful reconstruction and the 3D images of the reconstruction in different time istant.

In the sixth chapter I present the conclusions of this work with possible future developments.

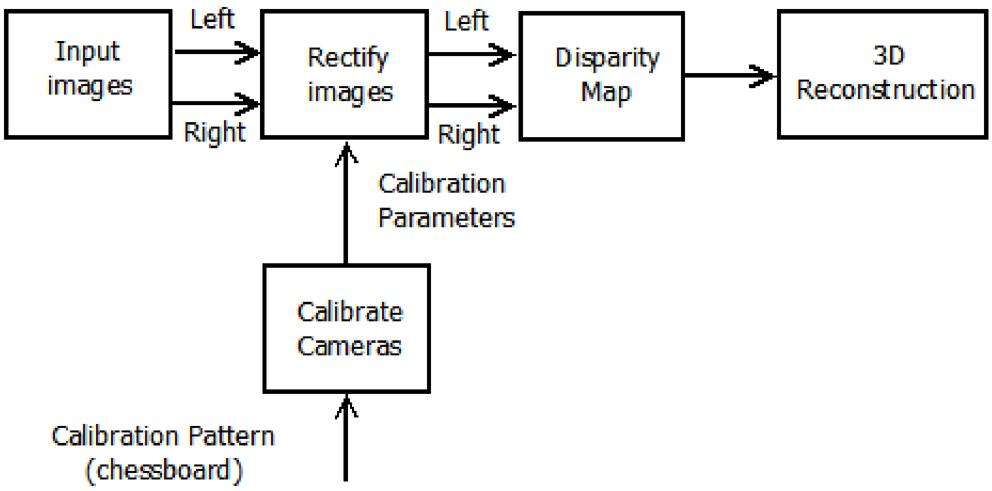


Figure 1.1: *System overview*

Chapter 2

Theoretical Review

In this chapter, I will introduce the fundamental mathematical tools and the algorithms needed to understand the workspace of the computer vision. In particular, I will describe the Camera Models, the Camera Calibration, the Epipolar Geometry, the Image Rectification and the Disparity Map.

2.1 Camera Models

The Camera Model defines the mapping between the 3-D points $\{x\}$ in a camera frame and the pixel $\{u\}$ in the image plane: $\Pi = \mathbb{R}^2 \rightarrow \Omega$.

The points on the image plane are the projection of the three-dimensional scene to a two-dimensional system; this projection depends from various parameters like the position and the orientation of the camera, the focal length of the lens, the optical center and the distortion. In particular, in geometric optics, distortion is a deviation from rectilinear projection; a projection in which straight lines in a scene remain straight in an image. We have vary models that can be correct the distortion all using the distortion coefficient (see section 2.2.3). It is a form of optical aberration.

2.1.1 Pinhole Model

Let the centre of projection be the origin of a Euclidean coordinate system, and consider the plane $Z = f$ the image plane. Under the *pinhole camera model*, a point in space with coordinates $\mathbf{X} = (X, Y, Z)^T$ is mapped to the point on the image plane where a line joining the point X to the centre of projection meets the image plane. All the light rays go throught a single point in the space $\mathbf{C}$ called *optical center* or *camera center*. Thus, the 2D image is the intersection between the straight line that connects the points in the space and the camera center with the image plane. The perpendicular line that joins the image plane is called *principal axis* and the distance between $\mathbf{C}$ and the image plane is called *focal length* [Fig. 2.1(a)].

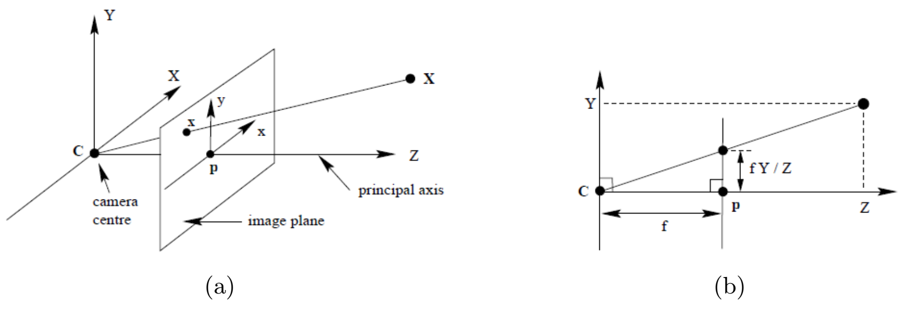


Figure 2.1: *The all model is called **Pinhole Model***
(a)sx:3-D view (b)dx:2-D view.

By similar triangles [Fig. 2.1(b)], one quickly computes that the point $(X, Y, Z)^T$ is mapped to the point $(fX/Z, fY/Z, f)^T$ on the image plane. Ignoring the final image coordinate, we have:

$$(X, Y, Z)^T \rightarrow (fX/Z, fY/Z)^T \tag{2.1}$$

this describe the central projection mapping from world to image coordinates. This is a mapping from Euclidean 3-space $\mathbb{R}^3$ to Euclidean 2-space $\mathbb{R}^2$ [1]. We can express the central projection (2.1) in matrix form:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \rightarrow \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.2}$$

The matrix in the equation (2.2) can be written as $diag(f, f, 1)[\mathbf{I} \mid \mathbf{0}]$ where $diag(f, f, 1)$ is a diagonal matrix and $[\mathbf{I} \mid \mathbf{0}]$ represents the identity matrix plus the zero column vector.

Defining $\mathbf{X}$ the world point represented by the homogeneous 4-vector $(X, Y, Z, 1)^T$, $\mathbf{x}$ the image point represented by the homogeneous 3-vector, and P the 3×4 homogeneous *camera projection matrix*, the camera matrix for the pinhole model of the central projection is described by:

$$\mathbf{x} = \mathsf{P}\mathbf{X} \tag{2.3}$$

which defines the camera matrix for the pinhole model of central projection as

$$\mathsf{P} = diag(f, f, 1)[\mathbf{I} \mid \mathbf{0}] \tag{2.4}$$

Equation (2.4) is valid if the origin of the coordinates in the image plane is at the principal point. If the assumption is not valid the relation (2.1) is:

$$(X, Y, Z)^T \rightarrow (fX/Z + \mathsf{p_x}, fY/Z + \mathsf{p_y})^T \tag{2.5}$$

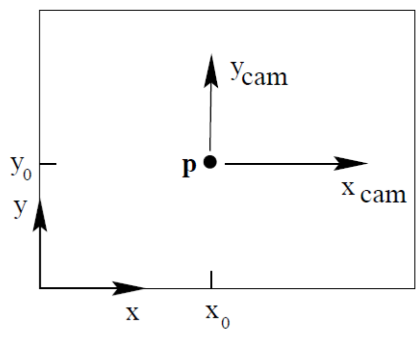


Figure 2.2: *Image (x, y) and camera (x_{cam}, y_{cam}) coordinate systems.*

where $(\mathsf{p_x}, \mathsf{p_y})^T$ are the coordinates of the principal point. Referring at the figure [2.2], in homogeneous coordinates the equation (2.2) becomes

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \to \begin{pmatrix} fX + Z\mathsf{p_x} \\ fY + Z\mathsf{p_y} \\ Z \end{pmatrix} = \begin{bmatrix} f & & \mathsf{p_x} & 0 \\ & f & \mathsf{p_y} & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.6}$$

Defining

$$\mathsf{K} = \begin{bmatrix} f & & \mathsf{p_x} \\ & f & \mathsf{p_y} \\ & & 1 \end{bmatrix} \tag{2.7}$$

and

$$\mathbf{X}_{cam} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.8}$$

then

$$\mathbf{x} = \mathsf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{X}_{cam} \tag{2.9}$$

where $\mathbf{X}_{cam}$ is the *camera coordinate frame.* Assiuming that the camera is positioned in the center of the Euclidean coordinate system (i.e. with the principal axis of the camera pointing straight down the Z-axis).

In general, the points in space are expressed in *world coordinate frame* obtained through rotation and translation, as we can see in the figure [2.3], $\hat{\mathbf{X}}$ is the vector representing the point in a world coordinate frame and $\hat{\mathbf{X}}_{cam}$ is the same point in the camera coordinate frame so

$$\hat{\mathbf{X}}_{cam} = \mathsf{R}(\hat{\mathbf{X}} - \hat{\mathbf{C}}) \tag{2.10}$$

where R is the rotation matrix representing the orientation of the camera coordinate frame and $\hat{\mathbf{C}}$ the orientation of the camera coordinate frame centre in the world coordinate frame. In homogeneous coordinate,

$$\mathbf{X}_{cam} = \begin{bmatrix} \mathsf{R} & -\mathsf{R}\hat{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathsf{R} & -\mathsf{R}\hat{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{X} \tag{2.11}$$

replacing equation (2.11) in equation (2.9):

$$\mathbf{x} = \mathsf{KR}[\mathbf{I} \mid -\hat{\mathbf{C}}]\mathbf{X} \tag{2.12}$$

where $\mathbf{X}$ is now in a world coordinate frame.

A general Pinhole Camera $\mathsf{P} = \mathsf{KR}[\mathbf{I} \mid -\hat{\mathbf{C}}]$ has 9 degree of freedom: 3 for K (the elements f, $\mathsf{p_x}$, $\mathsf{p_y}$), 3 for R, and 3 for $\hat{\mathbf{C}}$. The parameters in K are the *internal camera parameters*, the parameters of R and $\hat{\mathbf{C}}$, which relate the camera orientation and position to a world coordinate system, are called the *external parameters* [1].

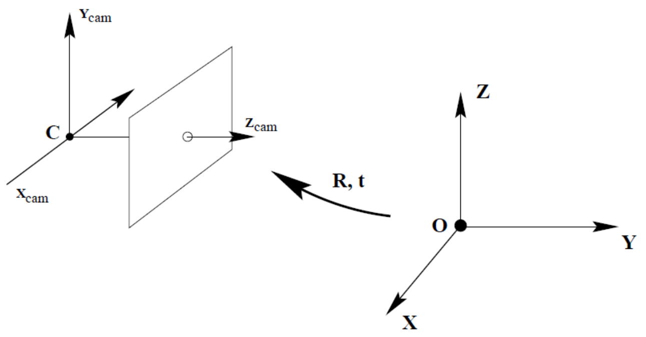


Figure 2.3: *The Euclidean Transformation.*

2.2 Camera Calibration

Geometric Camera Calibration is the process of obtaining the parameters of a lens and image sensor of an image or video camera [2]. These parameters are used to correct the distortion due to the lens, to measuring the dimension of an object in the world units or to determine the position of the camera in the scene. The algorithm calculates the camera matrix $\mathsf{P} = \begin{bmatrix} \mathsf{R} & \mathsf{t} \end{bmatrix}^T \mathsf{K}$ using the extrinsic and the intrinsic parameters. In particular the world points are transformed to camera coordinates using the extrinsics parameters and the camera coordinates are mapped into the image plane using the intrinsics parameters.

2.2.1 Extrinsic Parameters

The Extrinsic Parameters are the rotation R and the traslation t and define the position of the camera center and the camera's heading in world coordinates. t is the position of the origin of the world coordinate system expressed in coordinates of the camera-centered coordinate system. The position, $\mathbf{C}$, of the camera expressed in world coordinates is $\mathbf{C} = \mathsf{R}^{-1}\mathsf{t} = -\mathsf{R}^{\mathsf{t}}\mathsf{t}$ [2]

2.2.2 Intrinsic Parameters

The Intrinsic Parameters are the focal length, the optical center (principal point) and the skew coefficient, used to express K in equation (2.12):

$$\mathsf{K} = \begin{bmatrix} \alpha_x & \mathsf{s} & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

this is obtained by multiplying equation (2.7) on the left by an extra factor $diag(m_x, m_y, 1)$, where m_x and m_y are the number of pixels per unit distance in image coordinates in x and y directions. Thus $\alpha_x = fm_x$ and $\alpha_y = fm_y$ represent the focal length of the camera in pixel dimension; $x_0 = m_x p_x$ and $y_0 = m_y p_y$ are the coordinate of the optical center in pixel dimension and $\mathsf{s} = f_y \tan\alpha$ is the skew coefficient (when $\mathsf{s} \neq 0$ the image axes are not perpendicular) [1][3].

2.2.3 Distortion in Camera Calibration

The pinhole camera model does not account for lens distortion because an ideal pinhole camera does not have a lens. To accurately represent a real camera, the full camera model used by the algorithm includes the radial and tangential lens distortion.

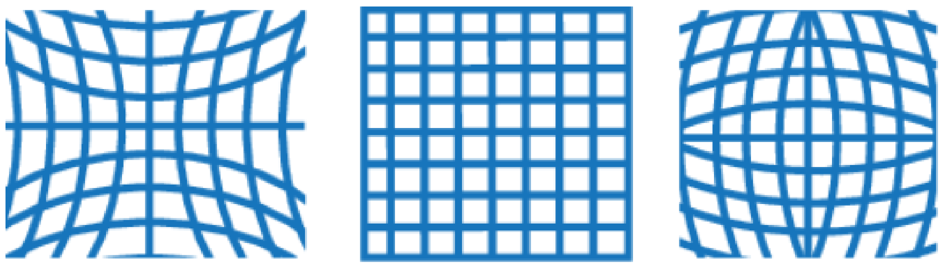


Figure 2.4: *Negative Distortion, No Distortion and Positive Distortion.*

Radial Distortion

The Radial Distortion occours when the light rays bend more near the edges of a lens than they do at its optical center. The distorted points are denoted as:

$$\begin{aligned} x_{distorted} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{distorted} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{2.14}$$

where (x, y) are the undistorted pixel in normalized coordinate; i.e. the coordinate computed from pixel coordinate by translating in the optical center and dividing by the focal length in pixel and k_n is the radial distortion coefficients of the lens [Fig. 2.4]

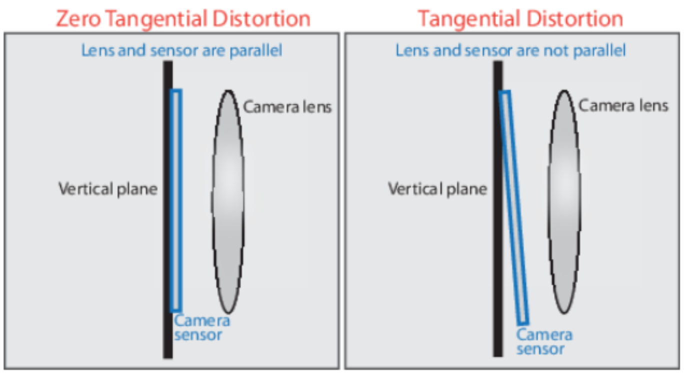


Figure 2.5: *Tangential Distortion*

Tangential Distortion

The Tangential Distortion occours when the lens and the image plane are not parallel [Fig. 2.5]. The distorted points are denoted as:

$$\begin{aligned} x_{distorted} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{distorted} &= y + [2p_2 xy + p_1(r^2 + 2y^2)] \end{aligned} \tag{2.15}$$

where p_1 and p_2 are the tangential distortion coefficients of the lens.

2.2.4 Single Camera Calibration

As mentioned earlier to calibrate a camera we need a correspondence between 3D world points and 2D image points, in particular the calibration of the intrinsic parameters is based on information obtained from homographies coming from the configuration of coplanar points. The problem is that the 3D points cannot be co-planar, so people build 3D calibration rigs, e.g. a box made of checkerboards [Fig. 2.6]. One image of a rig like that would

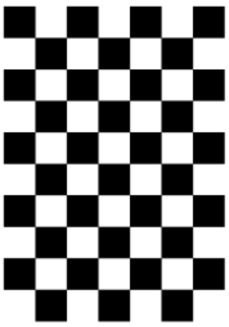


Figure 2.6: *Example of a calibration pattern*

be enough to calibrate, but those rigs are hard to build, because you have to get the planes to be at exactly right angles to each other. So we use a pattern with known dimensions and, in particular, the chessboard because its realization doesn't require high requirements, it can be produced by any graphic software and printed using a good quality laser printer. It is very important that the pattern is really flat infact we had to choose a good surface where apply the printed page.

One of the most important concepts for self-calibration is the *Absolute Conic* (AC) and its projection in the images (IAC).

Absolute Conic

The absolute conic (AC) Ω_∞ is a (point) conic on the plane at infinity Π_∞ [1]. The points that lie on Ω_∞ must be satisfy

$$\left.\begin{array}{r} x_1^2 + x_2^2 + x_3^2 \\ x_4^2 \end{array}\right\} = 0 \tag{2.16}$$

This two equations are required to define Ω_∞. For directions on Π_∞ (i.e. points with $x_4 = 0$) the defining equation can be written:

$$(x_1, x_2, x_3)\mathbf{I}(x_1, x_2, x_3)^T = 0 \tag{2.17}$$

so that Ω_∞ corresponds to a conic C with matrix $\mathsf{C} = \mathbf{I}$. The conic Ω_∞ is a fixed conic under any similarity transformation. More formally, the absolute conic, Ω_∞, is a fixed conic under the projective transformation $\mathbf{H}$ if, and only if, $\mathbf{H}$ is a similarity transformation.

Since it is invariant under Euclidean transformations, its relative position to a moving camera is constant. For constant intrinsic camera parameters its image will therefore also be constant. This is similar to someone who has the impression that the moon is following him when driving on a straight road. Note that the AC is more general, because it is not only invariant to translations but also to arbitrary rotations. It can be seen as a calibration object which is naturally present in all the scenes. Once the AC is localized, it can be used to upgrade the reconstruction to metric. It is, however, not always so simple to find the AC in the reconstructed space. In some cases it is not possible to make the difference between the true AC and other candidates.

Image of the Absolute Conic

Consider two camera projections P and P' corresponding to the same camera (internal parameters) but different poses. The *Image of the Absolute Conic* (IAC) is independent of the camera pose. Infact, the IAC is directly related to the intrinsic parameter matrix K of the camera defined in equation (2.13).

To derive this relation we first must determine the map between the plane at infinity Π_∞, and the camera image plane. Points on Π_∞ can be writtes as $x_\infty = (d^T, 0)^T$, and are imaged be a general Pinhole camera P as

$$x = \mathsf{P}x_\infty = \mathsf{KR}[\mathbf{I} \mid -\hat{\mathbf{C}}] \begin{pmatrix} d \\ 0 \end{pmatrix} = \mathsf{KR}d \tag{2.18}$$

this show that the mapping between Π_∞ and an image is given by the planar homography. This map is independent of the position of the camera and depends only on the camera internal calibration and orientation with respect to the world coordinate frame, as we said previously. Since the absolute conic Ω_∞ is on Π_∞ we can compute its image under $\mathsf{H} = \mathsf{KR}$, and find that the IAC is the conic

$$\omega = (\mathsf{KK}^T)^{-1} = \mathsf{K}^{-T}\mathsf{K}^{-1} \tag{2.19}$$

Example: A Simple Calibration Device [1]

The image of three squares (on planes which are not parallel, but which need not be orthogonal) provides sufficiently many constraints to compute K. Consider one of the squares. The correspondences between its four corner points and their images define the homography $\mathbf{H}$ between the plane Π of the square and the image. Applying this homography to circular points on Π determines their images as $\mathbf{H}(1, \pm i, 0)^T$. Thus we have two points on the ω and a similar procedure applies to the other squares (for a total of 6 points on ω).

First of all, for each square we compute the homografy $\mathbf{H}$ that maps the corner points $(0,0)^T, (1,0)^T, (0,1)^T, (1,1)^T$ to their imaged points. Then we compute the imaged circular points for the plane of that square as $\mathbf{H}(1, \pm i, 0)^T$; being $\mathbf{H} = [h_1, h_2, h_3]$ so the circular points are $h_1 \pm ih_2$. Thus we fit a conic ω to the six imaged circular points, in particular if $h_1 \pm ih_2$ lies on ω then $(h_1 \pm ih_2)^T \omega (h_1 \pm ih_2) = 0$, and the imaginary and real parts give respectively:

$$\begin{aligned} h_1^T \omega h_2 &= 0 \\ h_1^T \omega h_1 &= h_2^T \omega h_2 \end{aligned} \tag{2.20}$$

which are equations linear in ω. Finally we compute the calibration K from $\omega = (\mathsf{K}\mathsf{K}^T)^{-1}$ using the Cholesky factorization [7].

Calibration Method

There are many different approaches to calculate the intrinsic and extrinsic parameters for a specific camera setup. MATLAB uses the Zhang's method [4]. To perform a full calibration by the Zhang method at least 3 different images of the calibration target are required, either by moving the gauge or the camera itself. If some of the intrinsic parameters are given as data (orthogonality of the image or optical center coordinates) the number of images required can be reduced to two.

In a first step, an approximation of the estimated projection matrix **H** between the calibration target and the image plane is determined using Direct Linear Transformation method [5]. Subsequently, applying self-calibration techniques (correspondence between the calibration points when they are in different positions) is needed to obtained the image of the absolute conic matrix.

In particular we assume that we have a homography **H** that maps points x_π on a "probe plane" π to points x on the image. The *circular points* $\mathbf{I},\mathbf{J} = \begin{bmatrix} 1 & \pm j & 0 \end{bmatrix}^T$ lie on both proble plane π and on the AC Ω_∞. Lying on the absolute conic means they are projected onte the IAC ω, thus $x_1^T \omega x_1 = 0$ and $x_2^T \omega x_2 = 0$. The circular point project as

$$x_1 = \mathbf{HI} = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} \begin{bmatrix} 1 \\ j \\ 0 \end{bmatrix} = h_1 + j h_2 \tag{2.21}$$

$$x_2 = \mathbf{HI} = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} \begin{bmatrix} 1 \\ -j \\ 0 \end{bmatrix} = h_1 - j h_2 \tag{2.22}$$

Ignoring x_2 and replacing x_1 we have

$$\begin{aligned} x_1^T \omega x_1 &= (h_1 + j h_2)^T \omega (h_1 + j h_2) \\ &= (h_1^T + j h_2^T) \omega (h_1 + j h_2) \\ &= h_1^T \omega h_1 + j(h_2^T \omega h_2) = 0 \end{aligned} \tag{2.23}$$

The circular points remain invariant under similarity transformations, more formally a transformation is a similarity transformation if and only if it preserves the circular points, $[1, \pm i, 0]$. This property makes them useful for determining the angle between two lines; in particular, assume that we have two lines $\mathbf{u}_1$ and $\mathbf{u}_2$ which intersect the ideal line at two points, say $\mathbf{p}_1$and $\mathbf{p}_2$. Then, the cross ratio between these two points and the two absolute points $\mathbf{I}$ and $\mathbf{J}$ yields the directed angle θ from the second line to the first:

$$\theta = \frac{1}{2i} \log Cr(\mathbf{p}_1, \mathbf{p}_2; \mathbf{I}, \mathbf{J}) \tag{2.24}$$

where Cr is the cross-ratio of four collinear points. (2.24) is known as the *Laguerre formula.*

2.3 Epipolar Geometry

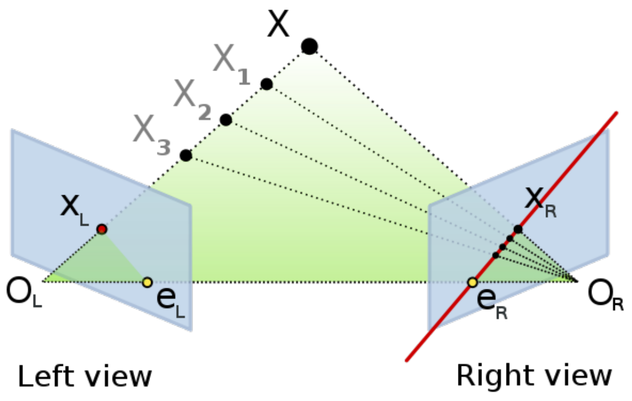


Figure 2.7: *Essencial elements of a Epipolar Geometry*

The Epipolar Geometry describes the relations and the geometric constraints between two 2-D images of the same 3-D scene, captured by two cameras with distinct position and orientation.

When we capture an element, that are in the point $\mathbf{X}$ [Fig. 2.7], with 2 cameras in $\mathbf{O}_R$ and $\mathbf{O}_L$ then $\mathbf{X}$ will be projected into $\mathbf{x}_R$ and $\mathbf{x}_L$ respectively. In particular

$$\begin{aligned} \mathbf{x}_L &= \mathsf{P}_L \mathbf{X} \\ \mathbf{x}_R &= \mathsf{P}_R \mathbf{X} \end{aligned} \tag{2.25}$$

where P_L and P_R are the projective transformations; i.e. a matrix that contains the position, the orientation and the formal parameters of the camera.

2.3.1 Epipole or epipolar point

Because the center $\mathbf{O}_L$ and $\mathbf{O}_R$ of the two cameras in [Fig. 2.7] are in distinct positions we can project one on the image plane of the other.

$$\begin{aligned} \mathbf{e}_L &= \mathsf{P}_L \mathbf{O}_R \\ \mathbf{e}_R &= \mathsf{P}_R \mathbf{O}_L \end{aligned} \tag{2.26}$$

The points $\mathbf{e}_L$ and $\mathbf{e}_R$ are the epipolar points.

2.3.2 Epipolar line

According the [Fig.2.7], the point $\mathbf{x}_L$ is the intersection between the line that connects $\mathbf{O}_L$ and $\mathbf{X}$ (projection ray) and the left plane. If we project this line on the right plane, then the line will connect the projections of $\mathbf{O}_L$ and $\mathbf{X}$ on the right plane, i.e. the line passing through $\mathbf{x}_R$ and $\mathbf{e}_R$

$$\mathbf{l}_R = \mathbf{x}_R \times \mathbf{e}_R \tag{2.27}$$

Analogously, we can define $\mathbf{l}_L$ The two line $\mathbf{l}_L$ and $\mathbf{l}_R$ are the epipolar line.

2.3.3 Epipolar plane

The Epipolar Plane is the plane where $\mathbf{X}$, $\mathbf{O}_L$ and $\mathbf{O}_R$ lie. This, associated with a $\mathbf{X}$ point, intercepts the image planes exactly in the epipolar line associated to $\mathbf{X}$

2.3.4 The Fundamental Matrix [6]

Let $\mathbf{x} = (x, y, t)^T$ be the homogeneous coordinates of a point in the first image and $\mathbf{e} = (u, v, w)$ be the coordinates of the epipole of the second camera in the first image. The epipolar line through $\mathbf{x}$ and $\mathbf{e}$ is represented by the vector $\mathbf{l} = (a, b, c)^T = \mathbf{x} \times \mathbf{e}$. The mapping $\mathbf{x} \to \mathbf{l}$ is linear and can be represented by a 3×3 rank 2 matrix $\mathbf{C}$:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} yw - zv \\ zu - xw \\ xv - yu \end{pmatrix} = \begin{pmatrix} 0 & w & -z \\ -w & 0 & u \\ z & -u & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{2.28}$$

The mapping of epipolar lines $\mathbf{l}$ from image 1 to the corresponding epipolar lines $\mathbf{l}'$ in image 2 can be represented (non-uniquely) as a collineation on the entire dual space of lines in the projective plane. Let $\mathbf{A}$ be a collineation such that $\mathbf{l}' = \mathbf{A}\mathbf{l}$. The constraints on $\mathbf{A}$ are encapsulated by the correspondence of 3 distinct epipolar lines. The first two correspondences each provide two constraints, because a line in the plane has 2 *dof.* The third line must pass through the intersection of the first two, so only provides one further constraint. Since $\mathbf{A}$ has eight degrees of freedom and we only have five constraints, it is not fully determined. Nevertheless, the matrix $\mathbf{F} = \mathbf{AC}$ is fully determined. Using (2.28) we get:

$$\mathbf{l}' = \mathbf{ACx} = \mathbf{Fx} \tag{2.29}$$

$\mathbf{F}$ is called the *Fundamental Matrix.* It is the algebraic representation of epipolar geometry; it is a 3×3 matrix which relates corresponding points in stereo images; as $\mathbf{C}$ has rank 2 and $\mathbf{A}$ has rank 3, $\mathbf{F}$ has rank 2. $\mathbf{F}$, also, defines a bilinear constraint between the coordinates of corresponding image points. If $\mathbf{x}'$ is the point in the second image corresponding to $\mathbf{x}$, it must lie on the epipolar line $\mathbf{l}' = \mathbf{Fx}$, and hence $\mathbf{x}'^T \cdot \mathbf{l}' = 0$. The epipolar constraint can therefore be written:

$$\mathbf{x}'^T \mathbf{Fx} = 0 \tag{2.30}$$

2.4 Image Rectification

The *Image Rectification* is the process of resampling pairs of stereo images taken from widely differing viewpoints in order to produce a pair of *"matched epipolar projections"*. In this projections the epipolar line is parallel with the x-axis and so disparities between the images in x direction only.

We apply a series of 2-D projective transformation to the two images, in this way we match the epipolar line; thus the matched points must be, more or less, the same x coordinate. Moreover the apply transformations subjects the images to a minimal distortion [1].

2.4.1 Mapping the Epipole to Infinity

In order to map the epipole to a point at infinity, a projective transformation is needed. If the epipolar line must be transformed into a line parallel to the x-axis so, the epipole should be mapped to the particupar infinite point $(1, 0, 0)^T$.

To choose a good transformation $\mathbf{H}$, it should act as a rigid transformation in the neighbourhood of a given selected point x_0 of the image. In this way, the neighbourhood of x_0 is subjected just to a rotation and a translation and they will look the same in the original and resampled images. A good choice may be the center of the image.

We suppose that x_0 is in the origin and that the epipole $\mathbf{e} = (f, 0, 1)^T$ lie on the x-axis. Considering also the transformation $\mathbf{G}$ defined as:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/f & 0 & 1 \end{bmatrix} \tag{2.31}$$

Relation (2.31) takes $\mathbf{e}$ to the point at infinity $(f, 0, 0)^T$ as required. A point $(x, y, 1)^T$ is mapped by $\mathbf{G}$ to the point $(\hat{x}, \hat{y}, 1)^T = (x, y, 1 - x/f)^T$.

With $|x/f| < 1$ then

$$(\hat{x}, \hat{y}, 1)^T = (x, y, 1 - x/f)^T = (x(1 + x/f + ...), y(1 + x/f + ...), 1)^T \quad (2.32)$$

The Jacobian is
$$\frac{\partial(\hat{x}, \hat{y})}{\partial(x, y)} = \begin{bmatrix} 1 + 2x/f & 0 \\ y/f & 1 + x/f \end{bmatrix} \quad (2.33)$$

plus higher terms in x and y. If $x = y = 0$ this is an *Identity Map*.

The required mapping $\mathbf{H}$ is defined as $\mathbf{H}$=$\mathbf{G}\mathsf{R}\mathsf{t}$, where t is the transaltion from x_0 to the origin, R is the rotation about the origin taking the epipole $\mathbf{e}'$ in the point $(f, 0, 1)^T$ on the x-axis and $\mathbf{G}$ is the mapping when we consider $(f, 0, 1)^T$ to the infinity.

2.4.2 Matching Transformation

We apply a mapping to the second image, in this way we match the epipolar line. Consider 2 images J and J'; we want resampling this two images according to transformation $\mathbf{H}$ for J and to $\mathbf{H}'$ for J'. So, the epipolar line in J is matched with the epipolar in J'; in particular $\mathbf{H}^{-T}\mathbf{l} = \mathbf{H}'^{-T}\mathbf{l}'$ where $\mathbf{l}$ and $\mathbf{l}'$ are the two epipolar lines.

To choose the transformation $\mathbf{H}'$, the sum-of-squadred distances between $\mathbf{H}x_i$ and $\mathbf{H}'x'_i$ should miniminized by

$$\sum_i d(\mathbf{H}x_i, \mathbf{H}'x'_i)^2 \quad (2.34)$$

Let J and K' be the images with fundamental matrix $\mathbf{F} = |\mathbf{e}'| \times \mathbf{M}$ and let $\mathbf{H}'$ the projected transformation of J'. The projective transformation $\mathbf{H}$ of J matches $\mathbf{H}'$ if and only if

$$\mathbf{H} = (\mathbf{I} + \mathbf{H}'\mathbf{e}'\mathbf{a}^T)\mathbf{H}\mathbf{M} \quad (2.35)$$

for some vector $\mathbf{a}$. We can therefore affirm that a transformation $\mathbf{H}$ of J matches $\mathbf{H}'$ is and only if $\mathbf{H} = \mathbf{H}_A\mathbf{H}_0$ where $\mathbf{H}_0 = \mathbf{H}'\mathbf{M}$. Is we consider that $\mathbf{H}'$ maps the epipole to the infinity, so we can choose the best transformation $\mathbf{H}$; i.e. that minimize the disparity.

Writing $\hat{\mathbf{x}}'_i = \mathbf{H}'\mathbf{x}_i$ and $\hat{\mathbf{x}}'_i = \mathbf{H}_0\mathbf{x}_i$, then the minimization problem (2.34) is to find $\mathbf{H}_A$ such that
$$\sum d(\mathbf{H}_A\hat{\mathbf{x}}'_i, \hat{\mathbf{x}}'_i)^2 \quad (2.36)$$

is minimized. In particupar, let $\hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i, 1)^T$ and $\hat{\mathbf{x}}'_i = (\hat{x}'_i, \hat{y}'_i, 1)^T$ and know $\mathbf{H}'$ and $\mathbf{M}$, so the two vector may be computed from the 2 matched points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. The (2.35) may be written as

$$\sum_i (a\hat{x}_i + b\hat{y}_i + c - \hat{x}'_i)^2 + (\hat{y}_i - \hat{y}'_i)^2 \quad (2.37)$$

Whit $(\hat{y}_i - \hat{y}'_i)$ constant, this is equivalent at

$$\sum_i (a\hat{x}_i + b\hat{y}_i + c - \hat{x}'_i)^2 \tag{2.38}$$

this is a simple linear least-squares parameter minimization problem, where we compute the 3 parameters a, b and c.

2.4.3 Algorithm Outline

Given a stereo pair, the intrinsic and extrinsic parameters, we want find the image transformation to achieve a stereo system of horizontal epipolar lines; we also assume that the cameras are calibrated.

1. Rotate both, left and right, camera so that they share the same x-axis: $\mathbf{O}_R - \mathbf{L} = \mathsf{t}$

2. Define a rotation matrix R_{rect}

    - Make the new x axis along the direction of the baseline [vector $\mathbf{e}_1$]
    - Make new y axis orthogonal to the new x and the old z which is along the old optical axis (cross product) [vector $\mathbf{e}_2$]
    - Make new z axis orthogonal to the baseline and the new y axis (cross product) [vector $\mathbf{e}_3$]
    - Rotation matrix is now complete

    $$\mathsf{R}_{rect} = \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{pmatrix}^T \tag{2.39}$$

    - Rotates left camera so that epipolar lines are parallel

3. Set $\mathsf{R}_L = \mathsf{R}_{rect}$ and $\mathsf{R}_R = \mathsf{R}\mathsf{R}_{rect}$ the two rotation matrix for left and right camera

4. For each left-camera point $\mathbf{x}_L = [x, y, z]^T$ compute $\mathsf{R}_L\mathbf{x}_L = [x', y', z']$ and the corresponding rectified point as $\mathbf{x}'_L = f/z'[x', y', z']$

5. Repeat the previous step for the right camera using R_R and $\mathbf{x}_R$

Basically rotate the point and then reproject it, in practice steps 4 and 5 require back projection. This is usually done with a homography [8] which is computed from the known rotation and calibration.

2.5 Disparity Map

One of the easiest methods to understand the disparity would be to blink your eyes, one at a time, alternating between your left and right eye. If you observe, the objects closer to you would appear to jump about their position more than the objects further away. This shift is would be negligible as the objects move away. Infact the brain uses binocular disparity to extract depth information from the two-dimensional retinal images in stereopsis.

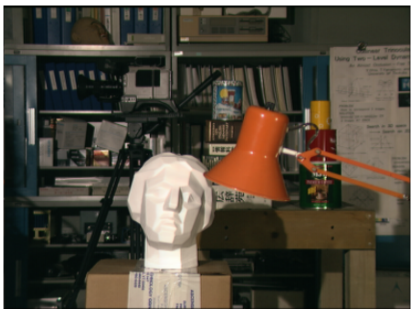
In computer vision, binocular disparity refers to the difference in coordinates of similar features within two stereo images [9]. Taking as a reference the image [2.7], if $\mathbf{X}$ projects to a point in the left frame $\mathbf{x}_L = (u, v)$ and to the right frame at $\mathbf{x}_R = (p, q)$ we can find the disparity for this point as the magnitude of the vector between (u, v) and (p, q). In particular, the disparity of $\mathbf{x}_L$ is defined as: $d(\mathbf{x}_L) = \mathbf{x}_R - \mathbf{x}_L$.

Obviously this process involves choosing a point in the left hand frame and then finding its match (the *corresponding point*) in the right hand image; often this is a particularly difficult task to do without making a lot of mistakes. If we perform this matching process for every pixel in the left hand image, finding its match in the right hand frame and computing the distance between them we would end up with an image where every pixel contained the disparity value for that pixel in the left image.

MATLAB uses the *Semi-Global Block Matching* algorithm [10] where the function computes disparity by comparing the sum of absolute differences (SAD) of each block of pixels in the image and additionally forces similar disparity on neighboring blocks. In particular, after the rectification process, i.e. there is no disparity in the y image coordinates (see section 2.4), the correspondence problem can be solved using the Semi-Global Block Matching algorithm that scans both the left [Fig. 2.8(a)] and right [Fig. 2.8(b)] images for matching image features. The algorithm, first of all, compute a measure of contrast of the image by using the Sobel filter [17]. Then it forms a smaller image patch around every pixel in the left image and these image patches are compared to all possible disparities in the right image by comparing their corresponding image patches. The comparison between these two patches is made throught the *Sum of Absolute Differences* (SAD):

$$\sum \sum |L(r, c) - R(r, c - d)| \tag{2.40}$$

where L and R refer to the left and right columns, r and c refer to the current row and column of either images being examined and d refers to the disparity of the right image. The disparity with the lowest computed value of (2.40) is considered the disparity for the image feature.


(a)

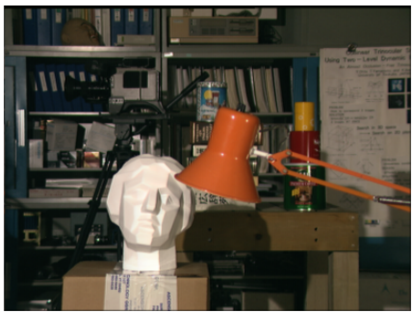

(b)

Figure 2.8: *(a)sx:left image (b)dx:right image*

As we saw in the initial example, knowledge of disparity can be used for depth calculation.

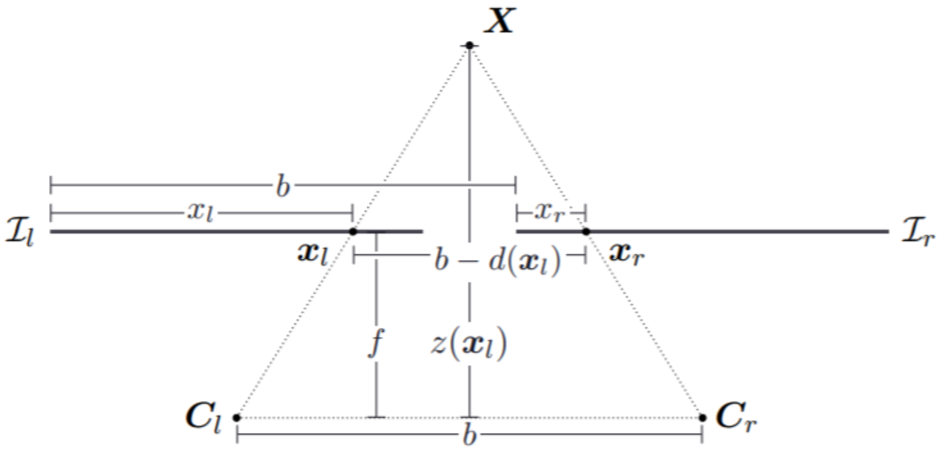


Figure 2.9: *A rectified setup viewed from above, showing the two camera centers, $\boldsymbol{C}_l$ and $\boldsymbol{C}_r$, and their respective image planes, $\mathcal{I}_l$ and $\mathcal{I}_r$.*

In particular, in rectified images, given a correspondence between one point of an image and one point of the other. Using the notation from the [Fig. 2.9] and similar triangles we get:

$$\frac{b - d(\mathbf{x}_l)}{z(\mathbf{x}_l) - f} = \frac{b}{z(\mathbf{x}_l)} \Leftrightarrow z(\mathbf{x}_l) = \frac{fb}{d(\mathbf{x}_l)} \tag{2.41}$$

for all pixel pairs in the two images, the focal length, f, and baseline, b, are constant.

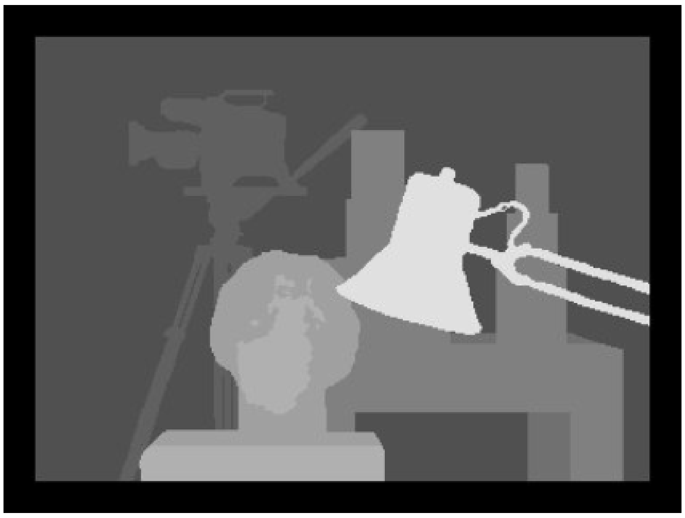


Figure 2.10: *Disparity Map*

We can see that the *depth*, $z(\mathbf{x}_l)$, is inversely proportional to the disparity, $d(\mathbf{x}_l)$. Therefore in the disparity map, the brighter shades represent more shift and lesser distance from the point of view (camera). The darker shades represent lesser shift and therefore greater distance from the camera [Fig. 2.10].

Chapter 3

Methodology

In this chapter, I will describe all the procedures that we use to perform the 3D view of the sequence that we have acquired. Using MATLAB, this is the procedure in particular:

1. Perform the stereo calibration of the two cameras;
2. Perform the rectification of the two images at the same time instant;
3. Compute the disparity map;
4. Reconstruct the scene in 3D;

3.1 Images Acquisition

A 2-camera system was employed to acquire experimental data based on image frames. The cameras adopted are colours C-Mount CMOS digital cameras and their resolution is 1920×1200. They are mounted onto the two holes of the microscope made for this purpuse.

Each camera records what the dentist sees respectively from the left eye and the second eye. The two cameras are *Power Over Ethernet* (POE), and to connect them to the PC, we used a Gigabit switch POE. We acquired the video images from the cameras at 20 frame per second. Cameras are synchronized enabling *Precise Time Protocol* (PTP) on them in order to acquire images couple by couple.

Images are saved on a Solid State Disk in Bayer BG format and then converted later in PNG format.

3.2 Compute the Calibration

As we have already seen in the section 2.2, we need to calibrate the cameras to avoid the distortion due to the lens, to compute the position of the two cameras (individually and each other) and to measuring the dimension of an object in the world units.

3.2.1 Camera Calibrator

Using the tool of MATLAB, *Camera Calibrator*, we calibrate the two cameras, *Camera0* and *Camera1*, individually. First of all, we need to prepare the images, the camera and the calibration pattern. The calibrator requires at least three images, to obtain a better results it is good practice to use a number of images between 10 and 20 images of the calibration pattern.

Harris points [14] or Shi-Tomasi corners [15] are a common choice for localizing junctions in an image. MATLAB,however, uses a procedure that gives more robust results with respect to image clutter, blurring artifacts and localization accuracy. In order to locate checkerboard corners in a grayscale image I, it compute a corner likelihood at each pixel in the image using two different $n \times n$ corner prototypes: one for axis-aligned corners and one for corners, which are rotated by 45 [16]. Each prototype is composed of four filter kernels $\{A, B, C, D\}$, which are convolved with the input image. For an ideal corner, the response of $\{A, B\}$ should be greater than the mean response of $\{A, B, C, D\}$, while the response of $\{C, D\}$ should be smaller, and vice versa for flipped corners.

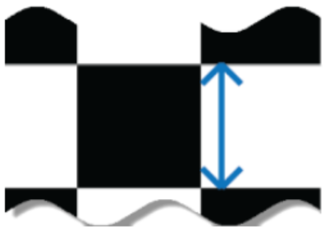


Figure 3.1: *Size of chessboad square*

The checkerboard pattern that we use must not be square, moreover, one side must contain an even number of squares and the other side must contain an odd number of squares. Therefore, the pattern contains two black corners along one side and two white corners on the opposite side. Also we need to measure the chessboard square [Fig. 3.1]. To calibrate, we use uncompressed images or lossless compression formats such as PNG of a chessboard (see section 2.2.4) taken by the *Camera0* in different position [Fig. 3.2]. The calibrator is able to rejects duplicate images, it also rejects images where the entire checkerboard could not be detected. Possible reasons for no detection are a blurry image or an extreme angle of the pattern. Detection takes longer with larger images and with patterns that contain a large number of squares.

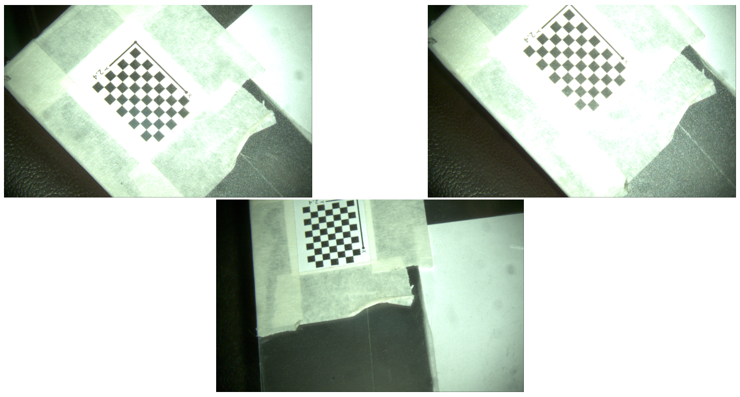


Figure 3.2: *Example of calibration frames*

During the calibration computing [Fig. 3.3], we note a certain mean error in the evaluation of the distance between detected and reprojected points. The tool calculates reprojected errors by projecting the checkerboard points from world coordinates, defined by the checkerboard, into image coordinates. The tool then compares the reprojected points to the corresponding detected points. As a general rule, mean reprojection errors of less than one pixel are acceptable [11]; usally a threshold is defined and images with an error above this value are discarded [Fig. 3.4].

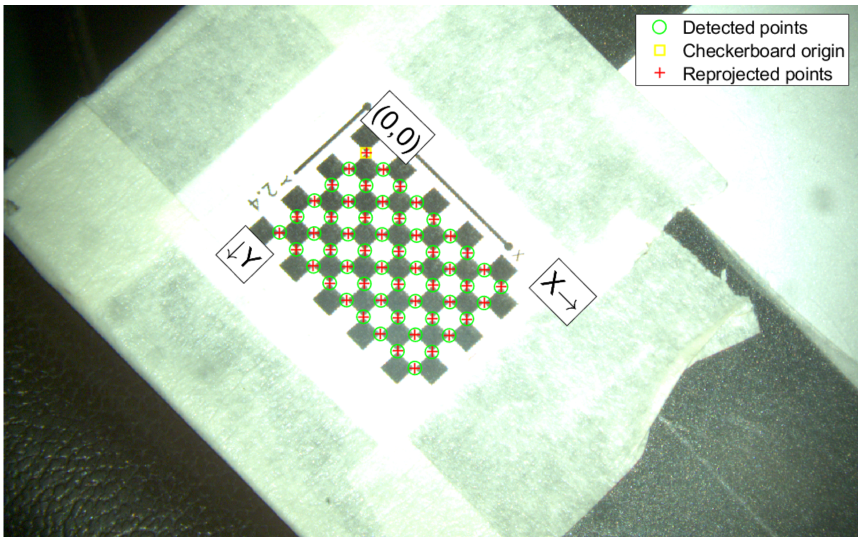


Figure 3.3: *Calibration of Camera 0*

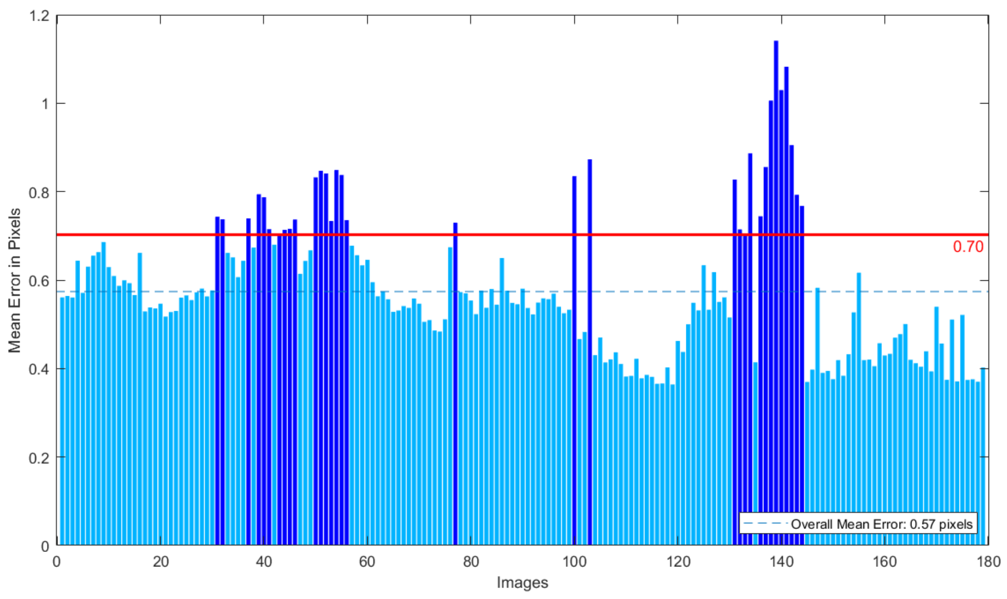


Figure 3.4: *The bar graph displays the mean reprojection error per image, along with the overall mean error. The bar labels correspond to the image IDs. The selected bar labels are the images that we delete*

Thanks to the described procedure, we obtain a good calibration of the cameras. In particual, we check the focal length, the rotation matrices and the intrinsic matrices to evaluate the calibration process.

3.2.2 Stereo Camera Calibrator

The calibration of the single cameras is important because we can use a fixed intrisic when we perform the *Stereo Camera Calibrator*. This type of calibration is useful because, in this way, we calculate the extrinsic patameters of the two cameras, i.e. the rotation and the traslation of the second camera with respect to the first one.

Using this MATLAB tool, we upload the same images of the chessboard used for the calbration of *Camera0* and *Camera1* coupling the images with the same point of view and, as previously done, we measure the chessboard square. This last value, in millimeters, is very important because we need it in the computation of the real world measures (see section 3.5). If the size of the chessboard square is correct then, the value of the baseline [Fig. 2.9] is in millimeters too.

Thus we forced the intrinsics of the two cameras previously calculated and then the stereo calibration is computed. We examine the reprojection errors of each image [Fig. 3.5] and, if the value is above the chosen threshold, the pair is delete.

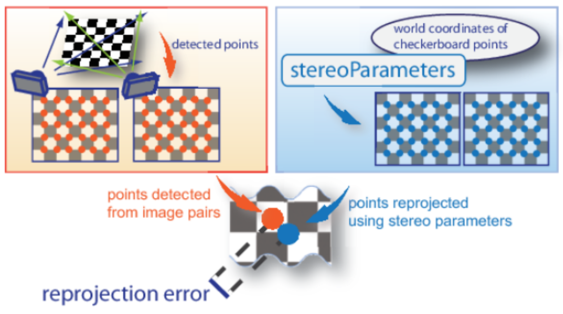

detected points
world coordinates of checkerboard points
stereoParameters
points detected from image pairs
points reprojected using stereo parameters
reprojection error

Figure 3.5: *Reprojection error*

3.3 Compute the Rectification

After the calibration and before the computation of the disparity map we need to rectify the pairs of images (see section 2.4).

So, we take two images from two cameras that capture the same scene but from two different point of view [Fig. 3.6] and put them in the rectification process [18]. Stereo image rectification projects images onto a common image plane in such a way that the corresponding points have the same row coordinates. This image projection makes the image appear as though the two cameras are parallel along the y direction .

The output of this process is an undistorted and rectified version of the left images, returned as an $M \times N \times 3$ truecolor image and an undistorted and rectified version of the right image, returned as an $M \times N \times 3$ truecolor image.

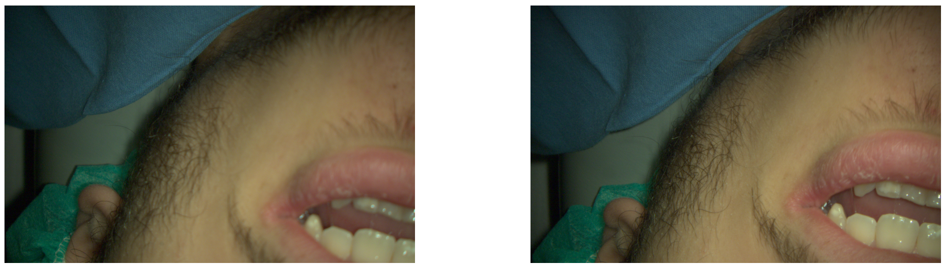


Figure 3.6: *Example of couple of images extract from left and right cameras, before the rectification*

The image pair, after the rectification, are displayed for convenience in a single plot. In particular, we combine left and right images into a red-cyan anaglyph (we also can view the output image with red-blue stereo glasses to see the stereo effect) [Fig. 3.7]. In this way, the rectification process can be evaluated checking if the two images are parallel, i.e. no differente in the vertical direction.

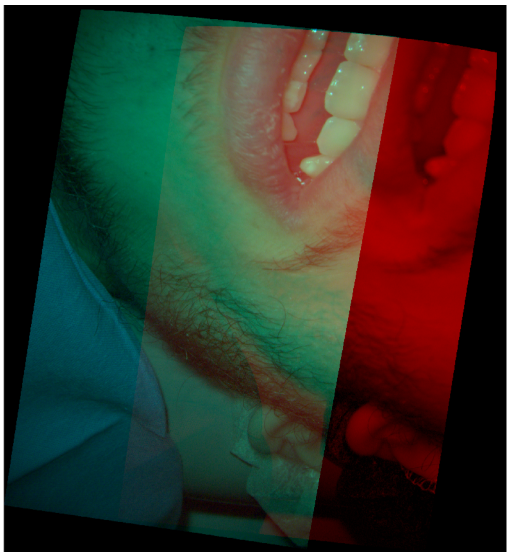


Figure 3.7: *Example of a red-cyan anaglyph*

For example, [Fig. 3.7] shows a difference in horizontal direction and only a very small difference in the vertical direction as result of a good rectificaiton process.

3.4 Disparity Map between Stereo Images

Now we must compute the disparity map.

First of all we convert the rectified images to grayscale and use them as input of the *disparity* function, then we compute the disparity using MATLAB disparity function. As we have alredy see in section 2.5, the function use the *Semi-Global Block Matching* algorithm. To improve the performance of the algorithm we can modify some parameters, like the *Range of disparity*, the *Contrast Threshold* and the *Distance Threshold.*

The disparity range depends on the distance between the two cameras and the distance between the cameras and the object of interest. The disparity range increases if the cameras are far apart and/or the objects are close to the cameras. In order to compute the value of the minimum distance, *mindis*, we have to take the disparity of a point far from the camera. Consequently to compute the value of the maximum distance, *maxdis*, we take a point near to the camera.

Defined a good set of parameters, the disparity map for a pair of stereo images, return an $M \times N$ 2-D grayscale image with the same size as the input images. Each element of the output specifies the disparity for the corresponding pixel in the image references as left image. The returned disparity values are rounded to $\frac{1}{16}$th pixel.

3.5 3-D Reconstruction of the Scene

The *Reconstruct Scene* function returns an array of 3-D world point coordinates that reconstruct a scene from a disparity map [18].

To implement the 3D reconstruction, the calibratior parameters used into the rectification process and the disparity map related to the stereo images are needed.

The disparity map can contain invalid values; these values correspond to pixels in the first image, which the disparity function did not match in the second image. The function sets the world coordinates corresponding to invalid disparity to *NaN*. Moreover the pixels with zero disparity correspond to world points that are too far away to measure, given the resolution of the camera, consequently the function sets the world coordinates corresponding to zero disparity to *Inf*.

The output is the coordinates of world points, returned as an $M \times N \times 3$ array. In particular the matrix $points3D(:,:,1)$ contains the x world coordinates of points corresponding to the pixels in the disparity map; $points3D(:,:,2)$ contains the y world coordinates, and $point3D(:,:,3)$ contains the z world coordinates. The third coordinate is very important because it tells us how far a pixel is from the camera in meters and it is crucial to reconstruct the scene in 3D.

In particular, taking as a reference [Fig. 2.9], we can compute the x,y and z world coordinates with this system:

$$\begin{cases} \mathbf{X} = -\dfrac{\mathbf{Z}x_1}{f} \\ \\ \mathbf{Y} = -\dfrac{\mathbf{Z}y_1}{f} \\ \\ \mathbf{Z} = \dfrac{fb}{x_1 - x_2} \end{cases} \tag{3.1}$$

where f is the focal length, b is the baseline, $(x_2 - x_1)$ is the disparity d, (x_1, y_1) are the coordinates of the first image, x_2 is the coordinate x of the second image and $\mathbf{X,Y,Z}$ are the world coordinates [12]. In particular the equations of this system are the inverse of the equation (2.6), with $x_1, y_1 = \mathsf{p_x}, \mathsf{p_y}$, for the first two equations and the inverse of the equation (2.41), with $x_1 - x_2 = d(\mathbf{x}_l)$, for the third equation.

This is possible only if we measure the focal length in pixel if the disparity is also in pixel; infact, making the dimensional analysis of (3.1), we have: $[\mathsf{pixel}][\mathsf{mm}]/[\mathsf{pixel}] = [\mathsf{mm}]$.

After the computation, we have all the parameters to reconstruct the scene.

Chapter 4

Results

In this chapter I specify the values of the parameters that we used and the results obtained.

4.1 Calibration

First of all, as we mentioned in section 3.2, after the upolad of the chessboad patter for the single and stereo calibration we need to exploit the size of the chessboad square. In the images that we used, we have a size of 2.4 [mm]. We upload 285 different images for each camera but the Calibrator tool accepts only 179 images for the *Camera0* and 187 images for the *Camera1*. For *Camera0*, the mean reprojection error of 0.57 pixel is eximated and it is considered an acceptable error [11]. However, thanks to the large amount of input images, the performance can be improved by eliminating all the images characterized by an error above 0.7 pixels. Deleting the bad images and after the recalibration, the mean error obtained is 0.52 pixels

Same procedure is applied for *Camera1* reducing the mean error from 0.71 pixels to 0.51 pixels.

Parameters obtained for *Camera0* and *Camera1* are, respectively:

1. Focal Lengths of $\begin{pmatrix} 8681.133 & 8623.96 \end{pmatrix}$ and $\begin{pmatrix} 8367.033 & 8327.11 \end{pmatrix}$
2. The Radial Distortion of $\begin{pmatrix} -1.189 & 15.879 \end{pmatrix}$ and $\begin{pmatrix} -0.656 & -1.989 \end{pmatrix}$
3. Intrinsic Matrices:

$$\begin{bmatrix} 8681.1 & 0 & 888.2 \\ 0 & 8623.9 & 921.6 \\ 0 & 0 & 1 \end{bmatrix} \ and \ \begin{bmatrix} 8367 & 0 & 1258 \\ 0 & 8327.1 & 937.4 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

The Tangential Distortion and the Skew are assumed to be 0 for both cameras. All this values are in [pixel].

Finally, we can compute the stereo calibration. So, we upload the 285 input couples of images and the tool accepts 168 pairs. As we have already said, we force the intrinsics of the two cameras recently calculated and then compute the stereo calibration. The mean reprojection error of the stereo calibration is around 0.72 pixels [Fig. 4.1]. In order to improve the performance of the stereo calibration, we manually deleted all the pair returning a mean errror above 0.7 in almost one image of the couple, obtaining a new mean error of 0.57 pixels.

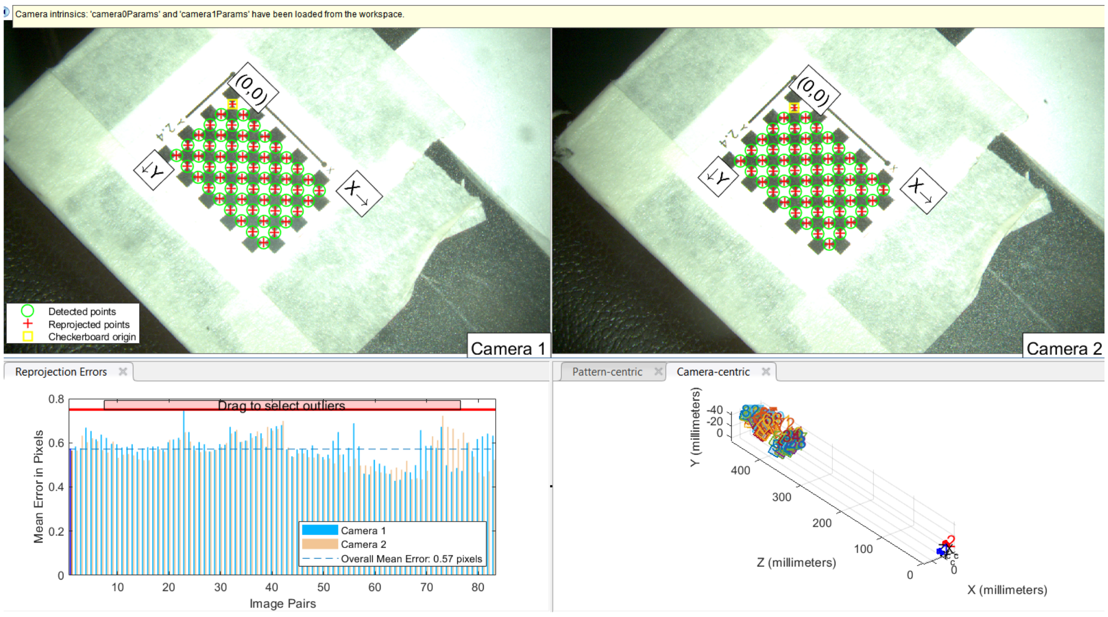

Camera intrinsics: 'camera0Params' and 'camera1Params' have been loaded from the workspace.

Figure 4.1: *Stereo Camera Calibrator tool. The Camera1 refers to the Camera0 and the Camera2 refers to the Camera1*

After the calculation we have the final stereo calibration useful for compute the next steps. In particular this results in a rotation matrix of the second camera with respect to the first one of:

$$\begin{bmatrix} 0.9989 & 0.0424 & 0.0192 \\ -0.0423 & 0.9991 & -0.0048 \\ -0.0194 & 0.0039 & 0.9998 \end{bmatrix} \tag{4.2}$$

4.2 Rectification

Now, we need to compute the rectification. So we extract from the video the couple of frame from the two cameras at the same time [Fig. 4.2] and put them in the rectification process.

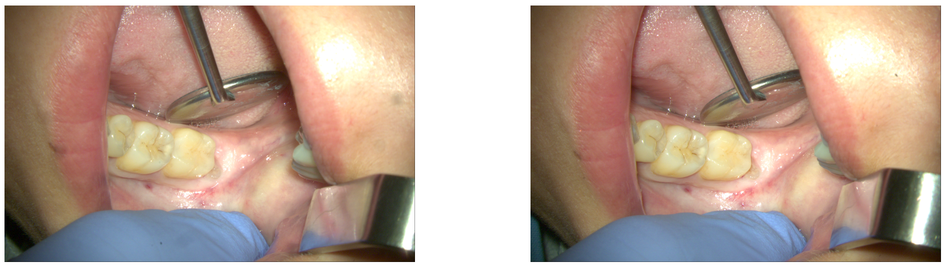


Figure 4.2: *Example of couple of frames extract from left and right cameras, before the rectification*

In [Fig. 4.3], an example of the rectification process is shown, the two images are parallel along the y coordinate.

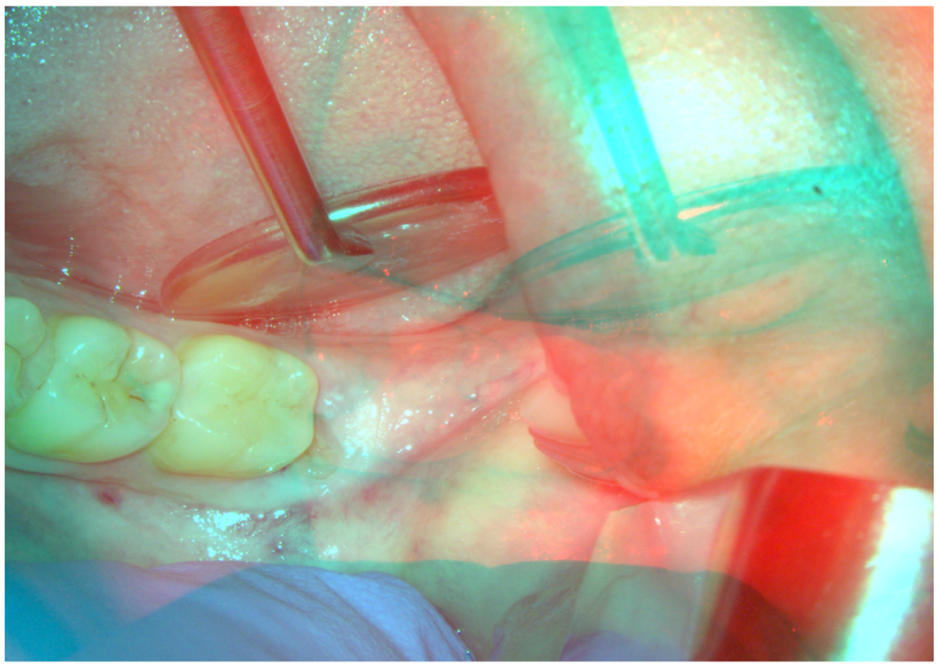


Figure 4.3: *Example of a red-cyan anaglyph*

This process is done for all frame pairs.

4.3 Disparity Map

After the rectification we must compute the disparity map for each pair. First of all we must set the parameters of the function, in particular the range of disparity. Thus we take the rectification plot of each couple and, thanks to *imdistline* [13], we calculate the values of *mindis* and *maxdis* In this set of frame, in order to compute the value of *mindis*, it is necessary to take the disparity of a point located far from the camera. In particular, a point in the mouth of the patient is chosen [Fig. 4.4] returning a value of 592 pixels. The value of the maximum distance (*maxdis*) is computed taking a point close to the camera. In particular, the choosen point is on the dentist's finger [Fig. 4.5] and the value returned is 656 pixels. We must choose a value for the range that is divisible by 16.

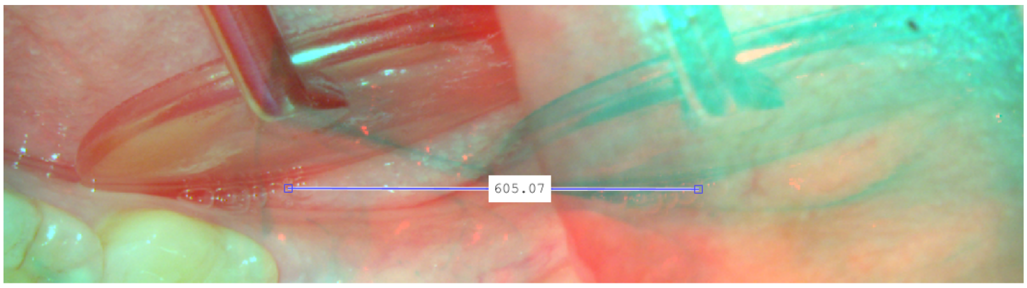


Figure 4.4: *Calculation of the minimum disparity*

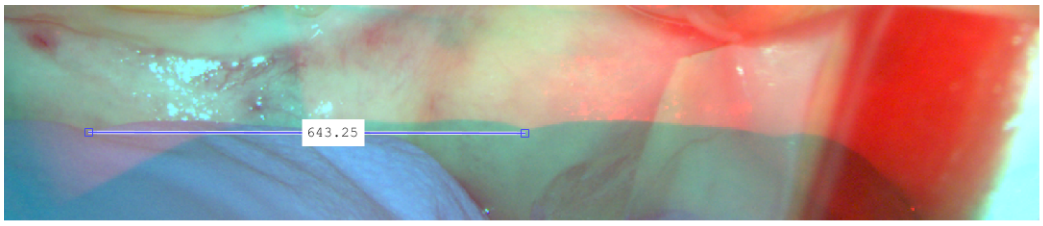


Figure 4.5: *Calculation of the maximum disparity*

In the set of video frames that we acquired, much attention is needed to take in to account the possibility that external factors may change the maximum and minimum distance evaluation. For example, during the operation, the dentist used several tools and the point closest/farther to the camera changes. Consequently the maximum disparity changes. Infact, in the following frames, we note that the value of *maxdis* change; in particular from the frame number 576 in the image appears a new tool [Fig. 4.6], this results in a new value of *maxdis* of 784 pixels.

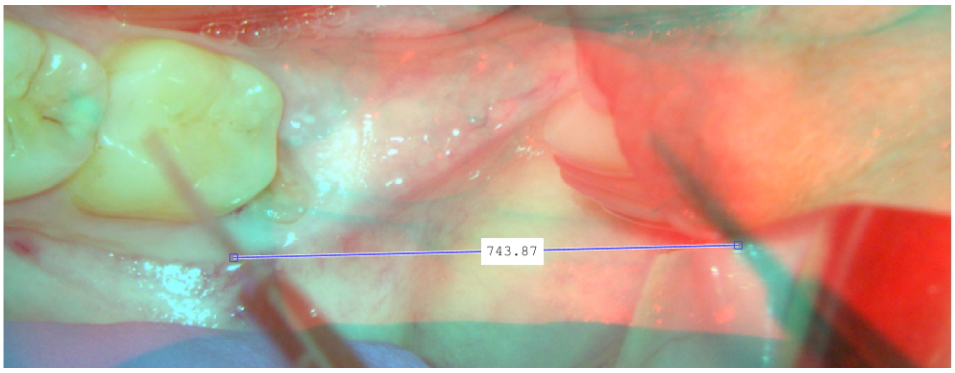


Figure 4.6: *Calculation of the maximum disparity after the appearance of the new tool*

These two parameters are the most important. We can vary the others based on the final result of the reconstruction of the scene.

After this we have a 1226 × 1739 disparity matrix for each couple of frame, [Fig. 4.7] and [Fig. 4.8], where each value exploit the distance along the x axis of each pixels of left image from the right image.

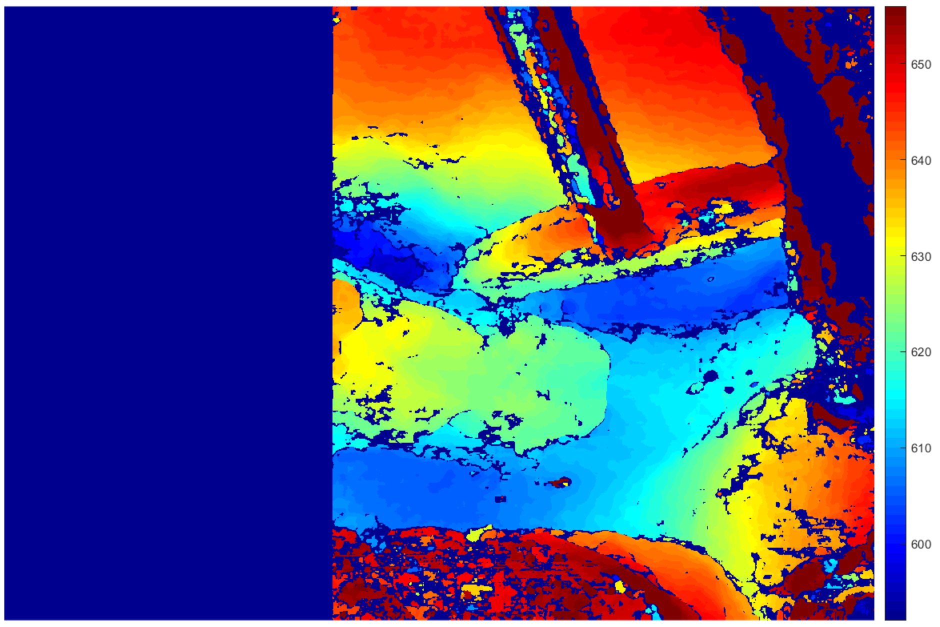


Figure 4.7: *Disparity map of the frame* 466

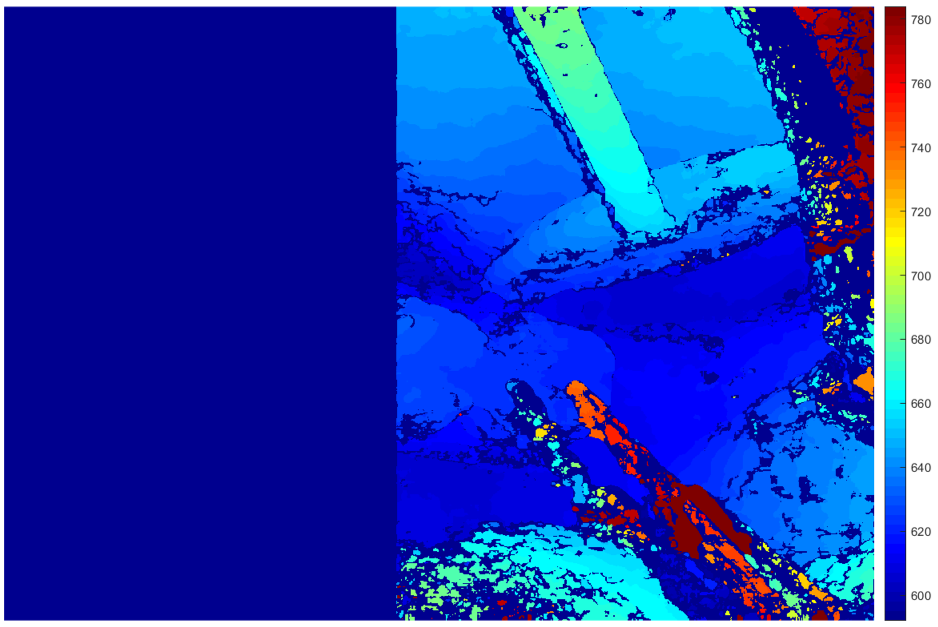


Figure 4.8: *Disparity map of the frame* 588

4.4 3D Reconstruction

Now we take this matrix, the stereo calibration parameters and we compute the 3D reconstruction of the scene for each frame.

After the computation we can see the final result in [Fig 4.9] and in [Fig. 4.10] for the first frame of the video. [Fig. 4.11] shows the goodness of the reconstruction procedure also when other tools are used by the doctor.

As mentioned in section 3.4 we compute the distance in [mm] of each pixel from the camera, but, as we can see in all the figures, the 3D reconstruction is degraded because the values used when we compute the distances do not represent the real image value. This is due to the fact that the images from which the 3D is extracted are modified. Infact, when we compute the rectification process, the rectified images are rescaled and the pixels of the resulting image come from pixel interpolation of the original image.

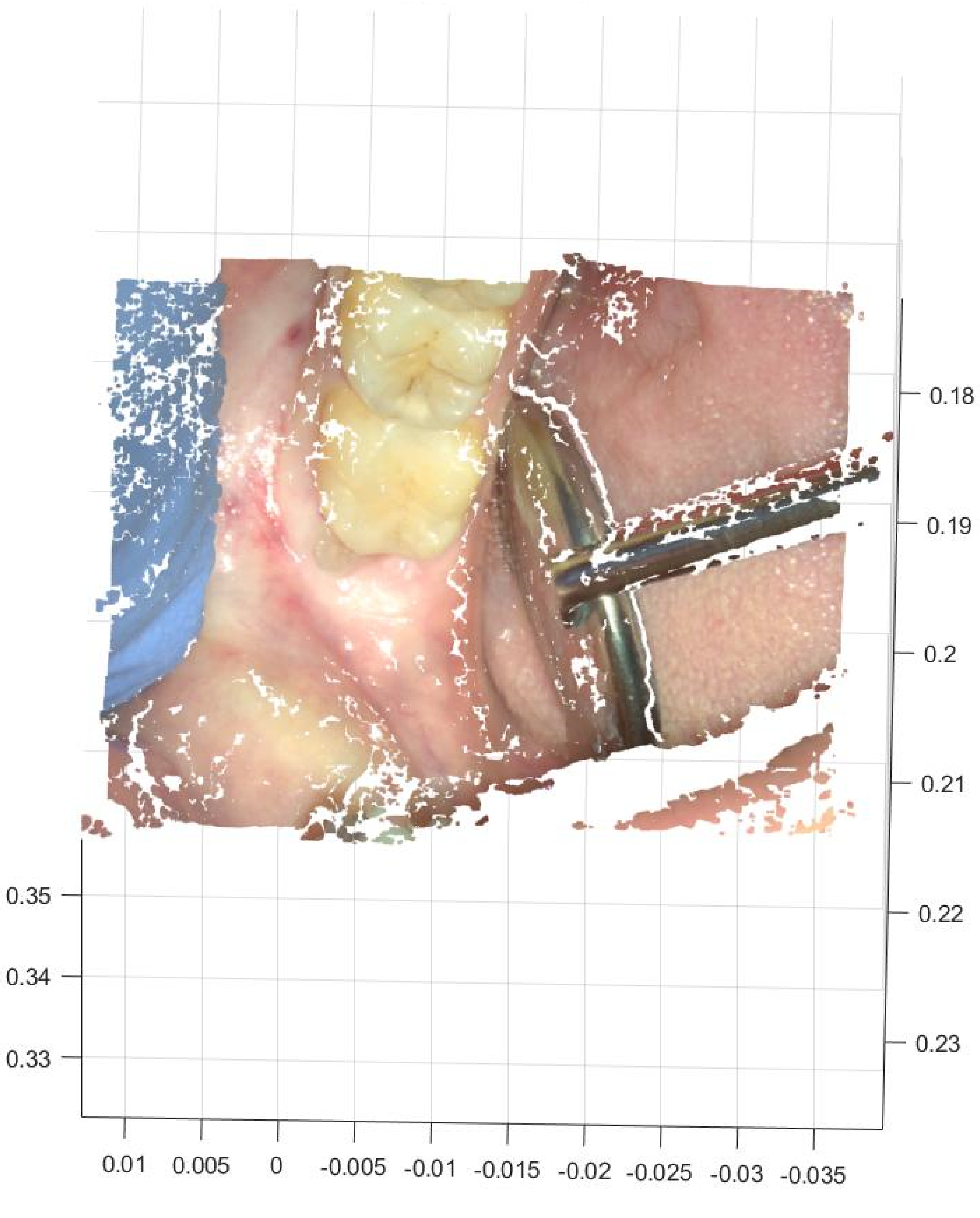


Figure 4.9: *3D reconstruction of the frame* 460

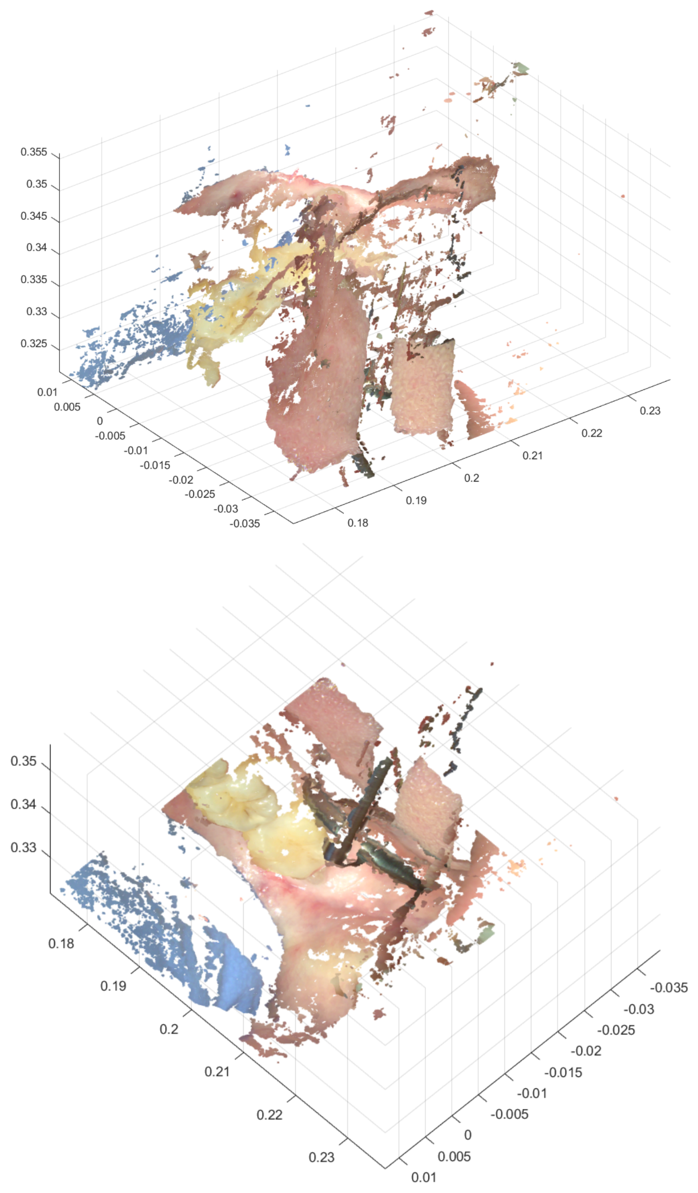


Figure 4.10: *Same 3D reconstruction from different POV*

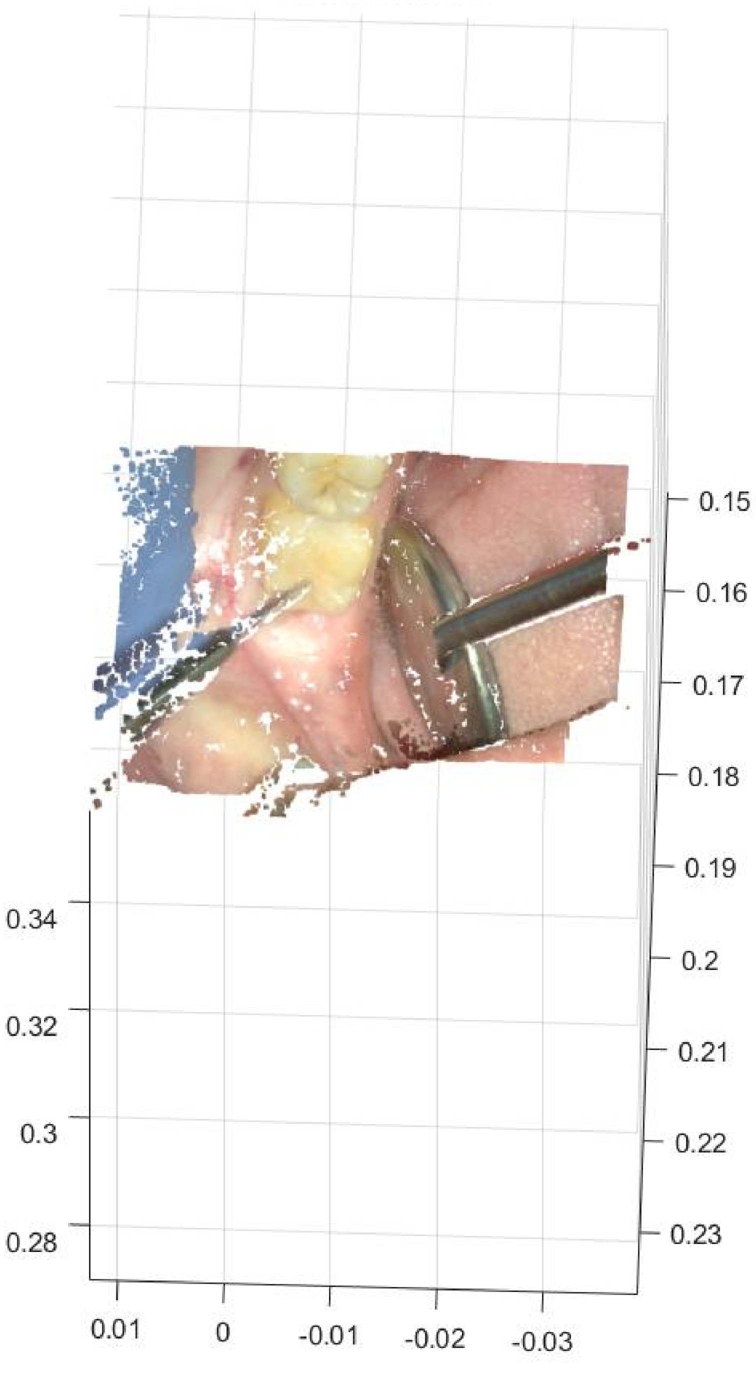


Figure 4.11: *3D reconstruction of the frame* 588

Conclusion

The purpose of this work is to project a system that can build a 3D reconstruction of the patient's mouth by acquiring stereo images from an odontoiatric microscope. This is useful because, thanks to 3D reconstruction, the dentist can notice a carcinoma in the patient's tongue or for teaching puproses.

The starting point is the acquisition of the stereo video thanks a 2-camera system, in particular the cameras adopted are a colours C-Mount CMOS digital cameras with a resolution of 1920×1200, that are mounted onto the two hole of the microscope. The position is congruent with what the dentist sees with his right and left eye.

So, in the first part of the work, we calibrated the two cameras individually and one with respect to the other; in this way, we obtain the parameters of a lens and image sensor of a video camera and thus correct the distortion due to the lens, measure the dimension of an object in the world units and determine the position of the cameras in the scene. This process is done through MATLAB, in particular we calibrate the two cameras individually with the App: *Camera Calibrator* and the stereo calibration with the App: *Stereo Camera Calibrator*. Using a total of 285 chessboard images, with a square size of 2.4 [mm], for each camera, we compute the intrisic parameters for the two camera individually, then we use this information to compute the stereo calibration using the 285 couples of images of the chessboard. This results in a rotation matrix of the second camera with respect to the first one different from the identity matrix. This means that the two video frame, take in the same time istant, have a different point of view because, in addition to being translated, they are also rotated. The rotation is not accepted for the 3D reconstruction, so we need to implement the rectification process.

The rectification is the process with whom the two images are modified in a way that, the only difference is the translation along the horizontal direction. However, this leads to a loss of information for 3D reconstruction because, when we compute the rectification process, the new images are rescaled and the pixels come from an interpolation of the pixels of the original image, thus we don't have the real image values. But, as we have just said, this process is necessary, so we take all the stereo frames from the video and we rectify the two images.

After this, we need to known the depth of each frame, thus we compute the disparity map. Infact, thanks to this map, we can extract depth information of the image; the process is the same as when we blink our eyes, one at a time, alternating between your left and right eye. The values in this map tell us the distance of the same pixel in the two different images, i.e. in the two different point of view. In particular, we convert the rectified images to grayscale and we find the best parameters to implement the Matlab function. The most important parameter is the disparity range, the values of the disparity map are inversely proportional to the real distance from the camera, thus, we are looking for a point far to the camera to compute the minumum disparity and inversely, we are looking for a point near to the camera to compute the maximum disparity. In particupar, we take, for the minimum disparity, a point on the patient's palate and, for the maximum disparity, a point on the dentist's finger. We noticed that, after some frame (in particular after the frame number 576), the dentist used a new tool, so the maximum disparity change. With all this information we have computed the disparity map of each frame [Fig. 4.12]

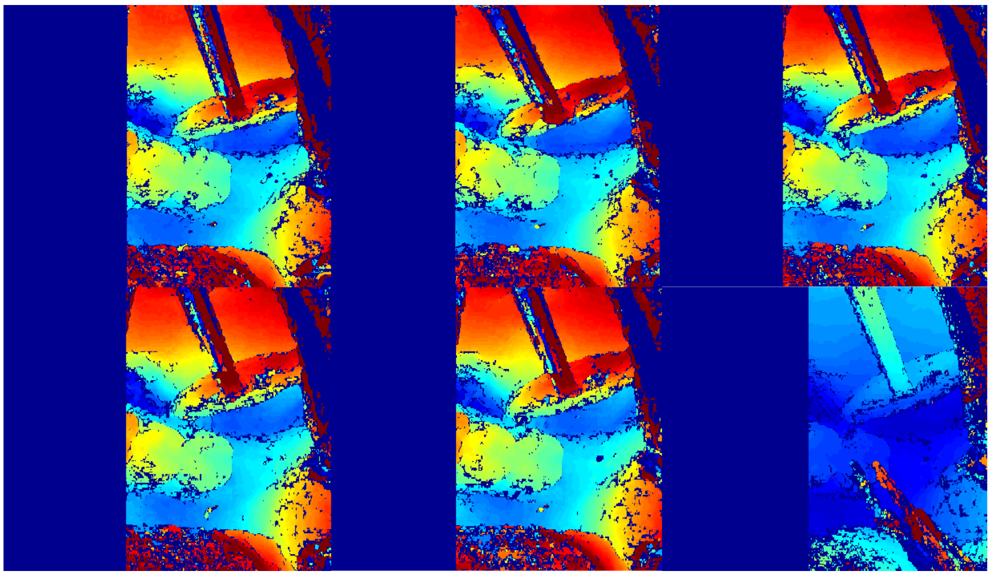


Figure 4.12: *Disparity map of different time istant*

Now we can compute the 3D reconstruction of the scene. Thanks to Matlab, throught the disparity map and the calibration, we have a good 3D reconstruction that, in particular, tell us the real distance from the camera of each pixel. This is not all true, infact, because of the rectification process, the 3D is degraded and the values of the depth ($point3D(:,:,3)$) in [mm] are not the real values.

	Depth Value [mm]
Patient's palate	347.6899
Dentist's tool	269.4527

How can wee see in the table above, we have a difference of about 0.1 [m] between the palate and the tool in the frame 588. This is a good approximation of the distance in the real world.

For a better 3D reconstruction it would be necessary to acquire the images so that the rectfication process was minimal, i.e. take frame from two cameras with, after the stereo calibration, a rotation matrix, of the second with respect to the first one, very similar to the identity matrix.

Bibliography

[1] Multiple view geometry in computer vision
R. Hartley and A. Zisserman, Cambridge university press, 2003.

[2] Camera resectioning, from Wikipedia, the free encyclopedia
`http://en.wikipedia.org/wiki/Camera_resectioning`

[3] What Is Camera Calibration?, MathWorks
`https://it.mathworks.com/help/vision/ug/camera-calibration.html`

[4] A flexible new technique for camera calibration
Z. Zhang, IEEE Transactions on Pattern Analysis and Machine Intelligence (pp. 1330–1334), Vol.22, No.11, 2000

[5] Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry
Abdel-Aziz, Y.I., Karara, H.M., Proceedings of the Symposium on Close-Range Photogrammetry (pp. 1-18), Falls Church, VA: American Society of Photogrammetry, 1971

[6] The Fundamental Matrix, Projective Geometry for Image Analysis
A Tutorial given at ISPRS, Vienna, July 1996
`http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MOHR_TRIGGS/node49.html`

[7] Cholesky decomposition, from Wikipedia, the free encyclopedia
`https://en.wikipedia.org/wiki/Cholesky_decomposition`

[8] Rectification, Dr. Gerhard Roth
`http://people.scs.carleton.ca/~c_shu/Courses/comp4900d/notes/rectification.pdf` (pp.9,10)

[9] Binocular disparity, from Wikipedia, the free encyclopedia
`https://en.wikipedia.org/wiki/Binocular_disparity`

[10] Small Vision Systems: Hardware and Implementation
Konolige, K., Proceedings of the 8th International Symposium in Robotic Research, (pp. 203-212), 1997.

[11] Single Camera Calibrator App, Evaluate Calibration Results, MathWorks
https://it.mathworks.com/help/vision/ug/single-camera-calibrator-app.html

[12] Reconstruction of three dimensional scenes, Paweł Pełczyński, Paweł Strumiłło
http://eletel.p.lodz.pl/pstrumil/po/3D_reconstruction.pdf

[13] imdistline, Distance tool, MathWorks
https://it.mathworks.com/help/images/ref/imdistline.html

[14] A combined corner and edge detector
C. Harris and M. Stephens, in Fourth Alvey Vision Conference, 1988

[15] Good features to track
J. Shi and C. Tomasi, in Computer Vision and Pattern Recognition, 1994.

[16] Automatic Camera and Range Sensor Calibration using a Single Shot
A. Geiger, F. Moosmann, O. Car, and B. Schuster, International Conference on Robotics and Automation (ICRA), St. Paul, USA, May 2012.

[17] Sobel Operator, from Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/Sobel_operator

[18] Learning OpenCV: Computer Vision with the OpenCV Library
G. Bradski and A. Kaehler, Sebastopol, CA: O'Reilly, 2008.