2016•2017
FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
master in de industriële wetenschappen: elektronica-ICT

Masterproef
3D position determination using a camera

Promotor :
Prof. dr. ir. Luc CLAESEN

Copromotor :
De heer Wout SWINKELS

Gezamenlijke opleiding Universiteit Hasselt en KU Leuven

Tom Bortels , Bart Bostijn
Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

2016•2017

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: elektronica-ICT

Masterproef

3D position determination using a camera

Promotor :
Prof. dr. ir. Luc CLAESEN

Copromotor :
De heer Wout SWINKELS

Tom Bortels , Bart Bostijn

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

universiteit hasselt | KU LEUVEN

Foreword

Pose estimation methods are used in several applications of our daily lives without us even knowing it. For example the cameras in our phones are calibrated using pose estimation techniques, several apps on our phones use virtual reality which relies heavily on pose estimation techniques. However we have never encountered these pose estimation techniques during our studies. We had no idea what to expect from this subject and where this thesis would take us. Working on this master's thesis gave us a chance to get familiar with these techniques and their (dis)advantages. We were also surprised by the amount of research done in this field, pose estimation definitely seems a hot topic.
First we would like to thank Prof. Dr. Ir. Luc Claesen and Ing. Wout Swinkels for their tremendous help and support with this thesis. Their help, ideas and support made this work possible. Their combined wisdom and experience helped us understand specific aspects of pose estimation techniques. Secondly we would also like to thank our friends and family for their support in this stressful time.

Tom Bortels and Bart Bostijn 2017

Table of content

List of tables

List of graphs

Abstract Nederlands

Deze masterproef voert onderzoek uit naar positiebepaling met behulp van een camera. Het gebeurde in samenwerking met onderzoeksgroep CoSenS van Universiteit Hasselt. Dit is een veel voorkomend probleem bij computer visie. In dit onderzoek worden verschillende methoden vergeleken voor het bepalen van de positieparameters van de zes vrijheidsgraden (6DOF). Hierbij worden voornamelijk methoden bestudeerd die gebruik maken van een camera zonder speciale bijkomende hardware vereisten. Dit onderzoek heeft een uitgebreid toepassingsdomein bijvoorbeeld in: augmented reality, robotica, gezondheidzorg…

In deze masterproef bespreken we de stappen die nodig zijn voor het detecteren van de feature points. Deze zijn in volgorde: de kalibratie/normalisatie van de camera, het detecteren van kenmerkpunten van een gekend patroon en de berekening van de positie. Verder bespreekt deze thesis klassiekere methodes zoal DLT, EPnP, OI, NPL, NLL en POSIT en modernere methodes zoals PRSOI en Moiré voor het bepalen van de 6DOF coordinaten. Deze thesis bestudeert ook welke methodes het best zijn onder bepaalde omstandigheden. Deze kunnen verschillen in hoeveelheid ruis en distorsie, beperkte zichtbaarheid van het patroon, resolutie…

DLT is een methode die gevoelig is voor ruis maar geen kalibratie nodig heeft. PRSOI is meestal de nauwkeurige methode voor het bepalen van alle 6 DOF. Moiré kan de rotaties met een zeer hoge nauwkeurigheid bepalen.

Abstract English

This master's thesis researches pose determination methods with a camera in cooperation with the research group CoSenS of Hasselt University. 3D position determination is a frequently occurring problem in the field of computer vision. In this thesis several methods for determining the 6 degrees of freedom are compared. Especially methods which do not require specialized or expensive hardware are studied. This research has a wide field of application, for example: augmented reality, robotics, healthcare, etc…

In this master's thesis we discuss the necessary procedures for detecting feature points. These procedures are, in order: the calibration and normalization of the camera, the detection of reference points in a known pattern and the calculation of the position of the object. This master's thesis researches classical methods like: DLT, EPnP, OI, NPL, NLL and POSIT and also some newer methods like PRSOI and Moiré patterns. These methods differ in robustness to noise and distortion, limited field of target, resolution, etc…

For example DLT is a method which is very susceptible to noise, but it does not require camera calibration. PRSOI is a one of the more accurate methods in determining the pose, however Moiré patterns can be used to detect the out-of-plane rotations more accurately.

1 Introduction

This master's thesis investigates 3D position determination using a camera and has been conducted in co-operation with the research group CoSenS of Hasselt University. 3D position determination is a frequently occurring problem in the field of computer vision. There is an extensive list of applications for 3D position determination: e.g. augmented reality, robotics, healthcare.
The thesis describes and compares several methods for acquiring the 6 degrees of freedom position parameters (3D coordinates and 3D orientations). This thesis mainly describes monocular position determination methods, which do not require specific extra hardware. But they always require points with known positions in a scene. They can consist of fiducials or known patterns. Often used currently, this pattern is a checkerboard but it can also be a pattern with circles, only one circle,…
There are several steps required to determine the pose of an object. The first step is taking an image of the calibration pattern. The second step is normalizing the image to reduce the influence of manufacturing defects of the camera. The normalization compensates for the radial and tangential distortion of the image. The next step is detecting the feature points of the pattern. Most patterns use corner or line detection to detect the feature points of the checkerboard in the image. The position of the feature point in the image is often refined till subpixel accuracy. The accuracy of this detection depends on the resolution, noise, and remaining distortion of the image. The forth step is calculating the six degrees of freedom of the pattern. There are both iterative and non-iterative methods.
The focus of this research is initially in favor of accuracy over speed. This thesis describes which methods are best used in specific circumstances. These circumstances can differ in the amount of noise and distortion, amount of outliers, the resolution, the necessary speed, the location and size of the calibration pattern…

2 Theory

To estimate the position of an object, a pinhole camera is used. However a pinhole camera is a theoretical concept. Implementing this model, by using a lens camera, causes distortions. The intrinsic camera parameters indicate these distortions which are introduced by the manufacturing of the camera. These distortion parameters, measured during calibration, depend on the focus distance of the camera. Once these parameters are known it is possible to calculate the undistorted image. This process is also called normalization. The pinhole model is the most frequently used camera model. There is also a study from 2015 of an incident ray tracking camera model [1]. This model tracks each light beam individually and thus is more precise but requires more calculations. This will be explained in section 4.5: Incident ray tracking camera model.

2.1 Pinhole model

There is a difference between a pinhole camera and a lens camera. An image captured with a pinhole camera has no distortion. The light reflects from an object and goes through an infinitly small hole in the front of the camera. The image is then projected on the back of the camera. This is shown in Figure 1. The image sensor is located at Q'P'. This implies that the image is perfectly displayed on the image sensor. However, in reality not enough light will reach the image sensor. To overcome this problem a lens is used. The lens bends the incoming light beams so that more light reaches the image sensor but it also causes distortion. This is illustrated in Figure 1 and Figure 2. [2] [3]

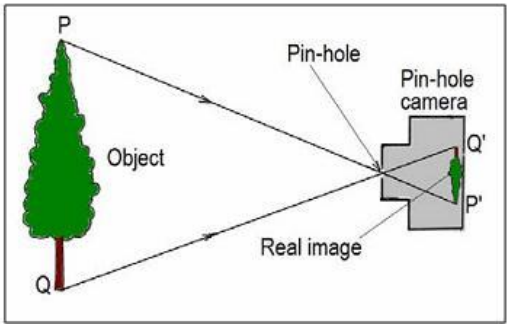

Figure 1: Real image by a pinhole camera [2]

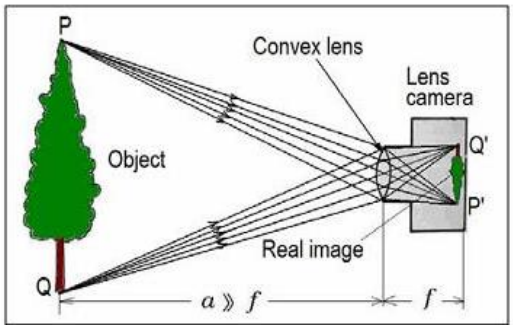

Figure 2: Real image by a lens camera [2]

2.2 Intrinsic camera parameters

There are three intrinsic camera parameters [4]

- Focal length (x , y)
- Principal point (u_0,u_0)
- Skew

Each of these parameters will be discussed briefly in the following sections.

2.2.1 Focal length

The focal length is a parameter which indicates how much the lens will bent the light. It indicates how far the clearest objects in the picture are removed from the lens. The range of the objects in focus is dependent on the aperture of the camera. This is the hole through which the light enters the camera. [5], [6]

2.2.2 Principal point

The principal point indicates the centre of an image and is determined by the lens. The light which passes through the lens without being bended indicates the optical axis of the lens. The spot where the optical axis crosses the image sensor is the principal point. Due to manufacturing inaccuracies this is not necessarily the centre of the image sensor. [7]

2.2.3 Skew

Due to the manufacturing process the lens is often not placed perfectly parallel with the image sensor. The angle between the optical axis of the lens and the image sensor is called the skew as shown in Figure 3. Most of the time this angle is so small that it is negligible as a calibration parameter but it directly influences the tangential distortion explained further in the text. [4]

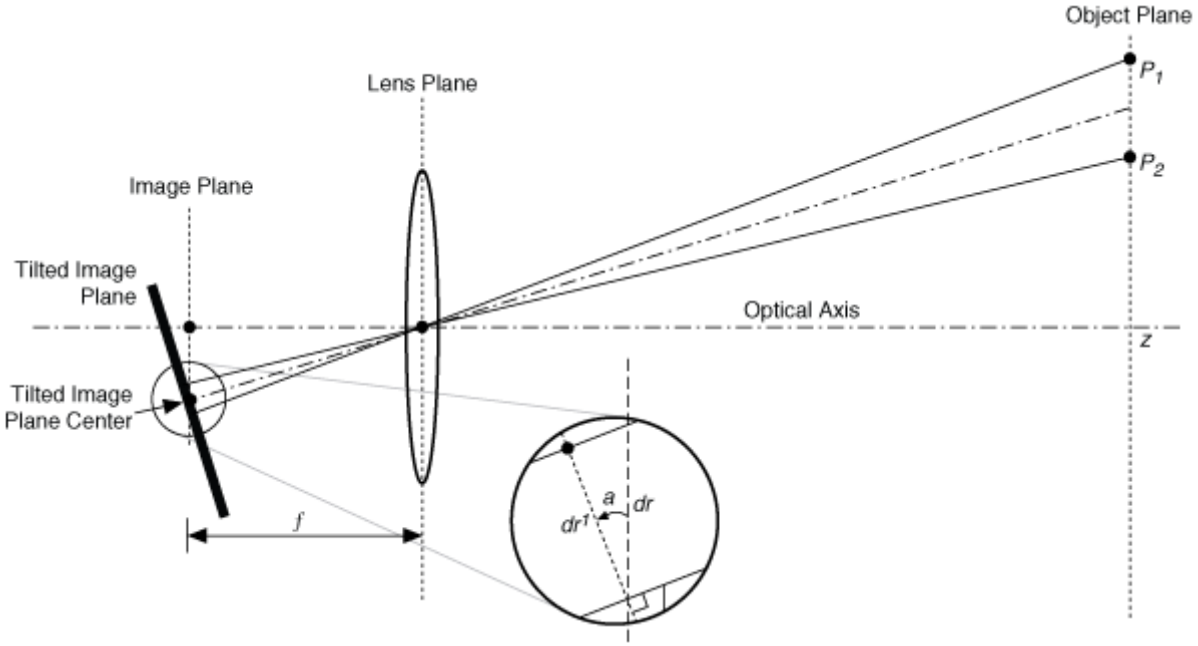

Figure 3: Skew of a non-perfect lens [8]

2.3 Calibration matrix

These intrinsic camera parameters are used in the calibration matrix. This matrix A represents the relationship between the undistorted 2D image coordinates (P^U) and the distorted 2D image coordinates (P^D), where both are in homogeneous coordinates.

$$P^D = A.P^U = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}.P^U \quad (1)$$

Hereby A is the calibration matrix, f_x, $f_y$the focus distance, c_x, c_y the coordinates of the principal point and γ the skew which is usually negligible and thus set to 0. Note that all parameters are written in pixel units except for the skew. [9]

2.4 Distortion parameters

There are two kinds of distortion: [10]

- Radial distortion
- Tangential distortion.

In the next sections these two distortions are discussed in more detail. The value of the distortion parameters is dependent upon the focal length, principal point and skew of each picture.

2.4.1 Radial distortion

Radial distortion is introduced by the fact that the lens is not perfectly parabolic. Therefore straight lines at the edges of an image are deformed. This is illustrated in Figure 4. Also the effect of the radial deformation gets stronger further away from the principal point.

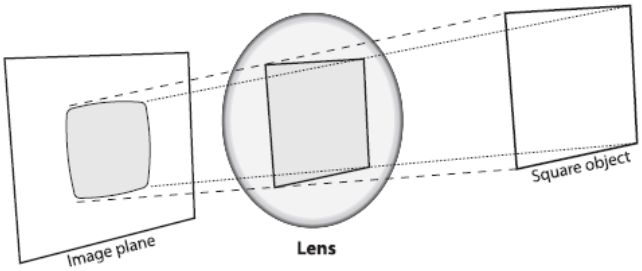

Figure 4: Radial Distortion [11]

The radial deformation can be subdivided in two categories. The first one is the pin cushion distortion also called negative radial distortion, shown in Figure 6. The second one is the barrel distortion also called positive radial distortion as shown in Figure 7. [11]

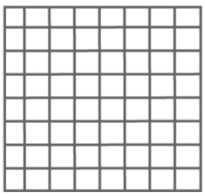

Figure 5: No Radial Distortion [12]

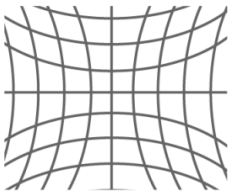

Figure 6: Negative Radial Distortion [12]

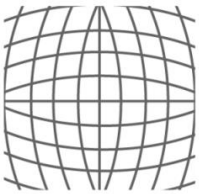

Figure 7: Positive Radial Distortion [12]

The relation between the distorted image coordinates P^D and the undistorted image coordinates P^U for radial distortion can be modelled by the following equations:

$$P_x^D = (1 + k_1 r_u^2 + k_2 r_u^4 + k_3 r_u^6) P_x^U \quad (2)$$

$$P_y^D = (1 + k_1 r_u^2 + k_2 r_u^4 + k_3 r_u^6) P_y^U \quad (3)$$

Where the factors k_1, k_2, k_3 are the radial distortion coefficients and where r_u is the distance between the principal point and the corrected pixel. [12]

2.4.2 The tangential distortion

In case the lens of the camera is not placed exactly parallel to the image sensor, a distortion is introduced in the image which is called the tangential distortion. This is illustrated in Figure 8.

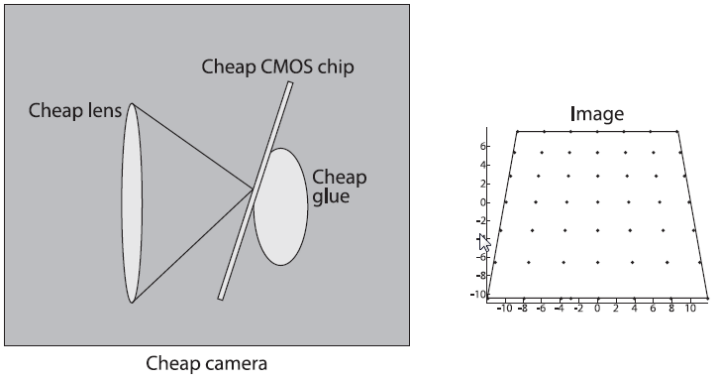

Figure 8: Tangential distortion due to imperfect placement of the lens. [13]

The angle between the lens and image sensor is indicated by the skew as mentioned before. The relation between the distorted image coordinates P^D and the undistorted image coordinates P^U for tangential distortion is given by the following equations:

$$P_x^D = P_x^U + p_2\left(r^2 + 2{P_x^U}^2\right) + 2p_1P_y^U \quad (4)$$

$$P_y^D = P_y^U + p_1\left(r^2 + 2{P_y^U}^2\right) + 2p_2P_x^U \quad (5)$$

Where $r^2 = (P_x^U)^2 + (P_y^U)^2$, p_1 and p_2 are the distortion coefficients.
The following equations take both kinds of distortion into account. [13]

$$P_x^D = (1 + k_1r_u^2 + k_2r_u^4 + k_3r_u^6)P_x^U + p_2\left(r^2 + 2{P_x^U}^2\right) + 2p_1P_y^U \quad (6)$$

$$P_y^D = (1 + k_1r_u^2 + k_2r_u^4 + k_3r_u^6)P_y^U + p_1\left(r^2 + 2{P_y^U}^2\right) + 2p_2P_x^U \quad (7)$$

2.5 Normalisation

Normalisation is the process in which a picture of a camera is transformed in a picture with theoretically no distortion. This is done with the following equations:

$$P_x^U = P_x^D + \left[p_1\left(r^2 + 2{P_x^D}^2\right) + 2p_2P_x^DP_y^D\right] * (1 + p_3r^2) \quad (8)$$

$$P_y^U = P_y^D + \left[2p_1P_x^DP_y^D + p_2\left(r^2 + 2{P_x^D}^2\right)\right] * (1 + p_3r^2) \quad (9)$$

Note that different calibration methods or options for these methods result in more or less distortion parameters. This is the equation for three distortion parameters. Theoretically a higher amount of parameters allow for a more accurate normalization. But practically this can result in a less accurate normalization because all measured points have a certain error in there subpixel position in the image. Because of these errors the normalization function is going to fluctuate heavier, if more parameters are used. A higher amount of parameters also results in an increasing calculation time. Therefore most methods only use two or three parameters. [11]

2.6 Coordinate systems

There are many different coordinate systems. The coordinate systems which correspond to a certain method are explained in the chapter describing the relevant method. The most frequently used coordinate systems in relation to cameras are: the (real) world coordinate system, the camera coordinate system, the image coordinate system and the object coordinate system. The difference between them is not always clear because the various methods highlighted in this thesis use distinct coordinate systems which can sometimes be coincident. This implies that a few methods have the same name for coordinates with different properties.
The object coordinate system is the coordinate system of a known object or pattern and is used for the calibration or pose estimation. This coordinate system has its origin on the pattern and its axes are aligned according to the pattern. The unit of the coordinate system is in mm. Often the object coordinate system is aligned with the world coordinate system. In Figure 9 the object or pattern is illustrated as gray and orange dots.
The world coordinate system is the coordinate system in which the result of the pose estimation is

expressed. The unit of the coordinate system is also in mm. The origin and axes of this coordinate system can be chosen anywhere relative to the camera or calibration pattern. This is illustrated in Figure 9.
The camera coordinate system has as origin the camera. The x and y axis are usually aligned with the borders of the image sensor. The z-axis is aligned with the optical as of the camera lens. The value of the z-axis can also be referred to as "distance" or "focus distance". The unit of the coordinate system is in mm. The pose of the pattern can also be represented with respect to this coordinate system as depicted in Figure 9.
The image coordinate system is the two dimensional system of the picture. The origin is on the top left of the image and often the axes are called u and v. The unit of this coordinate system is expressed in pixels.

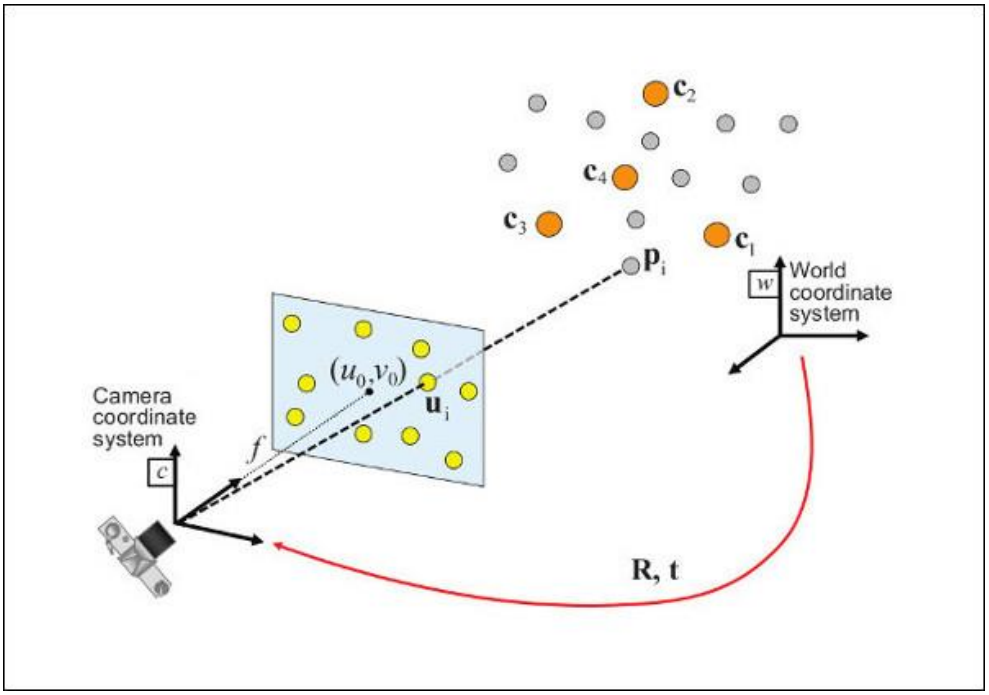

Figure 9: Overview of the different coordinate systems [14]

2.7 Pose estimation

In order to perform pose estimation the camera calibration calculation is inversely executed. The first step is to calculate the rotation and translation from the 2D camera coordinates to the 3D world coordinates by using the 2D coordinates of the distorted image. This results in a transformation between the origin of the camera coordinate system and the origin of the world coordinate system. The relation between the real world 3D coordinates and the camera's 2D coordinates is illustrated in Figure 10.

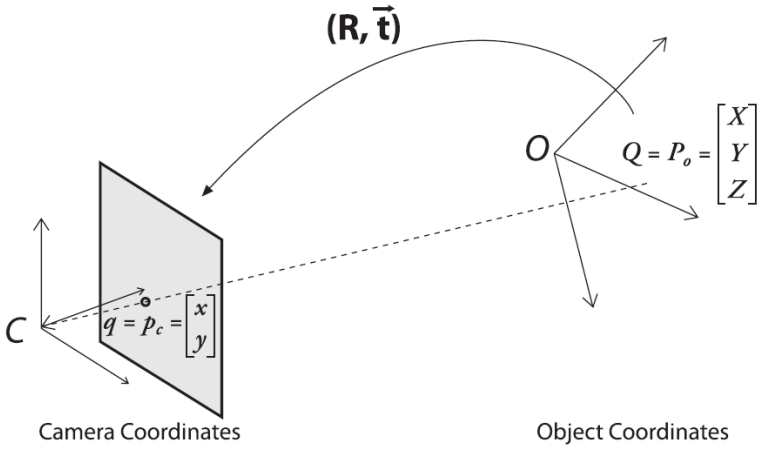

Figure 10: Relation between 3D world coordinate system and the camera coordinate system. [15]

2.7.1 Rotation

Rotating a point around an axis is the same as rotating the coordinate system by the same angle in the reverse direction. For each axis in the 3D real world coordinate system there is a different rotation angle.
Rotations around the x, y and z axis correspond respectively to the angles Ψ, φ and θ.

$$R_x(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix} \tag{10}$$

$$R_y(\varphi) = \begin{bmatrix} cos\ \varphi & 0 & -sin\ \varphi \\ 0 & 1 & 0 \\ sin\ \varphi & 0 & cos\ \varphi \end{bmatrix} \tag{11}$$

$$R_z(\theta) = \begin{bmatrix} cos\ \theta & sin\ \theta & 0 \\ -sin\ \theta & cos\ \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

The total rotation is the product of these three rotation matrices:

$$R = R_x(\psi)R_y(\varphi)R_z(\theta) \tag{13}$$

2.7.2 Translation

The translation vector describes the relation between the origins of two coordinate systems. In this case the translation of the world coordinate system origin with respect to the camera coordinate system origin can be described as [16]

$$T = origin^W - origin^C \tag{14}$$

2.7.3 Transformation

The following equation is applied to convert the vector of a point in the 3D world coordinate system (P^W) to its equivalent point in the undistorted 2D camera coordinate system (P^C):

$$P^c = R_w^c\,(P^w - T_w^c) \tag{15}$$

or

$$P^c = [R_w^c\ T_w^c]P^w \tag{16}$$

The vector of a point in the distorted 2D camera coordinate system is given by the equation:

$$sP^D = A.P^c = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}.[R_w^c\ T_w^c]P^w = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11}\ r_{12}\ r_{13}\ t_1 \\ r_{21}\ r_{22}\ r_{23}\ t_2 \\ r_{31}\ r_{32}\ r_{33}\ t_3 \end{bmatrix}P^w \tag{17}$$

Where S, the scale factor, equals to p_z^C. To solve this equation the same mathematical principles are used as in Direct Linear Transform (DLT) (chapter 4.1). Because the pattern is located in a flat plane, a simplified coordinate system can be used namely the marker coordinate system (Ψ_M). This means that the equation can be rewritten to:

$$sP^U = P^c = [R_M^c\ T_M^c]P^M = \begin{bmatrix} r_{11}\ r_{12}\ r_{13}\ t_1 \\ r_{21}\ r_{22}\ r_{23}\ t_2 \\ r_{31}\ r_{32}\ r_{33}\ t_3 \end{bmatrix}\begin{bmatrix} P_X^M \\ P_Y^M \\ P_Z^M \\ 1 \end{bmatrix} \tag{18}$$

Now P_Z^M can be removed because all the points are in the same plane.

$$sP^U = [R_M^c\ T_M^c]P^M = \begin{bmatrix} r_{11}\ r_{12}\ t_1 \\ r_{21}\ r_{22}\ t_2 \\ 0\ \ 0\ \ 1 \end{bmatrix}\begin{bmatrix} P_X^M \\ P_Y^M \\ 1 \end{bmatrix} = H.\begin{bmatrix} P_X^M \\ P_Y^M \\ 1 \end{bmatrix} \tag{19}$$

$$sP^D = A.P^c = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} r_{11}\ r_{12}\ t_1 \\ r_{21}\ r_{22}\ t_2 \\ 0\ \ 0\ \ 1 \end{bmatrix}\begin{bmatrix} P_X^M \\ P_Y^M \\ 1 \end{bmatrix} \tag{20}$$

The calibration matrix A is known because of the calibration estimation. The rotation and translation matrix have to be determined to calculate the 3D position of the camera. For each known point in the real world and their corresponding distorted image coordinates two equations can be composed each containing 3 unknown variables. This implies that the coordinates of at least six points are needed to find a solution. The more points are known the higher the accuracy of the estimation [16] [17].

2.8 Pose estimation error

There are two ways to measure the pose estimation error. The RMS error and physically measuring the position of the object or pattern. It is not very practical to measure the physical position of the object for each picture. Often electric motors are used to translate and/or rotate the object over a regular distance. This way only a few positions have to be measured and the others can be calculated. The advantage of this system is that it is the closest testing to a real application and the error is returned in

6DOF. The disadvantage is that there are many things that will introduce an error in the pose estimation. First of all there are inaccuracies in the position of the pattern relative to the camera. This is possible due to stability issues, a measuring mistake in the setup etc.
The second way is the RMS error. When the pose of the object is calculated it is possible to reproject the feature points on to the image. The location of all the feature points in the image are calculated by using their position in the calculated object. This results in a difference between the detected feature points and the projected feature points as illustrated in Figure 11.

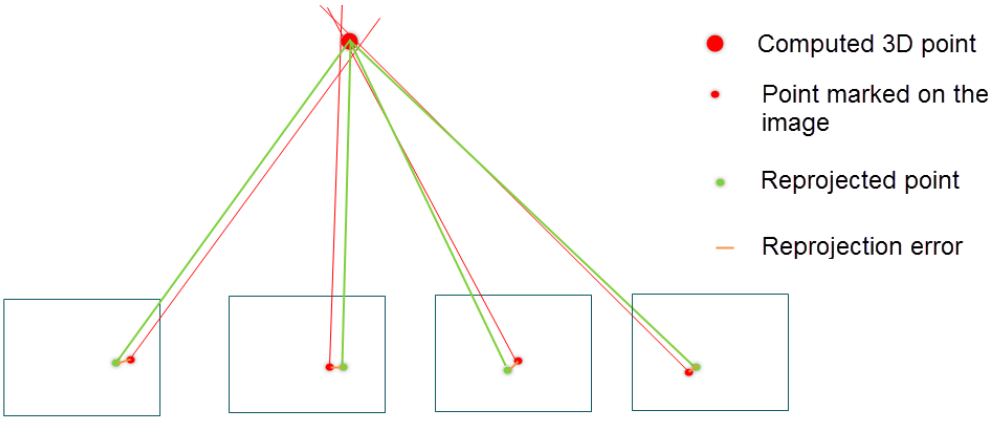

Figure 11: Visualization of reprojection [18]

The Euclidean distance between the detected feature point and the reprojected point is used to calculate the RMS error value. The RMS error is calculated with the following equation:

$$RMS\ error = \sqrt{\frac{\sum_{i=1}^{n}(\sqrt{\left(x_{fpi} - x_{rpi}\right)^2 + \left(y_{fpi} - y_{rpi}\right)^2}}{n}} \tag{21}$$

Were n is the total number of detected points, (x_{fpi}, y_{fpi}) are the coordinates of a feature point and (x_{rpi}, y_{rpi}) are the coordinates of the reprojection of that feature point. This value is a representation of the average distance between the feature point and the projected point.
The advantage of using the RMS error to test the algorithm is that all other sources of errors are eliminated and it can be simulated on a computer. A disadvantage of the RMS error is that it is sensitive to outliers. [18]

3 Influence on the accuracy of pose estimation by slight deviations

In 2008 a study was done by Thomas Luhmann [19] to find the influence of slight variation or errors on the accuracy of a pose estimation method. The method used in this study uses space resection to find the 6DOF pose of an object with relation to a reference frame. This method is similar to the DLT method discussed in section 4.1. The study uses a test with virtual data. An image resolution of 1300 x 1030 pixel was used with a focal length of 8 mm. On this image an object with the size of 300 x 300mm with 9 reference points was captured. Behind the object was a reference grid of 1200 x 1200 mm with 49 control points. The viewing distance was variable from 500 to 1500 mm. Figure 12 gives an overview of the virtual test environment.

A Monte-Carlo simulation was used to provide an efficient method to detect the numerical properties of a model while introducing artificial noise [20].

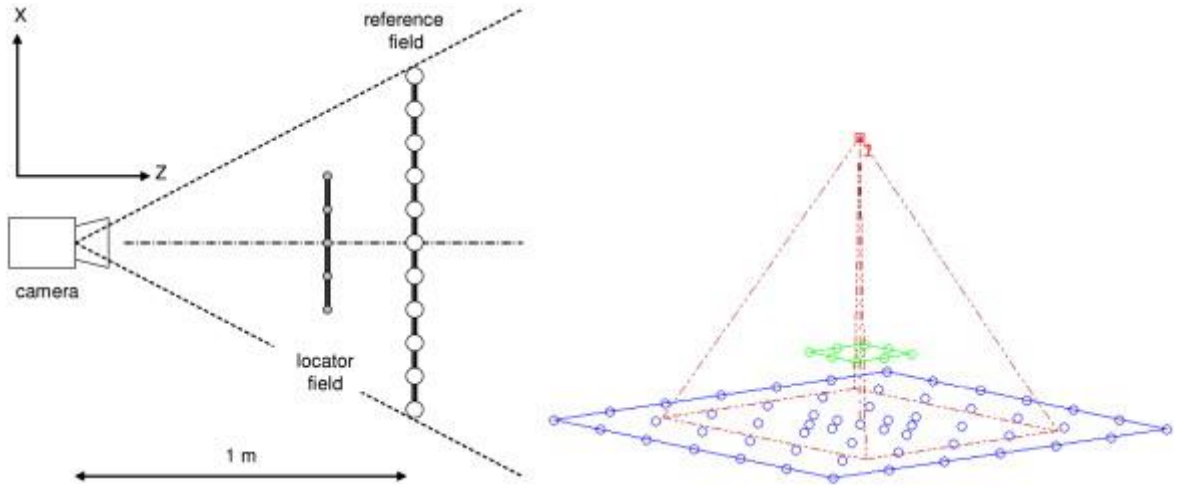

Figure 12: Overview of the virtual test environment [20]

3.1 Variation in planar points

In a first test, an image error was introduced increasing the image coordinate measurement noise from the original 0.35 µm to 1.7 µm. As is clearly visible in Figure 13, the accuracy of the detected x, y and z measurement decrease linearly with a decrease in image measurement accuracy. The accuracy of the depth translation is influenced noticeably more by image noise than in-plane translations where a maximum precision of 0.1 mm can be achieved.

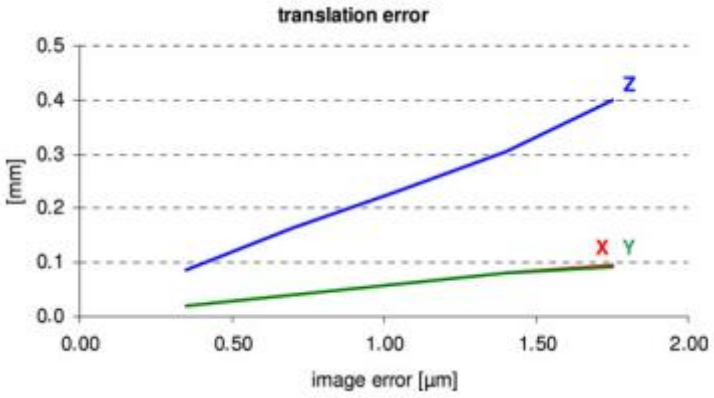

Figure 13: Translation error in function of image error [19]

The same test is performed for the rotation (roll κ, pitch ω, and yaw φ). The results are plotted in Figure 14. It's clear from this graph that the roll is affected far less by a decrease in image measurement accuracy compared to the out-of-plane pitch and yaw. The roll can be calculated up to an accuracy of 0.02 degrees. Both out-of-plane rotational accuracies are also affected nearly identical with an angular precision of up to 0.05 degrees.

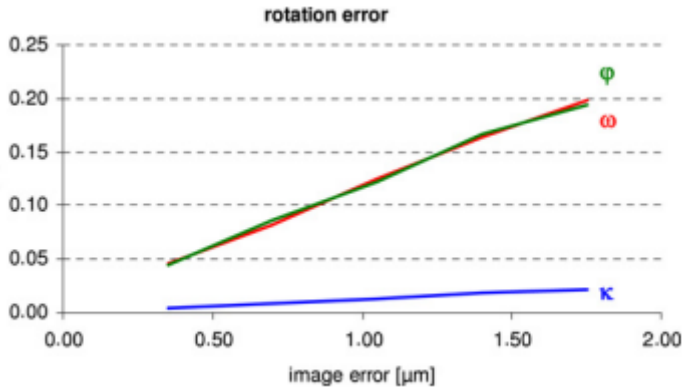

Figure 14: Rotation error in function of image error [19]

3.2 Variation in object size

If the object is scaled down from 100 % (1300 mm x 1030 mm) to 20%, roughly 60 mm x 60 mm, the accuracy of the rotation and translation also decreases as shown in Figure 15. Important to note here is that the decrease in accuracy is not linear like in the previous test. The accuracy of the translation and rotation appear linear up to around 50 % scaling factor. After that, the decrease in accuracy of out-of-plane rotations and translation occurs drastically. One can also note that the in-plane translations and rotation are barely affected by a change in scaling factor.

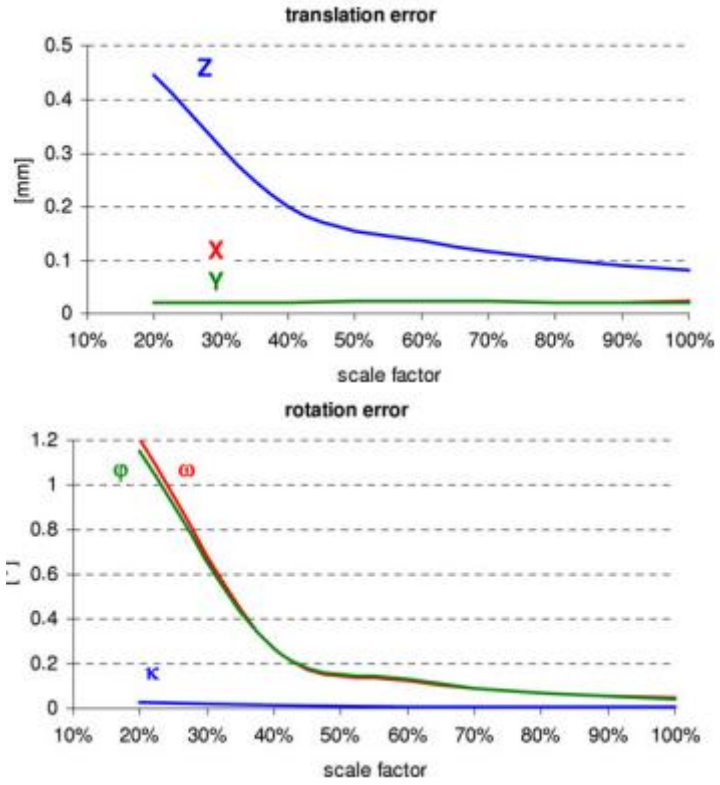

Figure 15: a) Translation error in function of scale factor of object
b) Rotation error in function of scale factor of object [19]

3.3 Variation on object tilt

For the third test, the object was rotated around the x axis. This is an out-of-plane rotation which would make the field of vision narrower. In this test, the x –axis was tilted at varying angles starting from 0° up to 50°. The results are plotted in Figure 16. Figure 16 a) shows the translation error in function of this tilt. The translations are barely affected by this tilt. The depth-estimation decreases slight in accuracy from 0^0 to 10° but stabilizes after that. The rotational error of the out-of-plane rotations increases when the object is tilted as seen in Figure 16 b). The reason is that a tilted target provides more information about the viewing direction [19].

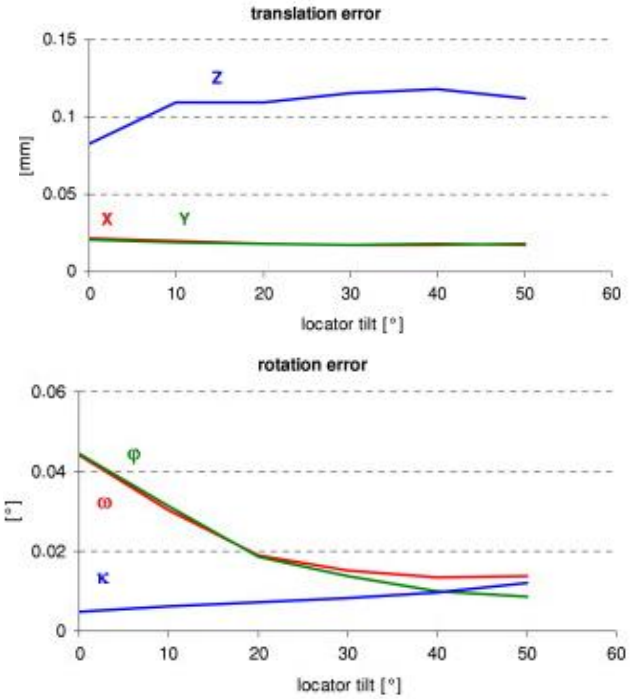

Figure 16: a) Translation error in function of tilt around x-axis of the object.
b) Rotation error in function of tilt around x-axis of the object. [19]

3.4 Variation in focal length

Changing the focal length while maintaining a fixed image distance will alter the image scale and viewing angle. In this test the focal length is varied between 4 mm and 16 mm. The result is shown in Figure 17. Increasing the focal length will decrease the translation and rotational errors. Please note that the out-of-plane rotations and translation will be affected more by an increase in focal length compared to the in-plane rotation and translations. Increasing the image scale and narrowing the field of view will improve the accuracy of the pose estimation [19].

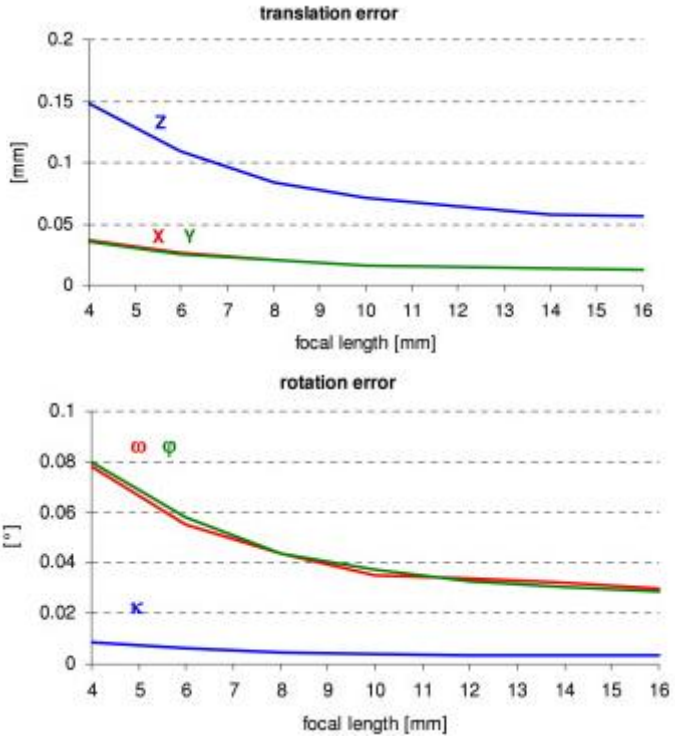

Figure 17: a) Translation error in function of a variable focal length
b) Rotation error in function of a variable focal length

4 Non-iterative pose determination methods

4.1 Direct Linear Transformation (DLT)

4.1.1 Introduction

Direct linear transformation (DLT) is an algorithm to solve a linear transformation of a set of points in multiple views [21]. It was first discussed in 1971 by Y.I. Abdel-Aziz and Dr H.M. Karara. It's one of the older but also most well-known position determination techniques.
DLT uses linear equations to find 11 coefficients describing the transformation [22].

4.1.2 Operation principles of DLT

Like the other pose estimation techniques discussed in this thesis, it starts with a reference point in the real world coordinate system (X, Y and Z) and its projection (u, v) in the image-plane reference frame as shown in Figure 18. In this reference point in the real world coordinates is denoted as O. Its projection onto the image plane is denoted as I. N identifies the projection center of the camera. All three points are collinear, as clearly shown in the figure. This forms the basic principle of the DLT method [23] .

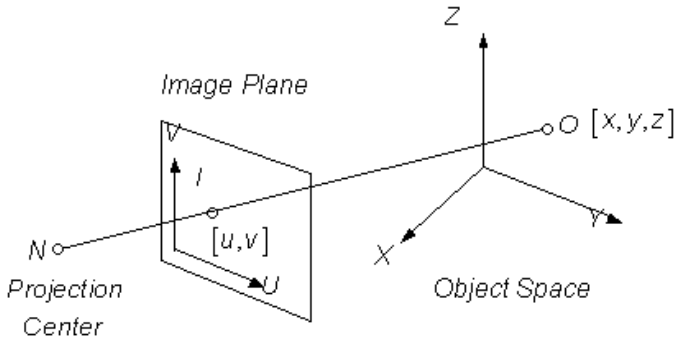

Figure 18: DLT projection overview [23]

A new axis, called W, is added perpendicular onto the image plane, effectively extending the 2D image plane, into a 3D space. The W-coordinates of all points in the original 2D image frame are always 0. Next a new point is added in the 2D image frame, called P. P is the principal point of the image located at (u_0, v_0) in the 2D image frame. The principal axis connects the projection center of the camera and the principal point and is perpendicular to the 2D image frame. The distance d is the principal distance and denotes the distance between N and P. The coordinates of the points I, P and N in the 3D image space coordinate system become $[u, v, 0]$, $[u_0, v_0, 0]$ and $[u_0, v_0, d]$ respectively. An overview of the new image space is shown in Figure 19 [23].

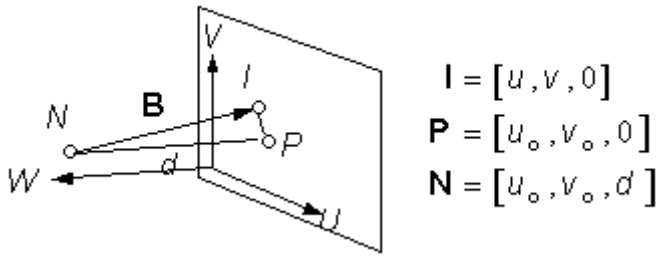

Figure 19: DLT extension of 2D image frame into 3D image space [23]

Now assume the vector A, connecting N and I. This vector will be denoted as $[x - x_0, y - y_0, z - z_0]$ where (x, y, z) are the real-world coordinate of the reference point O and (x_0, y0, z0) the real-world coordinates of the projection center N. A second vector is assumed, connecting the projection center and the principal point, both represented in the 3D image frame reference system. This vector is called B and denoted as.$[u - u_0,\ v - v_0,\ -d]$. As mentioned earlier, the points N, I and O are collinear, as seen in Figure 18. This implies that the vectors A and B are equal up to an unknown scalar multiplication c shown in equation (22).

$$B = cA \tag{22}$$

Vector A uses the 3D real-world or object-space reference frame and vector B uses the 3D image reference frame. For this reason, a 3x3 transformation matrix $T_{I/O}$ is introduced to relate these coordinates.

$$T_{I/O} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{23}$$

A transformation from vector A in the image reference frame to the object-space reference frame is shown in (24) where A^I is the vector A in the image reference frame and A^O the same vector in the object reference frame.

$$A^I = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} A^O \tag{24}$$

Using equation (24) in (22) yields equation (25) where vectors A and B are written in full [22].

$$\begin{bmatrix} u - u_0 \\ v - v_0 \\ -d \end{bmatrix} = c \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} * \begin{bmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{bmatrix} \tag{25}$$

Using the last row of equation (25) the scaling factor can be found and substituting it in the first two rows results in in two other equations (26) and (27).

$$u - u_0 = -d\ \frac{r_{11}(x - x_0) + r_{12}(y - y_0) + r_{13}(z - z_0)}{r_{31}(x - x_0) + r_{32(y - y_0)} + r_{33}(z - z_0)} \tag{26}$$

$$v - v_0 = -d \frac{r_{21}(x - x_0) + r_{22}(y - y_0) + r_{23}(z - z_0)}{r_{31}(x - x_0) + r_{32(y-y_0)} + r_{33}(z - z_0)} \quad (27)$$

Please note that u, u_0, v and v_0 in equation (26) and (27) use 3D image reference frame, however the units in this frame are not expressed in pixel, so a conversion factor λ is added shown in (28) and (29).

$$u - u_0 = -\frac{d}{\lambda_u} \frac{r_{11}(x - x_0) + r_{12}(y - y_0) + r_{13}(z - z_0)}{r_{31}(x - x_0) + r_{32(y-y_0)} + r_{33}(z - z_0)} \quad (28)$$

$$v - v_0 = -\frac{d}{\lambda_u} \frac{r_{21}(x - x_0) + r_{22}(y - y_0) + r_{23}(z - z_0)}{r_{31}(x - x_0) + r_{32(y-y_0)} + r_{33}(z - z_0)} \quad (29)$$

Lastly equations (28) and (29) are rearranged for x, y and as seen in (30) and (31). There the coefficients are labeled as $L_{n|n=1,2,\ldots,11}$. These are the 11 DLT parameters that need to be found and represent the relation between the real-world coordinate system and the image reference system [22].

$$u = \frac{L_1 x + L_2 y + L_3 z + L4}{L_9 x + L_{10} y + L_{11} z + 1} \quad (30)$$

$$v = \frac{L_5 x + L_6 y + L_7 Z + L_8}{L_9 x + L_{10} y + L_{11} z + 1} \quad (31)$$

To obtain the 11 DLT parameters an overdetermined system is needed and, as seen in (30) and (31), each reference point is expressed in 2 equations. This means that for finding the DLT parameters a minimum of 6 reference points are required [23].

4.1.3 Accuracy

Since DLT is an old and well known method for pose estimation, it has been used several times to compare other pose estimation methods to. In this thesis we will use the data from a study done in 2008 by Thomas Peterson at Aalborg University [24]. The study uses synthetic data with even distribution of reference points on a 1200 x 800 pixel image with focal length 882 pixels on both axis. A virtual camera positioned at (0, 0, -25) in 3D space is used.
Figure 20 a) and b) give an overview of the influence of Gaussian noise on the DLT estimated translation and rotation respectively. As is clearly shown in this graph, the DLT method is highly susceptible to noise. The x-axis of these graph shows an increasing noise level standard deviation in pixel, starting at 0 and going up to 10 pixel standard deviation noise. The y-axis shows the distance between the estimated position and the ground truth position of those points expressed as a relative camera position. An error of 1 indicates an error of 100 % the distance between the camera and the object. In Figure 21 b) the y axis shows the same distance to the ground truth expressed in rotations. 10 measurement points were used to create these graphs.

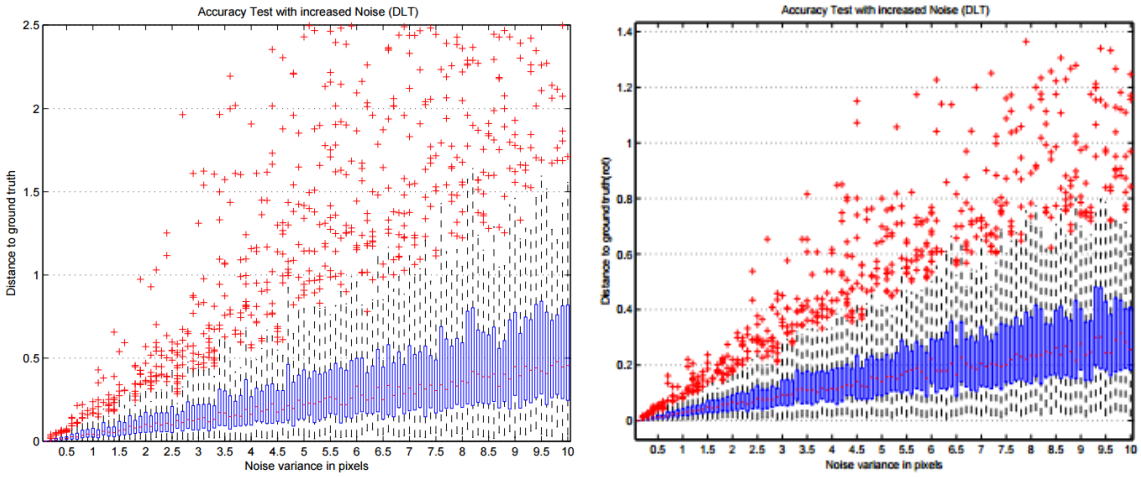

Figure 20: a) DLT translation accuracy in function of noise level. b) DLT rotation accuracy in function of noise level [24].

In case of no noise, the maximum accuracy achieved by this study is 0.14% error for translation. The maximum rotational accuracy achieved by DLT in this study is 0.0008 rotation or 0.288 degrees [24].

As mentioned earlier, DLT needs a minimum of 6 points to estimate as position, but like nearly all pose estimation methods, if more points are used, the accuracy increases. Figure 21 shows the influence of extra points on the accuracy of DLT. As seen in these graphs, using the minimum requirement of 6 reference points gives a very poor pose estimation. A median translation error of 4 % and a median rotational error of 9-10 degrees occur. However, increasing the amount of reference points also increases the quality of the pose estimation greatly [24].

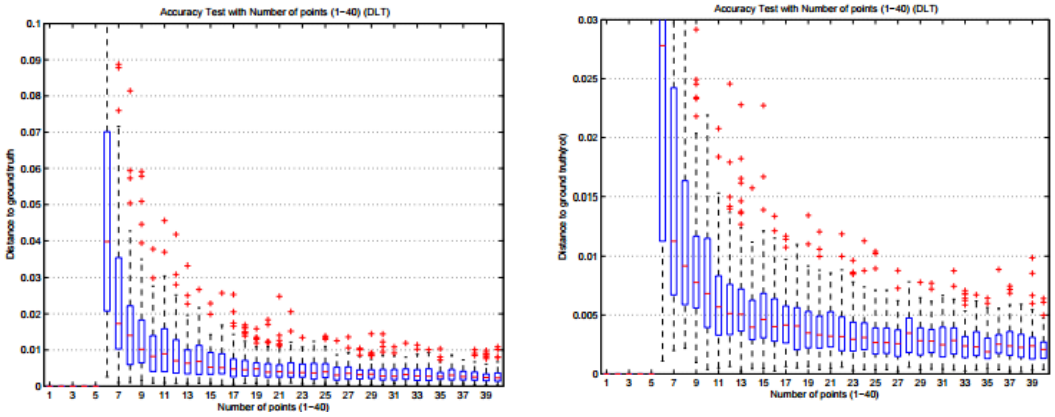

Figure 21: a) DLT translation accuracy in function of increasing number of reference points.
b) DLT rotation accuracy in function of increasing number of reference points [24].

4.1.4 Conclusion

DLT is one of the earlier pose estimation methods and that is clearly reflected in its performance. First of all, it requires a minimum of 6 pose estimation points where most other methods require 4. This influences the speed of execution of this method. Even though it is not an iterative method, finding 11 parameters requires more time than the 6 parameters EPnP(see chapter 4.3.2) for example need. In case of low noise DLT is fairly accurate, however DLT is highly susceptible to noise. Slight noise increases the translation and rotation error noticeably. Also using the minimum required 6 reference points yields a poor accuracy, starting at 15 reference points, the accuracy seems to stabilize, but this will drastically influence the execution time [22] [23] [24].

4.2 Perspective-n-Point problem (PnP)

Perspective-n-Point (PnP) is a problem which finds its origin in camera calibration. It is a mathematical problem that tries to find the position and orientation of a calibrated camera using n 3D reference points in the real world coordinate system viewed as 2D points in an image. The position of the camera is defined as 6 degrees-of-freedom by a translation (X, Y and Z coordinate) and a rotation (roll ϑ, pitch θ and yaw φ). An overview of the problem is shown in Figure 22.

Equation (32) shows the mathematical approach to the PnP problem. In this formula $[x_i \quad y_i \quad z_i \quad 1]^T$ are the real world coordinates of a reference point i. $[u_i \quad v_i \quad 1]^T$ is the corresponding homogeneous point detected in the image frame. w_i stands for the scaling factor of this image point. The 3 x 3 matrix (A) on the right hand side is the calibration matrix, containing the scaled focal lengths (f_u and f_v) and the principal point (u_c, v_c). The skew γ is often very small and often assumed 0, as seen in equation (39). The 3x4 matrix is the transformation matrix [R|T] with a 3 x 3 submatrix denoting the rotation and a 3 x 1 matrix identifying the translation. The transformation matrix is the unknown that PnP methods try to find. [25] [26]

$$w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} \tag{32}$$

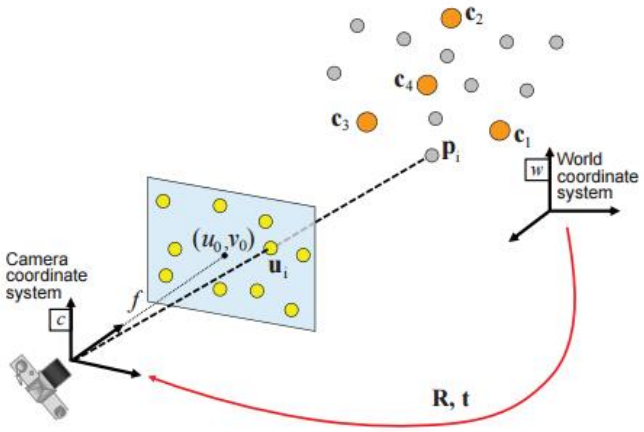

Figure 22: Overview of the PnP problem [14].

There exist several methods to solve the PnP problem. Each method uses a slightly different technique to solve this problem. The methods can be non-iterative, where a solution comes after the first

estimation or iterative where first an initial guess is made which is then refined using an iterative process. In the next section the Efficient Perspective n Point (EPnP) method will be discussed in detail. It is a fairly accurate and computationally fast non-iterative method. EPnP is one of the most well-known solutions for the PnP problem and often used to compare with newer techniques [27].

4.3 Efficient PnP (EPnP)

4.3.1 Introduction

Perspective-n-Point problems are already described in the previous section. In 2008 Vincent Lepetit et al. proposed an improved method for the PnP problem [27]. It provides a non-iterative solution which decreases the time and computation power needed compared to iterative methods which do an initial guess and refine the estimated pose with every iteration. The proposed method has a computational complexity of O(n). This means that the required amount of computations to solve the equations scales with the amount of reference points (n) that need to be detected in an image frame. The method does require n to be larger than or equal to 4 points in order to get a position estimation. Most other methods for solving the PnP have a higher complexity. For example another low complexity method for PnP by Fiore, introduced in 2001, has a complexity of $O(n^2)$ where the computational power scales exponentially with the amount of feature points n [28]. Some higher accuracy methods like those proposed by Ansar and Daniilidis have a complexity which can go up to $O(n^8)$ [29].
This paper claims even better accuracy and noise rejection compared to those other non-iterative methods [27].

4.3.2 EPnP algorithm

The EPnP method starts with a set of known feature points in the 3D world coordinate system and corresponding points in the 2D image coordinate system. It aims to retrieve the coordinates of those feature points in the 3D camera coordinate system. If those coordinates are found, the Euclidian motion that aligns those sets of coordinates can be found. This motion contains the translation and rotation. Contrary to other iterative PnP methods which start by estimating the depth of the reference points in the camera coordinate system, this method expresses those coordinates as a weighted sum of Virtual control points. If the reference points are coplanar, the minimum number of points needed to estimate this motion is 3. Otherwise at least 4 points are needed. The coordinates of the weighted sum points in the camera coordinate system are unknown. Because this technique uses weighted sums and not just the depth estimation of every reference point, it requires less computational power when a large amount of reference points (n) is used [27].

First, a set of reference points in the real world coordinate system are needed. These points are denoted as:

$$p_i, i = 1, 2, \dots, n. \tag{33}$$

Each reference point is expressed as a weighted sum of 4 virtual control points.

The control points are denoted as:

$$c_j, j = 1, 2, 3, 4. \tag{34}$$

The relationship between the control points and the reference points is shown in the following formula:

$$p_i^w = \sum_{j=1}^{4} \alpha_{ij} c_j^w \tag{35}$$

In equation (35), the w superscript specifies in which coordinate system the points are viewed. The symbol w signifies the world coordinate system, c is the camera coordinate system. The relationship for the world coordinate system is the same for the camera coordinate system, as shown in equation (36)

$$p_i^c = \sum_{j=1}^{4} \alpha_{ij} c_j^c \tag{36}$$

In both equations is the symbol α_{ij} present. This identifies the homogeneous barycentric coordinates of the reference points using the 4 control points. Barycentric coordinates use 3 points in a 2D system and 4 points in a 3D system do describe coordinates. The coordinates are given as a homogeneous combinates of the 3 or 4 reference points. More information about Barycentric coordinates can be found in [30]. The 4 control points form a tetrahedron and each reference point can be expressed as a combination of the 4 control points. An example is shown in Figure 23. In this figure, point B can be expressed as a homogeneous combination of points 1, 2, 3 and 4.

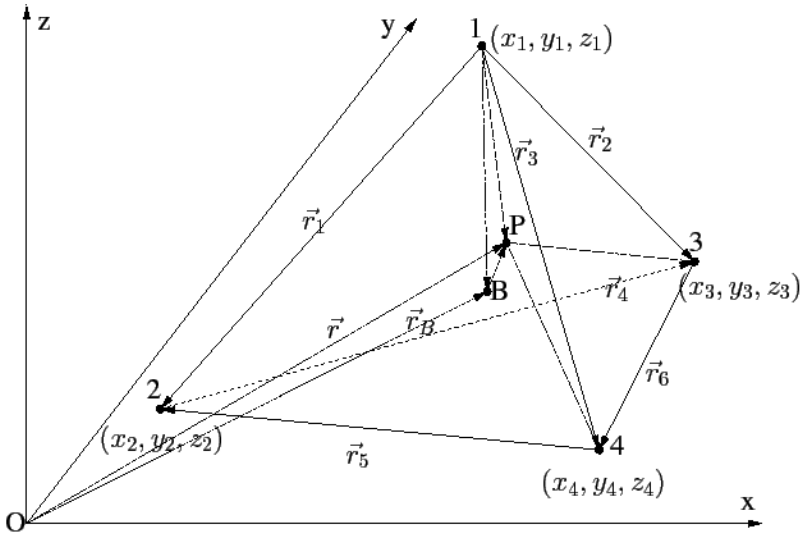

Figure 23: Tetrahedral barycentric coordinates [31]

Please note that the sum of the coefficients has to be 1.

$$\sum_{j=1}^{4} \alpha_{ij} = 1 \tag{37}$$

Theoretically the position of the control points can be chosen randomly, however the author of this method claims that one of the most stable ways to choose these control points is using one of the control points in the centroid of the reference points as the origin of a reference coordinate system and the other 3 points to construct the principal direction (x, y and z) [27].

The next step is constructing a matrix M. The solution of the problem lies in the kernel of this matrix, expressed as a vector. The dimension of the matrix M can be 2n x 12 or 2n x 9, depending on the amount of control points (4 control points with each 3 parameters (x, y and z) equal to 12 vectors describing each reference point). The 2n dimension identifies the dimension of the image coordinate system (2D). So each reference point has 12 vectors describing its x coordinate in the image coordinate system and 12 vectors describing its y coordinate in the image coordinate system, resulting in 24 vectors describing each point. Equation (38) shows the connection between the 2D projections of a reference point pi, represented as qi and the same reference point viewed in the camera coordinate system, notated as p_i^c. A is a matrix that holds the camera internal calibration parameters. A detailed description of these parameters is given in section 2.3. In short, these internal camera parameters are the 2D focal length coefficients fu and fv and the 2D principal point coefficients uc and vc.

$$\forall i, w_i \begin{bmatrix} q_i \\ 1 \end{bmatrix} = A p_i^c \tag{38}$$

In equation (38) w_i contains scalar projective parameters. Using equation (36) , equation (38) can be rewritten, shown in (39). The 4 control points are now written by using their x, y and z coordinates in the camera frame.

$$\forall i, w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix} \tag{39}$$

This equation results in 12 unknown parameters from 4 control points $\{x_j^c \quad y_j^c \quad z_j^c\}_{j=1,2,3,4}$ and another n unknown parameters in w_i, the scalar projective parameters, with n the amount of reference points used. However the parameters in w_i can be calculated using equation (39). Using the last row of this equation yields: [32]

$$w_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c \tag{40}$$

Equation (40) is substituted in the first two rows of equation (39). This results in two equations (41) and (42), which show the relation between the 2D projection coordinates of a reference point I (u_i and v_i) and the 3D coordinates of the same reference point, expressed as barycentric coordinates by using 4 control point in the camera coordinate system.$(x_j^c, y_j^c, z_j^c)_{j=1,2,3,4}$.

$$\sum_{j=1}^{4} \alpha_{ij} z_j^c u_i = \sum_{j=1}^{4} \left(\alpha_{ij} f_u x_j^c + 0 + \alpha_{ij} u_c z_j^c\right) \Leftrightarrow$$
$$0 = \sum_{j=1}^{4} \left(\alpha_{ij} f_u x_j^c + \alpha_{ij}(u_c - u_i) z_j^c\right) \quad (41)$$

$$\sum_{j=1}^{4} \alpha_{ij} z_j^c v_i = \sum_{j=1}^{4} \left(0 + \alpha_{ij} f_v y_j^c + \alpha_{ij} v_c z_j^c\right) \Leftrightarrow$$
$$0 = \sum_{j=1}^{4} \left(\alpha_{ij} f_v y_j^c + \alpha_{ij}(v_c - v_i) z_j^c\right) \quad (42)$$

With this substitution, the unknown parameters from w_i disappear from the equation. As mentioned earlier, the goal is to create a matrix $M \in \mathbb{R}^{2n \times 12}$ where 2n identifies the 2D coordinates from the image coordinate system of each reference point i, and the 12: x, y and z coordinates from the same reference point represented as barycentric coordinate using 4 control points in the camera coordinate system. To solve the equation a singular vector x is constructed where:

$$Mx = 0 \quad (43)$$

In equation (43) M represents the matrix which contains the coefficients from equation (41) and (42), x is a 12-vector containing the unknowns $\left[c_1^{cT}, c_2^{cT}, c_3^{cT}, c_4^{cT}\right]^T$. These are the coordinates of the 4 control points in the camera coordinate system. In order to solve equation (43), x is rewritten as shown in equation (44).

$$x = \sum_{i=1}^{N} \beta_i k_i \quad (44)$$

In this equation, k_i represents the null singular values of M. These are the same as the null eigenvectors of the matrix $M^T M$. The solution of equation (43) can be written as a linear combination of eigenvectors of the matrix $M^T M$ using the $\beta_{i \mid i=1,\ldots,N}$ coefficients.
According to the author, this method works even when there are less equations than unknowns. For example, if 4 reference points are used, matrix M will consist of 8 equations to find the 12 unknown parameters of x.
If there are 6 reference points, matrix M will consist of 12 equations to solve the 12 unknown parameters. And the dimension N of the null space of $M^T M$ is exactly one. In the case of an orthographic camera instead of a perspective one, the dimension N is increased to 4. A change in depth of the 4 control points does not alter the projections of the reference points. An orthographic camera shows a scene without the perspective distortion [33]. This effect can be achieved by increasing the focal length of the camera. In small focal lengths $M^T M$ has one zero eigenvalue. Increasing the focal length will result in the camera behaving more like orthographic than perspective, and the four smallest eigenvalues of $M^T M$ will reach near zero.
This means that there are 4 possible solution for N: 1, 2, 3 or 4. This depends on the configuration of the reference points, the amount of noise in the image and the focal length. This also means that there

are 4 possible solutions. All 4 solutions are calculated and reprojected. Reprojection is explained in section 2.8. The N which results in the smallest reprojection error is kept. The solution is different for each N. The central idea is using quadratic equations to calculate the distance as shown in equation (44).

$$res = \sum_i dist^2 \left(A[R|t] \begin{bmatrix} p_i^w \\ 1 \end{bmatrix}, q_i \right) \tag{45}$$

In this equation matrix A contains the internal camera parameters. [R|t] is the transformation matrix consisting of the rotation and translation. p_i^w is the reference point, viewed in the world coordinate system and q_i is the 2D projection of the reference point. *dist( )* is the 2D distance between both points in homogeneous coordinates.

The EPnP method is one of the most accurate non-iterative PnP solution. However, compared to some iterative methods its accuracy is slightly less. The author proposes an optimization to the EPnP method to increase its accuracy using the Gauss-Newton optimization [34]. This method will refine the $\beta_{i|i=1,2,3,4}$ values in equation (44) by picking a value that has the smallest change in distance between the control points [27].

4.3.3 Accuracy

4.3.3.1 Non planar

The accuracy of the EPnP method is calculated for a planar case (minimum 3 reference points) and non-planar case (minimum 4 reference points) separately. The EPnP method is compared to other recent iterative and non-iterative pose estimation methods. The setup for measuring the accuracy of this PnP method uses a 640 x 480 image from a calibrated virtual camera. On this image synthetically 3D-2D reference points are produced. First, the rotational accuracy is discussed as illustrated in Figure 24. A boxplot representation of the EPnP method and its rotational error in function of the Gaussian noise in the image is shown. 6 reference points are used for this measurement. The red plus signs in the boxplot are the high errors detected by the method. In total 300 measurements (20 for each Gaussian noise) are calculated and plotted.

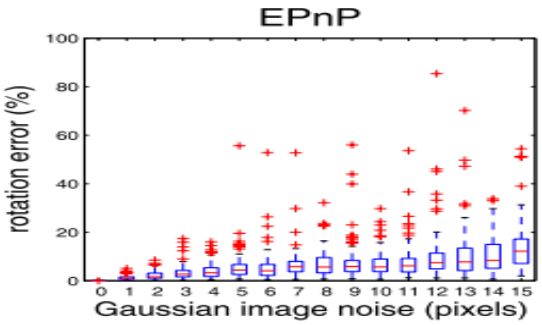

Figure 24: EPnP rotation error in function of image noise [27].

Figure 25 shows a comparison is made among Ansar and Daniildis, (AD), Clamped DLT, Lu's et al. (LHM) , LHM using EPnP's data for a good initial guess (EPnP + LHM) and the previously discussed EPnP using Gauss-Newton optimization (EPnP+GN).

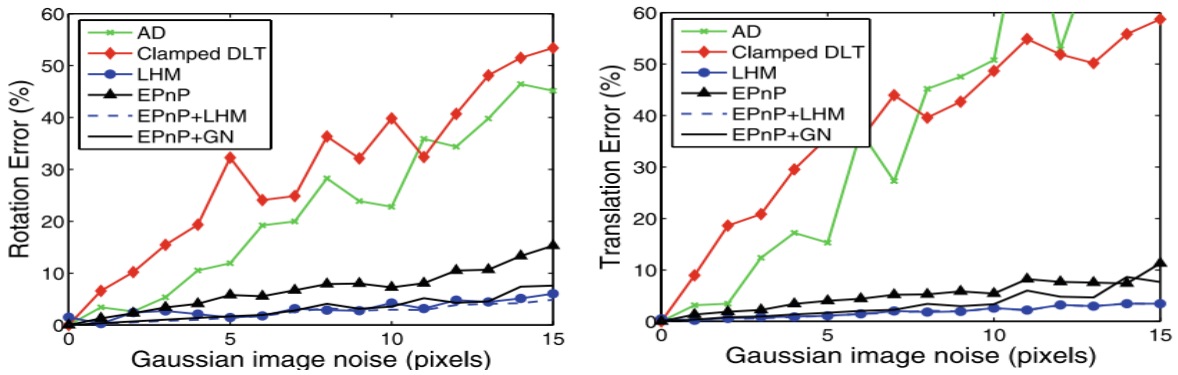

Figure 25: Non planar mean rotation and translation error comparison in function of image noise [27]

Figure 25 depicts both the rotational and translational errors using a fixed number of reference points (n = 6) with an increasing standard deviation of Gaussian noise from 0 to 15.

As mentioned earlier in this section, the amount of reference points used increases the accuracy of this method. This connection is shown in Figure 26 for the rotational error and the translation error. A Gaussian noise with standard deviation of 5 pixels is fixed and the amount of reference points is increased from 5 to 20 [27].

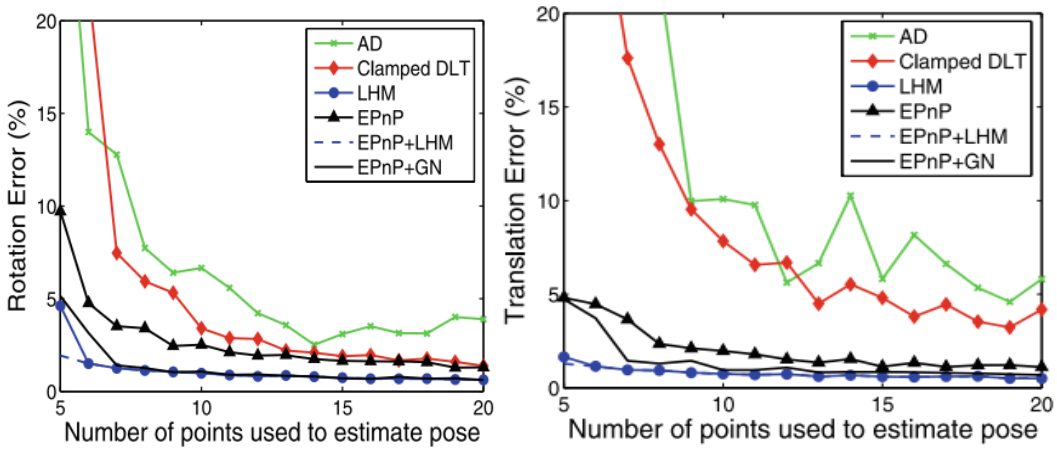

Figure 26: Non planar rotation and translation error in function of reference points [27].

4.3.3.2 Planar case

Reference points, lying on a plane result in serious instability when a camera pose based on those points is estimated due to an ambiguity. An overview of the rotational and translational errors in a planar case in function of the Gaussian noise are depicted in Figure 27. Please note that Clamped DLT is no longer present because this method is not applicable in planar configurations.

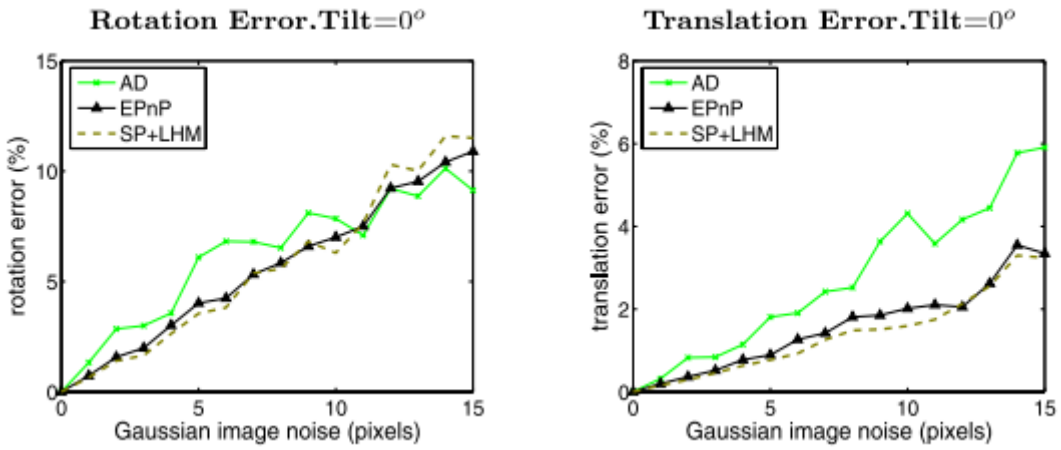

Figure 27: Planar rotation and translation error in function of Gaussian Noise [27].

However, a study, conducted in 2013 by Zheng et all. [35] shows that compared to more recent PnP algorithms, EPnP falls short in the planar case as shown in Figure 28. In the left figure, EPnP is not even represented in the graph.

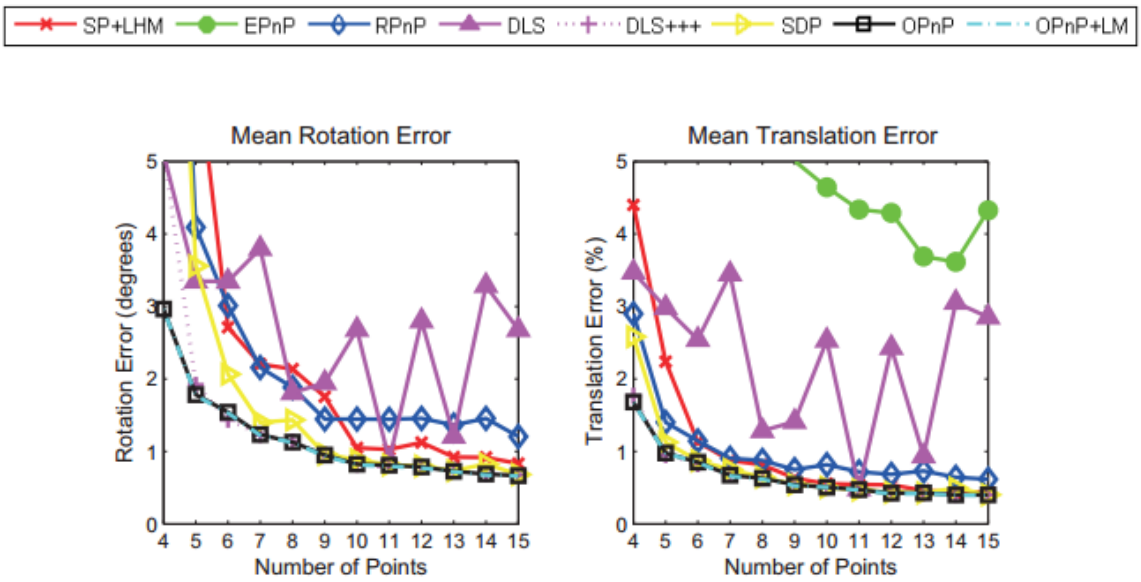

Figure 28: Comparison of EPnP with recent technologies [35].

4.3.4 Conclusion

EPnP is a well-known method for solving the PnP problem and is fairly accurate compared to other non-iterative methods and even some iterative methods. The O(n) complexity makes it an ideal choice when a fast position estimation needs to be implemented where errors of 5 % or more are acceptable.

4.4 Linear pose estimation from points or lines

This method has been developed in 2003 [36] for fast pose estimation with few world objects to determine the pose. There are two variations. One of them uses points, n point linear or NPL, another one uses lines, n line linear or NLL. This is the first non-iterative pose estimation method that uses lines. It has been developed for the National Aeronautics and Space Administration of The United States of America. The result of this n point or lines solution is given in camera coordinates. It can give the solution of the pose estimation if there are at least 4 known points and if they do no't lie in a critical configuration [36].

4.4.1 Principals pose estimation

In this chapter we describe the pose estimation method. Unfortunately the original paper [36] is written for researchers with in depth knowledge of mathematics. However, this is outside the scope of this thesis and therefore this section only explains the principals that are used. Interested readers can find more information in [36].
The method is based upon depth recovery of the world points. This is illustrated in Figure 29.

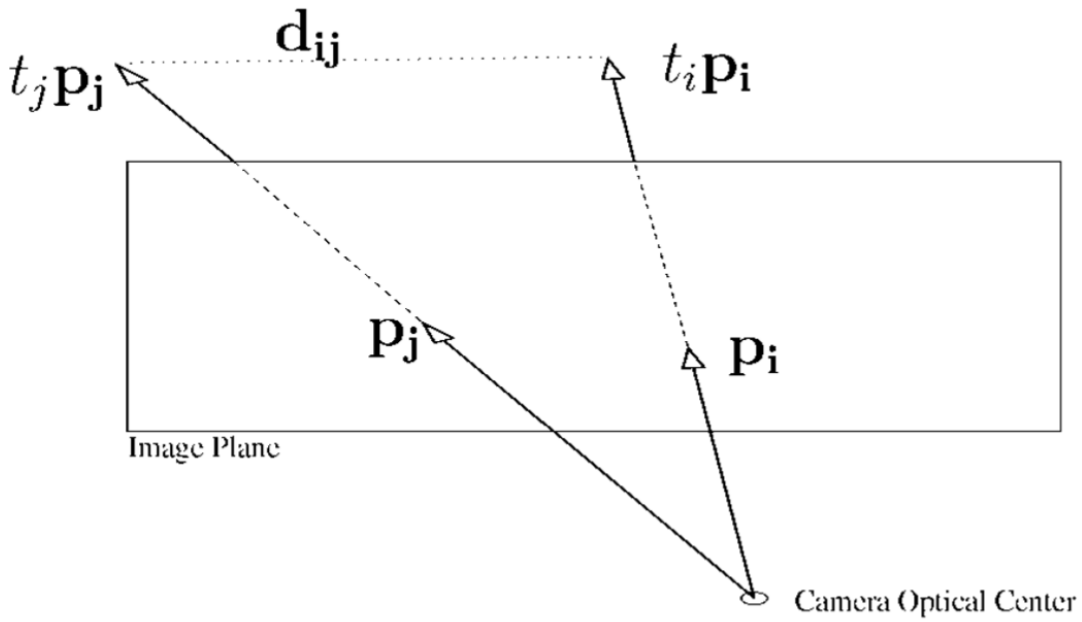

Figure 29: Visualization of depth recovery of the world points. [36]

The basic geometric constraint used in this method relates the distance between points in world coordinates d_{ij} and the scale factors t_i and t_j associated with the projections P_i and P_j. This method avoids a degree increase in the equations derived from these constrains and all couples all n points in a single system of equations which are solved simultaneously. It requires no initialization and can be used for a small number of points or lines. It also guarantees to find a unique solution, if there is one. This is in contrast with iterative solutions such as Lowe [37], which typically have a slow conversion with bad initialisations. These iterative solutions can even converge to a local minima and require more points for stable results [36].

4.4.2 Test setup

The researchers who developed this method use Matlab to simulate the measurements. It is assume that the point method has a focal length of 1500 and that the line case has a focal length of 600. A random pose is generated and the correct image points are calculated. Then an error is introduced in both the x and y direction for each point. The error is introduced as Gaussian noise between minus three and three times the standard deviation (σ). Further they use these points to calculate the pose of the object and the relative error that this pose has. Most simulations are repeated over 400 times. They comparison is conducted between different point techniques and one other line technique. The point techniques are:

1. PM: direct recovery and decomposition of the full projection matrix from six or more points by singular value decomposition(SVD) methods. SVD is a way to factorize a matrix. A triangle(△) is used to indicate this method on the graphs.
2. F: the n point linear algorithm of Fiore. This algorithm is indicated by a square (□).
3. QL: the n point linear algorithm of Quan and Lan [38]. This is signified by a diamond(◇).
4. LHM: the iterative algorithm of Lu et al (see chapter 5.1). initialized at ground truth. This is indicated by the circle(○). This algorithm is used as a reference and is expected to have the highest accuracy.

The line technique is

1. KH: the iterative algorithm of Kumar and Hanson. KH is initialized at the ground truth translation and rotation (KHRT indicated by a triangle △) this is done to evaluate the absolute performance.
2. KH: the iterative algorithm of Kumar and Hanson. KH is initialized at the ground truth translation and identity rotation (KHT indicated by a square □). This shows the disadvantage of having initialisation of the algorithm. [36]

4.4.3 Results point simulation 1

The first simulation tests the dependence on the noise level. The Gaussian noise varies between σ = 0.5 and 4. For each pose six points are calculated at a distance between 0 and 200 mm. The translation of the object is always smaller then 100. The results are illustrated in Figure 30.

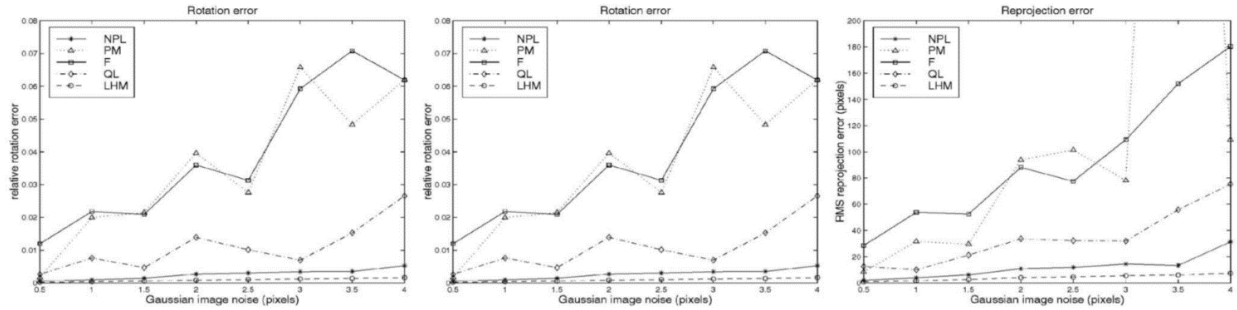

Figure 30: Rotation, translation, and reprojection errors for six points by different noise levels [36]

From these graphs we can see that this method (NPL) outperforms the other non-iterative methods. [36]

4.4.4 Results point simulation 2

In this simulation, the dependence on the number of points is tested. The number of points varies between 5 and 11 points. PM and F for 5 points are not plotted because these methods require at least 6 points. A 1,5 x 1,5 Gaussian noise is added. The results are displayed in Figure 31.

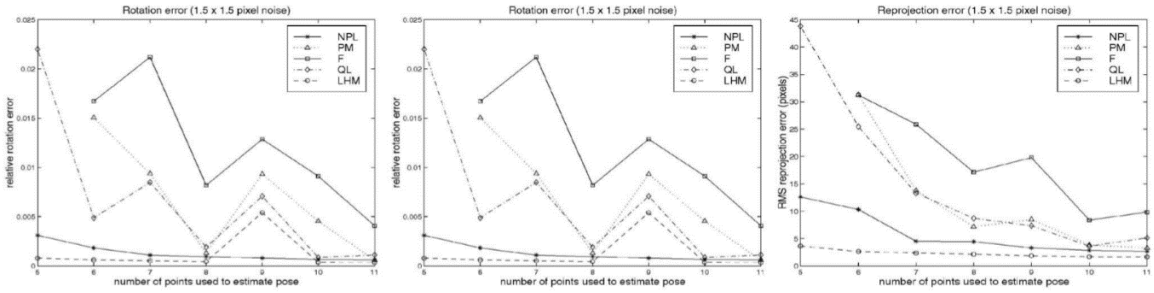

Figure 31: Rotation, translation, and reprojection errors for six points by different number of points [36]

Again NPL outperforms the other linear algorithms but the difference is the most significant with a lower number of points. [36]

4.4.5 Results point simulation 3

This simulation tests the dependence on the effective field of view. The points are generated in the same way as in the previous simulations but now there are only six points. The position of these points are limited to the vertices of a 10 x 10 x 10 cube centered around the optical axis. The same Gaussian noise is applied. This means that all the detected points are in a small portion around the center of the image. The results are shown in Figure 32. In which the relative error and the relative object size (distance/size) are plotted. [36]

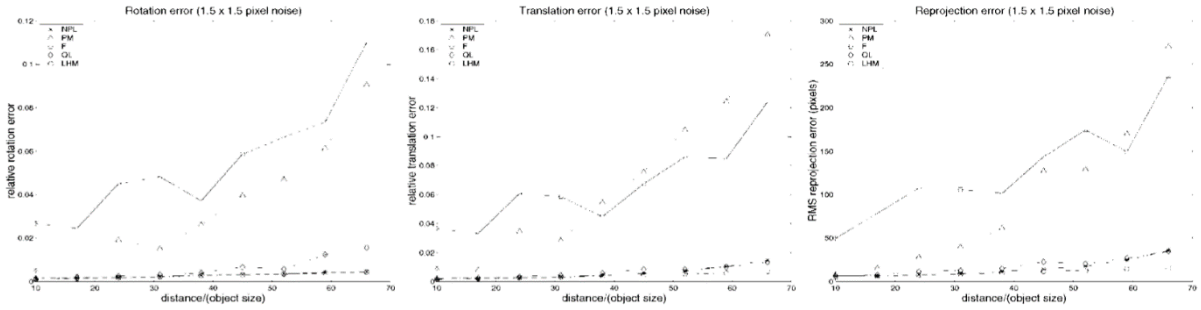

Figure 32: Rotation, translation, and reprojection errors for six points versus extend of object given as distance/size [36]

From this figure can be concluded that NPL outperforms QL, PM and F for pose estimation when the object is approximately seven time as far away as its extent. So NPL works better when the object is smaller or further away from the image. [36]

4.4.6 Results line simulation 1

In this simulation the dependence on the noise level is also tested. The pixel noise varies between σ = 0.5 and σ = 5. For each pose six line segments are generated. The world line segments are contained in a 20 x 20 x 20 box in front of the camera. This time the translation is limited to 10. The relative rotation and translation errors for NLL and KH are plotted in Figure 33.

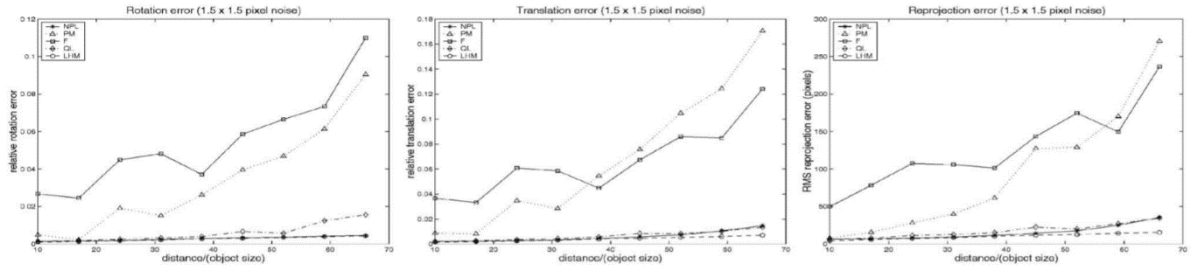

Figure 33: Rotation, translation, and reprojection errors for six lines by different noise levels [36]

The KHRT performs the best but this was expected because it is initialized with the correct position. From this simulation it is also possible to conclude that the NLL has the best mean performance. This is because even at ground truth initialization the iterative algorithm can converge on a local minima. [36]

4.4.7 Results line simulation 2

This simulation tests the dependence on the number of lines. The line simulation is the same as in the previous simulation but now the number of lines varies between 4 and 11. Also a fixed noise of 1,5 x 1,5 pixels is added. This is illustrated in

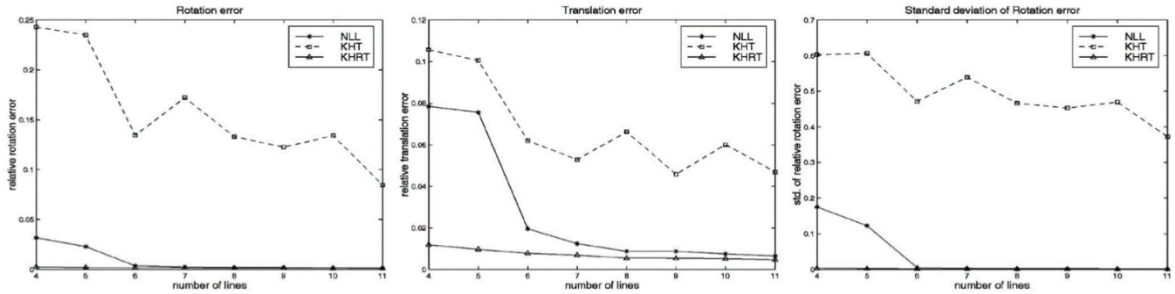

Figure 34: Rotation, translation, and reprojection errors for six lines by different number of points [36]

It is clear that for both algorithms the accuracy increases with increasing lines. The absolute performance of NLL is comparable to KH but it is more likely to converge to al local minima. [36]

4.4.8 Results timing simulation

The timing simulation makes clear that NPL is a slow method compared to the other linear algorithms but it preforms relatively better when there are less points. This is in sharp contrast to NLL which is ten to hundred times faster than its iterative counterparts. This was of course to be expected. Also this variation performs relatively better when there are less points. [36]

4.4.9 Conclusion

The NPL algorithm works relatively better when there are only a few known world points and when all these points are located near the center of the image. It is a relative slow algorithm but it has a high accuracy.
As expected The NLL is a fast algorithm for pose estimation with lines. It also has a comparable accuracy with respect to the iterative algorithm but it will not converge on a local minimum. [36]

4.5 Incident ray tracking camera model

Most applications use the pin hole camera model. In this model, all incident rays are directly projected onto the detection plane through the pinhole. However in reality the light rays go through compound lenses.
In a general imaging model, virtual sensing elements are used to make the linear mapping between the incident rays and the image plane. These elements have three parameters, an image projection, the yaw and the pitch directions of the projective ray. The calibration of these elements strongly depends on how accurate the rotation can be determined. This camera model simplification results in a less accurate position estimation.
The lens geometry model calculates, the geometric relationship between images and objects via Snell's Law and skew ray tracking. Snell's Law describes the angle of an incidence light ray with respect to the refraction of this light ray when it passes through the boundary between two different materials like glass and air. Figure 35 gives an overview of the path of an incident ray in an imaging system. This is used to make the connection between incident rays and the image pixels.

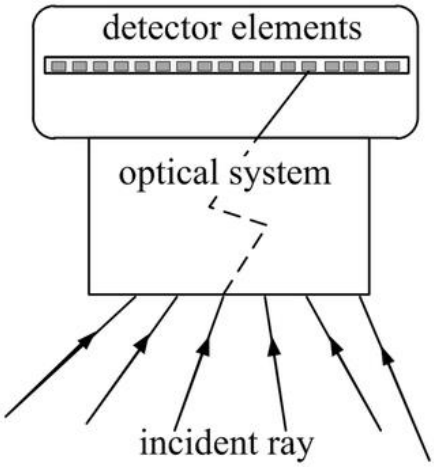

Figure 35: Representation of an incident ray passing through a camera [1]

Each pixel corresponds with an incident ray in the imaging system. However the path of an incident ray through the imaging system can be complex and is therefore replaced by the abstract mathematical equivalent which is call the perspective ray. The corresponding camera model is the incident ray tracking camera model (IRT). [1]

4.5.1 Parameters of the incident ray tracking camera model

The ray to image mapping for the camera is parameterized as depicted in Figure 36.

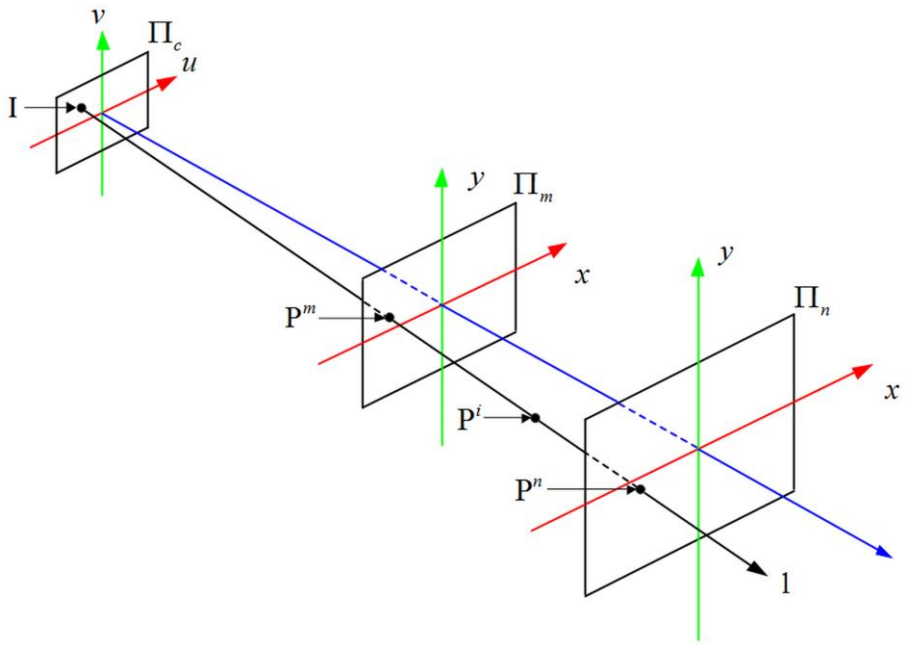

Figure 36: Geometrization of the IRT [1]

A point P^i (x, y, t) with image coordinates (x, y) and at depth t is located on the perspective ray l. The depth is the distance between P^i and π_c. The most convenient way to define the perspective rays, such as l, is to let them intersect two parallel reference planes. Here π_m and $\pi_n$are used. One point in each reference plane, namely P^m and P^n will define all perspective rays. The reference planes are defined as:

$$\begin{cases} \pi_m(x,y) = \{P^m\} \\ \pi_n(x,y) = \{P^n\} \end{cases} \tag{46}$$

The parameters used in IRT are calculated from the reference planes. The mapping from the reference frame π_n to the camera sensor plane is illustrated in Figure 37.

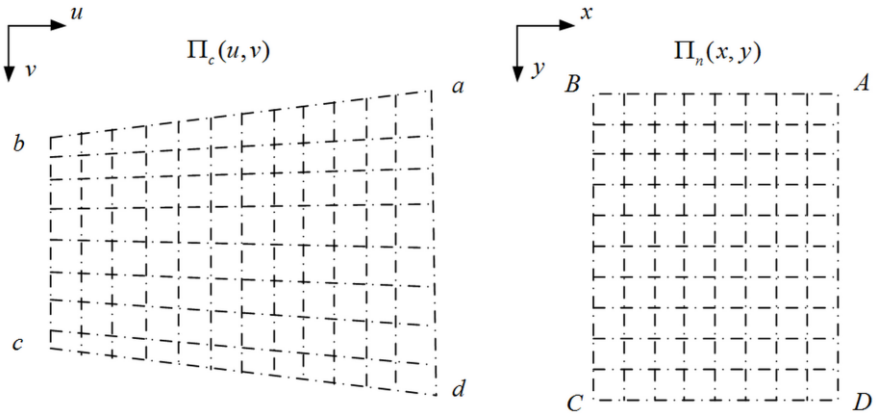

Figure 37: Mapping between reference plane and image plane. [1]

The mapping between the two planes is given by the equation:

$$\begin{cases} x = \sum_{i=0}^{n} \sum_{j=0}^{n-1} C_{ij} u^i v^i \\ y = \sum_{i=0}^{n} \sum_{j=0}^{n-1} D_{ij} u^i v^i \end{cases} \quad (47)$$

Where (C_{ij}, D_{ij}) are the mapping parameters, n is the order of the mapping, (x, y) are the space coordinates of the points in the reference plane and (u,v) are the image coordinates. The mapping parameters (C_{ij}, D_{ij}) are calculated with the Levenberg-Marquard method. This is a least-square estimation of non-linear parameters which is a combination between a Taylor series method and a gradient method. [1]

4.5.2 Incident ray camera model

Figure 38 depicts the perspective rays used for the camera pose estimation.

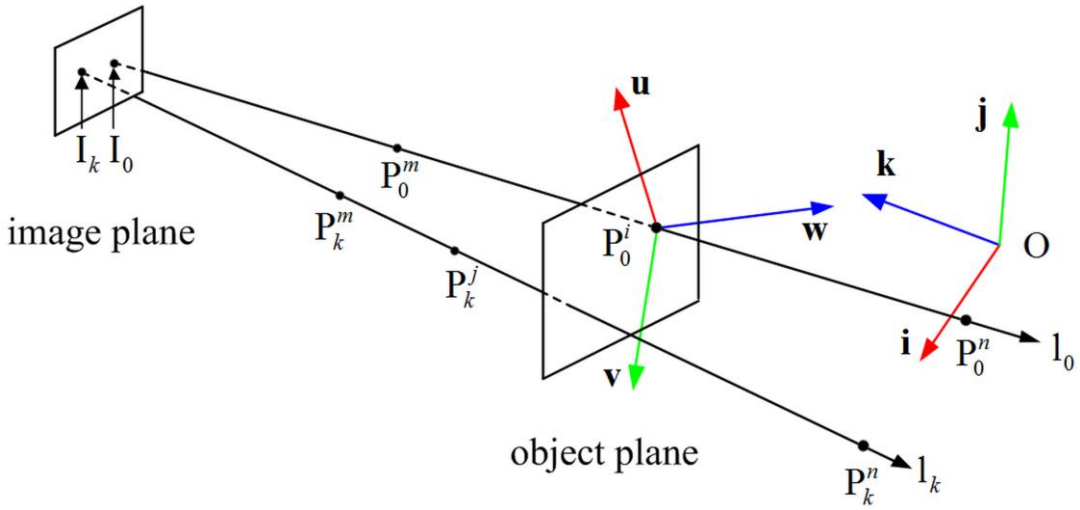

Figure 38: The perspective rays used for object pose [1]

The superscript of the point in the figure means that the points are an element of that reference frame. The subscript means that they are part of that perspective ray. $l_0(I_0,\ p_0^m,\ p_0^n)$ is the perspective ray through the origin of the object coordinate system p_0^i-u v w. The perspective ray $l_k(I_k,\ p_k^m,\ p_k^n)$ goes through p_k^j which is located on the object with known coordinates in the object coordinate system. The origin of the reference plane coordinate system is O -i j k. Object coordinates like $\overrightarrow{p_0^i p_k^j}$ can be transformed by the rotation matrix in to coordinates of a reference frame like $\overrightarrow{p_0^n p_k^n}$. The Rotation matrix R is defined as:

$$R = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \quad (48)$$

Where $i_u,\ i_v,\ i_w$ are the coordinates of unit vector i in the object coordinate system. The same applies for the other unit vectors of the object coordinate system. The vector k is the cross-product of i and j. So only the transformations of i and j have to be estimated to be able to estimate the rotation.
The translation vector is determined by the vector $\overrightarrow{Op_0^i}$ This is the vector between the origins of the coordinate systems. p_0^i is an element of the perspective ray l_0 which is defined as:

$$\begin{cases} f_x^0(z) = g_x^n(u_0, v_0) + (g_x^m(u_0, v_0) - g_x^n(u_0, v_0))(z - z^n)/(z^m - z^n) \\ f_y^0(z) = g_y^n(u_0, v_0) + (g_y^m(u_0, v_0) - g_y^n(u_0, v_0))(z - z^n)/(z^m - z^n) \end{cases} \quad (49)$$

z^m and z^n are z coordinates of the reference planes π_m and π_n respectively. This means that p_0^i can be defined as:

$$\overrightarrow{Op_0^i} = (f_x^0(z^i) - g_x^n(0,0), f_y^0(z^i) - g_y^m(0,0), z^i - z^n) \quad (50)$$

Here z^i is the z coordinate of the plane π_i. This plane is parallel to both reference planes and goes through the origin of the object coordinate system p_0^i. When analysing the formula above it can be conclude that only the z^i coordinate needs to be determined to know the object translation. However, there are three parameters that need to be computed to know the object pose namely, i, j and z. [1]

4.5.3 Projection on the perspective ray

Figure 39 shows an imaging model of the perspective rays with a scaled orthographic projection. Only two image points correspond with two feature points, p_0^i and p_k^i, are in the π_i plane.

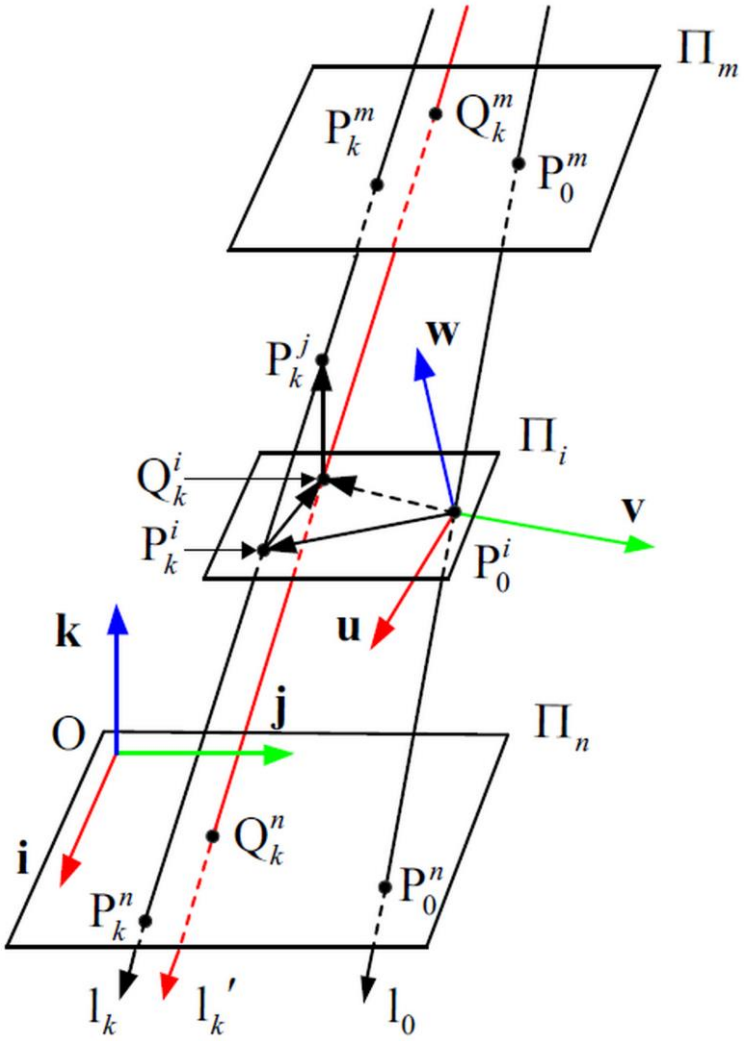

Figure 39: The imaging model of the perspective rays [1]

The perspective rays l_0 and l_k, are the rays of the points p_0^i and p_k^i respectively. They are determined with the calibration parameters. p_0^i is the origin of the object coordinate system, therefore the coordinates of p_0^i in the object coordinate system are known. π_m and π_n are the reference planes, the plane π_i is parallel to both reference planes and goes through p_0^i. The point p_k^i is projected on the plane π_i at Q_i according to the k axis. $l_k{}'$ is the perspective ray trough Q_i. [1]

4.5.4 Formulation of projections

The next equation defines the relationship between three unknowns, the unknown vectors i and j of the rotation matrix and the unknown z^i coordinate, and the known vector the $\overrightarrow{p_0^l p_k^j}$ in the object coordinate system.Figure 39 shows that vector $\overrightarrow{p_0^l p_k^j}$ is the sum of three vectors.

$$\overrightarrow{p_0^l p_k^j} = \overrightarrow{p_0^l p_k^l} + \overrightarrow{p_k^l Q_k^l} + \overrightarrow{Q_k^l p_k^j} \tag{51}$$

The perspective rays l_0 and l_k constrain the vector $\overrightarrow{p_0^l p_k^l}$. If (f_x^0, f_y^0) and (f_x^k, f_y^k) are the functions of l_0 and l_k, the coordinates of this vector can also be defined as:

$$\overrightarrow{p_0^l p_k^l} = (f_x^k(z^i) - f_x^0(z^i), f_y^k(z^i) - f_y^0(z^i), 0) \tag{52}$$

The perspective rays l_0 and $l_k{}'$ constrain the vector $\overrightarrow{p_k^i Q_k^i}$which can be defined as:

$$\overrightarrow{p_k^i Q_k^i} = (f_x^k(z^{i\prime}) - f_x^k(z^i), f_y^k(z^{i\prime}) - f_y^k(z^i), 0) \tag{53}$$

Hereby is $z^{i\prime}$ the z coordinate of p_k^j. $z^{i\prime} = (1 + \varepsilon^i)\ met\ \varepsilon^i = \overrightarrow{p_0^i p_k^j} \bullet \frac{k}{z^i}$.
The vector $\overrightarrow{Q_k^i p_k^j}$ is perpendicular to plane π_i. The vector can be expressed as:

$$\overrightarrow{Q_k^i p_k^j} = (0,0, z^i * \varepsilon^i) \tag{54}$$

The sum of the three vectors can then be expressed as:

$$\overrightarrow{p_0^i p_k^j} = (f_x^k(z^{i\prime}) - f_x^0(z^i), f_y^k(z^{i\prime}) - f_y^0(z^i), z^i * \varepsilon^i) \tag{55}$$

The dot products of $\overrightarrow{p_0^i p_k^i}$ with the unit vectors i and j are defined as:

$$\begin{cases} \overrightarrow{p_0^i p_k^j} \bullet i = f_x^k(z^{i\prime}) - f_x^0(z^i) \\ \overrightarrow{p_0^i p_k^j} \bullet j = f_y^k(z^{i\prime}) - f_y^0(z^i) \end{cases} \tag{56}$$

Solving this equation would make it possible to calculate the object pose.

4.5.5 Iteration for scaled orthographic projection

The dot products of $\overrightarrow{p_0^i p_k^j}$ can also be expressed as:

$$\begin{cases} \overrightarrow{p_0^i p_k^j} \bullet i = f_x^k\left(z^i(1 + \varepsilon^i)\right) - f_x^0(z^i) \\ \overrightarrow{p_0^i p_k^j} \bullet j = f_y^k\left(z^i(1 + \varepsilon^i)\right) - f_y^0(z^i) \end{cases} \tag{57}$$

Because the points p_0^i , p_0^n, p_k^j and p_k^n are located on the perspective rays l_0 and l_k, this equation can be written as:

$$\begin{cases} \overrightarrow{p_0^i p_k^j} \bullet I = f_x^k(z^n) - f_x^0(z^n) \\ \overrightarrow{p_0^i p_k^j} \bullet J = f_y^k(z^n) - f_y^0(z^n) \end{cases} \tag{58}$$

Hereby is $I = s_i * i,\ J = s_j * j$. I and J are the only unknowns in this linear system. The norm of I and J are respectively the scaling factor s_i and s_j between the vectors $\overrightarrow{p_0^i p_k^j}$and $\overrightarrow{p_0^n p_k^n}$. The length of the two vectors can be written as:

$$\left|\overrightarrow{p_0^n p_k^n}\right| = \frac{s_i + s_j}{2}\left|\overrightarrow{p_0^i p_k^j}\right| \tag{59}$$

This can be rewritten as:

$$\left|\overrightarrow{p_0^n p_k^n}\right| = \frac{s_i + s_j}{2}\sqrt{(f_x^k(z^{i\prime}) - f_x^0(z^i))^2 + (f_y^k(z^{i\prime}) - f_y^0(z^i))^2 + (z^i * \varepsilon^i)^2} \tag{60}$$

If the values of ε^i and z^i are given then this is the algorithm for estimating the pose. It is called the perspective-ray-based scaled orthographic projection (PRSO). The solution of the algorithm is only an estimation because the term ε^i can not exactly be determined. However, the calculated values of i and j can be used to obtain a more accurate value of ε^i. The iterative version of this algorithm is called PRSOI (PRSO with Iterations) and it converges towards the correct pose.
Initially the term ε^i is set to zero, this means that it is assumed that p_k^i and Q_k^i coincide. When tracking a moving object is tracked the value of ε^i from the previous frame is used. [1]

4.5.6 Solving the system of the PRSO algorithm

From all the equation above there is still an equation which is difficult to solve namely:

$$\begin{cases} \overrightarrow{p_0^i p_k^j} \bullet I = f_x^k(z^n) - f_x^0(z^n) \\ \overrightarrow{p_0^i p_k^j} \bullet J = f_y^k(z^n) - f_y^0(z^n) \end{cases} \tag{61}$$

If we define ξ^i as $f_x^k(z^n) - f_x^0(z^n)$ and η^i as $f_y^k(z^n) - f_y^0(z^n)$. The dot products use vector coordinates in the object frame. This can be rewritten as

$$\begin{cases} [u_i \quad v_i \quad w_i][i_u \quad i_v \quad i_w]^T = \xi^i \\ [u_i \quad v_i \quad w_i][j_u \quad j_v \quad j_w]^T = \eta^i \end{cases} \tag{62}$$

The coordinates of I and J are the only unknowns in these linear equations. f_x^0, $f_y^0$and f_x^k, f_y^k are the functions of the perspective rays l_0 and l_k, and u_i, v_i, w_i are the known coordinates of p_k^j in the object coordinates. Substitute the n feature points for the previous equation.

$$\begin{cases} A \bullet I = x' \\ A \bullet J = y' \end{cases} \tag{63}$$

In this equation A is the object points coordinates matrix in the object coordinate frame, $x' = \left[\xi_0^i \dots \xi_j^i \dots \xi_n^i\right]$, $y' = \left[\eta_0^i \dots \eta_j^i \dots \eta_n^i\right]$. If there are at least four non-coplanar points, the least square solution is given by:

$$\begin{cases} I = \mathrm{B} * x' \\ J = \mathrm{B} * y' \end{cases} \tag{64}$$

Here matrix B is the pseudo inverse of object matrix A. Once I and J are determined i and j can be calculated by normalizing I and J.
The translation vector T of the object can now be computed. It is equal to vector $\overrightarrow{Op_0^i}$. This vector can be calculated with the equations:

$$\overrightarrow{Op_0^i} = (f_x^0(z^i) - g_x^n(0,0), f_y^0(z^i) - g_y^m(0,0), z^i - z^n) \quad (65)$$

And

$$\left|\overrightarrow{p_0^n p_k^n}\right| = \frac{s_i + s_j}{2}\sqrt{(f_x^k(z^{i\prime}) - f_x^0(z^i))^2 + (f_y^k(z^{i\prime}) - f_y^0(z^i))^2 + (z^i * \varepsilon^i)^2} \quad (66)$$

Which were previously mentioned in this chapter. [1]

4.5.7 Camera calibration results

In this experiment a 768 x 576 CCD camera is used, with a pixel size of 0,0083 mm x 0,0086 mm and a View angle of 60°. A Zolix KSA300-11-X actuates the linear movement of the camera based on a solid evenly distributed 7 x 9 circular pattern.
The size of the pattern is 500 x 600 mm² with a distance of 60 mm between adjacent points.
In this experiment there are six images taken each 30 mm apart. The first and last image, at distance 0 mm and 150 mm, are used for the calibration. These are shown in Figure 40 and Figure 41. If the calibration is complete, the space error of the calibration points is calculated using IRT. The results are shown in Figure 42.

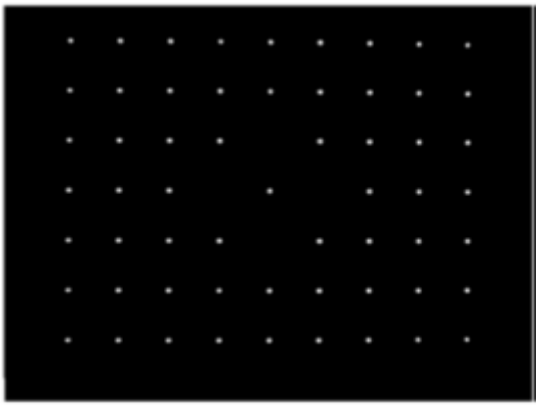

Figure 40: Checkerboard pattern at distance of 0mm [1]

Figure 41: Checkerboard pattern at distance of 150mm [1]

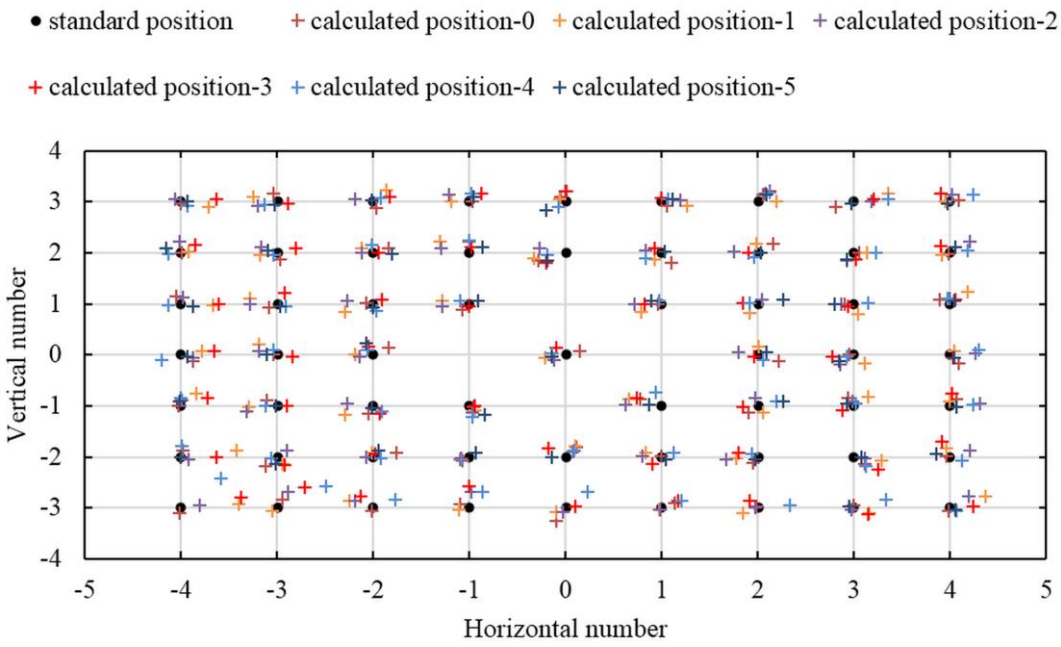

Figure 42: Position distribution of the calibration points [1]

The root mean square error (RMSE) (explained in chapter 2.8) represents of how far the reprojected point differs from the measured points. This measurement is used to determine the accuracy of the calibration parameters. In this experiment the average RMSE is 0,17 mm in horizontal direction and 0,12 mm in vertical direction. From all these points the four points with the lowest RMSE are selected and used for the pose estimation. This is a P4P method because it only uses four carefully selected points to determine the pose. [1]

4.5.8 Pose estimation results

The experimental setup for pose estimation uses a Zolix RAK-200 for the yaw rotation and Zolix RAD-100 for the pitch and roll directions. The two-phase stepping motor is controlled using the Zolix MC600-4B in closed-loop control.
During this experiment a picture is taken every 1°. The target rotation angel is measured by using the image of the initial position and the image of the current position. The same method is used for measuring the translation every 2 mm.
To test the accuracy of the POSIT algorithm the position is also calculated with different PnP algorithms using a pinhole model. These other algorithms include the geometric configuration solution by Liu ML and Wong KH denoted by LW and POSIT. POSIT is also explained in chapter 5.2. The LW can also incorporate the incident ray tracking camera model. This is denoted as LW+IRT. Table 1 gives an overview of the average RSME for the six degrees of freedom for the different algorithms.

Table 1: The RMSE for each degree of freedom of the different PnP algorithms

method	Rotation yaw (deg.)	Rotation pitch (deg.)	Rotation roll (deg.)	Translation in x direction (mm)	Translation in y direction (mm)	Translation in z direction (mm)
POSIT	0.290	0.243	0.071	0.369	0.241	0.552
LW	0.258	0.258	0.110	0.340	0.248	0.448
LW+IRT	0.201	0.176	0.083	0.146	0.130	0.362
PRSOI	0.136	0.115	0.062	0.152	0.128	0.272

Table 1 illustrates that the accuracy of LW+IRT is higher than LW for each degree of freedom. This indicates that the IRT model has improved the accuracy. From the table can also be derived that the accuracy of PRSOI is higher than POSIT for all degrees of freedom except for the roll rotation. For the roll rotation the accuracy is almost the same. PRSOI also has a higher accuracy than LW+IRT for the rotations and the z translation. The accuracy for the x and y translation are almost the same.
To conclude, PRSOI has a higher accuracy using the perspective ray model but it uses more calculating power. However, it is a recent study (2015) so there is not any other algorithm who uses this camera model yet. [1]

4.6 Moiré patterns

4.6.1 Introduction

Another way to determine the 6-DOF pose of an object is to utilize interference patterns. More specifically the moiré effect which occurs when two different sets of opaque lines or dots with transparent gaps between the lines are superimposed. A pattern of dark and light lines with a lower spatial frequency than the original set of lines can be observed. This is called a moiré pattern. The pattern appears when the two sets are not identical. The cause of the moiré fringes can be due to two distinct mechanisms. The first one is because of a slightly different frequency in the lines of both sets. The second cause is the inclination between the sets. An example of a moiré pattern is shown in Figure 43. Here there are two sets of lines with equal distance P between the lines. However one set is rotated by an angle α. An advantage of a moiré pattern is that a slight deviation in translation or rotation between the two sets results in a significant change in the pattern [39].

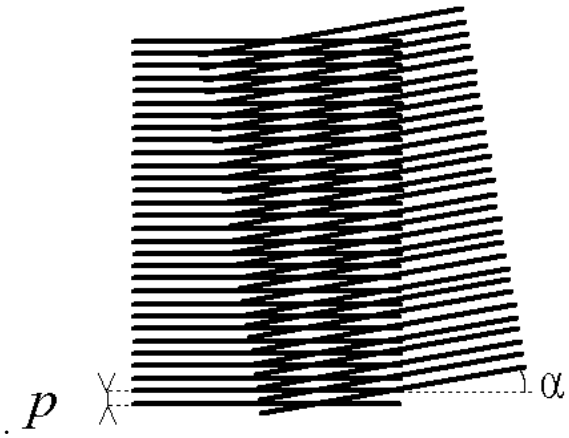

Figure 43: Example of a Moiré pattern [40]

Consider two sets of parallel lines. The first set consists of lines separated by distance p. The second set is separated by a distance $p + \delta p$. The middle of the first dark line that appears in the moiré fringe appears at the overlap of the nth line of both sets. The relationship between n and the distance p is shown in the next formula.

$$n = \frac{p}{2\delta p}. \tag{67}$$

This is only the case if there is no rotation between both sets. The symbol ϕ is the angle between the visible dark moiré fringe and the direction of the primary set of lines. θ is the angle between the first set of line and the secondary set. Using the next formula one can calculate ϕ from θ [41].

$$\phi = \frac{\pi}{2} + \frac{\theta}{2} \Leftrightarrow \theta = 2\phi - \pi. \tag{68}$$

4.6.2 Working of RGR-6D

The paper utilizing moiré patterns to determine an accurate 6-DOF estimation uses Retro-Grate Reflectors (RGR). These RGRs reflect most of the incoming light back to the source with a minimum of scattering. The RGRs are fitted with a small gratings. These gratings act as the black and transparent lines of the moiré pattern. The test pattern used for pose estimation, also called the “target” consists of 4 RGRs in the horizontal and 4 in the vertical direction as shown in Figure 44.

Each direction has 3 high resolution and one low-resolution grating.

The corners of the target consists of StarBurst marks which are only used for the detection of the target in noisy environments. In order to create a moiré effect, there need to be two slightly different gratings overlapping each other. The horizontal and vertical gratings in the target do not overlap (the vertical gratings are split in two parts). The moiré effect is achieved by using an identical pattern at the back of target. The front gratings have 2480 cycles per meter whereas the back gratings have 2500. Both layers are separated by a transparent glass substrate of 5.6 mm. This technique focuses on the accurate estimation of out-of-plane rotations and depth. [42]

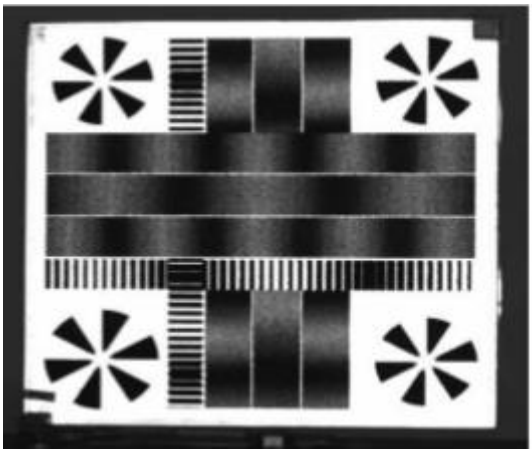

Figure 44: RGR target [42]

As mentioned in section 4.2, there are three coordinate systems needed to estimate the pose of an object from an image: the image (2D), camera (3D) and real world coordinate system. (3D).

4.6.3 Out-of-plane rotation estimation

For accurately estimating a small out-of-plane yaw rotation this technique looks at the displacement of the moiré pattern in the horizontal gratings. A 1 degree rotation in the positive yaw direction results in a right shift of the moiré pattern in the outer region gratings, while the center grating will shift to the left. The difference in phase shifting direction is due to the grating difference between the center and outer region. The outer region gratings are the first and third RGR starting from the top. The center region is the second RGR starting from the top. All back RGRs have grating of 2500 cycles/m. The outer region front RGRs have a 2480 cycles/m grating whereas the center front RGR has a 2520 grating. Both the outer and inner grating have 20 cycle/m difference with the back grating. But because the center grating has a larger spatial frequency than the back grating, the shifting is in the opposite direction. This is called the near-field effect which will be explained later in this section. Figure 45 shows how a positive + 1 degree yaw rotation influences the shifting in the outer and central region of the RGRs. A more precise measuring method of this shifting is shown in Figure 46. The luminosity of both regions are measured and displayed in a 8 bit gray-scale. The graph is measured using a Kodak KAI-0372M Imager by using a microlens.

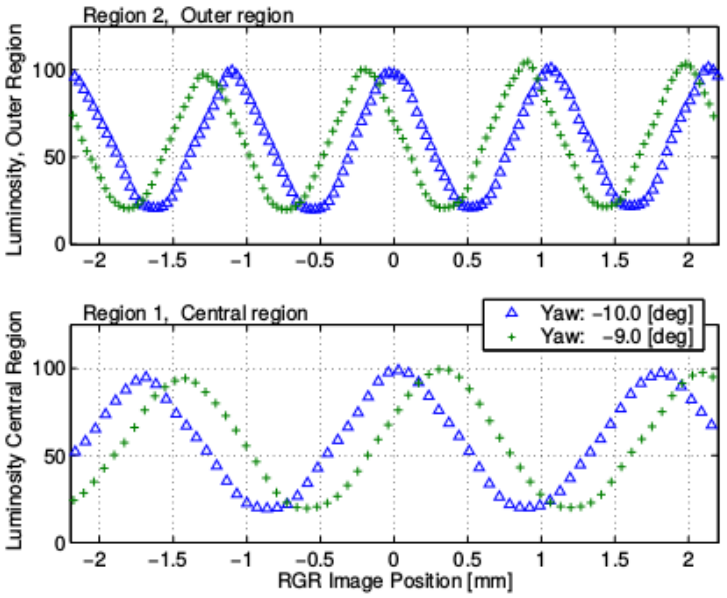

Figure 45: RGR-6D, luminosity shift due to + 1 degree yaw rotation [42]

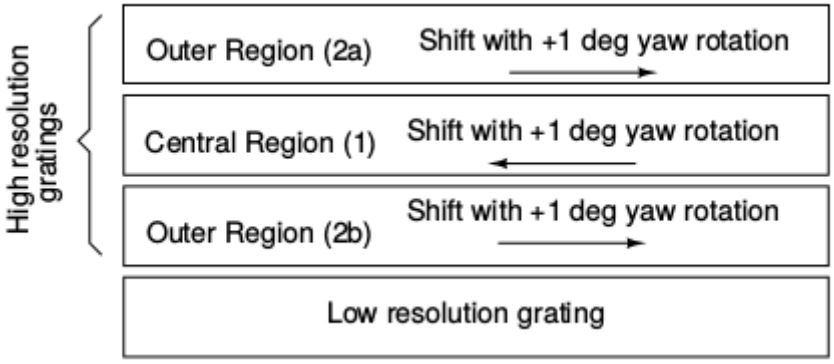

Figure 46: RGR-6D, Moiré pattern shift due to +1 degree yaw rotation [42].

The green curve has a +1 degree yaw rotation compared to the blue curve. This results in a 60 degree phase difference between the green and blue curve. The sensitivity of this system is 60 degrees shift to a 1 degree rotation. The pitch rotation is measured in a similar way by using the vertical RGRs.

As mentioned before the phase shift in the center region is opposite compared to the outer region. The back RGR of the center region has a larger spatial frequency in its gratings compared to the front RGR while the outer region back RGR has a smaller spatial frequency compared to the front RGR. This difference in phase shifting direction is used to perform a differential phase measurement between both regions in order to increase the accuracy. This method claims to achieve a yaw rotation estimation accuracy of +- 1 arc minute (1/60th of a degree) [42].

4.6.4 Depth estimation

Figure 44 shows clearly a lower spatial frequency of the central region compared to the outer region.

This effect can be used to measure the distance between the target frame and the origin of the camera frame (the optical center of the camera) also called the depth. Other techniques to estimate the depth of a target in a camera frame uses the true size of the target and the principal distance or lens' focal length. However, a variable focus or zoom make it harder to know the principal distance beforehand. Estimation of the depth using this method is divided in two parts: depth estimation in near field and depth estimation in far field. A camera is in near field when $\|{}^{c}P_{t}^{\cdot}\| < 1000 * d_t$ where d_t is the target thickness and ${}^{c}P_{t}^{\cdot}$ indicates the target coordinate system origin, viewed in the camera frame. The distance between the origin of the target frame a and the camera frame origin is represented as $\|{}^{c}P_{t}^{\cdot}\|$. If the camera is in the near field, the depth can be immediately estimated using the distribution of the moiré patterns, because a variation in depth results in a variation of the observed moiré fringes.

If the distance from the principal point to the origin of the target is much larger than the thickness of the target itself ($\|{}^{c}P_{t}^{\cdot}\| \gg d_t$) the camera is said to be in far field.

In the far field there is a linear relation between the spatial frequencies of the front and back RGR gratings and the observable moiré pattern frequency shown in the next formula

$$rF_m^{\infty} = F_b - F_f \tag{69}$$

In this formula rF_m^{∞} represents the spatial frequency of the observed moiré pattern seen in the far field. In the near field this relation is different as the apparent frequency of the back RGR grating increases. The origin of this increase in observable spatial frequency is the way that the rays of light are collected by the camera chip. Figure 47 gives a clear overview. The dotted lines are rays of incident light. It's obvious that the light coming from behind the target shows a bit more of the back RGR grating then the front RGR. This increases the apparent spatial frequency of the back RGR grating. In far field the two dotted lines can be seen as nearly parallel and this problem does not occur. [42]

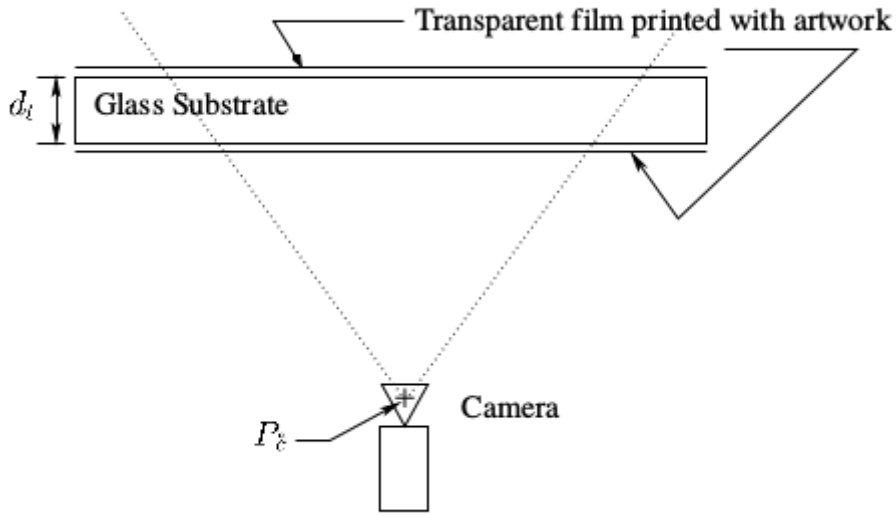

Figure 47: RGR-6D test setup [42]

This increase in spatial frequency will also alter the apparent spatial frequency of the observable moiré pattern. This apparent frequency changes with the point of observation. Using the images the camera produces to measure the moiré pattern can only result in the magnitude of the moiré pattern. Figure 48 gives an overview of the depth influences on the apparent moiré frequency in near field.

The x-axis indicates the distance (m) from the optical center of the camera to the target frame in both graphs. In the upper graph, the y-axis signifies the observable moiré fringe frequency (cycle/m). The blue and green line represent the outer region and central region respectively. The lower graph shows the ratio of the apparent moiré fringe frequency of the outer region to the center region. Please note that $|rF_m|$ represents the apparent moiré frequency in the real world coordinate system. $|iF_m|$, however represents the apparent moiré frequency in the image coordinate system. The relationship between $|iF_m|$ and $|rF_m|$ will be discussed in the next section.

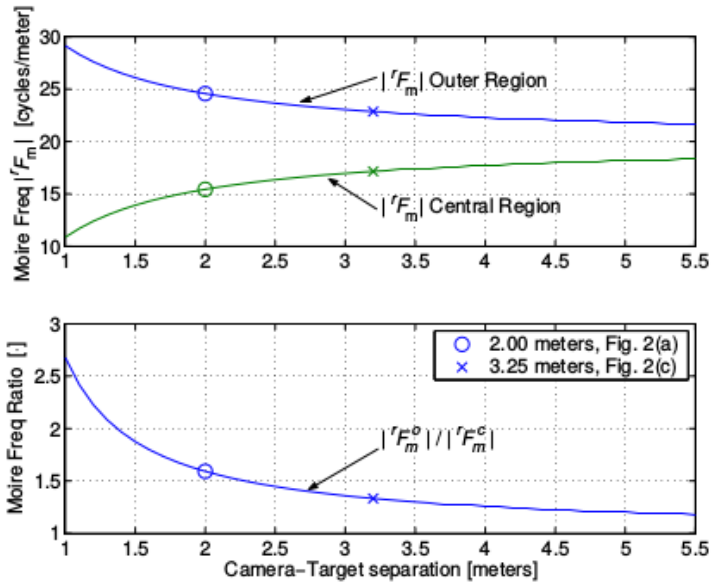

Figure 48: a) Moiré frequency as a function of the camera-target distance;
b) Ratio of real world image coordinate system moiré frequency in function of the distance [37].

The transition from $|iF_m|$to $|rF_m|$, which requires camera calibration parameters and the unknown pose of the target, can be quite complex. Luckily the outer and center region RGR are on a common plane. This means that the ratio of the outer and center region in both the image coordinate system and real world coordinate system should be the same. This ratio is shown in the next formula.

$$\frac{|rF_m^O|}{|rF_m^C|} = \frac{|iF_m^O|}{|iF_m^C|} \tag{70}$$

O and C identify the outer and center region respectively. This ratio is also shown in the y-axis of the lower graph of Figure 48. The result of the depth estimation is not as accurate as the yaw and pitch estimation. The authors claim to achieve an accuracy of 1-2% on a distance from 2 to 3.25 meter. However this technique does not require a calibrated or a high resolution camera. A yaw rotation accuracy of 4 arc minutes (1/15 th of a degree) could be achieved by using only a target image of 60 x 60 pixels [42].

4.6.5 Conclusion

This technique seems very promising as it achieves one of the highest accuracy in out-of-plane rotations of all the techniques in this thesis. An accuracy up to 1 arc minute can be achieved for out-of-plane rotations. Measuring depth can be accurate up to 1 % - 2 % on a distance from 2 to 3.25 m. The lack in need of a high resolution camera or camera calibration is also very appealing. However, in-plane rotation and translations need other (standard) photogrammetry functions. The depth estimation accuracy is accurate enough for most applications but if the need for high-accuracy depth estimation emerges, another method should be used. This technique is ideal if high-accuracy out-of-plane rotations need to be estimated. Combined with other techniques which provide high depth and in-plane accuracy this technique might be ideal.

5 Iterative pose determination methods

5.1 The orthogonal iteration algorithm

The orthogonal iteration (OI) algorithm is also often referred to as method of Lu's et al. It is a globally convergent iterative algorithm from 1998. Which is optimized for pose determination unlike the algorithms before it. It is one of the first algorithms for which there is a mathematical proof that it is globally convergent and which effectively account for the orthonormal structure of the rotation matrices. They minimize the error matric based on collinearity in object space as opposed to the image space which most other existing algorithms before did. This means that it is a fast algorithm which is as accurate as existing methods and which has a high robustness against outliers.
The OI algorithm is inspired by the work of Haralick et al. He developed a method who calculates simultaneously the object pose and the depth of the feature points. The main advantage of this algorithm is that the non-linearity due to perspective projection is eliminated through the inclusion of the depth variables. The main disadvantage of this method is that it has a slow convergence rate.
In the OI algorithm they use the minimizing of the object-space collinearity error to estimate the pose. It first estimates the rotation and then it calculates the translation. The orthogonality constraint is enforced by using singular value decomposition (SVD). [43]

5.1.1 Principals orthogonal iteration algorithm

In this algorithm the object position is determent in the camera coordinate system. The symbol for the feature points in the object coordinate system is P_i. The symbol for the corresponding camera coordinates is Q_i. The coordinates of P_i are projected to the plane with a depth of 1 called the normalized image plane. V_i is the image point on the normalized image plane which is the projection of P_i. According to the pinhole model (see chapter 2.1). V_i, P_i and the origin of the camera coordinate system should be collinear. This results in the equation:

$$V_i = \frac{1}{r_z P_i + t_z}(RP_i + t) \tag{71}$$

Hereby is R the rotation matrix, is r_z the rotation round the z-as, is t the translation matrix and t_z the translation of the z-as. Another approach to the collinearity is that the orthogonal projection of Q_i on V_i should be Q_i itself. This is expressed in the following equation:

$$Q_i = RP_i + t = F_i(RP_i + t) \tag{72}$$

Were

$$F_i = \frac{V_i V_i^t}{V_i^t V_i} \tag{73}$$

The difference between the image space error and the object space error is visualized in Figure 49.

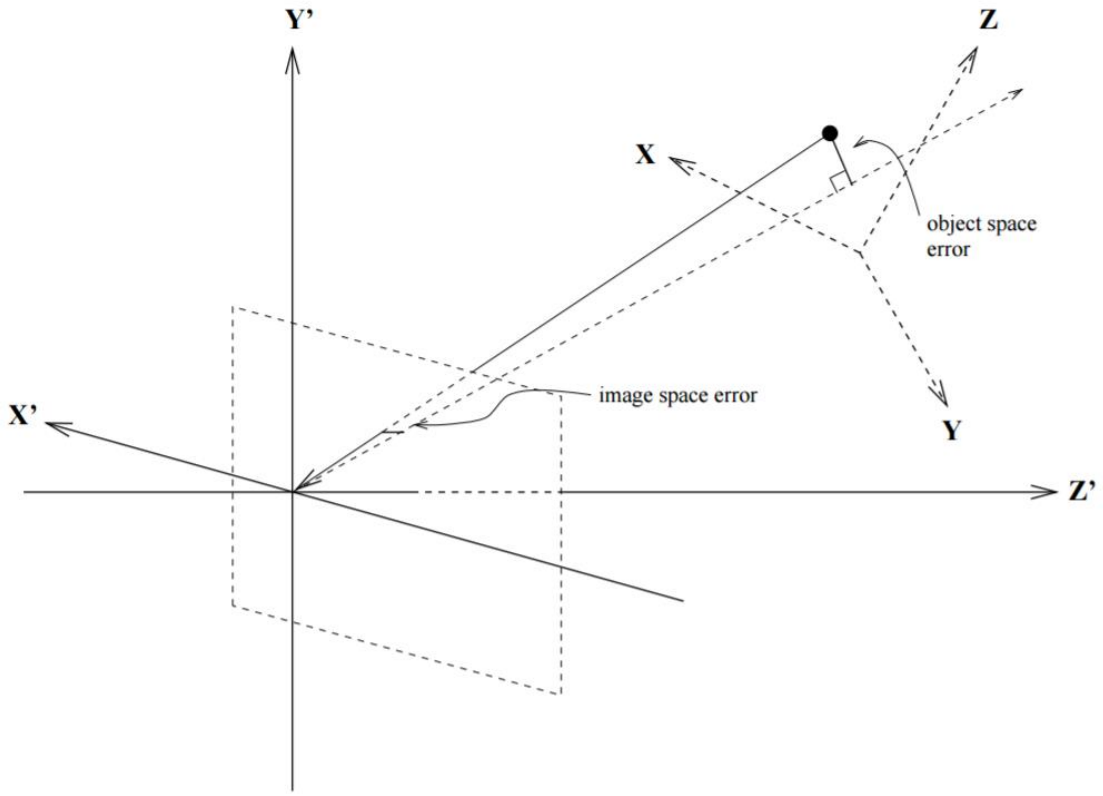

Figure 49: Object-space and image-space collinearity errors. [43]

To determine the pose the algorithm uses the object space error function:

$$e_i = (I - F_i)(RP_i + t) \tag{74}$$

This equation is then rewritten so it admits an iteration based on the solution to the 3D-3D pose estimation or absolute orientation problem. This is solved using singular value decomposition(SVD).

The mathematical proof of the algorithms global convergence is described in the original paper. Since it is globally convergent only the speed is effected by the initialisation. The original paper preforms absolute orientation between the set of reference points and the set of image points considered as coplanar 3D points. This is equivalent to using the weak perspective camera model. Often the other non-iterative methods discussed in this paper will provide a better initialisation.
The problem that this idea has is that it will give more weight to the reference points that are farther away. This is because the object-space collinearity error is greater farther away for the same pixel error in the image. This can cause problems for the pose estimation when the object is very close to the camera or when the depth of the object is similar as the distance between the object and the camera. This bias can be reduced by slightly modifying the equation. [43]

5.1.2 Test setup

To test the accuracy they generated the feature points. They had 3 control parameters the number of points N, the signal to noise ratio (SNR) and the percentage of outliers (PO).
The reference points were uniformly generated in a 10 x 10 x 10 box where the centre of the box was the centre of the object space. Also the rotation and the translation were uniformly generated. The x and y were selected in the interval [5, 15] and the z component in the interval [20, 50]. After this a fraction (according to PO) of the reference points was replaced by another point in the same box. The reference points are then projected onto the image plane. Gausian noise with a variance σ. σ is determend according to the equation:

$$SNR = -20 \log\left(\frac{\sigma}{0.3}\right) dB \tag{75}$$

There were three tests conducted on the simulation data:
C1: N = 20, PO = 0, the SNR varies between 30 db till 70 dB with 10 dB steps. This test the resistance against noise.
C2: N = 20, SNR = 60 dB, PO varies between 5 % till 25 % with 5 % steps. This is used to investigate how resistant the algorithm is against outliers.
C3: PO = 0, SNR = 50 dB, N varies between 10 till 50 with steps of 10. The purpose of this test is to see how the accuracy changes with more reference points.

The accuracy is determent by the mean errors of translation and rotation of 1000 trials for each case. the OI algorithm is compared to a linear method using full perspective camera model and a method using Levenberg-Marquardt (LM) minimalization. In this test LM uses the same initialisation as the OI algorithm. With poor initial guesses, LM has a slow convergence rate like a steepest descent method. This is the main reason why LM is slower than OI when there is more SNR or PO. When there is a good initialisation both algorithms have about the same speed. [43]

5.1.3 Results

Figure 50 shows the average runtime for each number of iterations for the different methods. The simulation has run 1000 times with SNR = 60 dB and PO = 0.

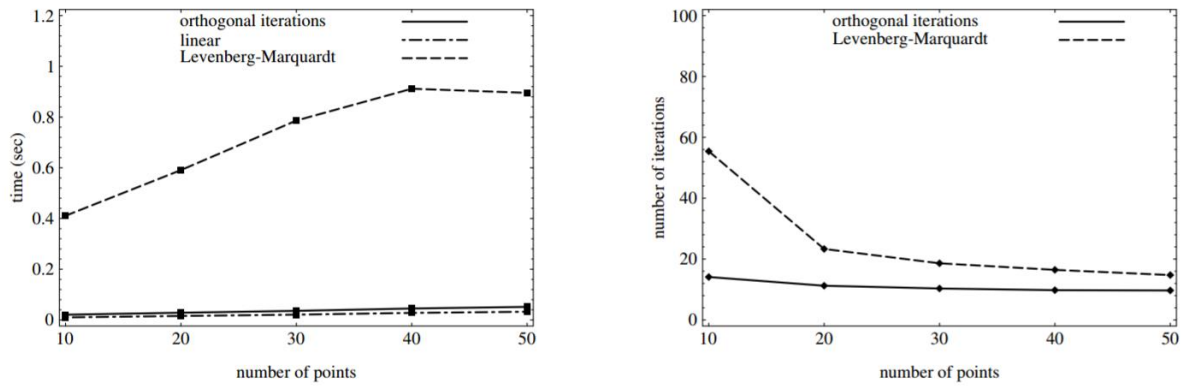

Figure 50: Running times and average number of iterations [43]

From this it is possible to determine that the OI algorithm needs les iterations to converge than LM. The results of test C1 are shown in Figure 51.

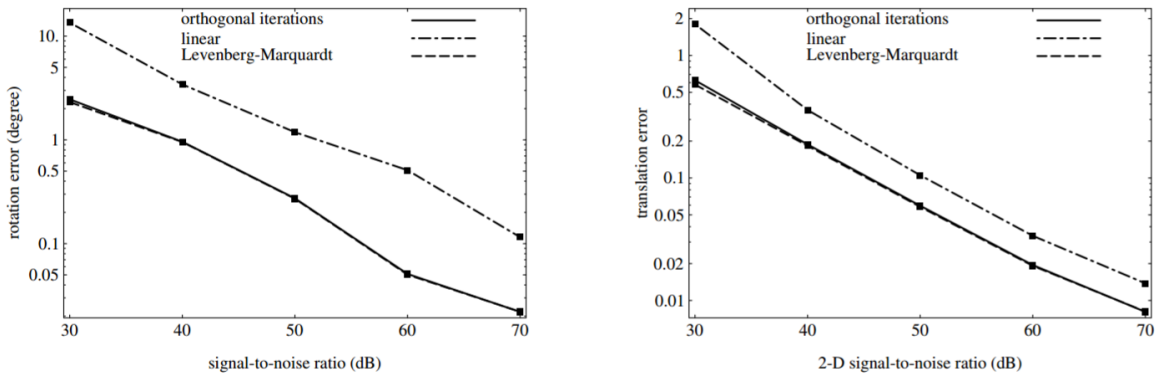

Figure 51: Result of simulation C1 error is in log scale. [43]

From these graphs it is possible to determine that the resistance against noise of LM is as good as OI. The results of test C2 are shown in Figure 52.

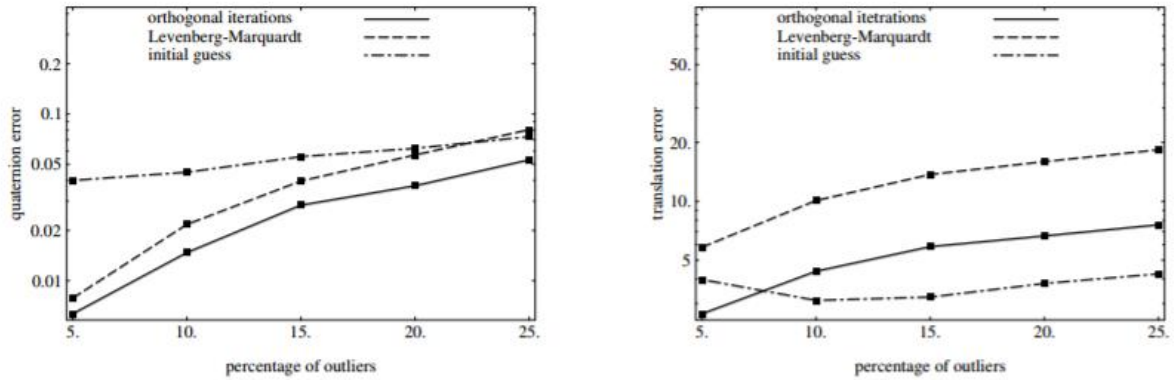

Figure 52: Results of simulation C2, error is in log scale. [43]

These graphs illustrate that OI is better resistant against outliers then LM. The results of the rotation error are expressed in quaternions. The advantage of quaternions is that they are more compact, numerically stable and may be more efficient than rotation matrices. For the rotation we can see that if there are few outliers OI is a lot more accurate as LM but when there are more outliers the difference is less significant but still present. By the translation error we see the same when there are few outliers OI preforms significantly better than LM. When there are more outliers OI is still better but the difference is less. Most surprising is that the initial guess has a higher accuracy for the translation. In Figure 53 are the results shown of test C3. Note that the rotation in the graph is this time in degrees.

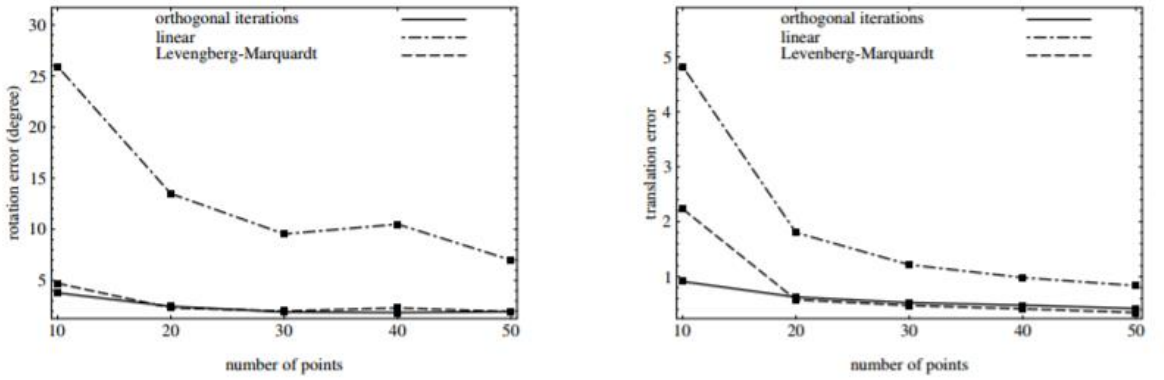

Figure 53: Results of simulation C3, error is in log scale. [43]

From these graphs we can determine that the influence on the number of points on the accuracy for LM and OI are almost the same. There is a fast improvement in accuracy until 20 feature points after that the improvement is less noticeable. [43]

5.1.4 Conclusion

The OI algorithm is a fast globally convergent iterative pose estimation algorithm. It has a high accuracy for its time, is robust against outliers and is fast. It is especially good when there is poor initialisation but this is unlikely with the more modern linear techniques. If there are more than 20 feature points the accuracy of the algorithm no longer improves significantly. [43]

5.2 POS and POSIT

5.2.1 Introduction

Another pose estimation method is called POSIT, this method was proposed by Daniel DeMenthon and Larry Davis in 1994. In this method, a minimum of 4 non coplanar points are needed to detect the objects pose [44]. The method consists of two algorithms. The first algorithm is called POS (Pose from Orthography and Scaling). This algorithm gives an approximation of the perspective projection using a scaled orthographic one. A scaled orthographic projection (SOP) is used instead of the conventional perspective projection. The rotation and translation of the object can be found by solving a linear system. The second algorithm is called POSIT (Pose from Orthography and Scaling with Iterations). This algorithms uses several iterations to refine the estimated pose, found by POS and increases its accuracy [44].

5.2.2 Working of POS and POSIT

Like all other pose estimation techniques, in the end, a rotation matrix and a translation vector needs to be found, describing the transformation between the camera coordinate frame and the real-world coordinate frame. Figure 54 shows a graphical overview of the coordinate systems and their projections used in this method.

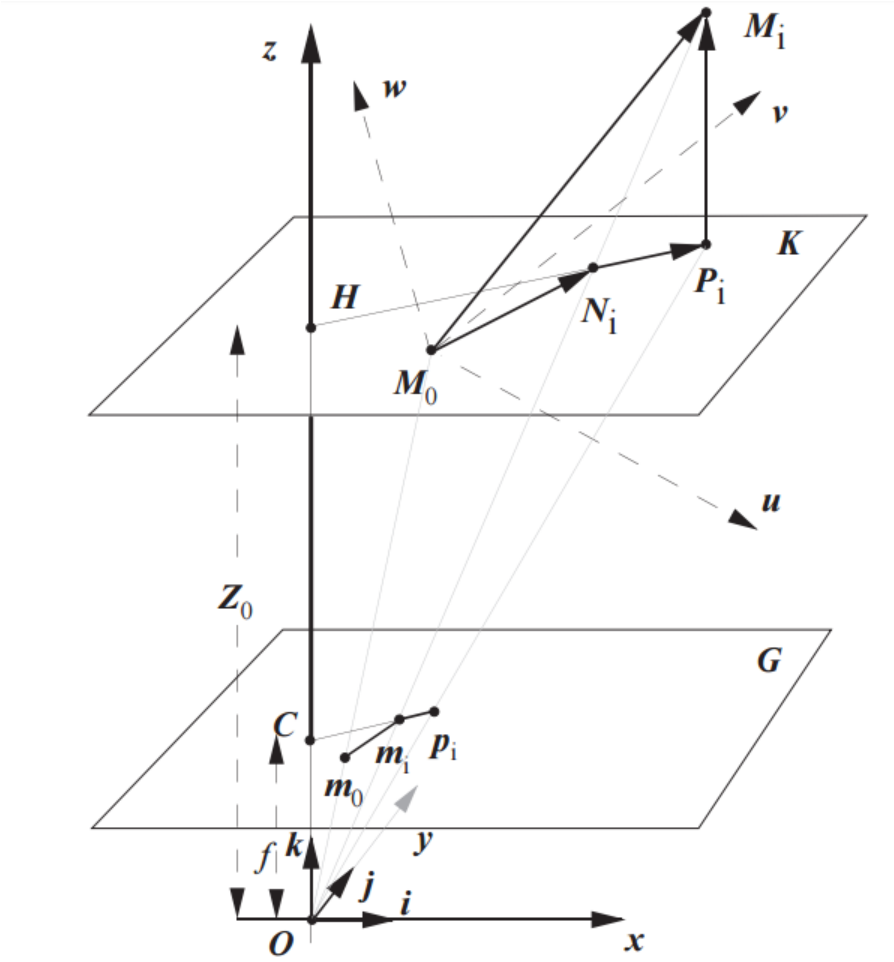

Figure 54: POSIT perspective and scaled orthographic projection of an object using feature points [44].

In this image the center of projection is denoted as O, this is used as the origin of the camera coordinate system where the vectors (**i**, **j** and **k**) represent the unit vectors of the axes of the camera coordinate system. The symbol *f* denotes the focal length and is also the distance between O and the 2D image frame. This 2D image frame is also called the image plane and is denoted in Figure 54 as G. Please note that the image plane G is parallel to the axes Ox and Oy of the camera reference plane, where Ox and Oy are the columns and rows of the camera's sensors. The camera has an object with several feature points in its field of view. In Figure 54, these feature points are shown as $M_0, M_1, M_2, \ldots, M_n$. The real world or object coordinate system uses M_0 as the origin of its coordinate system and use $M_0u, M_0v \text{ and } M_0w$ to form its axes. M_0 is also called the reference point, however, in other techniques, feature points are often called reference points and so we will use the name main feature point for M_0 to avoid confusion. Only two feature points (M_0 and M_i) are shown in Figure 54. Since the shape of the object is known beforehand, the coordinates of point M_i $(U_i,\ V_i,\ W_i)$ in the object reference system are known. However, the coordinates of this point $(X_i,\ Y_i,\ Z_i)$ in the camera coordinate system are unknown. The projection of M_i on the image reference frame gives the point m_i. The coordinates $(x_i,\ y_i)$ of this projection are also known.

As mentioned in the introduction, a rotation matrix and translation vector need to be found. Where the rotation matrix, denoted R, will use the unit vectors of the camera frame reference system (**i**, **j**, **k**) and the coordinates of the object coordinate system $(M_0u,\ M_0v,\ M_0w)$ to depict the rotation between these coordinate systems. This rotation matrix can be written as seen in equation (76) [44].

$$R = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \tag{76}$$

In this equation $i_u,\ i_v,\ i_w$ describe the coordinates of unit vector **i** in the object coordinate system. Keep in mind that only **i** and **j** need to be calculated in the object coordinate system. The reason behind this is that vector **k** can be calculated as the cross-product of **i** and **j** (**i** x **j**). The translation vector is denoted as **T** and it's the vector $\mathbf{OM_0}$. This vector is also shown in Figure 54 and connects the projection center of the camera O with the main feature point M_0, which is the center of the object coordinate system. This translation vector connects the origins of both coordinate systems. The coordinates of the translation vector are (X_0, Y_0, Z_0). The translation vector **T** is aligned with the vector $\mathbf{Om_0}$ (please note the lowercase m) where m_0 is the perspective projection of M_0 on the image plane G. The translation vector T forms a right triangle O-H-M_0 as seen in Figure 55 [44].

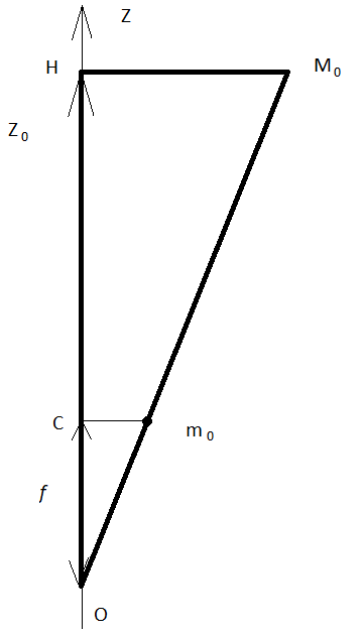

Figure 55: Trigonometric ratios in right triangles

In this triangle is a smaller triangle O-C-m_0.In this triangle is the distance $|OC|$ the focal length f and the distance $|OH|$ is the Z_0 coordinate of the main feature point M_0. Using the properties of trigonometric ratios in right triangles equation (77) can be found.

$$\begin{aligned} &\frac{OM_0}{Om_0} = \frac{Z_0}{f} \\ &\Leftrightarrow Om_0 = \frac{Z_0}{f} Om_0 \end{aligned} \tag{77}$$

As described earlier is the translation vector **T** = OM_0. Using equation (77) this results in

$$T = \frac{Z_0}{f} Om_0 \tag{78}$$

Because O and m_0 are assumed to be known. So to get the pose of the object **i**, **j** and Z_0 are required. POSIT uses scaled orthographic projection (SOP) instead of the conventional perspective projection. In orthographic projection the projection lines are orthogonal to the image plane [45].

Now assume the feature point M_i. In normal perspective projection the x coordinate in the image reference plane of point M_i would be defined as show in equation 4. This equation uses the method visible in equation (77) and Figure 55.

$$x = f\frac{X_i}{Z_i} \tag{79}$$

Figure 54 also shows a plane K. This is plane through M_0 and perpendicular on the Z –axis of the camera coordinate system. This plane is also parallel to the image plane G. Every point on the plan K has a Z coordinate in the camera reference system of Z_0. In order to get an SOP of M_i, M_i is first projected orthographic onto the plane K to form point P_i in K. Then the perspective projection of P_i is used to form the point p_i in the image plane. The x coordinate in the 2D image reference system of this point P_i can be written as seen in equation (80).

$$x_i' = f\frac{X_i}{Z_0} \tag{80}$$

In this equation, the ratio $\frac{f}{Z_0}$ is called the scaling factor of the SOP and is written with the symbol s.
The y coordinates of points M_i, in perspective projection and P_i can be found in a similar manner.
As mentioned earlier, the translation vectors **i** and **j** need to be found. Equations (81) and (82) show a connection between the unknown parameters (the translation vectors **i, j** and Z_0) and the known parameters (the vectors $\mathbf{M_0M_i}$ in the object reference system and x_i, y_i, x_0, y_0,the coordinates the projections of Mi and M_0 in the image reference system). This is a connection in the perspective projection. The connection between these parameters in POS will be described in (84) and (85).

$$\boldsymbol{M_0M_i} * \frac{f}{Z_0}\boldsymbol{i} = x_i(1+\epsilon_i) - x_0 \tag{81}$$

$$\boldsymbol{M_0M_i} * \frac{f}{Z_0}\boldsymbol{j} = y_i(1+\epsilon_i) - y_0 \tag{82}$$

The proof of these equation is shown in the original paper and will not be discussed in this thesis.
In these equation ϵ_i is defined as shown in (83).

$$\epsilon_i = \frac{1}{Z_0}\boldsymbol{M_0M_i} * \boldsymbol{k} \tag{83}$$

The connection between the unknown and known parameters in POS is shown in equation (84) and (85). These equation start from equation (81) and (82) where $x_i(1+\epsilon_i) = x_i' \; and \; y_i(1+\epsilon_i) = y_i'$.
X'_i is the POS projection of M_i onto the image reference plane, which is the same as the perspective projection of P_i onto the image reference plane.

$$\boldsymbol{M_0M_i} * \frac{f}{Z_0}\boldsymbol{i} = x'_i - x_0 \tag{84}$$

$$\boldsymbol{M_0M_i} * \frac{f}{Z_0}\boldsymbol{j} = y'_i - y_0 \tag{85}$$

The proof of this is also assumed valid and omitted in this thesis.
Equations (81) and (82) are rewritten as shown in (86) and (87).

$$\boldsymbol{M_0 M_i} * \mathbf{I} = x_i(1+\epsilon_i) - x_0 \\ \Leftrightarrow \boldsymbol{M_0 M_i} * \mathbf{I} = \zeta_i \tag{86}$$

$$\boldsymbol{M_0 M_i} * \mathbf{J} = y_i(1+\epsilon_i) - y_0 \\ \Leftrightarrow \boldsymbol{M_0 M_i} * \mathbf{J} = \eta_i \tag{87}$$

With $\mathbf{I} = \frac{f}{Z_0}\boldsymbol{i}$, $\mathbf{J} = \frac{f}{Z_0}\boldsymbol{j}$, $\zeta_i = x_i(1+\epsilon_i) - x_0, \eta_i = y_i(1+\epsilon_i) - y_0$

Using the definition of dot product of vectors. The left-hand side of (86) and (87) are rewritten as visible in (88) and (89).

$$\boldsymbol{M_0 M_i I} = [U_i\ V_i\ W_i] * [I_u\ I_v\ I_w]^T = \zeta i \tag{88}$$

$$\boldsymbol{M_0 M_i J} = [U_i\ V_i\ W_i] * [J_u\ J_v\ J_w]^T = \zeta i \tag{89}$$

In this equation we use $\boldsymbol{M_0 M_i} = [U_i\ V_i\ W_i]$. Because there are n feature points in the object reference system. A linear system of equations can be found as seen in (90).

$$\boldsymbol{AI} = x', \boldsymbol{AJ} = y' \tag{90}$$

In this equation **A** signifies the coordinates of the feature points M_i in the object reference system, $x' = (\zeta_1,\ \zeta_2,\ \dots,\ \zeta_n)^T$ and $y' = (\eta_1,\ \eta_2, \dots, \eta_n)^T$. A minimum of 4 visible feature points is require that are noncoplanar. This is necessary because a matrix of rank 3 is needed to solve this system. If these conditions are met equation (91) give the solution.

$$\boldsymbol{I} = \boldsymbol{B}x', \boldsymbol{J} = \boldsymbol{B}y' \tag{91}$$

Matrix B is the pseudoinverse matrix of matrix A. Because A is not a square matrix, the pseudoinverse needs to be found. Matrix B is found using Singular Value Decomposition (SVD). If **I** and **J** are found, **i** and **j** can be calculated by normalizing **I** and **J**. **k** can be found by **i** x **j**. The translation vector $OM_0 = \frac{Z_0}{f}\ Om_0 = \frac{Om_0}{s}$ with s the scaling factor. The scaling factor can be found by calculation the norm of vector **I** and **J**. If the ϵ_i is accurate, the estimated pose will also be accurate otherwise POSIT will use ϵ_i to refine the previously estimated pose [44]. In the first iteration is $\epsilon_i = 0$, then the pose is estimated. This will result in an estimated translation vector, rotation matrix and a new value of ϵ_i. If higher accuracy is needed, the new value of ϵ_i will be used in another iteration of POS to achieve higher accuracy. In most cases, 4 or 5 iteration are used to achieve an accurate result.

5.2.3 Accuracy

Several tests have been done using the POSIT algorithm. In this thesis, a test performed by Philips in 2002 will be discussed. In the first iteration, ϵ_i is set as 0. The reprojection of the estimated pose of the first iteration is shown in Figure 56. The blue lines represent the original model of the object. The corner points are used as reference points. The red lines are the reprojection of the estimated pose in the first iteration of posit. As is clearly visible in this image, the estimated pose in the first iteration is near the original pose but not accurate enough for some applications. In this test, the camera is placed at a distance of 60 mm from the object. [46]

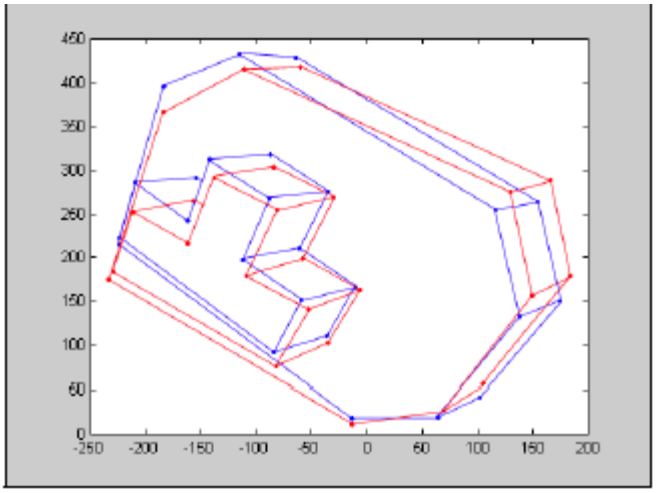

Figure 56: Reprojection error in POSIT test

The reason of this accuracy loss is the fact that POSIT uses scaled-orthographic perspective, which is a weak perspective camera model, in order to get linear equations. Pose estimations with $\epsilon_i = 0$ only produce a fairly accurate result when the distance between the camera and the object is much larger than the size of the object. This effect is seen in Figure 57. In this Figure the x-axis indicates the ratio of the distance between the camera and the object and the size of the object itself. The y-axis of Figure 57 a) indicates the absolute translation error expressed in mm. Figure 57 b) shows the rotational accuracy of POSIT. This graph shows clearly that accuracy starts to improve when the distance is at least 8 times greater than the size of the object. The improvement occurs exponential until the ratio is around 20. After that, the accuracy appears to stabilize. [46]

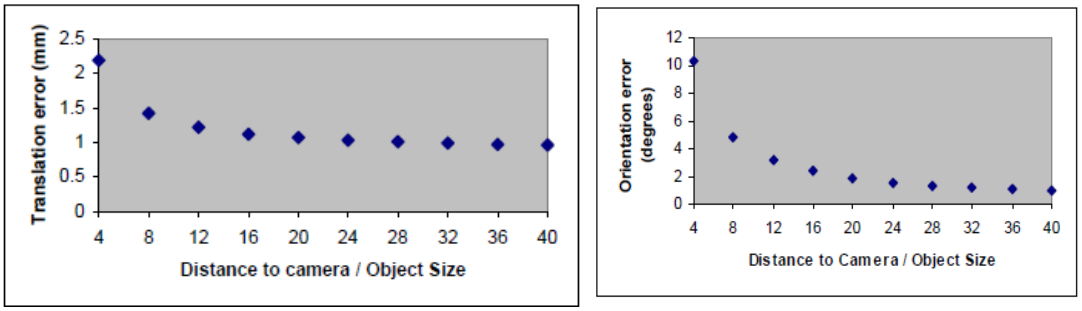

Figure 57: a) POSIT translation error in function of distance/object size ratio
b) POSIT rotation error in function of distance/object size ratio

5.2.4 POSIT Conclusion

POSIT is an interesting method since it uses a Scaled Orthographic Projection instead of the usual perspective projection. Its accuracy appears not to accurate, average errors of over 2 mm in translation and 10 degrees in rotation occur if the object is not far enough from the camera. Another comparison between POSIT and PRSO can also be seen in 4.5.8. However, POSIT also has advantages, as it does not need an initial guess other iterative methods do. Its computational speed is also fairly fast, especially for an iterative method. [46] [24]

6 Conclusion

There are many different algorithms for pose estimation. Each one has its strong and weak points. It is difficult to design a single test to compare each algorithm. Because the algorithms are optimized for different criteria and circumstances. Therefore we use the tests of the original papers and give an overview of the strong and weak points of each algorithm. Sometimes the test displays the results in a similar manner but due to different test conditions it is still difficult to compare them. It is especially difficult to compare the absolute speed of the algorithms. The reasons are the improvements of the processors over the years which have a big influence on the final speed of an algorithm. We can compare the complexity of the algorithms with the big O notation.

Direct Linear Transform (DLT) is one of the earlier pose estimation methods from 1971. DLT is very susceptible to noise. When there is almost no noise the accuracy is fairly high. An advantage of DLT is that it also determines the intrinsic camera parameters. This means that there is no need to calibrate the camera. However it reduces the accuracy of the algorithm. The algorithm can calculate the position by using a minimum of 6 reference points but the accuracy seems to starts to stabilize when there are 15 reference points.

Efficient perspective endpoint (EPnP) from 2008 is a well-known method for pose estimation. It is fairly accurate compared to other non-iterative methods of its time. The O(n) complexity makes it a fast algorithm but it is not very accurate.

The n point linear (NPL) algorithm works relatively better when there are only a few known world points which are located near the center of the image. It is a slow algorithm but is has a high accuracy.

The n linear lines (NLL) algorithm from 2003 is a fast algorithm for line detect. It has a comparable accuracy to the iterative algorithm. The main advantage is that it will not converge on local minima.

The perspective ray camera model has proven to improve the accuracy of the method of Liu ML and Wong KH. The algorithm perspective-ray-based scaled orthographic projection with iterations (PRSOI) has under the original test conditions a higher accuracy then POSIT. Because PRSOI preforms an iterative process to find the four points with the least RMS error the method is probably highly resistant against outliers but is likely more susceptible to image noise. The difference in accuracy was most noticeable in the yaw, pitch and the translations. The roll was only slightly more accurate. The main disadvantage of the perspective ray camera model is that it is more complex.

Because this is a relative recent study of 2015 we have not found any other methods that are integrated with the perspective ray camera model.

Moiré patterns seems a very promising technique to determine the pitch and yaw with a very high accuracy. This accuracy can even be up to 1 arc minute. The measuring depth can be accurate up to 1 % - 2 % on a distance from 2 to 3.25 m. The method works perfect with low resolutions. The disadvantage is that it does not calculate the other pose parameters. This is a method developed in 2016.

The orthogonal iteration algorithm is a globally fast converging iterative pose estimation algorithm from 1998. It is robust against outliers and was one of the first to not converge on local minima. For an optimal performance there needs to be 20 feature points. Adding more points does no longer improve the accuracy significantly.

Scaled Orthographic Projection with Iteration is from 1994 and is one of the most popular methods of its day. It is not a very accurate iterative algorithm but it does not need an initial guess and it is fairly fast for an iterative method.

References

[1] S. Pengfei, S. Changku, L. Wenqiang and W. Peng, “A New Pos eEstimation Algorithm Usinga Perspective-Ray-Based Scaled Orthographic Projection with Iteration,” PLOS ONE, 2015.

[2] J. Weng, P. Cohen and M. Herniou, “Camera Calibration with Distortion Models and Accuracy Evaluation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, no. 10, pp. 969 - 980, 1992.

[3] T. Emanuele and V. Alessandro, Introductory Techniques for 3-D Computer Vision, New Jersey: Prentice Hall, 1998.

[4] K. Simek, “Dissecting the Camera Matrix, Part 3: The Intrinsic Matrix,” 13 08 2013. [Online]. Available: http://ksimek.github.io/2013/08/13/intrinsic/. [Accessed 16 10 2017].

[5] T. Blakesley, “A new Definition of Focal Length, and an Instrument for determining it,” *Proceedings of the Physical Society of London,* vol. 44, no. 267, pp. 144-145, 2009.

[6] R. Raskar, *computational camera & photofraphy:,* MIT Medi Lab, 2009.

[7] T. Clarke, J. Fryer and X. Wang, “THE PRINCIPAL POINT AND CCD CAMERAS,” [Online]. Available: http://www.optical-metrology-centre.com/Downloads/Papers/Photogrammetric%20Record%201998%20Principal%20Point.pdf. [Accessed 2017 11 17].

[8] “In-Depth Discussion,” National Instruments, June 2011. [Online]. Available: http://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/spatial_calibration_indepth/.

[9] J. Caarls, “Pose estimation for mobile devices and augmented reality,” Technische Universiteit Delft, Delft, 2009.

[10] J.-Y. Bouguet, “Camera Calibration Toolbox for Matlab,” 14 10 2015. [Online]. Available: https://www.vision.caltech.edu/bouguetj/calib_doc/htmls/parameters.html. [Accessed 26 10 2017].

[11] Z. Zhang, “A flexible new technique for camera calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 22, no. 11, pp. 1330-1334, 200.

[12] T. Vass, “Applying and REmoving Lens Distortion in Post Production,” in *The Second Hungarian Conference on Computer Graphics and Geometry*, Budapest, 2003.

[13] D. C. Brown, “Decentering distortion of lenses,” *Photogrammetric Engineering,* vol. 32, no. 3, pp. 444-462, 1966.

[14] “Real Time pose estimation of a textured object,” OpenCV, [Online]. Available: http://docs.opencv.org/trunk/dc/d2c/tutorial_real_time_pose.html. [Accessed 28 05 2017].

[15] M. Jo, “camera calibration part I,” 19 August 2015. [Online]. Available: http://marcelojo.org/marcelojoeng/2015/08/camera-calibration-part-i.html.

[16] R. Hartley and A. Zisserman, Multiple View Geometry in computer vision., Cambridge: Cambridge University Press, 2003.

[17] J. Stals, “Pose estimation with a camera for orthognathic surgery planning,” Uhasselt, Diepenbeek, 2015.

[18] “Reprojection Error,” PIX4D, 2 May 2017. [Online]. Available: https://support.pix4d.com/hc/en-us/articles/202559369-Reprojection-Error#gsc.tab=0. [Accessed 10 May 2017].

[19] T. Luhmann, “Precision potential of photogrammetric 6DOF pose estimation with a single camera.,” *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 64, pp. 275-284, 2009.

[20] G. Casella, Monte Carlo Statistical Methods, New-York: Springer, 2002.

[21] T. D.S., “Direct Linear Transformation (DLT),” [Online]. Available: https://me363.byu.edu/sites/me363.byu.edu/files/userfiles/5/DLTNotes.pdf. [Accessed 29 05 2017].

[22] Y. Abdel-Aziz and H. Karara, “Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry,” in *Proceedings of the Symposium on Close-Range Photogrammetry,* Urbana, Illinois, 1971.

[23] Y.-H. Kwon, “DLT Method,” Kwon3D, 1998. [Online]. Available: http://www.kwon3d.com/theory/dlt/dlt.html. [Accessed 29 05 2017].

[24] T. Petersen, A comparison of 2D-3D Pose Estimation Methods, Ballerup, Denmark: Aalborg University, 2008.

[25] G. Xiao-Shan, H. Xiao-Rong, T. Jianliang and C. Hang-Fei, “Complete Solution Classification for the Perspective-Three-Point Problem,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 8, pp. 930-943, 2003.

[26] P.-S. Adrian, A.-C. Juan and M.-N. Francesc, “Exhaustive Linearization for Robus Camera Pose and Focal Length Estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 35, no. 10, pp. 2387-2400, 2013.

[27] V. Lepetit, F. Moreno_noguer and P. Fua, “EPnP: An Accurate O(n) Solution to the PnP Problem,” *International Journal of computer Vision,* 2008.

[28] P. Fiore, “Efficient linear solution of exterior orientation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 23, no. 2, pp. 140-148, 2001.

[29] A. Ansar and K. Daniilidis, “Linear Pose Estimation from Points or Lines,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 5, pp. 578-589, 2003.

[30] W. Eric, “Barycentric Coordinates,” MathWorld- A Wolfram Web Resource, [Online]. Available: http://mathworld.worlfram.com/BarycentricCoordinates.html. [Accessed 01 06 2017].

[31] A. Nentchev, “Linear Shape functions on Tetrahedral Elements,” [Online]. Available: http://www.iue.tuwien.ac.at/phd/nentchev/node30.html. [Accessed 25 05 2017].

[32] M. Weinmann, Reconstruction and Analysis of 3D Scenes, Karlsruhe: Springer, 2016.

[33] M. Patric, Drawing distinctions, the varieties of graphic expression, Cornell: Cornell University Press, 2005.

[34] C. H. Rycroft, "Gauss-Newton Method," 2014. [Online]. Available: http://iacs-courses.seas.harvard.edu/courses/am205/fall14/slides/am205_lec06.pdf. [Accessed 27 05 2017].

[35] Z. yinqiang, K. Yubin, S. Shigeki, A. Kalle and O. Masatoshi, "Revisiting the PnP Problem: A fast, General and Optimal Solution," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, Sydney, Australia, 2013.

[36] A. Adnan and K. Daniilidis, "Linear Pose Estimation from Points or Lines," IEEE, 2003.

[37] D. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 441-450, 1991.

[38] L. Quan and Z. Lan, "Linear N-point Camera pose determination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 7, pp. 774-780, 1999.

[39] I. Amidror, The Theory of the Moiré Phenomenon, Kluwer Academic Publisher, 2000.

[40] "Moiré ecart angulair - Moiré pattern," Wikipedia, 20 06 2005. [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/b/b1/Moire_ecart_angulaire.png. [Accessed 27 04 2017].

[41] W.-C. Wang, "Moiré Method," 2011. [Online]. Available: http://depts.washington.edu/mictech/optics/me557/moire_a.pdf. [Accessed 26 04 2017].

[42] B. Armstrong, T. Verron, L. Heppe and R. Karonde, "RGR-6D: Low-cost, High accuracy Measurement of 6-DOF Pose from a Single Image," 2016.

[43] L. Chien-Ping, D. H. Gregory and M. Eric, "Fast and Globally Convergent Pose Estimation From Video Images," 1998.

[44] D. DeMenthon and L. Davis, "Model-Based Object Pose in 25 Lines of Code," *International Journal of Computer Vision,* vol. 15, no. 2, pp. 123-141, 1995.

[45] L. Khen, "Camera Models and Imaging," [Online]. Available: https://www.comp.nus.edu.sg/~cs4243/lecture/camera.pdf. [Accessed 01 06 2017].

[46] I. Lopez, A. Frangi, B. van der Wijst and H. Broers, "Pose Estimation From 2D to 3D For Computer Vision In An Assembry Node," Philips, 12 03 2002. [Online]. Available: http://www.dtic.upf.edu/~afrangi/articles/Philips-CTB500-02-0000.pdf. [Accessed 01 06 2017].

[47] "Reprojection Error," PIX4D, 2 May 2017. [Online]. Available: https://support.pix4d.com/hc/en-us/articles/202559369-Reprojection-Error#gsc.tab=0. [Accessed May 2017].

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
3D position determination using a camera

Richting: **master in de industriële wetenschappen: elektronica-ICT**
Jaar: **2017**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

Bortels, Tom **Bostijn, Bart**

Datum: **6/06/2017**