3
Computer Vision

3.1.
Introduction

The TA-40 manipulator studied in this work is attached to a ROV (Remote Operating Vehicle) that takes it to its work environment. Every time it reaches its work position, the relative position of the manipulator base needs to be calibrated before the end-effector position can be estimated relative to the work environment. The primary goal of this thesis is to find a way to estimate this position by the use of digital cameras mounted on the manipulator.

Any form of automation requires that the robot is capable of finding its position in the environment. When the position is estimated, the operator can get a feed-back by means of virtual reality that can give a more detailed and comprehensive overview of the work environment. Through the use of the inverse kinematic model certain tasks can also be automated.

In this chapter, the mathematic camera model will be described, as well as how the parameters are obtained through calibration. Then a description will be given about how features are extracted from the images and how they are recognized in other images.

3.2.
Mathematic Camera Models

3.2.1.
The Pinhole Model

The pinhole model is one of the simplest and most widely used camera models. It is very useful in computer vision, due to its simple linear geometric interpretation. A schematic interpretation of the model is shown in Figure 13. The pinhole represents the camera center. The distance from the pinhole to the image plane is called the focal distance, usually denoted as f.

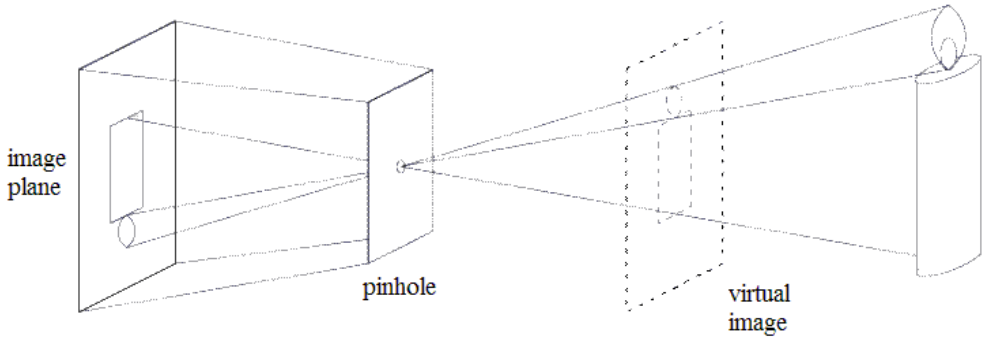

Figure 13 - Schematic representation of the pinhole model

Figure 14 shows the basic concept of the normalized pinhole model. The image coordinates in the normalized image plane are found by a simple geometric interpretation. The point *P* has the world coordinates *(x,y,z)* relative to the camera center or pinhole and is projected onto the normalized image plane in the point $\hat{p} = (\hat{u}, \hat{v})$. Their relationship can be expressed by the following equation:

$$\begin{cases} \hat{u} = \dfrac{x}{z} \\ \hat{v} = \dfrac{y}{z} \end{cases} \Leftrightarrow \hat{p} = \begin{bmatrix} \hat{u} \\ \hat{v} \\ 1 \end{bmatrix} = \frac{1}{z}\begin{bmatrix} Id & 0 \end{bmatrix}\begin{bmatrix} P \\ 1 \end{bmatrix} \qquad (46)$$

.

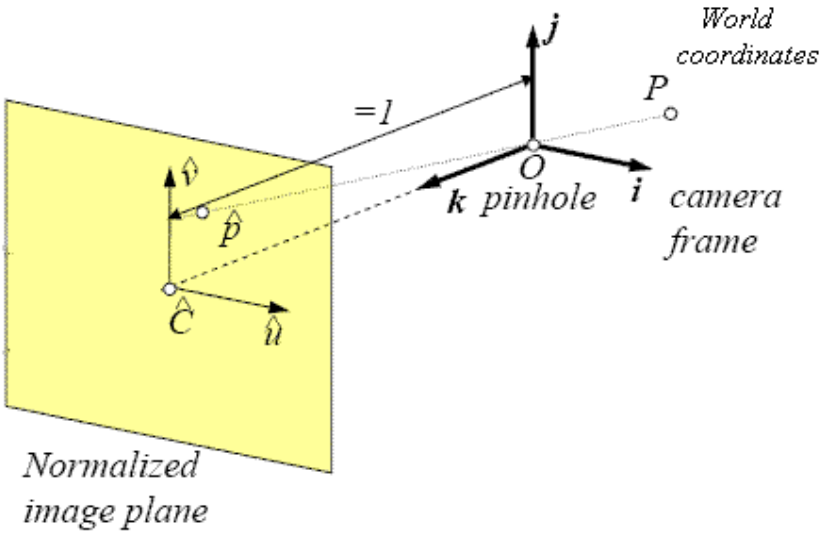

Figure 14 – Geometry of the pinhole model [14]

3.2.2. Intrinsic Parameters

Equation (49) assumes an ideal camera, where the optical axis (axis going through the center of the lens coincides with the center of the photo sensors in the image plane. It also assumes that the photo censors form a perfect square. Unfortunately, real cameras have certain discrepancies that deviate from this model. Therefore a camera has to be calibrated to estimate its intrinsic parameters. Figure 15 shows how the shape of a CCD panel can deviate from the ideal square shape. Figure 16 shows the modified pinhole model, incorporating the shape of the CCD panel.

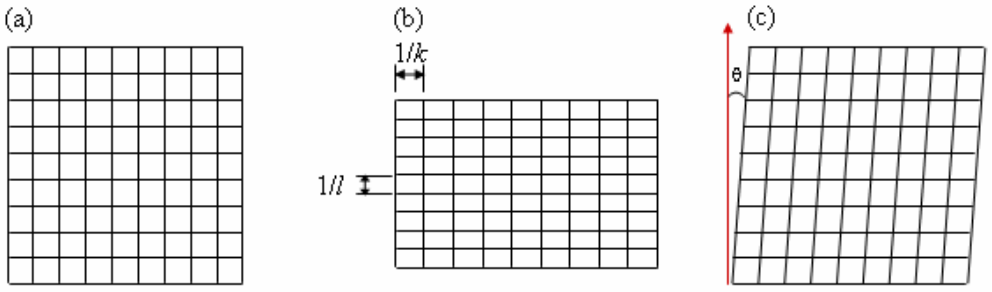

Figure 15 - CCD layout. (a) shows an ideal square, (b) shows that the scale in x and y direction can differ, (c) shows that the axes might not be perpendicular [14].

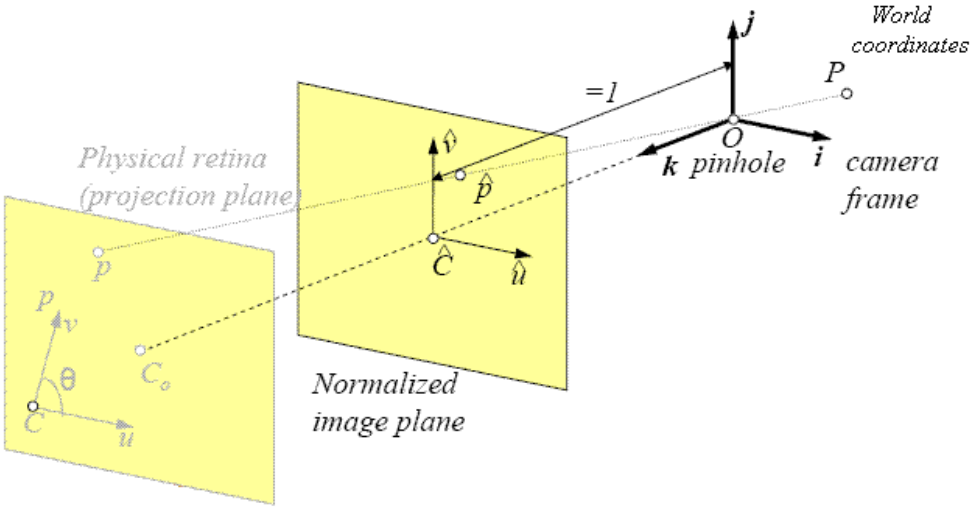

Figure 16 - The modified pinhole model. [14]

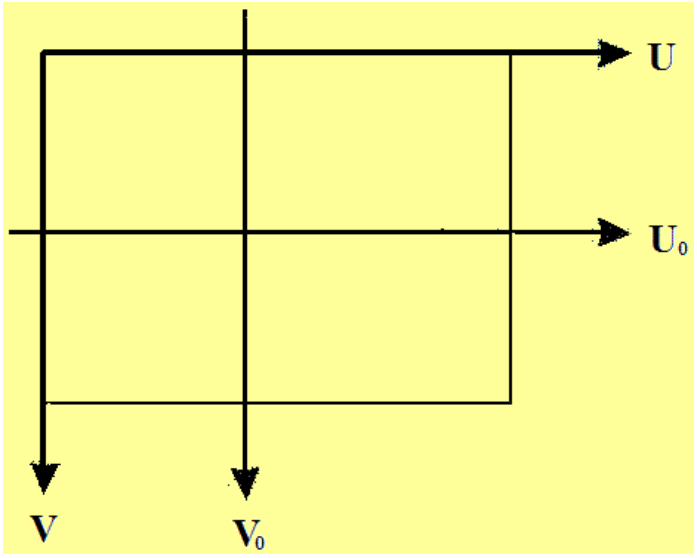

Figure 17 - The image center is not always in the middle of the sensor since the lens normal does not intersect with the middle of the sensor panel.

Taking into account that the scaling between the sensor axes might be different, defining *1/k* and *1/l* being the dimensions in the horizontal and vertical direction, gives:

$$\begin{cases} u = k\, f\, \hat{u} \\ v = l\, f\, \hat{v} \end{cases} \tag{47}$$

Then translating the coordinates relative to the image center (u_0, v_0):

$$\begin{cases} u = \alpha\, \hat{u} + u_0 \\ v = \beta\, \hat{v} + v_0 \end{cases} \tag{48}$$

where $\alpha = kf$ and $\beta = lf$

Taking into account that the sensors can be distorted by an angle θ, gives:

$$\begin{cases} u = \alpha\,\hat{u} - \alpha\cot\theta\,\hat{v} + u_0 \\ v = \beta\,\hat{v} + v_0 \end{cases} \tag{49}$$

All these intrinsic parameters define the calibration matrix of the camera which relates the actual image coordinates, p, with the ideal normalized coordinates, $\hat{p}$.

$$\boldsymbol{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & -\alpha\cot\theta & u_0 \\ 0 & \beta/\sin\theta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{v} \\ 1 \end{pmatrix} = \boldsymbol{K}\ \hat{\boldsymbol{p}} \tag{50}$$

Estimating the normalized pinhole coordinates is important in computer vision since they give the true line of sight to any object in the image. The normalized image coordinates can then be found by:

$$\hat{p} = K^{-1} p \tag{51}$$

3.2.3. Extrinsic Parameters

The previous section described how an object with known coordinates relative to the camera center is projected onto image plane. Often the coordinates of an object are given with respect to another reference frame than the actual camera center. Therefore, it is necessary to estimate how the other reference frame is expressed in terms of the camera reference frame. Figure 18 shows a schematic interpretation of the two reference frames.

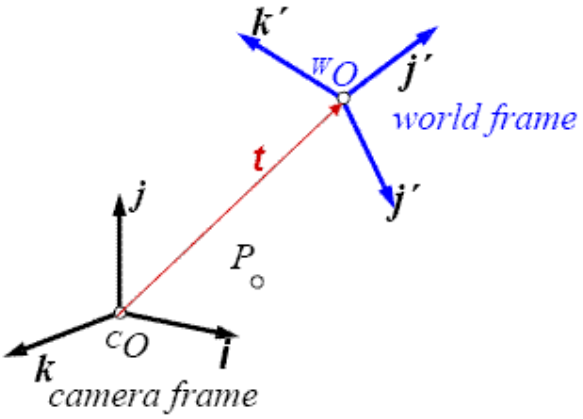

Figure 18 – Transformation of world coordinates to camera coordinates. [14]

The transformation can be expressed by a homogeneous matrix:

$$ {}^{C}P = \begin{pmatrix} {}^{C}_{W}R & {}^{C}O_{W} \\ 0^{T} & 1 \end{pmatrix} {}^{W}P = \begin{pmatrix} {}^{C}_{W}R & t \\ 0^{T} & 1 \end{pmatrix} {}^{W}P \qquad (52) $$

The six parameters describing the translation and rotation are called the extrinsic parameters of the camera. By combining Eqs. (46), (50) and (52), the projection matrix (M) is obtained:

$$ p = \frac{1}{z}\begin{pmatrix} K & 0 \end{pmatrix} \begin{pmatrix} {}^{C}_{W}R & t \\ 0^{T} & 1 \end{pmatrix} {}^{W}P = \frac{1}{z} K \begin{pmatrix} {}^{C}_{W}R & t \end{pmatrix} {}^{W}P \qquad (53) $$

$$ M = K \begin{pmatrix} {}^{C}_{W}R & t \end{pmatrix} \rightarrow p = \frac{1}{z} M \, {}^{W}P \qquad (54) $$

The projection matrix gives the relationship between the image coordinates and any 3D coordinate in the reference frame used.

3.3. Camera Calibration

In order to use a camera as an accurate measuring instrument, all the intrinsic parameters need to be found by calibration. There are several calibration algorithms available. The most common way is to estimate the projection matrix (M) through a least square estimate, using known 3D coordinates as inputs together with their corresponding image projections. The calibration procedure

used to find the intrinsic parameters for the pin-hole model in this thesis is based on the method described in [15]. Getting a good result is highly dependent on the accuracy of the measured coordinates. To perform the calibration, a calibration rig is generally used with many distinct and easily recognized coordinates, like for example a chess board pattern. A typical calibration rig is shown in Figure 19.

The coordinates can be found manually or automatically. An automatic detection algorithm is faster and potentially more accurate than the manual method. In this thesis, a semi automatic calibration technique is used, that uses a k-mean line fitting technique to find the exact coordinates of the corners.

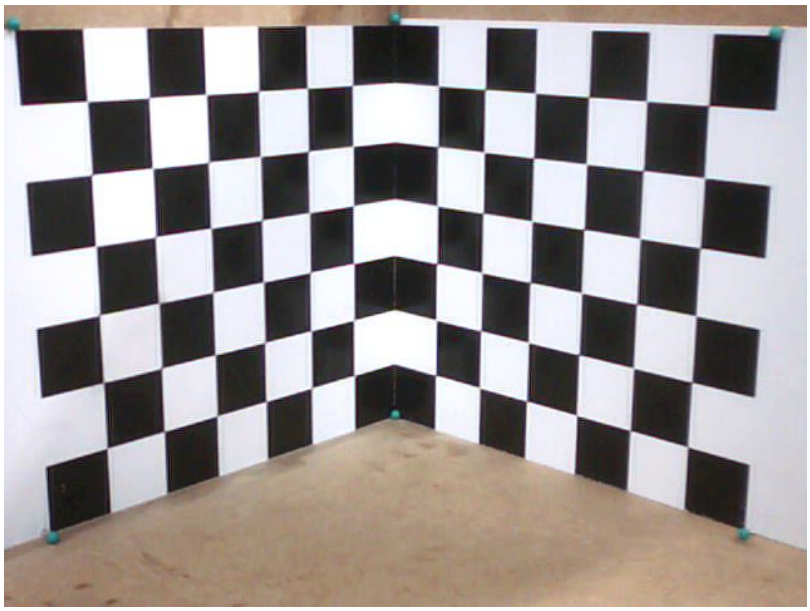

Figure 19 – Calibration rig

Figure 20 shows a geometric representation of the calibration procedure.

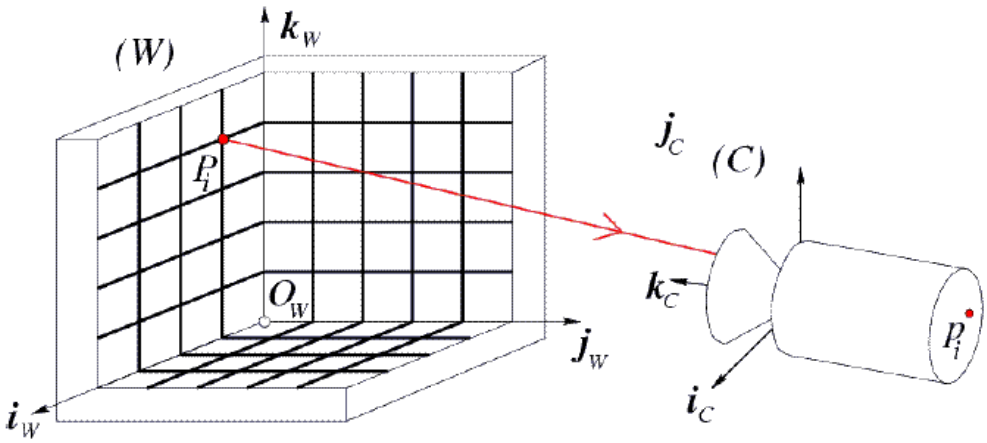

Figure 20 - Transformation from world coordinates to picture coordinates. [14]

The calibration procedure starts by estimating the projection matrix from a set of known world coordinates ${}^{w}P = \begin{bmatrix} {}^{w}X & {}^{w}Y & {}^{w}Z \end{bmatrix}^{T}$ with their corresponding image coordinates (*u,v*). The projection matrix is then estimated using a least square approximation.

Having n image coordinates (corners in the calibration rig) and their corresponding 3D coordinate pairs, the observation matrix is obtained:

$$L_o = \begin{bmatrix} {}^{w}X_1 & {}^{w}Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -{}^{w}X_1u_1 & -{}^{w}Y_1u_1 & -{}^{w}Z_1u_1 & -u_1 \\ 0 & 0 & 0 & 0 & {}^{w}X_1 & {}^{w}Y_1 & {}^{w}Z_1 & 1 & -{}^{w}X_1v_1 & -{}^{w}Y_1v_1 & -{}^{w}Z_1v_1 & -v_1 \\ : & : & : & : & : & : & : & : & : & : & : & : \\ {}^{w}X_i & {}^{w}Y_i & {}^{w}Z_i & 1 & 0 & 0 & 0 & 0 & -{}^{w}X_iu_i & -{}^{w}Y_iu_i & -{}^{w}Z_iu_i & -u_i \\ 0 & 0 & 0 & 0 & {}^{w}X_i & {}^{w}Y_i & {}^{w}Z_i & 1 & -{}^{w}X_iv_1 & -{}^{w}Y_iv_i & -{}^{w}Z_iv_i & -v_i \\ : & : & : & : & : & : & : & : & : & : & : & : \\ {}^{w}X_n & {}^{w}Y_n & {}^{w}Z_n & 1 & 0 & 0 & 0 & 0 & -{}^{w}X_nu_n & -{}^{w}Y_nu_n & -{}^{w}Z_nu_n & -u_n \\ 0 & 0 & 0 & 0 & {}^{w}X_n & {}^{w}Y_n & {}^{w}Z_n & 1 & -{}^{w}X_nv_n & -{}^{w}Y_nv_n & -{}^{w}Z_nv_n & -v_n \end{bmatrix} \quad (55)$$

The projection matrix, *M*, can be written as:

$$M = \begin{bmatrix} m_1 & m_2 & m_3 & m_4 \\ m_5 & m_6 & m_7 & m_8 \\ m_9 & m_{10} & m_{11} & m_{12} \end{bmatrix} \quad (56)$$

Ideally, the elements of the projection matrix should satisfy the following equation.

$$L_o \cdot M_v^T = L_o \cdot \begin{bmatrix} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} & m_{11} & m_{12} \end{bmatrix}^T = 0 \quad (57)$$

where M_v represents the projection matrix in vector form.

However, this is not possible to achieve, so the projection matrix has to be found by a least square estimate, meaning finding the minimum of the expression: $\min\left(M_v L_o^T L_o M_v^T\right)$

The solution to this problem is given by:

$$M_v = eig\left(L_o^T L_o\right) \quad (58)$$

where *eig* means the eigen vector corresponding to the smallest eigen value.

Once the projection matrix has been found, the intrinsic parameters can be estimated. The estimated projection matrix has to be multiplied by a scaling factor to give the projection matrix of Eq. (54).

$$\rho M = K\begin{pmatrix} R & t \end{pmatrix} = K\begin{pmatrix} r_1 & t_x \\ r_2 & t_y \\ r_3 & t_z \end{pmatrix} \tag{59}$$

where ρ is the scaling factor, r_1, r_2 and r_3 are the rows of the rotation matix.

The intrinsic parameters in the calibration matrix, *K*, can be found with the following procedure. The procedure is deduced in [15].

First finding the scaling factor, ρ.

$$\rho = \frac{\varepsilon}{\left\| \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix} \right\|}, \quad \varepsilon = \pm 1 \tag{60}$$

The first row of the rotation matrix between the camera reference frame and the calibration rig is given by:

$$r_3 = \rho \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix}, \quad \varepsilon = \pm 1 \tag{61}$$

The horizontal coordinate of the image center is:

$$u_0 = \rho^2 \begin{bmatrix} m_1 & m_2 & m_3 \end{bmatrix} \begin{bmatrix} m_9 \\ m_{10} \\ m_{11} \end{bmatrix} \tag{62}$$

The vertical coordinate of the image center is:

$$v_0 = \rho^2 \begin{bmatrix} m_5 & m_6 & m_7 \end{bmatrix} \begin{bmatrix} m_9 \\ m_{10} \\ m_{11} \end{bmatrix} \tag{63}$$

The distortion angle, θ, is:

$$\theta = \cos^{-1}\left(-\frac{\left(\begin{bmatrix} m_1 & m_2 & m_3 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix}\right) \cdot \left(\begin{bmatrix} m_5 & m_6 & m_7 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix}\right)}{\left\| \begin{bmatrix} m_1 & m_2 & m_3 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix} \right\| \left\| \begin{bmatrix} m_5 & m_6 & m_7 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix} \right\|} \right) \tag{64}$$

The magnification in *x*-direction is:

$$\alpha = \rho^2 \left\| \begin{bmatrix} m_1 & m_2 & m_3 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix} \right\| \sin\theta \tag{65}$$

The magnification in *y*-direction is:

$$\beta = \rho^2 \left\| \begin{bmatrix} m_5 & m_6 & m_7 \end{bmatrix} \times \begin{bmatrix} m_9 & m_{10} & m_{11} \end{bmatrix} \right\| \sin\theta \tag{66}$$

3.3.1.
Radial Distortion

Due to the fact that the projection through a lens is not linear, the projected image coordinates will often suffer from non linear distortion. In some cases the distortion is small so that the linear pin-hole model is sufficient to estimate the relationship between world coordinates and the image coordinates. Other cameras do have big distortion effects. The effect is most evident on wide angle lenses where the light travels through the lens at a larger angle of incidence. A thick lens or glass covering the camera will aggravate the effect. This may be evident in cameras used at great sea depths. The radial distortion can be either barrel shaped or pin cushion shaped. Examples of these two types of distortion are shown in Figure 21.

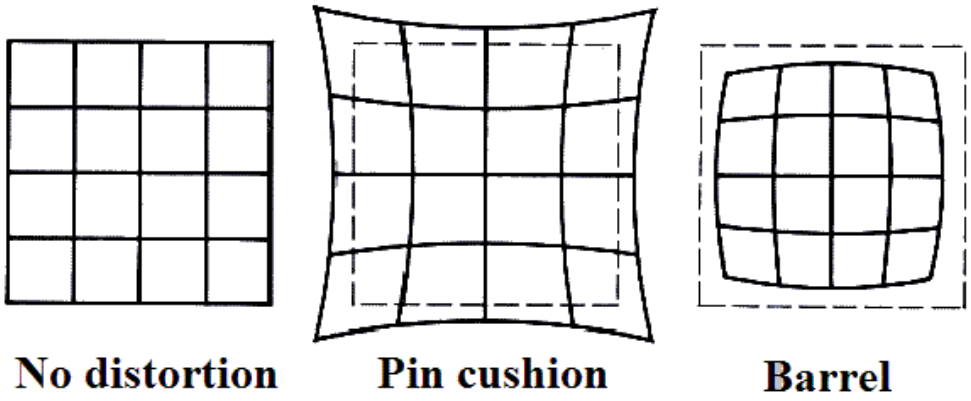

Figure 21 - Types of radial distortion.

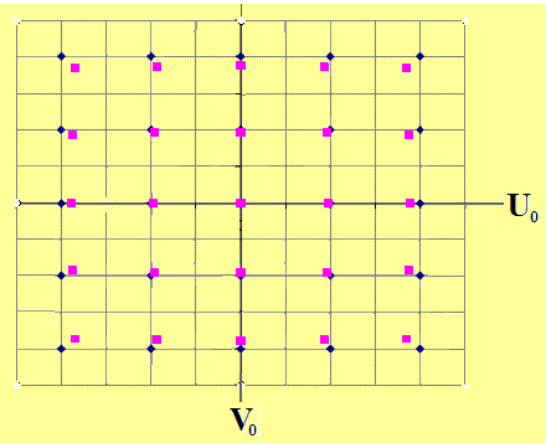

Figure 22 – Principle of barrel distortion. The black coordinates represent the image coordinates for a camera without distortion. The violet coordinates show the distorted image coordinates.

The procedure to estimate the effect of radial distortion is taken from [16]. When the skew angle is small, the ideal image coordinates can be estimated by the equations:

$$\breve{u} = u + (u - u_0)\left(k_1\left(\hat{u}^2 + \hat{v}^2\right) + k_2\left(\hat{u}^2 + \hat{v}^2\right)^2\right) \tag{67}$$

$$\breve{v} = v + (v - v_0)\left(k_1\left(\hat{u}^2 + \hat{v}^2\right) + k_2\left(\hat{u}^2 + \hat{v}^2\right)^2\right) \tag{68}$$

Where $(\hat{u}, \hat{v})$ are the ideal nonobservable normalized pinhole coordinates, (u, v) are the ideal nonobservable image coordinates estimated by the linear pinhole model, $(\breve{u}, \breve{v})$ are the corresponding observable distorted image coordinates, and (u_0, v_0) are the estimated image center coordinates. The ideal image coordinates can be given by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z} MP = \frac{1}{z}\left({}^{c}_{w}R \quad t\right)P \tag{69}$$

The ideal normalized pinhole coordinates, $\hat{u}$ and $\hat{v}$, refer to the respective 3D world coordinates given by:

$$\begin{bmatrix} \hat{u} \\ \hat{v} \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{70}$$

The parameters can then be estimated using alternation:

$$D_r k_r = d_r \tag{71}$$

Equations (66) and (67) can be given in matrix form:

$$\begin{bmatrix} (u-u_0)(\hat{u}^2+\hat{v}^2) & (u-u_0)(\hat{u}^2+\hat{v}^2)^2 \\ (v-v_0)(\hat{u}^2+\hat{v}^2) & (v-v_0)(\hat{u}^2+\hat{v}^2)^2 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} \breve{u}-u \\ \breve{v}-v \end{bmatrix} \tag{72}$$

Given *m* points and *n* images, the equations can be stacked together to obtain *2mn* equations. The least-square solution is given by:

$$k_r = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \left(D_r^T D_r\right)^{-1} D_r^T d_r \tag{73}$$

This procedure is repeated until it converges.

3.3.2. Sophisticated calibration

This section gives a description of a more sophisticated calibration method. Generally, if a camera is built symmetrically, the effect of radial distortion is enough to develop an adequate mathematical model for the camera projection. However, in some cases the radial distortion is not sufficient. This can be due to an imprecise construction of the camera or by the fact that the camera is covered by a casing with asymmetric characteristics. To deal with this complex distortion, a more sophisticated calibration method needs to be applied. For this application a calibration method presented in [17], is more appropriate. It presents a nonlinear, iterative procedure to minimize the squared error

$$F = \sum_{i=1}^{N} \left(\left(\tilde{u}_i - \breve{u}_i \right)^2 + \left(\tilde{v}_i - \breve{v}_i \right)^2 \right) \tag{74}$$

where $(\overset{\vee}{u}_i, \overset{\vee}{v}_i)$ are the estimated image coordinates using the mathematical model and $(\tilde{u}_i, \tilde{v}_i)$ are the observed image coordinates. The mathematical model is quite complicated, using fifth order polynomials to estimate the distortion.

$$\overset{\vee}{u}_i = s_u f\left(k_2\hat{v}_i^5 + 2k_2\hat{u}_i^3\hat{v}_i^2 + k_2\hat{u}_i\hat{v}_i^4 + k_1\hat{u}_i^3 + k_1\hat{u}_i\hat{v}_i^2 + 3p_2\hat{u}_i^2 + 2p_1\hat{u}_i\hat{v}_i + p_2\hat{v}_i^2 + \hat{u}_i\right) + u_0 \quad (75)$$

$$\overset{\vee}{v}_i = f\left(k_2\hat{v}_i^5 + 2k_2\hat{u}_i^2\hat{v}_i^3 + k_2\hat{u}_i^4\hat{v}_i + k_1\hat{u}_i^2\hat{v}_i + k_1\hat{v}_i^3 + p_1\hat{u}_i^2 + 2p_2\hat{u}_i\hat{v}_i + 3p_1\hat{v}_i^2 + \hat{v}_i\right) + v_0 \quad (76)$$

Having the observed image coordinates $(\tilde{u}_i, \tilde{v}_i)$ and the estimated image center coordinates (u_0, v_0), the optimization is done using the Levenberg-Marquardt algorithm. The iterative method stipulates that some camera parameters are known. The resolution of the image sensor and its size needs to be given. Also an initial guess of the focal length needs to be given in order to give a good starting point for the iterations.

Using the equations (75) and (76), the image coordinates can be estimated from a given 3D coordinate relative to the camera. However, in computer vision the coordinates $(\hat{u}_i, \hat{v}_i)$ are the ones of interest. It is therefore necessary to find a way to estimate $(\hat{u}_i, \hat{v}_i)$ from $(\tilde{u}_i, \tilde{v}_i)$. An implicit camera model results in:

$$\begin{bmatrix}\hat{u}_i \\ \hat{v}_i\end{bmatrix} = \frac{1}{G}\begin{bmatrix}u_i^* + u_i^*\left(g_1 r_i^2 + g_2 r_i^4\right) + 2g_3 u_i^* v_i^* + g_4\left(r_i^2 + 2u_i^{*2}\right) \\ v_i^* + v_i^*\left(g_1 r_i^2 + g_2 r_i^4\right) + g_3\left(r_i^2 + 2v_i^{*2}\right) + 2g_4 u_i^* v_i^*\end{bmatrix} \quad (77)$$

where

$$G = \left(g_5 r_i^2 + g_6 u_i^* + g_7 v_i^* + g_8\right) r_i^2 + 1 \quad (78)$$

$$u_i^* = \frac{\tilde{u}_i - u_0}{s_u f}, v_i^* = \frac{\tilde{v}_i - v_0}{f} \quad (79)$$

$$u_i^* = \frac{\tilde{u}_i - u_0}{s_u f}, v_i^* = \frac{\tilde{v}_i - v_0}{f} \quad (80)$$

f is the nominal focal length of the camera, s_u is the aspect ratio of the image. The vector $\mathbf{g}=\begin{bmatrix} g_1 & g_2 & g_3 & g_4 & g_5 & g_6 & g_7 & g_8 \end{bmatrix}$ is estimated by a least square approximation.

Defining the observation vectors:

$$U_i = \begin{bmatrix} -u_i^{*2} r_i^2 & -u_i^* r_i^4 & -2u_i^* v_i^* & -\left(r_i^2 + 2u_i^{*2}\right) & u_i^* r_i^4 & \hat{u}_i u_i^* r_i^2 & \hat{u}_i v_i^* r_i^2 & \hat{u}_i r_i^2 \end{bmatrix} \quad (81)$$

$$V_i = \begin{bmatrix} -v_i^{*2} r_i^2 & -v_i^* r_i^4 & -\left(r_i^2 + 2v_i^{*2}\right) & -2u_i^* v_i^* & v_i^* r_i^4 & \hat{v}_i u_i^* r_i^2 & \hat{v}_i v_i^* r_i^2 & \hat{v}_i r_i^2 \end{bmatrix} \quad (82)$$

$$T = \begin{bmatrix} U_1 & V_1 & \ldots & U_i & V_i & \ldots & U_N & V_N \end{bmatrix} \quad (83)$$

where

$$e = \begin{bmatrix} u_1^* - \hat{u}_1 & v_1^* - \hat{v}_1 & \ldots & u_1^* - \hat{u}_1 & v_1^* - \hat{v}_1 & \ldots & u_N^* - \hat{u}_N & v_N^* - \hat{v}_N \end{bmatrix} \quad (84)$$

then

$$e = Tg \quad (85)$$

$$\hat{g} = \left(T^T T\right)^{-1} T^{-1} e \quad (86)$$

When the vector $\hat{g}$ is estimated, the ideal coordinates of the image can be estimated using Eq.(86). However, the output of the calibration software gives the ideal image coordinates for a nondistorted image, (u, v). The ideal normalized coordinates can be estimated by Eq. (87).

$$\begin{bmatrix} \hat{u}_i \\ \hat{v}_i \\ 1 \\ \end{bmatrix} = \begin{bmatrix} fs_u & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \quad (87)$$

3.3.3. Coordinate extraction

The coordinates of the corners of the calibration rig are most easily extracted using an algorithm that automatically detects the edges of the image and finds the coordinates of the corners. The image of the calibration rig offers strong contrasts in the chess patterned surface of the rig. This means that the edges form distinct lines that can be found using edge detection. The corners can be found using a Harris corner detector, however this method has proven to be inaccurate because the strength of the edges vary so that the detected corner does not fall exactly in the crossing between two edges. This is the incentive for the proposed extraction method using nonmaximum suppression together with k-means line fitting to extract the coordinates. The procedure starts by manually finding the six outer corners of the calibration rig. An initial projection matrix can be estimated from these six coordinates. From the known geometric structure of the calibration rig, the approximate coordinates of all the corners can be estimated using Eq. (54). When the approximate coordinates are found, k-means line fitting is used to estimate the actual position of the corners.

3.3.4. Nonmaximum Suppression

Nonmaximum suppression is a technique that creates a binary map representing of the edges in an image. The image is first filtered with a Gaussian filter to eliminate noise. The gradients of the image are then estimated in x and in y direction using the Sobel mask. The Sobel mask used to estimate the gradient component in x direction is given below:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{88}$$

The Sobel mask used to estimate the gradient component in the y direction is:

$$S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{89}$$

The edge magnitude is given by the equation:

$$\left|G(u,v)\right| = \sqrt{G_x(u,v)^2 + G_y(u,v)^2} \tag{90}$$

The gradient angle is given by:

$$e_o(u,v) = \tan^{-1}\left(\frac{G_y(u,v)}{G_x(u,v)}\right) \tag{91}$$

The gradient angles are divided into 4 different directions, as shown in Figure 23.

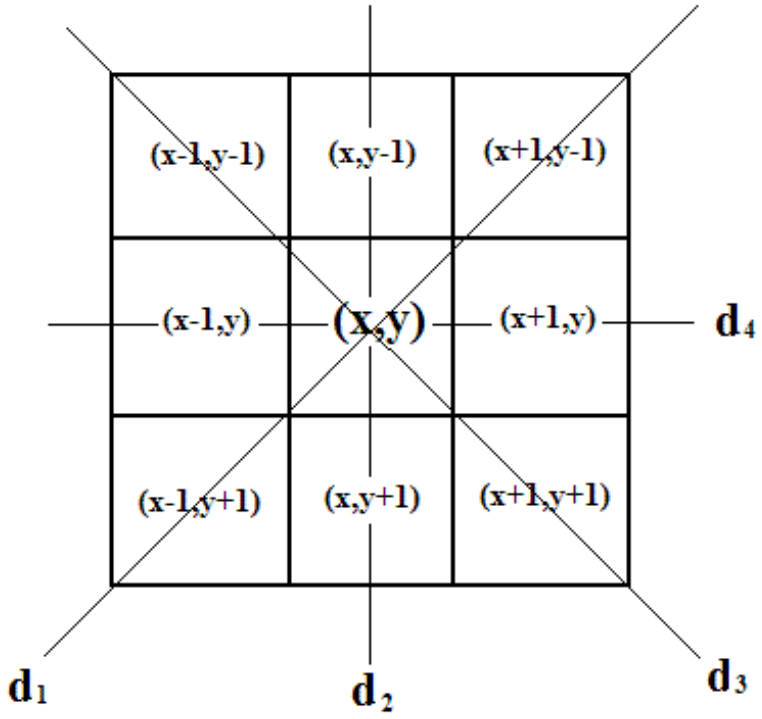

Figure 23 – The 4 directions used in nonmaximum suppression

The edge amplitude for every pixel is then compared to its two neighbours in the assigned gradient direction. The selected edge points are the coordinates which have the highest edge amplitude compared to its two neighboring pixels. This means that only the edge maximums will be assigned as edge points, giving a binary image with thin lines, preferably 1 pixel thick edges. In addition, a

threshold is added so that only the edges above certain strength are accepted as true edge points. When choosing the right threshold, the output image shows the exact location of the desired edges used to estimate the corner coordinates of the calibration rig. Figure 24 shows the typical binary output image of the algorithm.

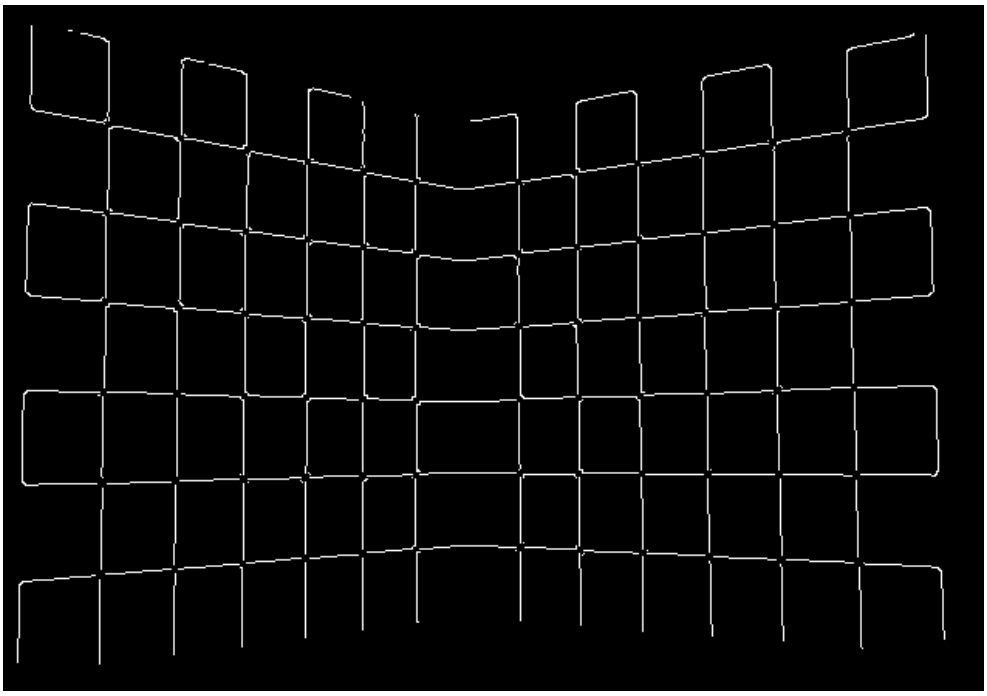

Figure 24 – Output of the nonmaximum suppression algorithm.

3.3.5. K-means Line Fitting

In order to find the coordinates of the intersection between two edges in an image, it is helpful to find a mathematical expression for the respective edges. K-means line fitting is a process of fitting an arbitrary number of lines to a binary image. In this case there are only two lines to be estimated. The process starts with an initial estimate of the two lines of the form $a_l x + b_l y + c_l = 0$, then allocates the non-zero image pixels to the closest line. The line parameters are then reestimated using the eigenvalue problem below:

$$\begin{bmatrix} \overline{x_l^2} - \overline{x_l}\,\overline{x_l} & \overline{x_l y_l} - \overline{x_l}\,\overline{y_l} \\ \overline{x_l y_l} - \overline{x_l}\,\overline{y_l} & \overline{y_l^2} - \overline{y_l}\,\overline{y_l} \end{bmatrix} \cdot \begin{bmatrix} a_l \\ b_l \end{bmatrix} = \lambda_l \begin{bmatrix} a_l \\ b_l \end{bmatrix} \tag{92}$$

where

$$\overline{x_l} = \frac{1}{N_l}\sum_{i=1}^{N_l} x_{l,i}, \qquad \overline{y}_l = \frac{1}{N_l}\sum_{i=1}^{N_l} y_{l,i}$$

$$\overline{x_l y_l} = \frac{1}{N_l}\sum_{i=1}^{N_l} x_{l,i} y_{l,i}, \qquad \overline{x_l^2} = \frac{1}{N_l}\sum_{i=1}^{N_p} x_{l,i}^2 \qquad \overline{y_l^2} = \frac{1}{N_l}\sum_{i=1}^{N_p} y_{l,i}^2$$

This problem has two solutions. The eigen vector corresponding to the smallest eigen value gives the best fitting line parameters. The constraint $a_l^2 + b_l^2 = 1$ is then imposed on the solution. The line parameter c_l is then found by:

$$c_l = -a_l\,\overline{x_l} - b_l\,\overline{y_l} \tag{93}$$

The process is repeated until it converges, meaning that all the samples are assigned to the same line in two consecutive iterations. The process depends on a good initial guess in order to converge. To achieve a good result using this method, it is required that the iterations start with initial line parameters that are relatively close to the true values.

3.4. Feature Matching

One of the most important and challenging problems in computer vision is finding distinct features in images that can be recognized in different scenarios. In order to get a reliable and accurate estimate of the relative pose and translation between two camera positions it is necessary to attain features that are both robust and precisely positioned. In a calibration rig this is fairly easy to achieve, but on an arbitrary object this is much more challenging. If image features are to be used to estimate the relative position of a camera mounted at the end-effector of a manipulator operating at great depths at sea, the feature matching has to be able to cope with noise and be robust against changes in scale, and rotation.

For this task an algorithm called SIFT ("Scale Invariant Feature Transform") will be used. It is a fairly new algorithm developed by Lowe in 1999 [18]. In this thesis, the more recent version from 2004 [3] is used. This algorithm satisfies all the criteria mentioned, and it is also fairly robust when it comes to changes in luminosity. The algorithm finds a high number of interest points in the respective images and calculates a descriptor vector from the neighboring pixels. The Euclidian distance between the descriptor vectors is used to match the feature points. The following section will give a more detailed description of the algorithm.

3.4.1. Detection of Interest Points

The first stage of the SIFT algorithm is to filter the image with a Gaussian filter to eliminate noise. The second stage is to find points that are most likely to give robust feature vectors that can be recognized in other scenarios. The interest points are found at the local extremes in the difference-of-Gaussian pyramid.

The Gaussian blurred image $L(x,y,\sigma)$ is formed by convolving the image $I(x,y)$ with the Gaussian mask $G(x,y,\sigma)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{94}$$

The function $G(x,y,\sigma)$ is given by:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\left(x^2+y^2\right)/2\sigma^2} \tag{95}$$

The algorithm becomes more efficient by subtracting the Gaussian masks instead of subtracting the filtered images:

$$D(x, y, \sigma) = \left[G(x, y, k\sigma) - G(x, y, \sigma)\right] * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{96}$$

1- The initial image is convolved with a Gaussian mask with a standard deviation which increases with the scale factor k every stage. These images are stacked in the left column.

2- For every octave the image scale is divided by an integer number, s, so that $k = 2^{1/s}$, giving $s+3$ images in the octave so that the detection of local maxima covers the whole octave.

3- The adjacent images are subtracted to obtain the difference of Gauss in the right columns.

4- When the octave is finished processing, the image is down sampled with a factor twice the size of the initial σ, then starting over with step 1.

The keypoints are the local maxima and minimas of the function *D(x,y,σ),* which are extreme points relative to the eight neighbors in its own scale and also the nine neighbors in the scale above and the nine neighbors in the scale below as shown in Figure 25 – Maxima and minima in the Difference-of-Gauss are compared to its 26 neighbors [3].

When a maximum is found in a lower octave, the image size is reduced and therefore also the precision of the position estimate. The position in the original image is therefore found by interpolation.

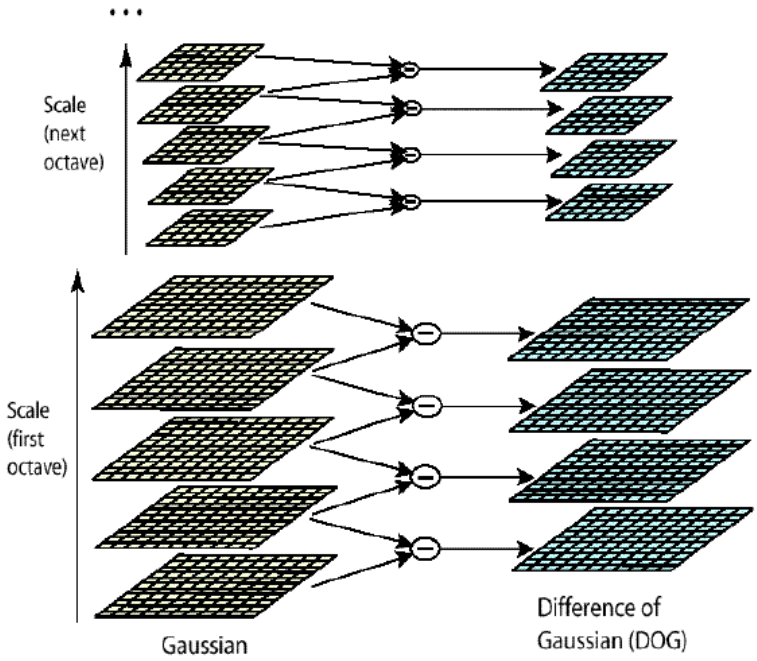

Figure 25 – Maxima and minima in the Difference-of-Gauss are compared to its 26 neighbors [3].

3.4.2. Elimination of Edge Responses

The DOG function has a high response on the edges. Keypoints situated on the edges are undesirable since they are likely to be affected by the angle of

incidence of the light. The curvature of an edge is strong in the gradient direction but weak in the perpendicular direction. The Hessian matrix is therefore used to eliminate the keypoints that are found on or close to an edge maximum. The Hessian matrix of the function D is given by the equation:

$$H = \begin{bmatrix} D_{XX} & D_{XY} \\ D_{XY} & D_{YY} \end{bmatrix} \tag{97}$$

The largest eigenvalue of H is denoted by λ_1 be and the smallest eigenvalue is denoted by λ_2. The ratio between the eigenvalues can be found by the trace and determinant of H.

$$\begin{aligned} Tr(H) &= D_{XX} + D_{YY} = \lambda_1 + \lambda_2 \\ Det(H) &= D_{XX} D_{YY} - D_{XY}{}^2 = \lambda_1 \lambda_2 \end{aligned} \tag{98}$$

Defining r_λ as the ratio between λ_1 and λ_2 giving $\lambda_1 = r_\lambda \lambda_2$, then:

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} = \frac{(r_\lambda \lambda_2 + \lambda_2)^2}{r \lambda_2{}^2} = \frac{(r_\lambda + 1)^2}{r_\lambda} \tag{99}$$

r_λ denotes the principal curvature. The principal curvature has to be above a certain value to ensure that the keypoint is not located on an edge. Elimination of the edge points is done by removing the keypoints that satisfy the following equation:

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r_\lambda + 1)^2}{r_\lambda} \tag{100}$$

In all the experiments in this thesis, a threshold of r_λ=10 is used.

3.4.3. Accurate keypoint localization

The next step is to perform a detailed fit of the keypoint to its nearby data. The characteristics of the DOG function are estimated using a Taylor expansion around the keypoint location show in the equation below:

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x \tag{101}$$

where D and its derivatives are evaluated at he same point and $x=(x,y,\sigma)^T$ is the offset from this point. The localization of the extremum, $\hat{x}$, is determined by setting the derivative of the Taylor expansion to zero, giving

$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2}\frac{\partial D}{\partial x} \quad (102)$$

The exact location of the extremum is used to determine the exact position of the keypoint, giving it a higher precision than one pixel. The peak value of, $D(\hat{x})$, is used to eliminate keypoints with low contrast since they are more easily affected by noise. The maximum value can be estimated by substituting Eq.(102) into Eq.(101) giving:

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D^T}{\partial x}\hat{x} \quad (103)$$

In the experiments in this thesis all keypoints with a contrast lower than 0.03 are rejected.

3.4.4. Assigning Orientation

To obtain a feature descriptor invariant to camera rotation, each keypoint is assigned an orientation estimated from local edge tangents in the image. For all the pixels in the image, *L(x,y),* the edge magnitude *m(x,y)* and edge orientation *θ(x,y)* are calculated by the following equations:

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2} \quad (104)$$

$$\theta(x,y) = \tan^{-1}\left(\frac{\big(L(x,y+1) - L(x,y-1)\big)}{\big(L(x+1,y) - L(x-1,y)\big)}\right) \quad (105)$$

In the surrounding, region there will be a prevailing orientation that is found by forming a histogram of 36 regions, shown in Figure 26. The prevailing orientation in the histogram will be the orientation of the keypoint. Finally, the exact orientation is found by interpolating a parabola to find the center of the group.

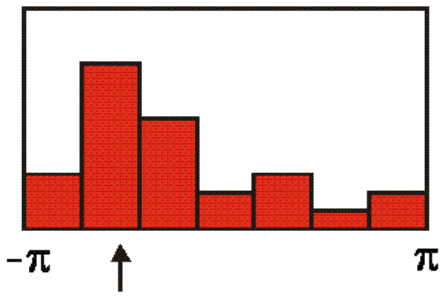

Figure 26 – Histogram of key-point orientation [3].

3.4.5. The Key-Point Descriptor

The final stage is the creation of key-point descriptors. Here the steps made to ensure distinctive descriptors that are invariant to viewpoint and illumination changes will be described.

A key-point descriptor is created by computing the gradients at each pixel in the region surrounding the key-point location. These are weighted by a Gaussian window to increase the emphasis on the gradients closest to the keypoint and avoiding sudden changes in the descriptor with small changes in keypoint position. The gradients are divided into 4x4 sub regions that are transformed into orientation histograms that constitute the elements of the keypoint descriptor. This process is shown in Figure 27, where the left side represents the gradient directions and the right side represents the histograms. The gradient histograms are divided into 8 different orientations. The length of the keypoint descriptor can vary depending on the algorithm. The algorithm used in this thesis uses a 4x4x8=120 element descriptor.

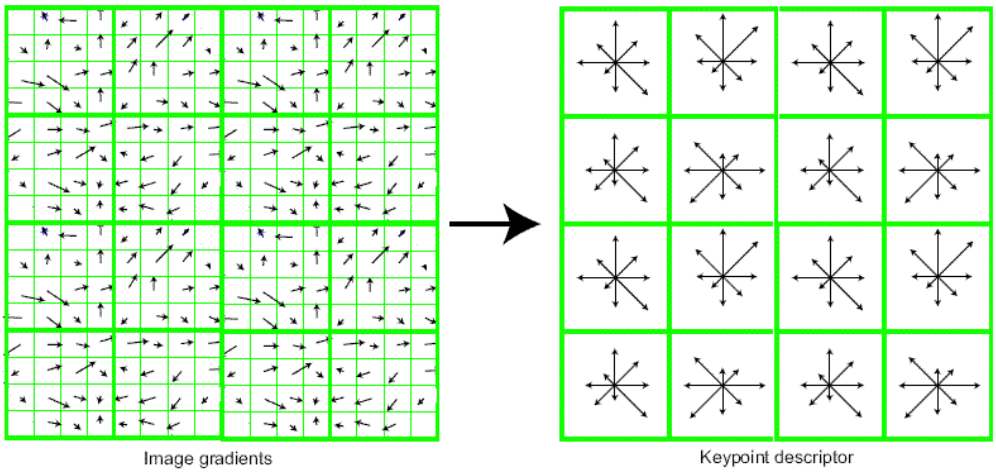

Figure 27 - Keypoint calculation process. Each 4x4 element of gradients (left) is referred to as a bin. For each bin a histogram of 8 directions is calculated (right) [3]

Since the gradient position within each bin does not affect the histogram, each gradient value is distributed to the adjacent bins by a factor depending on the distance to the respective bins. This measure is introduced to avoid the effect of sudden changes when a gradient falls into a neighboring bin.

3.4.6. Invariance to orientation and illumination

To achieve orientation invariance, each orientation in the descriptor histogram is calculated relative to the keypoint orientation. This means that the keypoints will suffer only minor distortions when camera orientation in the image plane is changed.

In order to achieve the invariance to lighting conditions, each descriptor is normalized to unit length. This means that any change in lighting that equally affects the whole image will have little effect on the descriptor. However, when a 3D object is subject to different lighting conditions there can be created sudden alterations in intensity due to shadow effects, etc. This is more likely to occur on the strongest gradients that are more likely to indicate a strong 3D curvature. Therefore a maximum threshold value is introduced to minimize the effect of the strongest gradients. After the strongest edges have been reduced, the descriptor is normalized again. This step enhances the emphasis on the gradient distribution rather than the orientation of the strongest edges.

3.4.7. Keypoint matching

After the SIFT algorithm is applied to two or more separate images, the keypoint descriptors can be matched. First, the Euclidian distances between the descriptors are calculated. When the closest match to each descriptor is found, the distance to the second closest match is compared with the best match. If the distance to the closest match is bigger than 0.8 times the distance to the second best match, the match is rejected.

In most cases, the match based on distance measures alone is not enough to filter out the false matches. The second stage is usually performed using the RANSAC algorithm (Random Sample Consensus) [19] , to eliminate the false matches. If the relative camera pose between the images is small, RANSAC can be used together with the Hough transform [20], but due to its 2D interpretation of the scenes it has a limited use when dealing with more complicated 3D applications.

RANSAC can be computationally heavy if the number of false matches is high, so it is advantageous to eliminate as many false matches as possible before the algorithm is applied. In most scenarios, the orientations between a keypoint and its correct match will change approximately proportionally to camera orientation in the image plane. This can be used to eliminate the matches that differ from the prevailing orientation change. Lowe has suggested eliminating all the matches that differ more than 20°. Figure 28 shows the relative orientation distribution between the keypoints found in two images. The peak at 0° confirms that the relative camera angle in the image plane, θ_z, is zero. If the relative camera angle, θ_z, is altered, the peak in the histogram will change accordingly.

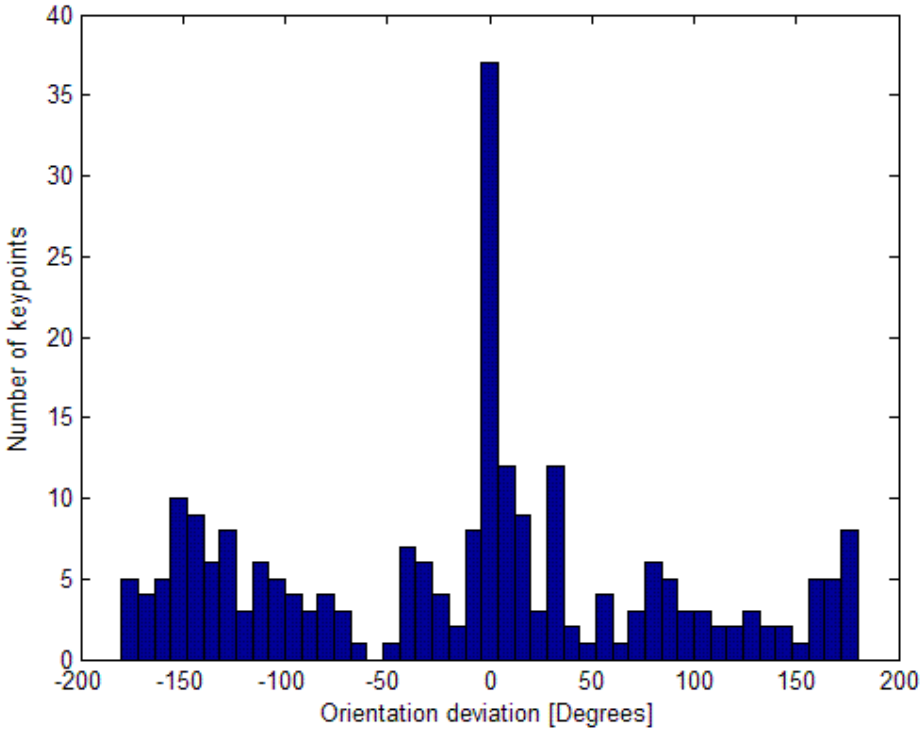

Figure 28 – Distribution of relative keypoint orientation.

The keypoint orientation is in the interval –180° to 180. This means that for keypoints with an orientation close to 180°, the match may have the orientation close to -180°. In order to bypass this ambiguity, the orientation difference is given by:

$$\Delta\theta_z = \min\left(\begin{bmatrix} \theta_{z,1} - \theta_{z,2} - 360 \\ \theta_{z,1} - \theta_{z,2} \\ \theta_{z,1} - \theta_{z,2} + 360 \end{bmatrix}\right) \tag{106}$$

where $\theta_{z,1}$ is the orientation of the keypoint in the first image, and $\theta_{z,2}$ is the orientation of the corresponding keypoint in image 2.

When the SIFT keypoints are used in stereo vision the relative orientation and position of the two cameras is known. By using the pinhole model and the known 3D geometry between the views, the correct keypoint matches will generate two 3D lines which almost intersect at a common 3D point. The matches that don't intersect within a certain distance will be rejected. After the keypoints have been filtered using orientation and triangulation, few false matches remain, if any. To do the final elimination of the false keypoints, RANSAC is used to find the proper geometric model. The use of RANSAC is explained in chapter 3.8.

3.5. Triangulation

When calibrating the manipulator base, cameras are used to calculate its position relative to the environment. The technique used in this thesis is called triangulation. The triangulation process stipulates that the relative translation and orientation between at least two cameras is known. The relative pose can be a fixed position, like a stereo pair, but it can also be estimated by the kinematics of the manipulator if the manipulator is calibrated beforehand.

Triangulation has a simple geometric interpretation, shown in Figure 29. This figure shows two calibrated cameras, or the same camera from two different positions. Point $\hat{p}$ in the first normalized image is correct match with point $\hat{p}'$ in the second normalized image. The camera 3D positions are O_{C1} e O_{C2} respectively. The 3D coordinate of the match can then be found by estimating the intersection between the continuation of the lines $\overline{O_{C1}\hat{p}}$ and $\overline{R\left(O_{C2}\hat{p}'\right)}$, where R is the rotation matrix between camera 1 and camera 2. This is an ideal case. In practical terms, these 3D lines never meet, due to errors in image coordinates and errors in camera positions. The solution is to find the coordinates, P_1 *and* P_2, where they have their closest distance to one another.

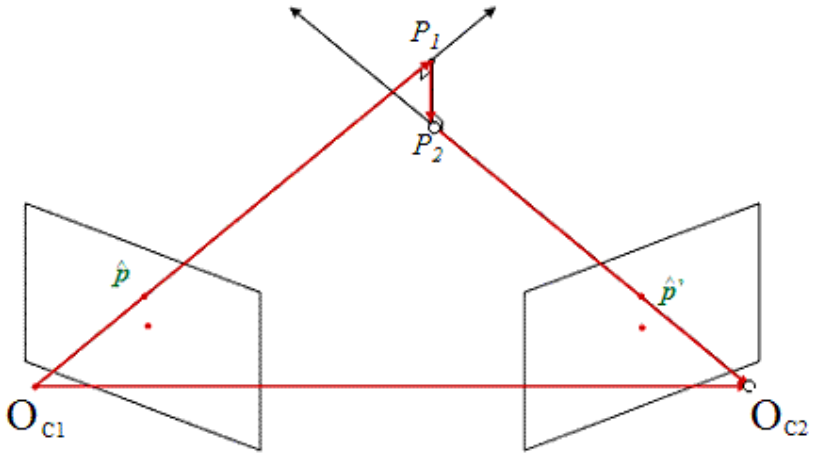

Figure 29 – Triangulation

In order to perform the triangulation, the parameters ${}^1\gamma$ and ${}^2\gamma$ that minimize the distance between P_1 and P_2. must be found. By treating the coordinates $\hat{p}$ and $\hat{p}'$ as vectors, the following equations are obtained:

$$P_1 = O_{c1} + {}^1\gamma \hat{p} \tag{107}$$

$$P_2 = O_{c2} + {}^2\gamma R \hat{p}' \tag{108}$$

The line between P_1 and P_2 is perpendicular to both lines defined in the Eqs. (107) and (108) giving:

$$\overline{(P_1 - P_2)} \cdot \hat{p} = 0 \tag{109}$$

$$\overline{(P_1 - P_2)} \cdot R\hat{p}' = 0 \tag{110}$$

Expanding these using the equations of the straight lines:

$$\left(\overline{(O_{c1} - O_{c2})} + {}^1\gamma \hat{p} - {}^2\gamma R \hat{p}' \right) \cdot \hat{p} = 0 \tag{111}$$

$$\left(\overline{(O_{c1} - O_{c2})} + {}^1\gamma \hat{p} - {}^2\gamma R \hat{p}' \right) \cdot R\hat{p}' = 0 \tag{112}$$

Four coordinates define the two lines:

$$O_{c1} = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}, \hat{p} = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}, O_{c2} = \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix}, O_{c2} + R\hat{p}' = \begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix}$$

Expanding these in terms of the 3D coordinates is not shown here, but the result is as follows:

$$d_{1321} + {}^1\gamma d_{2121} - {}^2\gamma d_{4321} = 0 \tag{113}$$

$$d_{1343} + {}^1\gamma d_{4321} - {}^2\gamma d_{4343} = 0 \tag{114}$$

where

$$d_{mnop} = (x_m - x_n)(x_o - x_p) + (y_m - y_n)(y_o - y_p) + (z_m - z_n)(z_o - z_p) \tag{115}$$

Finally, solving for ${}^{1}\gamma$ gives:

$$ {}^{1}\gamma = \frac{d_{1343}d_{4321} - d_{1321}d_{4343}}{d_{2121}d_{4343} - d_{4321}\cdot d_{4321}} \tag{116} $$

Back substituting gives γ_2

$$ {}^{2}\gamma = \frac{d_{1343} + {}^{1}\gamma d_{4321}}{d_{4343}} \tag{117} $$

By using Eqs. (107) and (108) the coordinates of the closest point can be estimated for both lines. This is handy when it comes to eliminating false keypoint matches.

3.6. Stereo Vision

Stereo vision is a widely used method in auto localization. The method estimates the cameras' relative position and orientation to a set of matched 3D image coordinates. The principle is shown schematically in Figure 30.

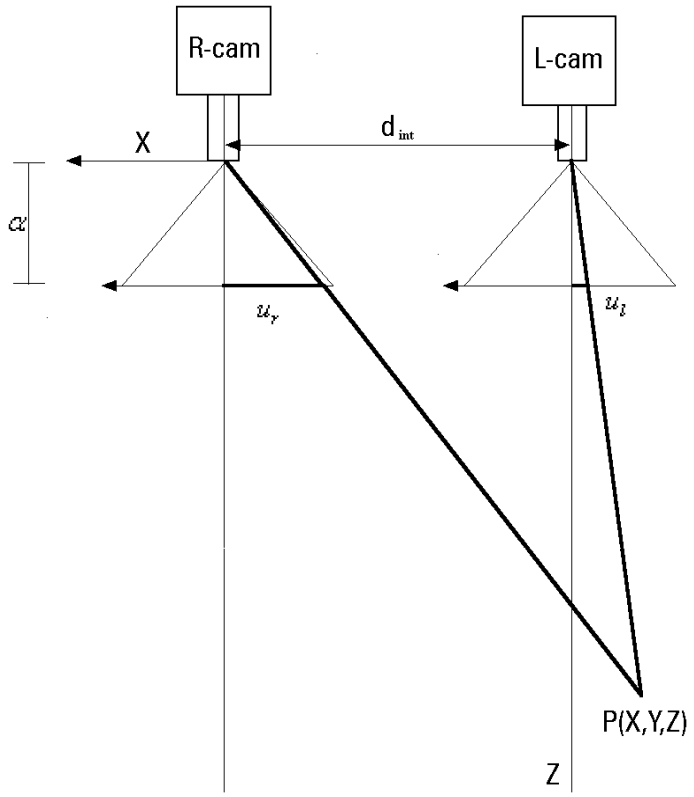

Figure 30 – Stereo Triangulation

Through a geometric interpretation, the following equations are deduced:

$$x = \left(\frac{u_l - u_{l0}}{\alpha}\right) z + \frac{d_{\text{int}}}{2} = \left(\frac{u_r - u_{r0}}{\alpha}\right) z - \frac{d_{\text{int}}}{2} \qquad (118)$$

$$y = \left(\frac{v}{f}\right) z = \left(\frac{v_r - v_{r0}}{\beta}\right) z = \left(\frac{v_l - v_{l0}}{\beta}\right) z \qquad (119)$$

$$z = \frac{d_{\text{int}}\alpha}{s} \qquad (120)$$

where d_{int} is the baseline distance between the two cameras, (u_{l0}, v_{l0}) are the coordinates of the image center of the left camera, (u_{r0}, v_{r0}) are the coordinates of

the image center of the right camera, (u_l, v_l) are the image coordinates of the left camera corresponding to the point P(x,y,z), (u_r, v_r) are the image coordinates of the right camera corresponding to the point P(x,y z), and $s = u_r - u_{r0} - (u_l - u_{l0})$ represents the x coordinate change [pixels] between the left and the right image.

To get an idea of the accuracy of the estimated coordinates, the sensitivity relative to a change in image coordinates is estimated. The position error of the point *P(x,y z)* is given by the following equations:

$$\Delta x = -\left(\frac{u - u_0}{\alpha^2 d_{\text{int}}}\right) z^2 \Delta s \qquad (121)$$

$$\Delta y = -\left(\frac{v - v_0}{\beta^2 d_{\text{int}}}\right) z^2 \Delta s \qquad (122)$$

$$\Delta z = -\left(\frac{1}{\alpha d_{\text{int}}}\right) z^2 \Delta s \qquad (123)$$

where Δx is the error in x direction, Δy is the error in y direction, Δz is the error in z direction, α is the magnification factor in x direction, β is the magnification factor in y direction and Δs is the is the error of s. The image coordinates *v, v_0, u* and u_0 are the image coordinates of the chosen reference camera. It can be either the left or the right camera.

It is obvious from these equations that the accuracy of the coordinates is highly dependent on the resolution of the camera and the intraocular distance between the two stereo cameras. Having a stereo pair attached at the end effector restricts the feasible size of the camera system and thus the possible intraocular distance. This is the incentive for the idea presented in Chapter 3.9, which uses the manipulator kinematics to place the camera in different positions, allowing an arbitrary intraocular distance.

The calibration of the manipulator structure increases the repeatability and absolute precision of the manipulator. This makes it possible to move the camera to any desired position relative to the object of interest and calculate the relative orientation and translation. The manipulator does not offer the same intraocular distance accuracy as a stereo pair camera, but the high versatility makes it a good alternative when the range to the object of interest is high.

3.7. Manipulator Base Calibration

In order to use stereo vision to estimate the relative movement of a camera, two sets of 3D coordinates has to be created by triangulation. The first set, ${}^1p$, is generated from known camera positions relative to the environment. To estimate the relative movement of the camera when found in another position, a second corresponding 3D coordinate set, ${}^2p$, has to be created. The sets ${}^1p$ and ${}^2p$ give the position of the same coordinates, but with different reference frames. Estimating the translation and rotation of the reference frames will give the relative movement of the camera.

Estimation of the reference frame can be done in several ways. The first method investigated here was to generate the 4 x 4 homogeneous transformation matrix using a minimum least square technique. However, this technique was very sensitive to noise and did not give an accurate result compared to other methods.

The second method that was implemented and tested was generation of the relative rotation matrix, R, by using a least square technique suggested in [21].

Having two sets of 3D coordinates ${}^1p$ and ${}^2p$ each containing m samples representing the same coordinates given two different reference frames. The relationship between the i-th coordinate sets is given by:

$$R \cdot {}^2p_i + t = {}^1p_i \qquad (124)$$

Where R is the rotation matrix between the coordinates and t is the translation vector relative to the reference frame of ${}^1p$. The rotation matrix can be calculated by using quaternions or by the least square estimate suggested in [21].

When the rotation matrix, R, is calculated, the translation can be found by:

$$t = \frac{1}{m}\sum_i^m \left({}^1p_i - R \cdot {}^2p_i \right) \qquad (125)$$

where m is the number of samples in each set of coordinates

The translation between the cameras is then estimated using Eq.(125). This method has shown a better result than the direct estimation of the homogeneous matrix.

In an attempt improve the position estimate even further, another method to estimate the rotation matrix has been tested. The method estimates the rotation matrix using quaternions. The relative position of the cameras is then estimated using Eq.(125). This method has proven to be more accurate than the least square method when there are outliers in the coordinate sets. The quaterninon techniques are deduced in Chapter 3.7.2.

3.7.1. Quaternion Algebra

A quaternion is an extension of the concept of complex numbers [22]. A quaternion can be written in the form:

$$q = a_0 + a_1 i + a_2 j + a_3 k \tag{126}$$

where i, j and k are complex quaternion units. The unit vectors are orthogonal 3D vectors and $i \times j = k, j \times k = i, k \times i = j$

The quaternion can be written in the simplified form:

$$q = \left[a_0, a^T \right]^T \tag{127}$$

The Hamilton conjugate is defined by:

$$q^* = \left[a_0, -a^T \right]^T \tag{128}$$

When working with quaternions it is necessary to know the basic rules of different operations. Define the two quaternions a=$\left[a_0, a^T \right]^T$ and b=$\left[b_0, b^T \right]^T$. Addition and subtraction of the two quaternions follow the following rule.

$$\mathrm{a} \pm \mathrm{b} = \left[a_0 \pm b_0, a^T \pm b^T \right]^T \tag{129}$$

Multiplication of two quaternions follow the following rule:

$$\mathrm{a} \circ \mathrm{b} = \left[a_0 b_0 - a \cdot b, \left(a_0 b + b_0 a + a \times b \right)^T \right]^T \tag{130}$$

The norm of a quaternion is defined by:

$$N(\mathrm{a}) = \mathrm{a}^{\mathrm{T}}\mathrm{a} = \|\mathrm{a}\|^2 \tag{131}$$

The inverse of a nonzero quaternion is given by:

$$\mathrm{a}^{-1} = \frac{\mathrm{a}^{*}}{N(\mathrm{a})} \tag{132}$$

When dealing with pure rotations, the quaternions are unit quaternions with norm=1. This means that for a quaternion $q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T$, $q^{-1} = q^*$ and it can be written in the following way:

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\mathrm{k} \tag{133}$$

If the 3D coordinate y can be expressed as a pure rotation of 3D coordinate x, the corresponding equation is:

$$y = R \cdot x \tag{134}$$

The same equation can be expressed in the following way using quaternions:

$$y = q \circ x \circ q^* \tag{135}$$

The rotation matrix (R) in Eq.(134) can be estimated from the corresponding quaternion using the following equation:

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{136}$$

3.7.2. Estimation of the Rotation Matrix Using Quaternions

Consider a 3D coordinate ${}^1p$. Written in another reference frame the same coordinates take the form ${}^2p$. Transformation between the coordinates is given by the following equation:

$$R \cdot {}^2p = {}^1p - t \tag{137}$$

When using quaternions, the following equation is used:

$$R \cdot {}^2p = q \circ {}^2p \circ q^* \tag{138}$$

Substituting Eq.(138) into Eq.(137) gives

$$q \circ {}^2p - \left({}^1p - t\right) \circ q = 0 \tag{139}$$

Expanding Eq.(139) gives:

$$\left(q \cdot {}^{2}p, q \cdot {}^{2}p - {}^{2}p \times q\right) = \left(q \cdot \left({}^{1}p - t\right), q\left({}^{1}p - t\right) \times \left({}^{1}p - t\right)\right) \qquad (140)$$

Because the scalar part , $q \circ {}^{2}p - \left({}^{1}p - t\right) \circ q = 0$, Eq.(140) is equal to:

$$\left({}^{2}p + \left({}^{1}p - t\right)\right) \times q = q\left({}^{2}p - \left({}^{1}p - t\right)\right) \qquad (141)$$

For the matrix R to be a rotation matrix, $q = \cos\left(\frac{\theta_q}{2}\right) + \sin\left(\frac{\theta_q}{2}\right)\mathbf{k}$, where **k** is the rotation axis and θ is the rotation angle of R respectively. Substituting this into Eq.(141) gives:

$$\tan\left(\frac{\theta_q}{2}\right)\left({}^{1}p - t + {}^{2}p\right) \times \mathbf{k} = {}^{2}p - {}^{1}p + t \qquad (142)$$

Defining the scew matrix $Q(v) = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}$, which has the following property for the vector $v = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}$.

$$Q(v) \cdot x_q = v \times x_q \qquad (143)$$

Equation (142) can then be written as:

$$Q\left({}^{2}p + {}^{1}p\right)x_q - Q(t)x_q - t = {}^{2}p - {}^{1}p \qquad (144)$$

where:

$$x_q = \tan\left(\frac{\theta_q}{2}\right)\mathbf{k} \qquad (145)$$

which is the Rodrigues parameter and

$$y_q = -Q(t)x_q - t \qquad (146)$$

Equation (144) can then be written as:

$$Q\left({}^{1}p - {}^{2}p\right)x_q + y_q = {}^{2}p - {}^{1}p \qquad (147)$$

Assuming that there are m available coordinate pairs, to estimate the rotation quaternion from these m measurements the variable y_q can be eliminated by substituting $\zeta_i = {}^{2}p_i - {}^{2}p_{i+1}$ and $\rho_i = {}^{1}p_i - {}^{1}p_{i+1}$, giving the equation:

$$Q(\zeta_i + \rho_i)x_q = \zeta_i + \rho_i \qquad (148)$$

which holds for all i=1:m measurements. This equation can be solved with a minimum of three measured coordinate pairs, giving the exact solution by solving the equation:

$$A_q \cdot x_q = b_q \tag{149}$$

where

$$A_q = \begin{bmatrix} Q(\zeta_1 + \rho_1) \\ Q(\zeta_2 + \rho_2) \end{bmatrix} \tag{150}$$

and

$$b_q = \begin{bmatrix} \zeta_1 - \rho_1 \\ \zeta_2 - \rho_2 \end{bmatrix} \tag{151}$$

Having more than three measurements the equation can be solved by means of the pseudo inverse matrix:

$$\mathrm{x_q} = \left(\mathrm{A_q}^{\mathrm{T}} A_{\mathrm{q}}\right)^{-1} A_{\mathrm{q}}^{T} b_{\mathrm{q}} \tag{152}$$

where

$$A_{\mathrm{q}} = \begin{bmatrix} Q(\zeta_1 + \rho_1) \\ . \\ . \\ . \\ Q(\zeta_{m-1} + \rho_{m-1}) \\ Q(\zeta_m + \rho_m) \end{bmatrix} \tag{153}$$

and

$$b_q = \begin{bmatrix} \zeta_1 - \rho_1 \\ . \\ . \\ . \\ \zeta_{m-1} - \rho_{m-1} \\ \zeta_m - \rho_m \end{bmatrix} \tag{154}$$

After solving for $\mathrm{x_q}$, the quaternion q can be estimated. The vector $\boldsymbol{k}$ from Eq.(133) is found by:

$$\mathbf{k} = \frac{\mathbf{x_q}}{\left\| \mathbf{x_q} \right\|} \tag{155}$$

The angle θ_q from Eq.(133) is found by:

$$\theta_q = 2\tan^{-1}\left(\frac{\max\left(\mathrm{x_q}\right)}{\max\left(\mathbf{k}\right)}\right) \tag{156}$$

The rotation quaternion is then found by Eq.(133) and the rotation matrix is given by Eq.(136).

3.8. Elimination of Keypoint Matches using RANSAC

If the correct orientation and translation between two views have been found, the samples in the coordinate sets should ideally satisfy Eq.(124). However, after filtering the keypoint matches by their orientation angles and their point of coincidence, there might still remain some false matches in the sets. To eliminate these remaining outliers, RANSAC is used together with Eq.(124) to find which samples satisfy the 3D model. The rotation matrix, R, can be estimated using three corresponding samples from the sets ${}^1p$ and ${}^2p$. The syntax of the algorithm is as follows:

a) $R \cdot {}^2p_i + t = {}^1p_i$

b) Randomly pick 3 corresponding samples from the sets ${}^1p$ and ${}^2p$ forming ${}^1p^*$ and ${}^2p^*$

c) Estimate the relative rotation matrix, R, between the samples using ${}^1p^*$ and ${}^2p^*$. This can either be done by using the quaternion technique deduced in section 3.7.2 or by the least mean square technique presented in [21].

d) Estimate the mean of ${}^1p^*$ and ${}^2p^*$ denoted by ${}^1\tilde{p}^*$ and ${}^2\tilde{p}^*$

e) Generate a new set of translated coordinates:

$${}^1_t p_i = {}^1p_i - {}^1\tilde{p}^* \tag{157}$$

$${}^2_t p_i = {}^2p_i - {}^2\tilde{p}^* \tag{158}$$

If the correct rotation matrix is estimated, the samples should satisfy the following equation:

$$R\,{}^2_t p_i = {}^1_t p_i \tag{159}$$

However, due to uncertainties in the estimated coordinates, a certain error has to be tolerated. The samples that satisfy the following equation are accepted by the model.

$$\left| R\,{}^{2}_{t}p_i - {}^{1}_{t}p_i \right| = d_{r,i} < d_r \qquad (160)$$

After a number of iterations the model that includes the highest number of samples is used to generate the final rotation matrix. The position is then estimated using Eq.(124).

After RANSAC is applied to the data sets, any outlier with a detrimental effect to the model is eliminated. The remaining samples all fit the model, some better than others. To further increase the accuracy of the measured position, a last filtering can be done. The samples that are close to the center of the cluster of samples are less important when it comes to estimating the relative rotation matrix. They might even have a negative effect on the accuracy. Defining the error ratio by the following equation:

$$r_{e,i} = \frac{\left| {}^{1}_{R}p_i - {}^{\tilde{1}}_{R}p \right|}{d_{r,i}} \qquad (161)$$

where $d_{r,i}$ is the Euclidian distance error from Eq.(160). ${}^{1}_{R}p_i$ are the coordinates of ${}^{1}p$ after RANSAC has eliminated the outliers, and ${}^{\tilde{1}}_{R}p$ is the center of the coordinate set ${}^{1}_{R}p$.

The sample with the highest value of $r_{e,i}$ is then eliminated and the rotation matrix is reestimated. The process is then repeated until all the samples in the sets have an error ratio less than a chosen limit, r_{lim}.

This final step has proven to improve the estimated position accuracy. This can be seen in the experiments in Chapter 5.

3.9. Triangulation Using the Kinematics of the Manipulator

In computer vision, the most common way to estimate 3D coordinates by triangulation is using a stereo pair camera, which generally consists of two parallel cameras with a fixed intraocular distance. The problem with this configuration is that it demands a high intraocular distance between the cameras to achieve a good depth resolution. Mounting a device close to the end effector might be difficult due to its size. In order to bypass this problem, it is desirable to be able to achieve a good depth resolution using a more simple system. This is the incentive for the use of the kinematics of the manipulator to estimate the relative pose between the respective views. The kinematics of the TA-40 manipulator which has six degrees of freedom, offers an increased precision after calibrated. But the orientation of the end effector is not precise enough due to the rotational errors in the last joint, which cannot be measured in a simple way with only one theodolite target at the end-effector. However, mounting the camera on the fifth link solves this problem. The calibrated kinematic model does not offer an exact orientation estimate for this link, but the orientation error caused by the repetitive errors will have a fixed bias relative to the true orientation for all configurations of the joints. This means that the relative pose of a camera mounted on link 5 can be estimated with high accuracy. The only pose errors will be caused by random errors, such as play or friction. Even though it doesn't offer the same precision as a stereo pair camera, its versatility makes up for the lack of precision. Performing a triangulation using configurations that form approximately equilateral triangles offers a great advantage when it comes to depth estimation.

The procedure starts by choosing a reference camera position. This must be a position where the camera has a good view of the whole object of interest, for example a manifold panel of valves. With the camera in this position an image called the reference image is taken. SIFT is then performed on the image. In order to estimate the 3D position of the SIFT keypoint found in the reference image, various images are taken in the area surrounding the reference position. The photos need to be taken from positions that offer the necessary intraocular distance with the reference position. The relative positions of the cameras are estimated using the kinematics of the manipulator. The SIFT keypoints of the

reference image are then matched with the keypoints of the other images. Using triangulation, the 3D positions of the keypoints are estimated relative to the reference position. This creates a set of 3D coordinates, ${}^{1}p$. This procedure is illustrated in Figure 31. There are usually certain landmarks in the work environment that the robot wants to use as 3D references. These objects don't always have keypoints found on their location in the image. In order to estimate these coordinates' positions relative to a known object, for example a valve or the corners of a panel, the coordinates of the desired objects need to be found manually in the images. The position of the reference camera relative to these objects can then be estimated by triangulation.

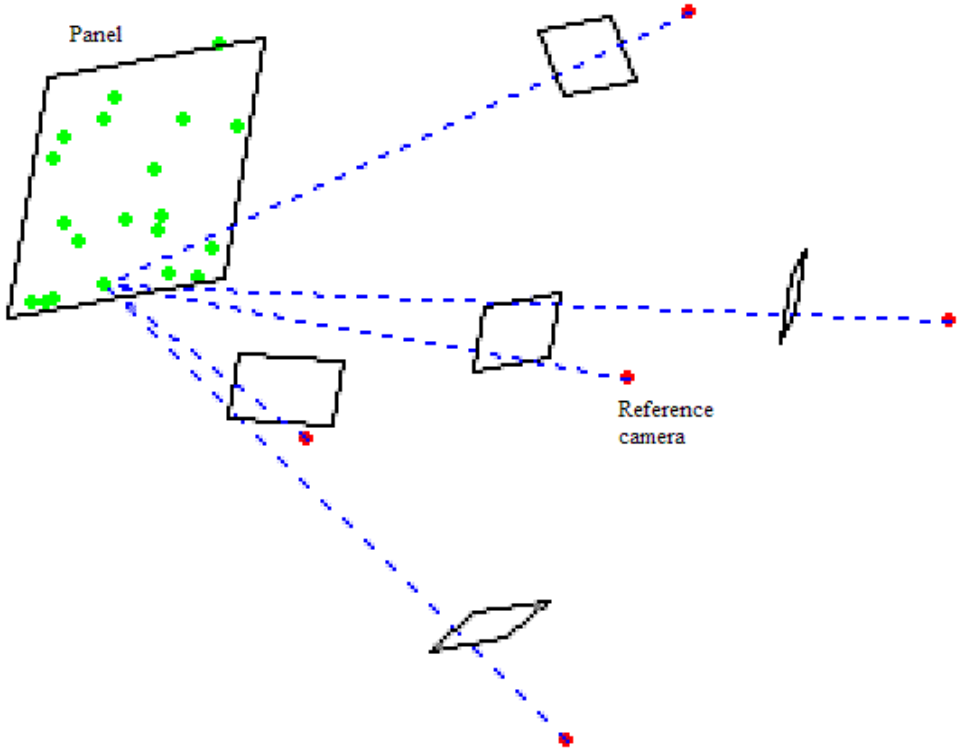

Figure 31 –Initial procedure to estimate the position of the reference camera relative to the keypoints. Creating a set of 3D coordinates, ${}^{1}p$.

The keypoints of the reference image will find matches in the other images. The most robust keypoints will find matches in many images. The corresponding 3D coordinate for each keypoint is then estimated as an average of the estimated coordinates. This is shown in Eq.(162) . Since the center of the reference camera

defines the origin of the reference frame, the equation does not need to include a rotation matrix.

$$ {}^{1}p_i = \frac{1}{N_{c1}} \sum_{j=1}^{N_{c1}} \left({}^{1}\gamma_{i,j} \hat{p}_{i,j} \right) \tag{162} $$

where N_{c1} is the number of images where each keypoint is found, $\hat{p}_i$ is the normalized image coordinate for the respective keypoint, and ${}^{1}\gamma_{i,j}$ is the multiplication factor deduced on Chapter 3.5 that denotes where the line through the normalized coordinate $\hat{p}_i$ intersects with the line through $\hat{p}_{i,j}$ of the j-th image. This gives:

$$ {}^{1}p_i = \frac{1}{N_{c1}} \sum_{j=1}^{N_{c1}} \left({}^{1}\gamma_{i,j} K^{-1} p_{i,j} \right) \tag{163} $$

To avoid that any false keypoint matches will ruin the estimated coordinate, the Euclidian distance between all the corresponding coordinates are estimated. The coordinates that don't have a neighbour within a distance d_n are not used in the estimate. After these coordinates have been eliminate, N_{c1} matches remain to estimate the final coordinate ${}^{1}p_i$.

The coordinates can also be estimated in an alternative way, by estimating where the line through $\hat{p}_{i,j}$ has its closest point to $\hat{p}_i$:

$$ {}^{1}p_i^{*} = \frac{1}{N_{c1}} \sum_{j=1}^{n} \left(t_j + {}^{2}\gamma_{i,j} R_j^T \cdot K^{-1} \cdot p_{i,j} \right) \tag{164} $$

where t_j is the position of the j-th camera relative to the reference camera, R_j is the rotation matrix between the chosen reference camera and the j-th camera position, K is the intrinsic calibration matrix of the camera, and $p_{i,j}$ is the i-th image coordinate of the j-th camera position.

Ideally, these methods should give the same result, but due to projection approximations and uncertainties in the relative camera positions, the estimates will differ. The false keypoint matches usually don't come close to one another, so by excluding the matches where ${}^{1}p_i^{*}$ and ${}^{1}p_i$ differ more that a certain distance, many false matches will be eliminated.

The more camera positions used to estimate these coordinates, the more accurate the estimate of the coordinate set, ${}^{1}p$. It is also desirable that the angles between the reference camera and the other used camera positions are relatively big, to increase the accuracy of the triangulation. The angles can, however, not be too large since it is more difficult to recognize SIFT keypoints when the relative angle between the cameras increase.

When the manipulator is repositioned and the position of a camera is to be estimated relative to the chosen object, two photos from different positions need to be taken. This is shown in Figure 32.

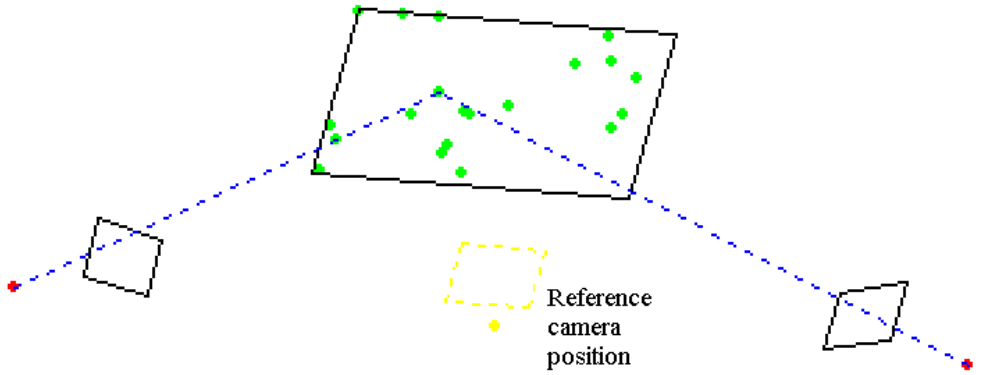

Figure 32 – Finding a second set of corresponding coordinates, ${}^{2}p$.

The relative position between the views is estimated using the kinematic manipulator model. Using triangulation to generate a set of 3D coordinate matches, ${}^{2}p$, the relative position of the camera and hence the end-effector can then be calculated. Estimating the second set of coordinates can be done by triangulation using more than two camera positions, but this step will be performed in real time. This means that the processing time will be important. Therefore only two camera positions are used. The second set of coordinates is given by the equation below.

$$ {}^{2}p_{i} = {}^{1}\gamma_{e,i} K^{-1} p_{e2,i} \qquad (165) $$

where $p_{e1,i}$ is the i-th image coordinate of the first image used in the position estimate. One of the cameras defines the reference frame for ${}^{2}p$, therefore

Eq.(165) contains no rotation matrix. The coordinate set can again be calculated in another way:

$$ {}^{2}p_{i}^{*} = t_{rel} + {}^{2}\gamma_{e,i} R_{rel} \cdot K^{-1} p_{e2,i} \qquad (166) $$

where R_{rel} is the relative rotation between the two camera positions and t_{rel} is the relative camera position.

Having two sets of 3D coordinates, ${}^{1}p$ and ${}^{2}p$ or ${}^{1}p^{*}$ and ${}^{2}p^{*}$ their relative orientation can be estimated. After the rotation matrix is found, the relative position of the camera can then be calculated using Eq. (125).

Before the coordinate sets are calculated, the false keypoint matches need to be eliminated. This is first done by the initial steps of keypoint orientation comparison and triangulation. When estimating the coordinate set ${}^{1}p$, the coordinates will be an average of several estimated coordinates. The most robust and reliable keypoints are the ones that are recognized in several images. By only accepting the keypoints that have matches in a certain number of images, the estimated coordinates become more robust. Since the set ${}^{1}p$ is estimated from an average of many measurements from various positions, the coordinates estimated from the same corresponding keypoint will form a cluster in space. By eliminating the contribution from the coordinates that fall outside a certain distance from this cluster center, the estimated coordinates will be more accurate.

In the next chapter, the presented methodologies are applied to a specific manipulator, the TA-40.