Calibration of robot kinematics using a double ball-bar with embedded sensing

Sebastian Brand

Niklas Nilsson

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6017
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2016

Abstract

Today's industrial robots are highly repeatable, but need to be calibrated to improve their absolute accuracy. This calibration can be done on many of the robot properties such as kinematic parameters, joint friction or bending stiffness. This thesis explores a calibration procedure for the kinematic parameters, that use a specialized piece of hardware - the double ball-bar.

The double ball-bar restricts the motion of the robot to a spherical surface. Sensors were added to the ball-bar joints, which made it possible to use the forward kinematic homogeneous transformation matrix from both the robot side and the ball-bar side, the matrices can be compared to each other and the parameters of the robot can be identified using a non-linear least-squares minimization algorithm.

The calibration proved promising in simulations and showed an increased robustness to error sources such as white noise and fluctuations of the gear ratio found in cycloid drives. It also provided an improved identification of the robot parameters compared to the calibration done using the sensor-less double ball-bar.

In experiments the identification showed some improvement in the identification over using the sensor-less double ball-bar, but also that the method needs to be further improved to be able to produce satisfactory calibration results.

Acknowledgements

We would like to thank everyone at Cognibotics for the opportunity to conduct this master thesis. Special thanks to Klas Nilsson, Martin Holmstrand and Sandra Collin for their involvement and support. Thank you Klas for allowing us to come along to the Comau factory in Turin and be involved in the practical testing; it was a lot of fun and a very rewarding experience.

Thank you Svante Bouvin for the help with the detail drawings and the manufacturing. Last but not least thank you Anders Robertsson for being an inspiring professor and setting us up with Cognibotics to start with.

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

As industrial robots are used more and more in machining, the need for high accuracy increases. Today's robots are highly repeatable but lack the absolute accuracy needed for machining with low tolerances. The tolerance can be defined as the maximum allowed error between the actual value and the desired value, like a dimension for example, where low or tight tolerance means small allowed error.

Repeatability can be defined as the robot's ability to return to the same pose (position and orientation) when doing repeated movements. This does not necessarily mean it is the absolute correct pose it reaches every time, but the pose will be the same every time. Accuracy on the other hand is defined as the robot's ability to reach a well known position in space, and exactly that position [*Accuracy and Repeatability in Industrial Robots*]. The difference between the two is illustrated in Figure 1.1.

When programming robots, two methods are used - online or offline programming. Online programming means that every robot is taught its targets in the application environment. Online programming requires a good repeatability to reach the taught pose successfully every time. A drawback with this method is that while the robot is being taught, it can not be used in production activities. If there are many poses to be taught, the method can be very time-consuming.

The other method is to use offline programming in a virtual environment. In this case there is a larger need for the robot to be accurate, because a difference in the actual robot parameters compared to the nominal parameters that the offline controller is using will result in an error on the real robot. This method is suitable for applications with many and/or complex poses, but it requires good absolute accuracy. Furthermore the physical robot does not lose any productive time with this approach since it can be used all the time up until reprogramming and deployment.

It was found that building the robots to very good repeatability and then compensate for accuracy errors by updating the kinematic model to represent the real robot was the better approach to calibration instead of manufacturing the robot with very high standards and exact dimensions [Benjamin et al., 1991]. This is the

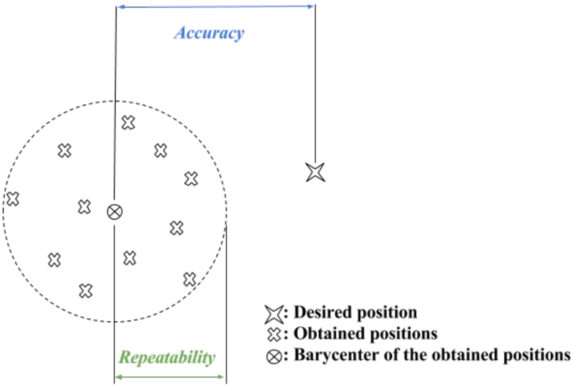


Figure 1.1 An illustration of repeatability and accuracy, adapted from [*Accuracy and Repeatability in Industrial Robots*].

motivation behind calibration, to find the actual parameters of the real robot which makes it behave closer to how it does in a virtual environment [ISO 9283:1998(en), 1998].

This thesis was conducted primarily at the company Cognibotics AB, which is a company that specializes in methods and services for high-performing and cost-effective determination of robot properties such as backlash, friction, and non-linear compliance.

1.2 Problem formulation

Double ball-bar calibration without ball-bar sensors in experiments conducted by Cognibotics AB has turned out to be very sensitive to unmodeled errors and in an effort to improve the calibration, ball-bar sensors are introduced. The improvement in the calibration robustness is tested using both simulations and experiments. The procedure to accomplish this is divided into smaller steps:

1. Selection of measurement method and sensors needed.

2. Simulations with some unmodeled error sources.

3. Design and manufacturing of hardware needed to attach the sensors to the ball-bar.

4. Experiments examining error sources and the potential improvement of calibration robustness.

1.3 Previous work

The area of robot calibration with or without passive end-point constraints is quite well explored. There are several methods using external measuring equipment to measure the robot position or robot pose, which along with the robot joint angles for each pose is used for parameter identification. However many of these methods use expensive optical instruments such as 3D cameras and laser tracking systems [Nubiola and Bonev, 2012] or measurement arms [Ginani and Motta, 2011]. The result achieved in those papers was reduced maximum/mean errors from 0.968 mm/2.158 mm to 0.364 mm/0.696 mm and improved accuracy from 1.5 mm to 0.3 mm.

A method using alternative external measurement equipment is presented in [Bennett and Hollerbach, 1991] where two serial manipulators are used; one to be calibrated and the other either locked as an end-point constraint or used for measurement.

There are also several methods that use semi-passive end-point constraints for calibration. In [Goswami et al., 1993] a method is presented that uses a telescopic ball-bar with a linear distance encoder as external measurement of the robot pose and in [Joubair and Bonev, 2015] the sides of a well defined cube is used as a passive constraint along with a touch probe for triggering the measurements.

A method for calibrating a robot using only a passive end point constraint can be found in [Collin, 2016] where a sensor-less double ball-bar is rigidly attached to a robot and constrains its movement to a sphere. The calibration reduced the position errors by up to 80% and rotational errors by up to 22%. The method and the report are used as a base for this thesis which will use the same passive constraint - the double ball-bar.

1.4 Robot calibration

To perform a calibration of a robot using this method, there are a number of steps to follow. To begin with, data from the robot joint angles and the ball-bar joint angles need to be collected. This is performed by attaching the robot rigidly to the cardan joint of the double ball-bar, which will restrict the robot's movement to a spherical surface in space and the setup can be seen in Figure 1.2. The robot is then run in a path that covers as much as possible of the robot's reach of the sphere while both the robot and the ball-bar joint angles are logged. This data collection makes it possible to formulate a non-linear optimization problem which can be solved using a suitable algorithm. The kinematic parameters will be identified by this optimization problem and will better represent the actual robot compared to the nominal kinematic parameters found in the robot specification.

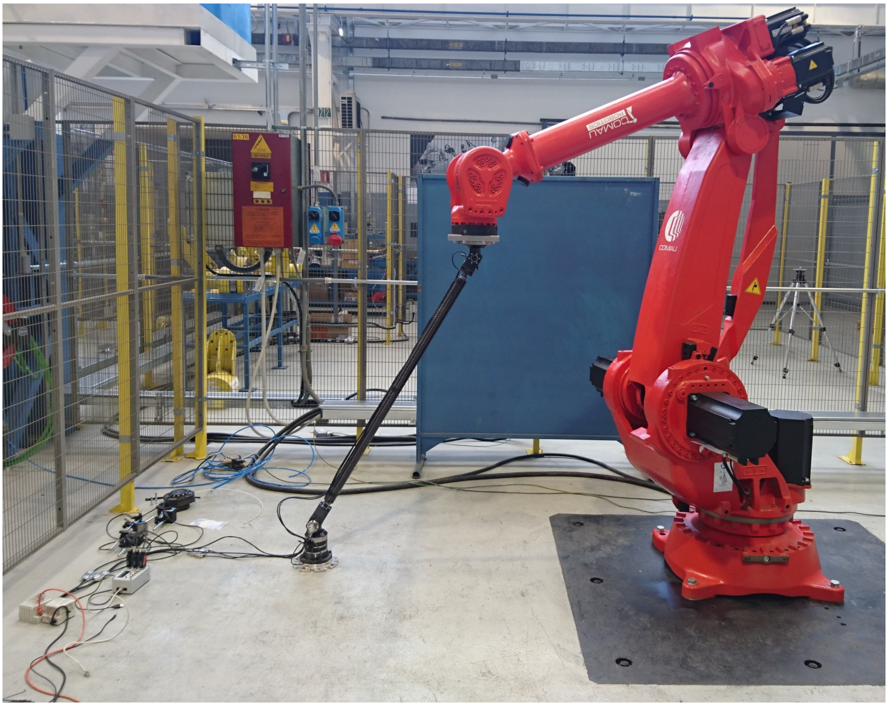


Figure 1.2 The experimental setup when using the Comau NJ220-2.7 robot.

1.5 Report outline

2 - Kinematic parameter identification Introduction to the concepts and mathematical principles behind the kinematic parameter identification

3 - Simulations Description and theory of the error sources used in simulation, and results from the kinematic parameter identification for the simulated data.

4 - Hardware development Design and considerations for the hardware that was developed and manufactured.

5 - Experiments Description of the experiments that were conducted and the results from them.

6 - Conclusion and discussion Conclusions for the thesis and discussion about the results and the method used to reach them.

7 - Further work Possible solutions to the problems found when moving forward in improving the method.

Appendix Contains descriptions of the robots that were used in the experiments and additional results from the experiments.

1.6 Work division

The total work in this thesis was equally divided. Both authors have been involved in all parts of the thesis, though some focus have been put on different parts. Niklas had some more focus on the hardware development and logging parts while Sebastian focused more on simulations and the identification algorithm.

2

Kinematic parameter identification

2.1 Serial manipulator description

There are a number of mathematical descriptions of a serial manipulator. They all provide a way to calculate the position and orientation of the tool center point, the TCP, expressed in the robot base frame coordinates. The one used for describing the robot and ball-bar in this thesis is the Denavit-Hartenberg parameters. The reason is that it is a common convention and is used in software and in industry today [Spong et al., 2006].

The Denavit-Hartenberg parameters, DH parameters, are used in one of the oldest descriptions of serial manipulators. Introduced in 1955 by Jacques Denavit and Richard Hartenberg [Hartenberg and Denavit, 1955], it is a convention to describe the transformation of coordinate frames when traversing a series of manipulator links. For every link, there are four parameters. These four parameters are used in the calculation of a homogeneous transformation matrix between the previous link and the current link. Using those transformations, it is possible to calculate the position and orientation of the TCP in the base frame coordinates. To be noted is that there are two conventions of indexing the DH parameters where the one used in this report often is referred to as the standard DH parameter, compared to Modified DH parameters which is the alternative. Since they produce different sets of parameters it is important to make sure that only one convention is used [Spong et al., 2006].

One drawback with using DH parameters is that it has a mathematical singularity when two links are nearly parallel, which results in large jumps in the parameters for small joint position changes. One way to solve this is using the Hayati modified DH parameters which introduces an extra rotation parameter for every link [Hayati and Mirmirani, 1985]. In this thesis the singularity is however assumed not to influence the simulations and experiments due to some parameters not being identified, which can seen in Section 3.2.

The standard Denavit-Hartenberg parameters of a serial manipulator, which are illustrated in Figure 2.1 are defined as follows:

1. a_i (Link length) - The distance along the x_i axis from O_i to the intersection of x_i and z_{i-1} axes.
2. α_i (Link twist) - The angle between the z_{i-1} and z_i axes measured around x_i.
3. d_i (Link offset) - The distance along the z_{i-1} axis from O_{i-1} to to the intersection of the x_i and z_{i-1} axes. If the joint is a prismatic joint, this parameter is a variable.
4. θ_i (Joint angle) - The rotation around the z_i axis. If the joint is a revolute joint, this parameter is a variable.

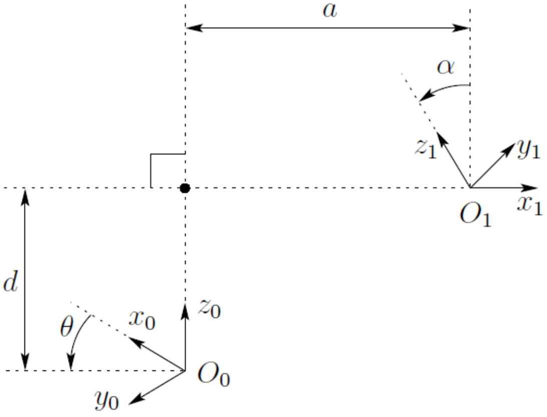


Figure 2.1 The Denavit-Hartenberg parameters $a_1, \alpha_1, d_1, \theta_1$ shown on a link transformation [Spong et al., 2006].

For each link a homogeneous transformation matrix can be calculated, which can then be multiplied together to get the total transformation matrix for several consecutive links. The elements of a homogeneous transformation matrix can be seen below.

$$T = \left[\begin{array}{ccc|c} X_x & Y_x & Z_x & P_x \\ X_y & Y_y & Z_y & P_y \\ X_z & Y_z & Z_z & P_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Where:

$$
\begin{aligned}
T &: \text{the resulting homogeneous transformation matrix} \\
X_{x,y,z}, Y_{x,y,z}, Z_{x,y,z} &: \text{3x3 matrix that describes the relative orientation} \\
P_{x,y,z} &: \text{the translatation in x, y and z coordinates.}
\end{aligned}
$$

2.2 Double ball-bar

The double ball-bar, which can be seen in Figure 2.2, consists of one gimbal joint at the base and one cardan joint at the end of a rod. The gimbal joint is physically built up by three single axis joints and the cardan is built up by two single axis joints, on which it is possible to mount angle encoders. This configuration gives a five degree-of-freedom mechanism, with an end point moving along a spherical surface. The sphere center is the center of the gimbal joint and the radius is the length of the ball-bar rod and the connecting parts of the joints. To be noted is that the ball-bar only defines a sphere if the center of the cardan joint is considered as the ball-bar end point, which in the mathematical model is solved by assigning tool changer, offset plate and part of the physical cardan joint parts as the robot tool. To completely define the ball-bar position the ball-bar origin is described using a homogeneous transformation matrix with a translation in the robot base coordinate system, which is used as the main coordinate system.

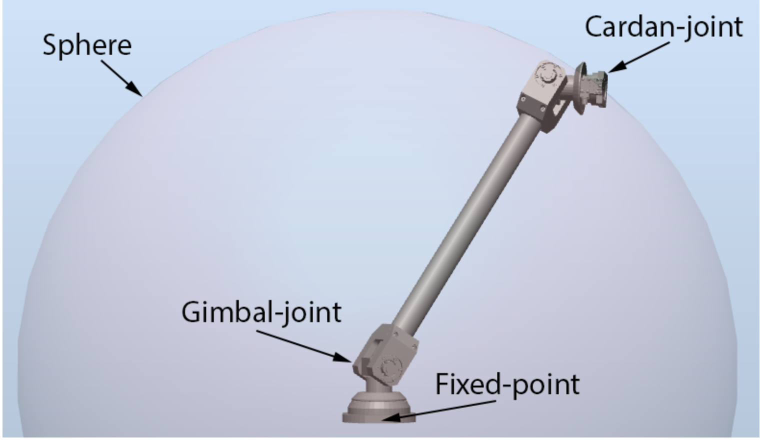


Figure 2.2 The spherical surface defined by the double ball-bar [Collin, 2016].

2.3 Error sources

When identifying robot parameters with double ball-bar identification it is the DH parameter errors that are identified. In addition to these there are also varying robot errors that depend on the robot position, payload and usage. Some of these varying errors are modelled and accounted for by Cognibotics, as robot deformation under load for the different robot joints as well as robot joint gear backlash, which means that they are not considered unmodeled errors and therefore not used in the simulations. There are however many types of unmodeled errors that might reduce the performance of the double ball-bar identification, some of which are briefly described below and also used in simulations.

Cycloid drives

A drawing of a cycloid drive can be seen in Figure 2.3 and briefly described the cycloid drive consists of an input shaft with an eccentrically mounted bearing. This bearing drives a larger cycloid disc with pin holes and stationary rollers along the edge of the disc. The rolling pins, directly connected to the output shaft, fit into the pin holes of the cycloid disc. The resulting rotation of the output shaft when spinning the input shaft is a reversion of the rotation direction and a large reduction in rotational speed, with the corresponding increase in output torque.

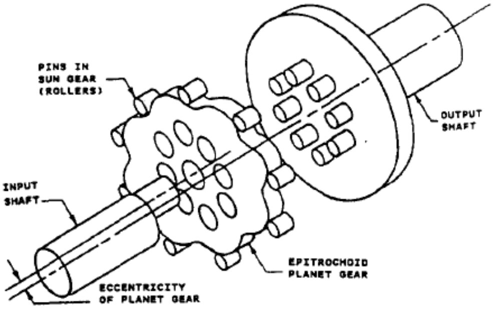


Figure 2.3 The schematic structure of the cycloid drive [Yang and Blanche, 1990].

The cycloid drive type transmission is used in the Comau NJ 220-2.7 robot that was used in some of the experiments. It is a compact transmission type with high gear ratio and high efficiency. It is however not a perfect transmission; compared to traditional involute gears, that are found for example in car transmissions, the cycloid drive has larger backlash and there is torque and gear ratio ripple present at the output shaft because of this backlash. An example of gear ripple in a cycloid drive can be seen in Figure 2.3. The size of the ripple depends on manufacturing tolerances and wear, and is often larger for higher gear ratios which comes from the smaller gear teeth needed for higher gear ratio [Yang and Blanche, 1990].

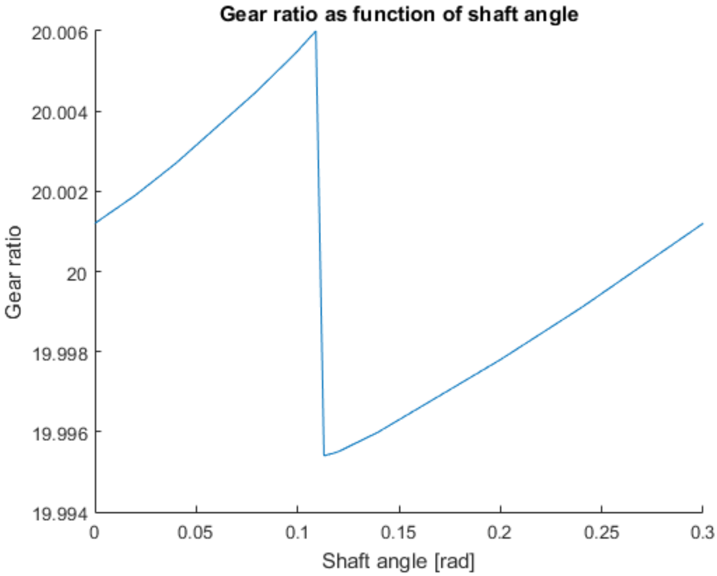


Figure 2.4 Gear ratio ripple in cycloid drives (created with data from [Yang and Blanche, 1990]).

Sinusoidal gear error

For other unmodeled gear errors a sine wave is used as a simple model. The idea is that some errors might arise from not perfectly concentric mounted motors or gear boxes, and these should intuitively have an error close to a sine wave which is periodic either with the input or the output axis. Sine errors might also be interesting as a substitute for gear ratio errors in cycloid drives since sine waves are simpler to simulate than gear ratio ripples.

Ball-bar backlash

The ball-bar is, in [Collin, 2016] and this thesis, assumed to perfectly follow the sphere surface, which might not be completely true. If the ball-bar is exposed to larger forces it will stretch or compress, or ball-bearings in the joints may have some backlash. Especially when using large forces in experiments the performance of the ball-bearings might be reduced. An example of this is one of the ball-bar joints previously used in experiments, which had an evident ball-bearing backlash.

Noise

Noise is a common error source in experiments and is in this report also used to simulate non-specific unmodeled errors. The type of noise used is Gaussian white noise, which is a common noise type with jointly Gaussian distribution with a spectral density that is equal across all frequencies [Gallager, 2013].

Synchronization error

Since the joint angles for the robot and for the ball-bar in the experiments in this report will be logged on different systems it is interesting to see the influence of non-perfect log synchronization, or of clock drift in the systems. This means that robot angles and ball-bar angles that are assumed to be logged at the same time, and therefore describe the same robot and ball-bar position and rotation, might have been logged with some time difference. The time error will depend on the method for synchronizing the log files but is assumed to be counted in milliseconds.

Truncated joint angle values

The limited accuracy and resolution of angle encoders is also an unmodeled error source. Truncated joint angle values are used as a simple model for limited accuracy, and mostly concerns the ball-bar angles where external angle encoders are to be attached.

2.4 Identification algorithm

When solving minimization problems there are several possible algorithms to use. Following previous double ball-bar based calibration the Levenberg-Marquardt is used. It is a sum-of-squares-based algorithm that can be used for non-linear problems, and it is a common algorithm for solving non-linear optimization problems. It should be noted that the Levenberg-Marquardt only finds a local minimum, which may not be the global minimum. The Levenberg-Marquardt algorithm uses a user-defined cost function and its Jacobian for solving the problem defined by the cost function [Araneda, 2004]. A simple example of a cost function is $C = (1-x)^2$, which is minimized when $x = 1$, and the cost functions used in this thesis can be seen in Equations 2.1-2.4.

2.5 Cost functions

Depending on the number of ball-bar sensors used in the double ball-bar based parameter identification the cost function looks different. Common to all cost functions used in this report is that they are differentiable, to make it possible to generate the Jacobian used by the algorithm.

Length

$$C_\ell = (R_x - o_x)^2 + (R_y - o_y)^2 + (R_z - o_z)^2 - bd_3^2 \tag{2.1}$$

where

$R_{x,y,x}$: the robot TCP position calculated with robot forward kinematics
$o_{x,y,x}$: the position of the ball-bar origin
bd_3 : the ball-bar length

This is the cost function used for sensor-less double ball-bar calibration in [Collin, 2016], and is used as a reference. Note that since the errors are squared in the Levenberg-Marquardt algorithm there is no need to use the square of the length difference.

Position

$$C_P = (R_x - B_x)^2 + (R_y - B_y)^2 + (R_z - B_z)^2 \tag{2.2}$$

where

$R_{x,y,x}$: the robot TCP position calculated with robot forward kinematics
$B_{x,y,x}$: the ball-bar position calculated with ball-bar forward kinematics

The idea behind this cost function is that by comparing errors in space instead of distance to the ball-bar origin, points where the distance is correct but the position is wrong are also penalized. One practical reason for considering this cost is that it is possible to implement when only using angle encoders on the first two joints, counted from the floor and up, on the ball-bar.

Position and ball-bar angles

$$C_{P\theta} = C_P + sin(\widehat{b\theta_4} - b\theta_4)^2 + sin(\widehat{b\theta_5} - b\theta_5)^2 \tag{2.3}$$

where

$b\theta_4$: the ball-bar angles for joint 4, measured with ball-bar sensors
$b\theta_5$: the ball-bar angles for joint 5, measured with ball-bar sensors
$\widehat{b\theta_4}$: the ball-bar angles for joint 4 calculated using ball-bar inverse kinematics with the transformation matrix from robot forward kinematics.
$\widehat{b\theta_5}$: the ball-bar angles for joint 5 calculated using ball-bar inverse kinematics with the transformation matrix from robot forward kinematics.
C_P : the cost based on the position error, which is described in Equation 2.2.

The reason for considering this cost function is that it uses all sensors that were available in the experiments. There are however some drawbacks with this cost function, which is that it is very sensitive close to ball-bar singularities. Another drawback is that the ball-bar inverse kinematics uses both four quadrant

inverse tangent and matrix inverse which makes the cost function and the Jacobian expressions long and computationally intensive to use.
The sine is used on the errors to make small errors and (2π - small errors) equally valid.

Full transformation

$$C_T = \sum((R - B) \odot (R - B)) \tag{2.4}$$

where

R : the robot forward kinematics transformation matrix
B : the ball-bar forward kinematics transformation matrix
$\odot$: element-wise multiplication
$\sum$: the sum of all elements in the matrix

In comparison to $C_{P\theta}$, C_T uses all joint values of the ball-bar. The reason for using the full transformation matrix instead of comparing the ball-bar angles as in $C_{P\theta}$ is partly to reduce the sensitivity to ball-bar singularities, and also to reduce the complexity of the cost function to make it less computationally intensive to use.

3

Simulations

The goal with the simulations was to compare the cost functions against each other for different types of errors, and also to compare how different types of errors affect the parameter identification. To be able to compare the simulation results with experiments all simulations were done with the ABB IRB2400 robot as model, with the parameters seen in Appendix A. The simulations were mostly done using Matlab, where the Matlab Symbolic Toolbox was used to generate the symbolic expressions for the cost functions, and a toolbox developed for Matlab called RVC Toolbox [Corke, 2011] was used for calculating much of the robot and ball-bar kinematics

3.1 Error model

Different errors added to the model are both static for all robot positions and varying between the robot positions. The static errors are errors in the DH parameters for the robot and ball-bar. These errors are the ones that should be identified, and are often a result of machining tolerances in the production chain or a result of wear of the robots. Varying errors are errors as encoder noise or unmodeled gear ratio ripple. These are assumed to have a mean of zero for the path used for the simulations, since varying errors with a non-zero mean are assumed to be modelled as a static error added to a zero mean varying error.

To generate robot angles for a path with added errors the following steps were performed.

1. Initial transformation matrix is calculated based on the initial DH parameters for the robot and the reference angles that are read from a file.

2. Adjusted transformation matrix is calculated. The position is updated based on errors in the ball-bar length and ball-bar origin. The rotation part of the transformation matrix is kept as it is.

3. Adjusted DH parameters for the robot and the ball-bar are calculated by adding errors to the initial DH parameters.

4. Inverse kinematics for both the robot and the ball-bar are calculated based on the adjusted transformation matrix and the adjusted DH parameters.

5. The calculated angles are written to a file in the same format as the reference file, and a file is created describing the errors used to generate the new file.

Since the reference is given in robot angles it is possible to use a path measured on a real robot and calculate a path that is almost equal but without errors, and introduce known errors. This way it is possible to in a simple way take into account the different number of log entries for different positions on the sphere, which is a result of the robot moving slower along the surface at some points and moving faster along the surface at some points.

Varying errors

The different varying errors used are described in Section 2.3. Errors that affect the robot joint angles are cycloid gear ratio ripple, sinusoidal gear error and noise. For cycloid gear ratio ripple, referred to as just gear error, and sinusoidal gear error the size of the errors seen in plots is the amplitude of the signals, and for noise the error size is the the standard deviation. Ball-bar joints in simulations are only affected by noise and truncated joint angle values, where the smallest angle difference possible in the truncated values is the size of the error. The ball-bar is also affected by ball-bar backlash, where the error size is the deviation from initial ball-bar length. Synchronization errors are defined by the time the logged points for the robot and ball-bar differs. Even if the distance the robot has moved in the time difference depends on how fast the robot was run for the specific simulation, it is possible to compare different cost functions against each other.

Kinematic parameter errors

Errors were introduced to ball-bar encoder offsets, ball-bar origin, ball-bar length and some of the DH parameters for the robot. Errors were only applied to the robot parameters that are identifiable for all cost functions, to simplify simulation and comparison between cost functions. No errors were applied to any other ball-bar parameters with the reason that the ball-bar joints are assumed to be well defined. The errors used in the simulations were constant for all simulations and were initially taken from a white Gaussian distribution with standard deviation of 1 *mm* (or *mrad*). This creates errors that can be seen in Table 3.1, which considering the errors in [Collin, 2016] are of reasonable amplitude, and the reason for Gaussian distribution is that many variations in manufacturing fit a Gaussian distribution [Geng, 2004]. Since the DH parameters consist of both translational and rotational parameters the unit is either *mm* or *mrad*, which can be shortened as *mm*|*mrad*.

i	ra_i	rd_i	$r\alpha_i$	$r\theta_i$	ba_i	bd_i	$b\alpha_i$	$b\theta_i$	$o_{x,y,z}$
1	-0.40	0	0.37	0	0	0	0	-0.74	0.16
2	0.14	0.27	0.78	-0.09	0	0	0	1.27	0.80
3	-1.52	0	-1.36	0.19	0	0.24	0	-1.90	-0.70
4	-1.88	-0.16	0.35	-0.12	0	0	0	-0.19	-
5	0.80	1.80	0.08	0.72	0	0	0	0.98	-
6	0	-1.47	0	-0.22	-	-	-	-	-

Table 3.1 Introduced static errors, the unit is $mm|mrad$.

3.2 Identifiability

Not all parameters are identifiable. One reason is that the parameter Jacobian is numerically zero and another reason is that the parameter depends on another parameter, where one example is the robot joint 1 angle and the ball-bar origin xy-position. To detect which parameters depend on each other QR decomposition of the Jacobian can be used, where small or zero diagonal elements in the Q-matrix generated by the QR decomposition comes from parameters which either depends on other parameters or have zero Jacobian [*QR-decomposition*]. Parameters with zero Jacobian do not affect the cost and can be removed without consequence. For parameters that depend on other parameters the dependencies are found by removing all non-identifiable parameters except the one to be examined and removing identifiable parameters until the one examined is no longer reported as non-identifiable.

Since static errors in the simulation are not applied to all parameters, some parameters initially have zero Jacobian. To avoid these parameters to be reported as non-identifiable 10 μrad noise is applied to the angles produced by the error model. Since the Jacobian and QR decomposition is done numerically all values less than $1 \cdot 10^{-6}$ are considered numerically zero.

The resulting parameter dependencies are found in Table 3.2 where there are one row for each of the cost functions used in the parameter identification. Each column in the rows represents a parameter dependency, where only one of the parameters separated by vertical bars can be identified while the others need to be set as non-identifiable. An example is that only one of ra_6, bd_3 and ba_5 can be identified while the other two must be set as non-identifiable for the cost functions C_P, $C_{P\theta}$, C_T. The parameters with zero Jacobian are found in Table 3.3, with a row for each of the cost functions. The parameters in the zero Jacobian table are ordered to simplify comparison between the cost functions.

These non-identifiable parameters are just the ones which initially depend on each other. Depending on the robot path used and the DH parameters for the robot and ball-bar additional non-identifiable parameters might be found. Especially with a parameter identification starting near or at the solution many parameters will have a zero-Jacobian and near-zero element in the QR decomposition, which is therefore

C_ℓ	$ra_6 \mid bd_3$	$rd_1 \mid o_z$	$rd_2 \mid rd_3$	$r\theta_1 \mid o_{x,y}$		
C_P	$ra_6 \mid bd_3 \mid ba_5$	$rd_1 \mid o_z \mid bd_1$	$rd_2 \mid rd_3$	$r\theta_1 \mid o_{x,y}$	$rd_6 \mid bd_5$	$bd_2 \mid b\alpha_2$
$C_{P\theta}$	$ra_6 \mid bd_3 \mid ba_5$	$rd_1 \mid o_z \mid bd_1$	$rd_2 \mid rd_3$	$r\theta_1 \mid o_{x,y}$	$rd_6 \mid bd_5$	
C_T	$ra_6 \mid bd_3 \mid ba_5$	$rd_1 \mid o_z \mid bd_1$	$rd_2 \mid rd_3$	$r\theta_1 \mid o_{x,y}$	$rd_6 \mid bd_5$	$r\alpha_6 \mid b\alpha_5$

Table 3.2 Identifiability dependencies for the cost functions seen in Equations 2.1-2.4.

C_ℓ	$r\alpha_6$	ba_{1-5}	bd_1,bd_2,bd_4,bd_5	$b\alpha_{1-5}$	$b\theta_{1-5}$
C_P	$r\alpha_6$			$b\alpha_{3-5}$	$b\theta_{3-5}$
$C_{P\theta}$				$b\alpha_{3-5}$	$b\theta_3$
C_T					

Table 3.3 Zero Jacobians for the cost functions seen in Equations 2.1-2.4.

not always a sign that the parameter can not be identified.

3.3 Non-identifiable influence

When removing parameters to identify to make all remaining parameters identifiable, the parameters to keep were chosen in the following order.

1. $b\theta_{neededjoints}$, where $neededjoints$ are the indexes for the joints needed for the different cost functions. $b\theta_{neededjoints}$ which effectively are the encoder offsets, can not be assumed to be known.

2. Ball-bar origin ($o_{x,y,z}$). Ball-bar origin is often not a known position.

3. Robot parameters ($ra_{1-6}, rd_{1-6}, r\alpha_{1-6}, r\theta_{1-6}$), which are the parameters of interest to identify.

4. Ball-bar length (bd_3). Even if it is possible to measure the ball-bar length, it is difficult to measure it exactly.

5. Ball-bar parameters not mentioned above ($ba_{1-5}, bd_{1,2,4,5}, b\alpha_{1-5}$). These parameters can often be assumed to be nominal, since the ball-bar joint manufacturing tolerances are tight.

Following the above order the following parameters were removed from the parameters to be identified: ra_6, ba_5, rd_1, bd_1, rd_3, $r\theta_1$, bd_5, $b\alpha_2$, $b\alpha_5$. An error in one of these parameters might influence one or more of the identifiable parameters, and to test this an error of $1mm|mrad$ was introduced to the parameters, one at the time. The identified parameters were considered significantly affected if they were off by more than $0.05mm|mrad$, which is 5% of the introduced error. The introduced error of $1mm|mrad$ was motivated by being the size of expected errors

in the robot parameters and large enough that the limit of $0.05mm|mrad$ did not come close to the simulation precision. As seen in Table 3.4 most of the parameters only affect other parameters in their non-identifiable groups except for $r\theta_1$ which also influences $b\theta_1$ where applicable, and the parameters ra_6 and ba_5.
ra_6 and ba_5 are in the end the same length, which is the offset plate length, and affect most of the identified parameters. The reason for this is the scalability problem, which means that without one length known in the identification it is possible to scale all robot and ball-bar lengths without changing their proportions. This makes the identification problem unconstrained, and therefore makes the identification result unreliable [Collin, 2016].

Error	ra_6 | ba_5	rd_1 | bd_1	rd_3	$r\theta_1$	bd_5	$b\alpha_2$	$b\alpha_5$
Affected	Most parameters	o_z	rd_2	$b\theta_1, o_x, o_y$	rd_6	bd_2, bd_3	$r\alpha_6$

Table 3.4 Influenced parameters by errors in non-identifiable parameters.

3.4 Comparing simulations

For each simulation the result is given as the deviation from the actual DH parameters, which are the nominal DH parameters plus the parameter errors. An example can be seen in Table 3.5, where the non-identifiable parameters are represented with a $'-'$.

i	ra_i	rd_i	$r\alpha_i$	$r\theta_i$	ba_i	bd_i	$b\alpha_i$	$b\theta_i$	$o_{x,y,z}$
1	-0.090	-	-0.012	-	-	-	-	-	0.037
2	-0.061	-0.60	0.005	0.108	-	-	-	-	0.124
3	0.048	-	0.047	-0.022	-	-0.126	-	-	-0.030
4	-0.020	0.000	0.040	-0.007	-	-	-	-	-
5	0.038	0.011	0.001	-0.041	-	-	-	-	-
6	-	0.015	-	0.054	-	-	-	-	-

Table 3.5 Example of errors between the identified DH parameters and the actual DH parameters. The unit is mm or mrad.

To be able to in a simpler way compare different errors and cost functions against each other the Euclidean distance $|e|$ and the maximum error $\hat{e}$ were used, where the error units is either milliradians or millimeter. Since 1 mrad angle difference on a 1 m long bar is of similar size as an 1 mm difference, and most lengths on the simulated robots are in the order of 1 meter, the different errors are treated equally. The Euclidean distance is used as a generic distance to the correct solutions, and therefore can be used as a measurement of how well the identification performed. The reason for also using the maximum error is to get a estimate of how the errors are distributed; if they are evenly spread out or concentrated to one parameter.

All parameters that were identifiable were set as identifiable in the simulations, and the calculated Euclidean distance includes all parameters and not just the robot parameters. The reason is that the large errors in the ball-bar parameters can not be said to not occur on a robot parameter next time. This was found by introducing the same type of error in a simulation but with different starting random seed each time, and the parameters were arranged according to error size. The result, which partly can be found in Table B.1 in Appendix B, showed that even if some parameters occur first in the list more than others the parameters are not confined to the ball-bar parameters.

Without running verification with identified parameters and checking the position error it is difficult to exactly define what resulting parameter error is considered acceptable. But to get a hint of what is acceptable the condition that the Euclidean distance should not be larger than 0.5 mm|mrad is used. This comes from that the needed precision in robots to be able to be used in simpler manufacturing is less than 0.5 mm [Low, 2007].

To validate the error generation and the solver with the different cost functions, a simulation was done with static but without varying errors introduced, and the resulting Euclidean distances were less than 2.0 µm|µrad and the resulting maximum error was less than 1.0 µm|µrad for all cost functions. This could have been improved by using tougher convergence conditions in inverse kinematics and the parameter identification but being smaller than 1% of the size of the introduced parameter errors were considered small enough.

3.5 Simulation results

Number of points

Since the number of points used strongly influences the solving time it is desirable not to use more than needed. To test this a path was generated with the static errors in Table 3.1 and robot and ball-bar noise of size 0.010 mm|mrad. The number of points used for solving was varied and the resulting Euclidean distance and maximum error for the identified parameters can be seen in Figure 3.1. It can be seen that the difference is small when using more than 750 points, but less points fast yields a large error. For simulations and experiments 1500 points were considered a fair trade-off between identification speed and some margin against the number of points needed.

An interesting thing about the plot is the relative large difference for the error for more than 750 points, which shows that the difference between simulations may vary. This is also evident in the other simulations, where the relatively high uncertainty is seen in the identification results. It is however possible to use the simulation for identifying general trends and results.

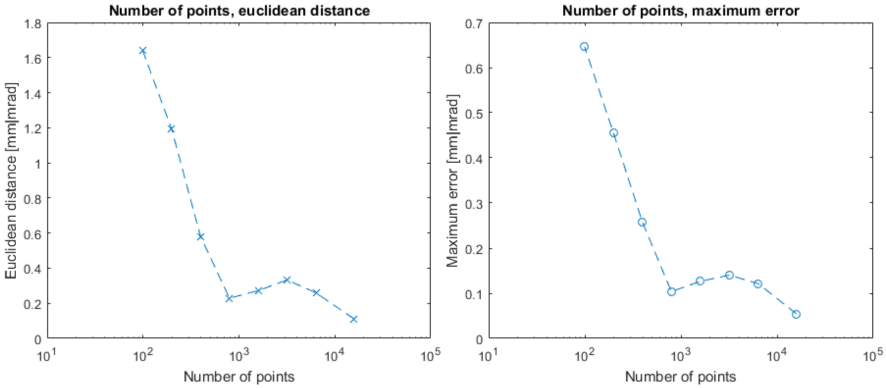


Figure 3.1 Number of points used for solver compared to the error in the identified parameters.

Robot and ball-bar joint angle noise

Simulations were done with equal amplitude error noise in both robot joint angles and ball-bar joint angles and the result can be seen in Figure 3.2. Even if the ball-bar ideally should not contain as large unmodeled errors it is interesting to compare how well the different methods handles disturbances. It can be seen that the C_ℓ is much more sensitive than the other solution methods and also that the error becomes large even for small noise amplitudes, larger even than the introduced error. It can also be seen that the aim of 0.5 mm|mrad Euclidean distance only is possible for very small amplitude noise. Comparing noise introduced only in robot joint angles and noise introduced only in ball-bar joint angles, which can be seen in Figure B.1 in the appendix, there is no greater difference in final error; and comparing to noise in both robot and ball-bar joint angles the difference is not much larger. To simplify further simulations, varying errors are therefore only applied to the robot joint angles.

Noise and other errors

Figure 3.3 shows the error types gear error, sinusoidal gear error and noise compared to each other. While the error size is not completely equivalent for the different types of error they are of comparable sizes. As for noise, the same trend can be seen for the gear error and the sinusoidal gear error, which is that the error is significantly lower for C_T than C_ℓ. The parameter identification for the cost functions C_P and $C_{P\theta}$ for these errors follow the same trend but are omitted to allow a clearer plot. Comparing the errors to each other it can be seen that they are of approximately the same size, even if gear error seems to influence the more using the C_ℓ cost function and noise seems to influence more when using the C_T (and C_P, $C_{P\theta}$) cost function(s).

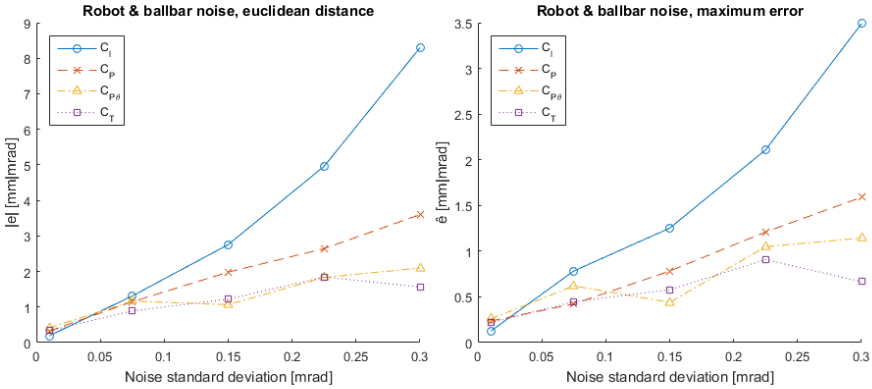


Figure 3.2 Resulting parameter identification error as result of robot and ball-bar noise.

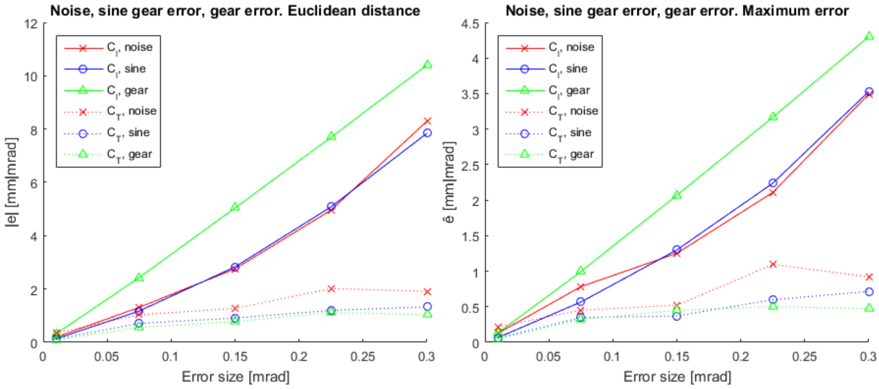


Figure 3.3 Gear error, sinusoidal gear error and noise compared.

Synchronization error

Since two logging systems, one for the robot and one for the ball-bar, need to be synchronized for experiments it is of interest to see the influence of a time difference between the logs that are combined. The result can be seen in Figure 3.4 and shows that the log synchronization is very important if the cost functions using ball-bar sensors should improve the result compared to the C_ℓ cost function.

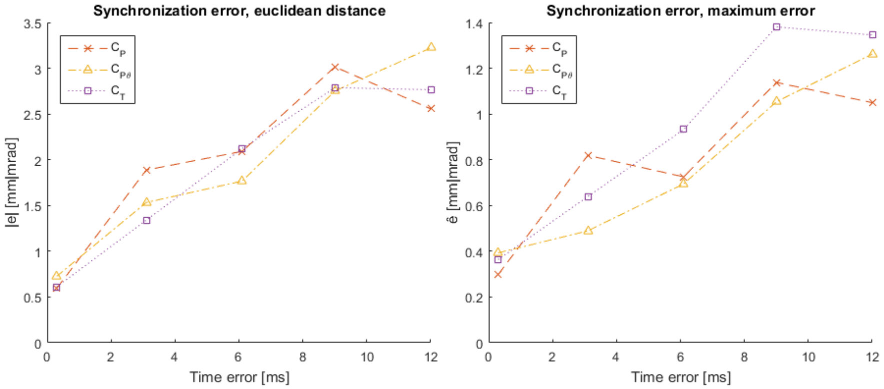


Figure 3.4 Time synchronization error

Backlash

The simulated result of introducing backlash can be seen in Figure 3.5. The impact on the final result for even small amplitude backlash is evident, even if the cost functions involving whole or part of the rotation as sub-costs produce better results. The simulation assumes pure backlash without ball-bar force dependent deviation. But the simulation result is assumed to be valid as a rough estimation of the parameter identification error coming from ball-bar force dependent deviations with similar size.

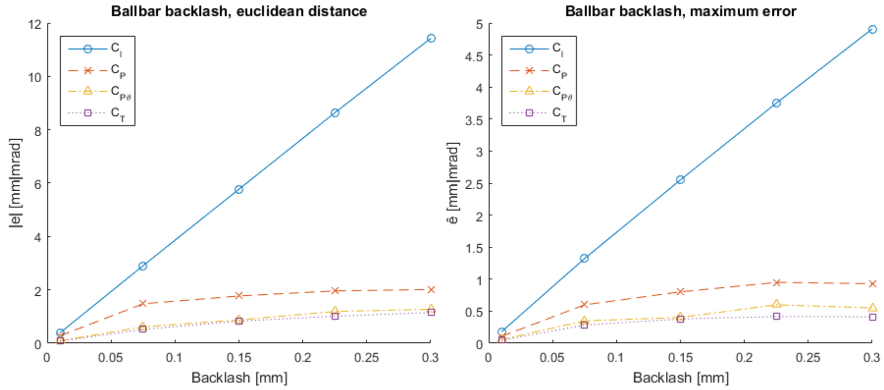


Figure 3.5 Ball-bar backlash error.

Truncated ball-bar angles

Instead of noise one simulation was done with truncated ball-bar angle values, to simulate limited resolution for the encoder. Figure 3.6 shows limited resolution compared to noise introduced in the ball-bar joints, which shows that the resulting error coming from limited encoder resolution is somewhat smaller than the error from encoder noise, but still of similar size.

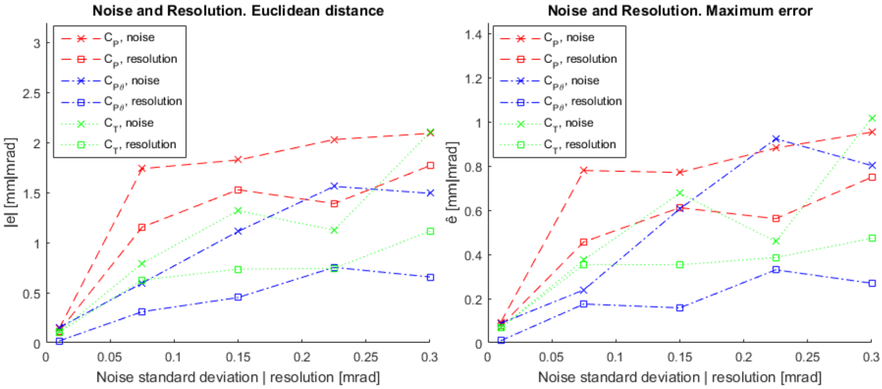


Figure 3.6 Ball-bar noise and ball-bar encoder resolution compared.

4

Hardware development

4.1 SolidWorks

The parts and drawings were designed using SolidWorks 2015, a popular 3D-CAD software. The software allows users to create 3D models of parts, insert the models into drawings for dimensioning and tolerancing, or create 3D assemblies to see how the finalized parts work together which can be seen in Figure 4.1. It is a practical tool to find out how many different parts will fit together before they are manufactured.

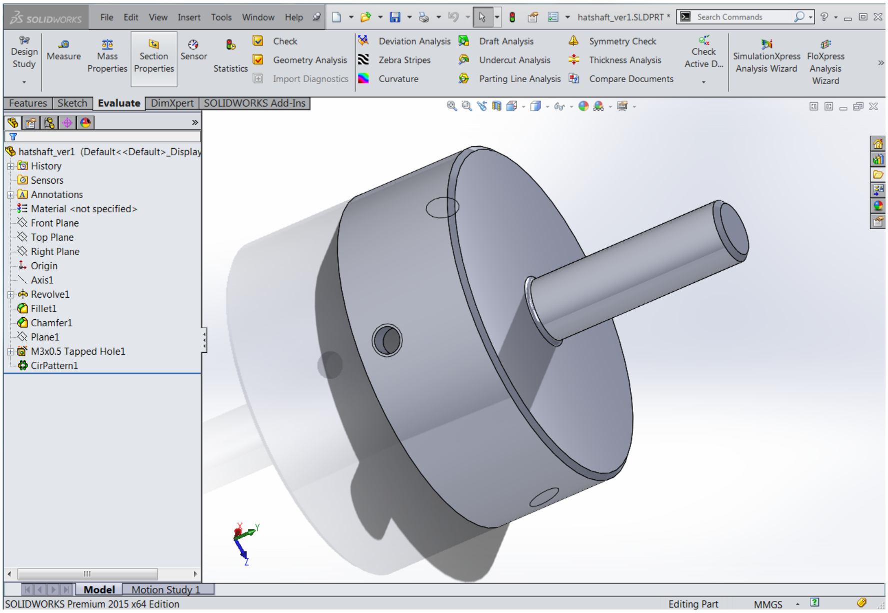


Figure 4.1 The Lock-nut adapter seen in the SolidWorks virtual workspace.

4.2 Design and mounting considerations

Designing hardware to be used for calibration comes with a number of practical considerations. The double ball-bar device is a well defined piece of equipment manufactured with low tolerances, and the hardware that connects to it needs to be equally well defined and manufactured to similar tolerances.

When it comes to mounting, every part must have a well defined mounting position. A common method of defining the mounting position in the plane parallel to the mounting surfaces is to use guiding pins in well-defined pin holes. Using one guiding pin locks the hardware in one rotational dimension in the plane, using two locks them completely in the plane. Then the two surfaces are tightened against each other using screws and locking the third dimension.

If there are any defects in the manufacturing or an error in the mounting, there is a risk the identification algorithm will be affected by that error which will result in an inaccurate calibration of some parameter.

4.3 Angle encoder considerations and choice

To measure the rotational axes of the ball-bar, rotary encoders were needed. There were a number of details to consider when picking the right encoder for this application. The varying properties of a rotary encoder can be the size of the encoder, the defined accuracy and the defined resolution.

In [Collin, 2016] the resolution of the robot joint angles is 0.01 degrees on the IRB140 robot and 0.001 to 0.005 degrees on the IRB6640 robot. The resolution for the IRB2400 robot joint angles is 0.01 degrees [*Product Specification IRB 2400*]. These numbers were used as an initial estimate for the required ball-bar encoder accuracy, with the notion that robot accuracy might be lower than the specified resolution. In the robot lab of the Automatic Control Department there were encoders that had previously been used on a parallel robot that was no longer in use. Those were Heidenhain ERN 1080 rotary encoders which can be seen in Figure 4.2, which are incremental encoders with 3600 lines per revolution and an accuracy of 18 arc seconds [Heidenhain, 2015], which corresponds to 0.005 degrees or 87 µrad. It was decided they were suitable for experiments in this thesis, since they matched the accuracy estimation.

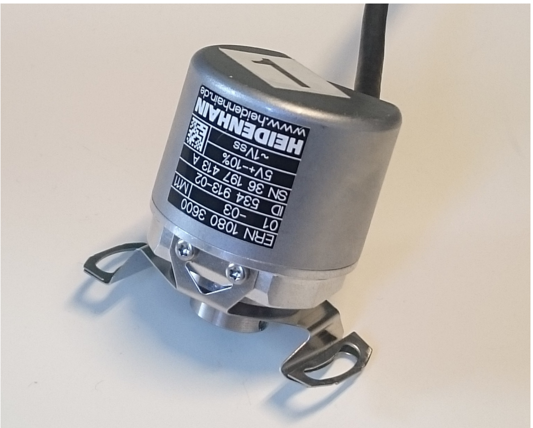


Figure 4.2 One of the Heidenhain ERN 1080 encoders used.

4.4 Encoder mounting considerations

The design of the double ball-bar device meant that there were some restrictions to where and how the encoders could be mounted. The main goal with the parts were:

- Extend the physical rotation axes so they could be measured by the encoders with relative ease.
- Easy to mount on both the double ball-bar and the encoder.
- Limited modification of the existing hardware.

One aspect that was considered when mounting the encoder was to make sure that the encoder was concentric with the axis it was mounted on. That was done with the help of its integrated bearing that helped correct alignment errors without reducing the accuracy significantly [Heidenhain, 2015].

Base rotation

To measure the center hub rotation there were two main placement options. The first option was to mount the encoder directly on the top of the rotating center hub, and extend the fork part of the gimbal joint. This had a number of advantages, mainly that only a small shaft to the encoder was needed. Another was that there was no modification done to the base and its mounting connection to the rigid environment, which was already well defined. However, the gimbal joint needed to be extended by at least the height of the encoder to make sure that the joint could rotate freely without interference from the encoder. This was not possible due to the gimbal joint already being assembled and modifications could not be done without manufacturing a completely new joint.

The second option was to design a cylindrical pedestal to mount the base on. This way the gimbal joint did not need to be modified, and the base could simply be raised up by mounting it on this new extension. The encoder would then be mounted upside down on the bottom of the base, inside the cylindrical pedestal. To transfer the rotation of the center hub to the encoder a shaft was needed. The shaft would transfer the rotation through the middle of the base where there already was a through hole, previously used for fastening the base. This option was considered to be desirable, because it required no modification of the existing hardware other than a new way to secure the base and a part to secure the shaft in its place. Ultimately this design was decided upon and made it into production.

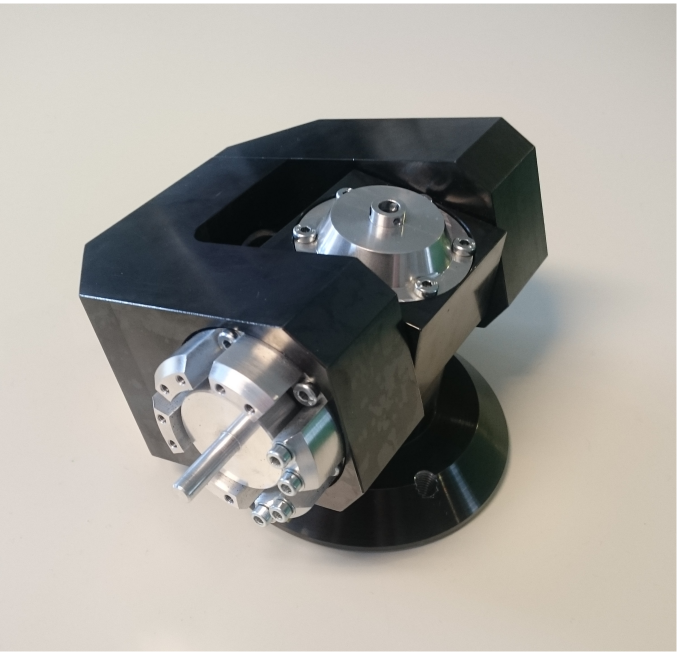


Figure 4.3 The base joints on the base-adapter with hardware mounted, excluding encoders. Not included in the image is the shaft transferring the rotation of the base to the encoder mounted on bottom. This shaft is fastened in the center of the base joint in the picture.

Fork rotation

To be able to measure the rotation of the fork part of both the gimbal joint and the cardan joint, no modifications of the joints had to be made. The encoder and the part for transferring the rotation could be mounted on the side of the joint without any interference with other parts of the ball-bar.

Rod rotation

There was very limited space on the gimbal joint where the rod was connected. To be able to fit the encoder, the protruding shaft the rod was mounted on would need to be extended which was an unwanted complication due to the already well defined mounting of the the rod. It was decided not to mount an encoder there for the experiments.

4.5 Manufactured parts

Offset plate

It was found in [Collin, 2016] that the rotation around the last robot joint, $r\theta_6$ was not identifiable. This was due to the exact alignment of joint six of the robot and joint five of the ball-bar mechanism, creating an infinite amount of solutions for the joint positions in the inverse kinematic calculation. By offsetting these two axes from one another, the problem could be avoided. It was also desirable to use the offset plate as the fixed distance used to avoid the scalability problem. In previous double ball-bar calibration the carbon-fibre rod had been used as the known constant length, but it is not always simple to measure or manufacture that component to an exact specification. That is not a problem with the offset plate as the distance between the holes can be very well defined using the correct equipment, such as a three degree-of-freedom milling machine.

The distance between the axes was specified to be 250 mm. It is a distance that does not result in large deformations during load, is large enough to make sure the angle of the sixth robot joint affects the position on the sphere, and considered sufficiently long as the scalability locking distance.

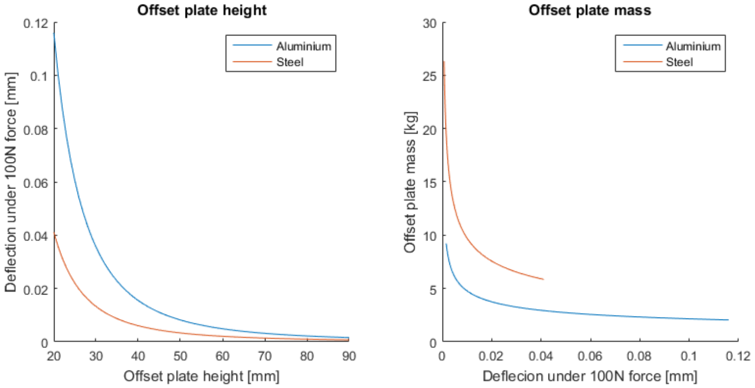


Figure 4.4 Deflection during load depending on material and thickness of the plate and the corresponding weight of the plate.

An estimation of the bending moment that the offset plate might experience was calculated to determine the maximum deflection of the plate during load. It was desirable to keep this deflection as small as possible to minimize the error that could influence the identification. The estimated force applied to the offset plate was 100 N, calculated from 50% of the handling capacity of the ABB IRB2400/16 which is 20 kg. The estimated maximum deflection allowed was 10 µrad, which is about 10% of the encoder resolution at 1 m distance and is considered small enough to not significantly affect the experiments. The deflection was calculated using deflection for a cantilever beam with uniformly distributed load and concentrated load at the free end, and the resulting plot can be seen in Figure 4.4. This showed that to keep the deflection under 10 µrad the thickness of the offset plate should be more than 45 mm and should be made using aluminium to minimize its weight.

The manufactured offset plate can be seen along with a TA20 tool changer part in Figure 4.5. Starting at the robot mounting point, there are mounting holes for a TA20 tool attachment. In the other end of the offset plate there are holes for the same guiding pins as the TA20 uses, but three M8 screws are used to fasten the plate to the base of the cardan joint. A reason for not using the four screw holes of the TA20 was that it might interfere with the mounting of the encoder. In hindsight that was not the case, and the interference could have been easily avoided by making sure there was enough clearance for the screw-heads and using a correct orientation of the encoder mounting holes.

Base-adapter

The base-adapter, seen in Figure 4.6, was designed with the primary goal to fit the encoder with a height and diameter of 42 mm. The encoder would be mounted on the existing base upside down. The base had to be fully supported by the adapter, so the diameter of the base was used as an existing dimension to base the rest of the design on. The 105 mm diameter cylinder was hollowed out to an inner diameter of 65 mm, which left enough space to fit the encoder.

To maintain the current mounting mechanism of the base on the mounting plate, slots for fitting the 6 mm bearing balls mounted to the bottom of the base were designed, as well as mounting holes for bearing balls at the same corresponding positions on the bottom of the base-adapter. That allowed the base-adapter to support the base with the encoder mounted to the bottom, as well as ensuring that the adapter was prevented from moving in any direction when running the experiments.

The base adapter was manufactured using steel, since it was a load-bearing component and should be strong enough to handle the loads exerted on it by the robot without deforming and influencing the identification.

Lock-nut adapter

The lock-nut adapter, seen in Figure 4.7 and Figure 4.3, was designed to be mounted on the lock nuts that tighten the ball-bearings on the side of the forks. Mounted

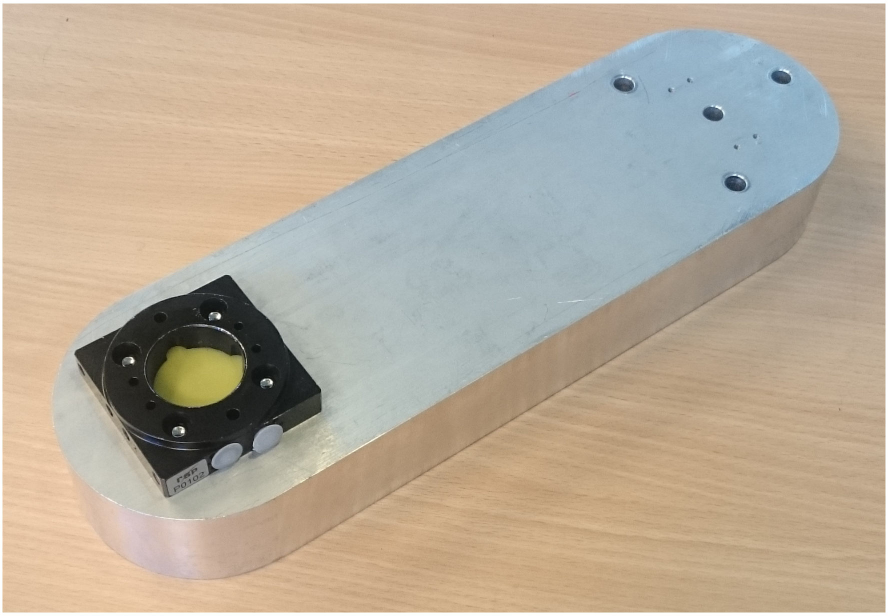


Figure 4.5 The offset plate with the TA20 tool attachment mounted.

there the adapter will extend the rotation axis of the second and the fourth joint of the ball-bar. It consisted of two parts, one protruding shaft that the angle encoder was clamped around, and one hollow cylinder that was mounted on the current lock nut with three tightening screws. The short shaft was first mounted on the encoder, and then the cylinder was slid on the lock-nut and the concentricity of the mounting was manually checked with a dial indicator and adjusted by adjusting the three screws. The lock-nut adapter was not a part that was subjected to any load, except the low starting torque of the encoder which was less than 0.001 Nm [Heidenhain, 2015], so no material strength considerations needed to be made. It was therefore manufactured in aluminium.

Adapter-ring

The adapter ring, seen in Figure 4.7 and Figure 4.3, was designed to to be mounted on the side of the fork, concetric with the stationary axis that is measured. The adapter-ring was moving relatively to the Lock-nut adapter, which was stationary when mounted. The ring was manufactured in aluminium, since it was not a part that was subjected to any load.

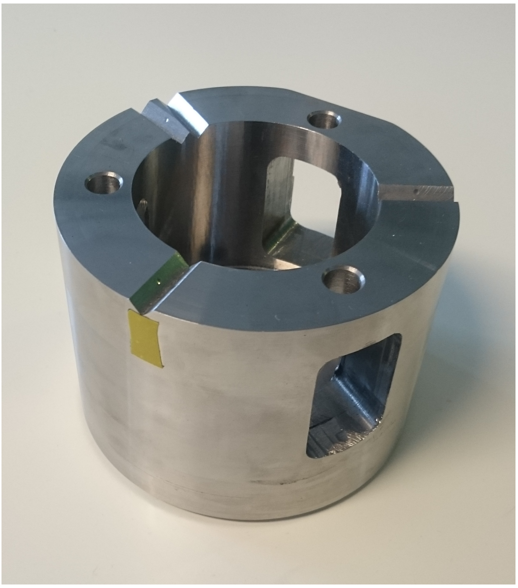


Figure 4.6 The base adapter.

Lock ring

The lock ring, seen in Figure 4.7 and Figure 4.3, was mounted on top of the center hub, in a limited space with the fork rotating right above it. The lock ring was used as a mounting point for the shaft that transferred the rotation through the base. The lock ring was also manufactured in aluminium, since it was not a part that was subjected to any load.

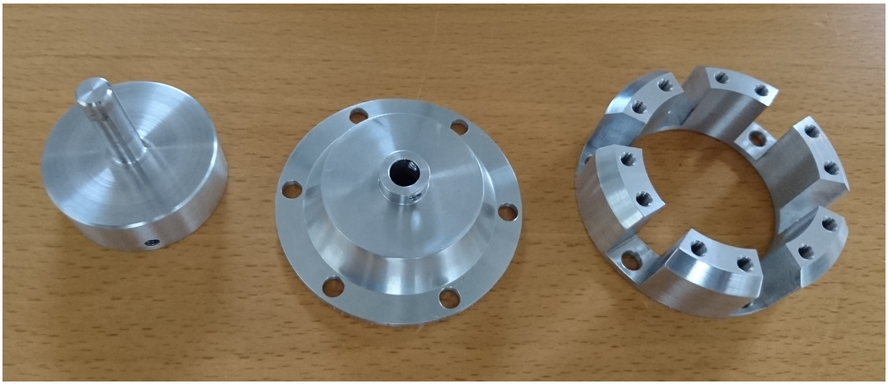


Figure 4.7 From left to right: lock-nut adapter, lock ring and adapter-ring.

4.6 Encoder interfacing

The encoder signals were analog 1 Volt peak-to-peak complementary signals. There were three complementary signals coming from the encoder; sine, cosine, and reference signal to determine the zero position of the encoder. To measure the signals and determine the angular position, Beckhoff EL5021 SinCos Encoder Interface Terminals were used. They were coupled to a Beckhoff EK1100 EtherCAT Bus Coupler which handles the EtherCAT connection to a PC. The EL5021 have the possibility to interpolate the signals to improve the resolution of the encoder. Depending on the speed of the rotating shaft up to 13 bits interpolation, 8 192 steps within each signal period, was possible. It was discovered during the development of the logging program that when 13 bits interpolation was used, encoder value fluctuations could be seen in the program even when the encoder shaft was stationary. Due to this noise, 12 bits interpolation was used when the experiments were run. The encoder interfacing units can be seen in Figure 4.8.

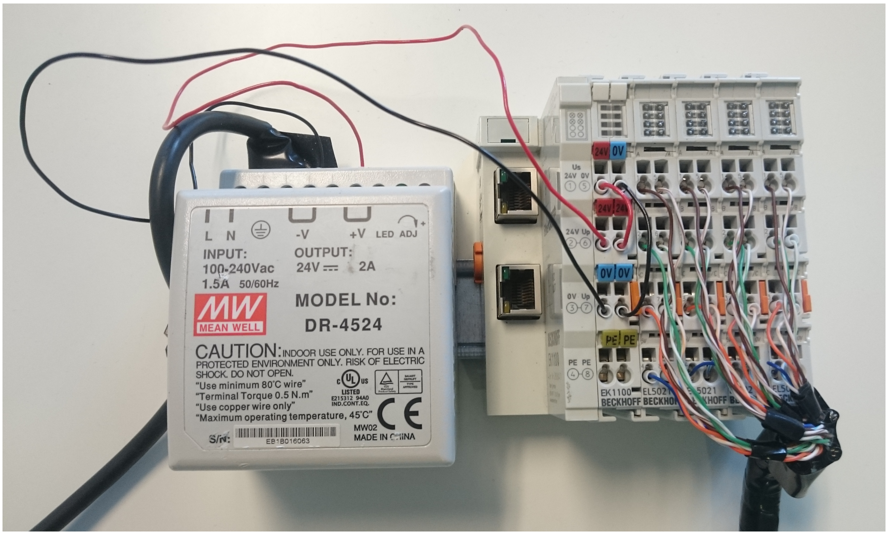
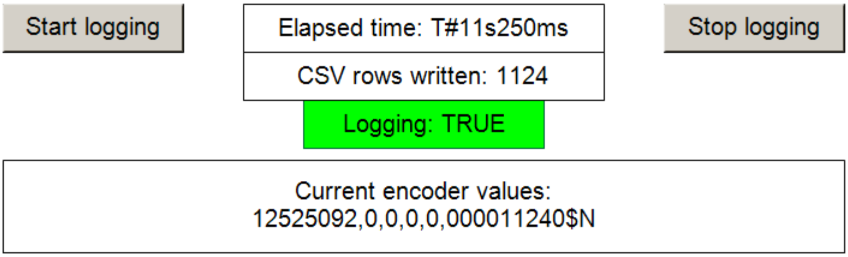


Figure 4.8 Power supply, EtherCAT Bus Coupler and four SinCos Interface Terminals.

Logging software

To log the encoder values and save them to file, a small program was developed. The program uses the EtherCAT 3 protocol to interface with the EtherCAT Terminals and was developed using TwinCAT 3 software for Windows.

The program was written in the Structured Text language, which is a language designed to work on PLCs, programmable logic controllers [IEC 61131-3:2013, 2013]. To simplify interfacing with the logging program a simple GUI was developed which can be seen in Figure 4.9.



Elapsed time: T#11s250ms

CSV rows written: 1124

Logging: TRUE

Current encoder values:
12525092,0,0,0,0,000011240$N

Figure 4.9 The logging software visualization during logging of one encoder.

Program design

The logging program was designed as a state machine. The states and what they did were built up as follows:

1. The first state is simply a wait state. Here the program waits for input from the operator for when the program should start logging.
2. The second state initialises the file writing by creating a timestamp of the starting time, creates a new file and opens that file for write access.
3. In the third state a line of all the encoder values available, separated by commas, are created. This line is written to file.
4. This state is a wait state for the file access, it waits until the buffer has finished writing the line created in the previous state. The state machine returns to state 3 as long as the logging should continue.
5. The fifth state is reached only when there is input from the operator to stop the logging of encoder values. This state handles the closing of the file access and then returns to state 1.

To minimize jitter and to make sure the program is running consistently in real-time, TwinCAT was configured to run on a single dedicated core of the processor, with the Intel Hyper-Threading Technology turned off.

The sampling frequency was 100 Hz. To save the counter values, the output file format .csv was used. Csv is an abbreviation that stands for comma-separated values and is a format used to store tabular data. Each line written to this file contained all the counter values and the timestamp when the sampling occurred. There were occasional overruns seen in the log file where state four wait for file writing to finish state, ran for two clock cycles instead of one, so the next sample was taken 5 ms later.

Since the timestamps were included in the ball-bar log file overruns did not influence the log synchronization against the robot log, which also contained timestamps for each measurement. The log synchronization might however been influenced by any clock drift present in either the robot logging or the ball-bar logging.

5

Experiments

5.1 Path generation

The experiments were done to evaluate the kinematic parameter identification using ball-bar angles and to get an estimate of the size of errors that occurs. The robots used in the experiment were the ABB IRB2400 and Comau NJ220-2.7. Structurally they are both of the same design; articulated robot with 6 degrees-of-freedom, a parallel rod between the second and third joint, and a 3 degree-of-freedom wrist. For the tests on the ABB IRB2400 the program RobotStudio [*RobotStudio*] was used to generate paths and RAPID-code for controlling the robot. For the tests on the Comau NJ220-2.7 a program developed by Cognibotics was used to generate the paths.

The spherical path generation was limited by four factors: the different available rod-lengths, the placement of the ball-bar origin, the reach of the robot, and the cable routing of the encoders mounted close to the robot. The two first factors completely define the position and size of the spherical surface that limits the movement of the robot. They are not always free to choose, which may complicate the path generation. Since the ball-bar encoders in the experiments had their cables attached on the outside of the ball-bar they had a limiting effect on the ball-bar joints. Even if the cables used in experiments allowed for ±360° movement it turned out to be a problem for some generated paths. Another consideration is ball-bar singularities, which occur when two or three axes are aligned and can be rotated around each other, which can result in very large and fast rotations. These singularities should be avoided to not damage the ball-bar or the robot.

The paths that were used tried to maximize the area of the sphere so that all the joints of the robot were sufficiently excited. To successfully be able to identify kinematic parameters, sufficient robot joint excitation is necessary. Two of the paths that were usable are seen in Figure 5.1, where path 2 have more sphere coverage. Both paths were used for experiments and path 2 was used in simulations.

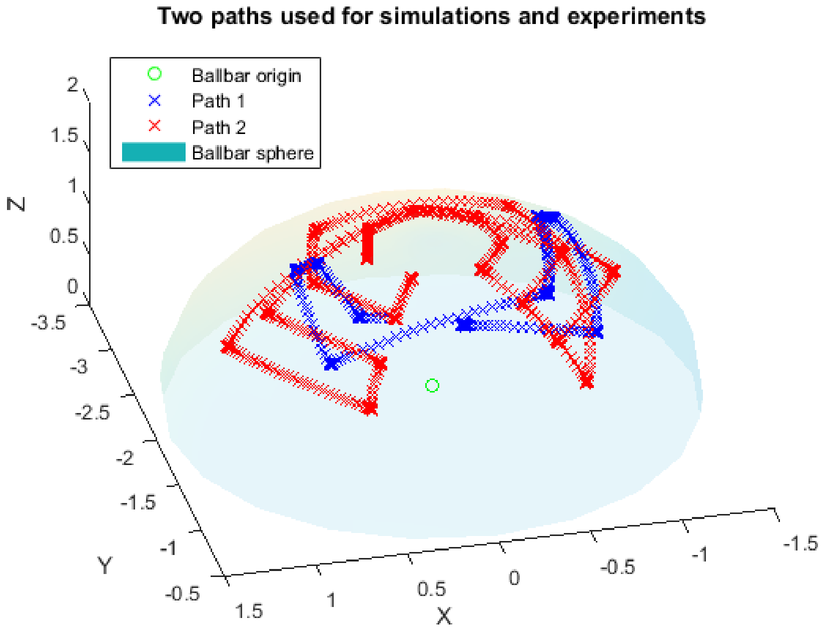


Figure 5.1 Two of the paths used for experiments, Path 2 was also used for simulations.

5.2 Measurement data processing

Robot joint angles

For the ABB IRB2400 there was an external logging system that logged the raw joint motor values, which were given in radians, from the motor encoders. These values were adjusted for motor offsets and multiplied with the transmission matrix for the IRB2400, which is a matrix containing gear ratios for every motor and joint angle dependencies if other joints influence the final joint value in any way. The transmission matrix can be found in Appendix A.

For the Comau NJ 220 - 2.7 the logging was done in the robot controller and the logged values were the joint values and not the raw motor encoder values. The logged joint values were however calculated for another set of DH parameter describing the same geometry. To be able to use the joint values without changing many of the DH parameters some joint angle signs were changed. Also the third joint angle values were compensated for influence from the second joint angle, coming from the design with the parallel rod. The relationships between logged joint values and joint values used for parameter identification are found in Appendix A.

Ball-bar joint angles

The measurements for the ball-bar joint angles were logged as counter values from the Beckhoff SinCos Interface units, which in the log handling were converted into joints radians. Since the starting encoder offset was not known an initial guess was manually found by calculating the forward kinematics from the robot and then the inverse kinematics for the ball-bar. The read angles were then offset until they matched the calculated values as good as possible for a few points equally spread along the path. The resulting angle errors between the calculated angles and the read angles were often less than ±5 mrad.

Since the ball-bar has multiple solutions, describing different configurations for the inverse kinematics, it turned out to be a bit of a challenge; especially for the cost function $C_{P\theta}$ which uses inverse kinematics. To solve this the inverse kinematics always assumed the ball-bar to be in the first configuration, and encoder angles from experiments with a ball-bar in any other configuration were converted to first configuration joint values. The ball-bar configuration in the experiments depended both on the starting condition and also on whether the ball-bar during the experiment passed through a singularity.

Filtering joint values

To find the amount of signal noise that was present on the measured joint angles, a part of a log with the ball-bar and robot standing completely still with the brakes applied was used. The joint signal jitter can be seen in Figure 5.2. To remove this jitter and to remove possible unwanted spikes in the measurements the joint signals were filtered with a moving average filter, with a filter period of 100 ms.

This period time was assumed to be a large enough value to remove the largest spikes and small enough value to preserve relatively fast changes in the joints, based on visual inspection of the logs.

Log synchronization

The synchronization of the robot and the ball-bar logs for the joint values was done in three steps. The first step was to roughly synchronize the logs in the range of ±100 ms. In this step the initial encoder offsets were not calculated, which meant that any kind of comparison of transformation matrices was not possible. Instead the joint values were differentiated to find points along the path where all joint speeds were close to zero. This way the rough synchronization was done based on points where the robot, and therefore also the ball-bar, was standing still or almost standing still. The resulting plot used for rough synchronization can be seen in Figure 5.3, where 1 means that all robot joints are moving with a speed of less than 10^{-4} rad/s and -1 means that all ball-bar joints are moving with a speed of less than 10^{-4} rad/s. This value was found to be a suitable value for enough, but not excessively many, synchronization lines in the plots. The resulting plots do not match up perfectly, but

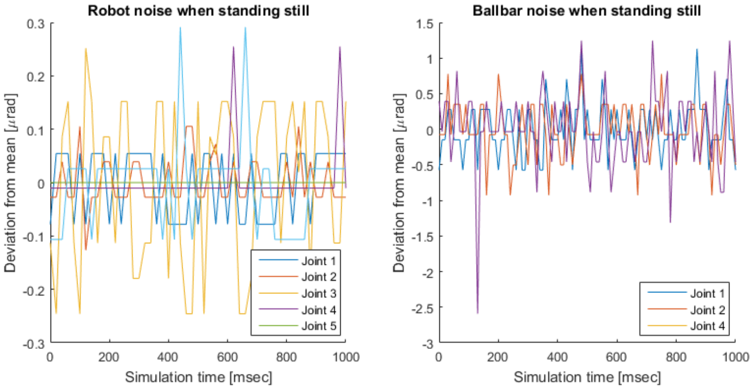


Figure 5.2 The jitter seen in the robot and ball-bar angles when both were stationary (brakes activated).

good enough to find the time the ball-bar logs needed to be shifted to match the robot log. The plots seen in the Figure are plots after finding the time difference.

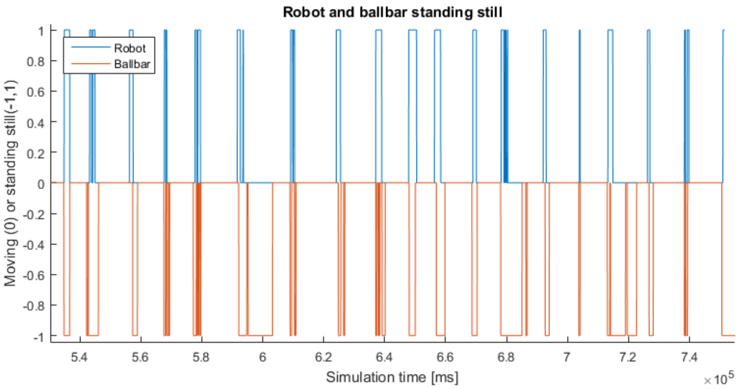


Figure 5.3 Differentiated robot and ball-bar angles. Plotting when all robot or ball-bar angles are less than 10^{-4}. The plot is an excerpt of the full path.

When the rough synchronization was done the initial ball-bar encoder offsets were found. These were then used to remove outliers in the logs, by calculating the cost based on the cost functions used for parameter identification. To avoid log entries near singularities, when the robot is standing still for a long time and other unwanted entries, the logs were trimmed which can be seen in Figure 5.4.

Using the cost function but only varying the synchronizing time difference between the robot and ball-bar logs, with interpolation of the ball-bar angles

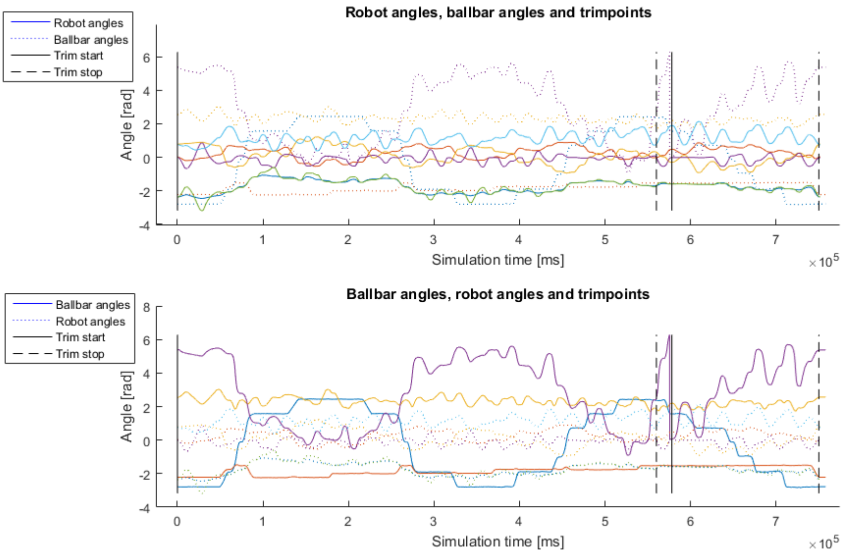


Figure 5.4 Trimming of log files; both robot joint angles and ball-bar joint angles are shown.

for higher time resolution, the cost could be plotted against the introduced time difference. As seen in Figure 5.5 it is possible to find a minimum for the time difference, and also that changing the synchronization more than 200 ms might not give good results. One of the problems with this method can however be seen by running two fine-tuning synchronizations after each other, where the time difference is changed according to the found minimum after the first run. The second run, with the now changed time difference, should ideally have its minimum at 0. However for the experiment used for Figure 5.5 the second run has its minimum at 8 ms. Even when repeating the process with more iterations the resulting minimum will vary about ±10 ms, which is considered the accuracy of this synchronization method.

5.3 Path forces and sphere deviations

One of the tests done with the ABB IRB2400 was to measure how much force the ball-bar is exposed to during a path and to measure the resulting sphere deviations. The setup can be seen in Figure 5.6 where a JR3 force-/torque-sensor used can be seen attached to the robot flange, and where the small orange and yellow spots are infra-red LEDs for measuring the deformation. The LEDs are connected to a Nikon K600-system which is a system for 3D coordinate measuring with an accuracy of <90 µm [*K-Series Optical CMM solutions*].

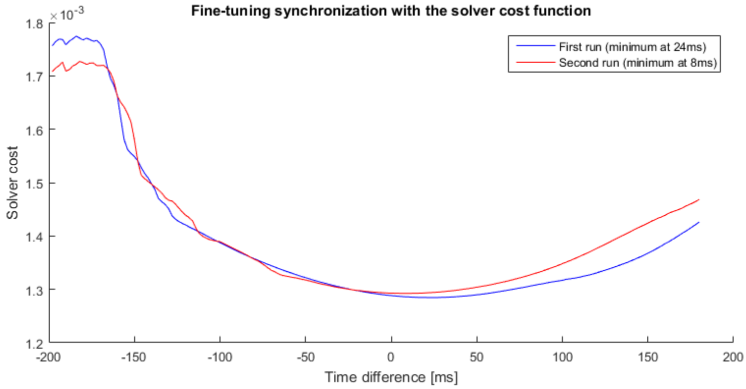


Figure 5.5 Cost as function of synchronization time.

Forces during an experiment

To estimate the ball-bar forces during a parameter identification experiment a path was generated for the ABB IRB2400. The path was performed with a SoftServo-value of 80% and with the JR3 force-torque sensor attached between the robot flange and the tool changer. SoftServo can be set to any value between 0% and 100% and is a ABB command that makes the robot compliant, in the way that joint torques are proportional to the deviation from the reference point. This is is done by removing the integral part of the robot joint PID-controller and changing the proportional value depending on SoftServo setting, where 100% is most compliance and means the lowest proportional factor. The main reason for using SoftServo is to avoid damaging the robot or ball-bar when there are differences between the generated reference path and the real path. The secondary reason is to minimize the ball-bar forces during a run, to minimize the sphere deviation. The resulting forces for the generated path can be seen in Figure 5.7. To be noted is that the path run contained many points close to ball-bar singularities, where the ball-bar force is increased compared to points further away from the singularities. This can also be seen in the force oscillations in some parts of the path.

Sphere deviations

Estimation of sphere deviations was done both by experiments measuring ball-bar strain and backlash, and by calculating the theoretical strain for the ball-bar. The reason for experiments measuring the backlash but no calculations was that the joints are ideally backlash-free, but if they are damaged a resulting non-zero backlash is possible. The possibility of backlash was evident from one of the older joints available that had an evident backlash.

The ball-bar strain was experimentally measured by using the robot to pull on the ball-bar and the backlash was measured by using the robot to both push and pull

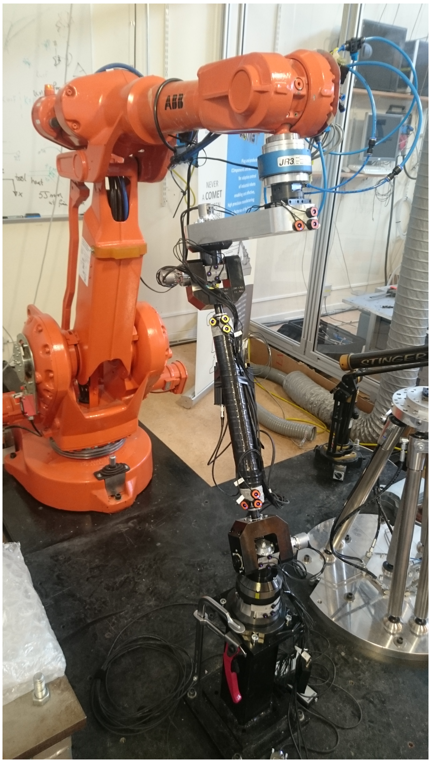


Figure 5.6 The ABB IRB 2400 robot with force-torque sensor, offset plate and the double ball-bar mounted.

on the ball-bar, while length differences in the ball-bar rod and joints was measured. The result can be seen in Figure 5.8 and Figure 5.9, which show that the estimated strain for the rod as well as each joint is less than 5 µm for a force of 150 N and the joint backlash is less than 5 µm. The result is however an estimation done with the assumption that the measurement accuracy is better for small deviations than the absolute accuracy of the Nikon system, and can therefore not be completely trusted.

The strain for the rod was calculated to 2.7 µm which was based on a 0.50 m carbon fiber rod. The strain for the joints was based on previous measurements of the ball-bearing strain performed by Cognibotics and the strain for the main parts of the joint, which summed becomes 2.8 µm for a force of 150 N.

Adding the rod and joints together results in a total sphere deviation of ±25 µm using the experimental results, and ±8.3 µm for the calculated results, based on the forces measured in Figure 5.7. Considering the simulations with ball-bar backlash ±25 µm is not a major problem when using the cost functions C_T and $C_{P\theta}$ but for the other cost functions, and especially C_ℓ, even ±8.3 µm might reduce the parameter

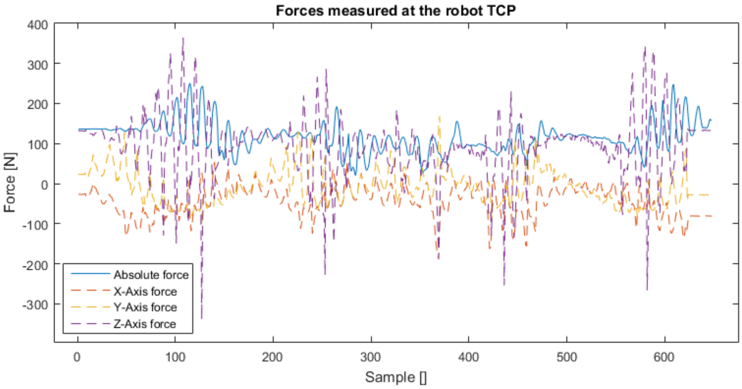


Figure 5.7 Excerpt of forces for a path run with the IRB2400.

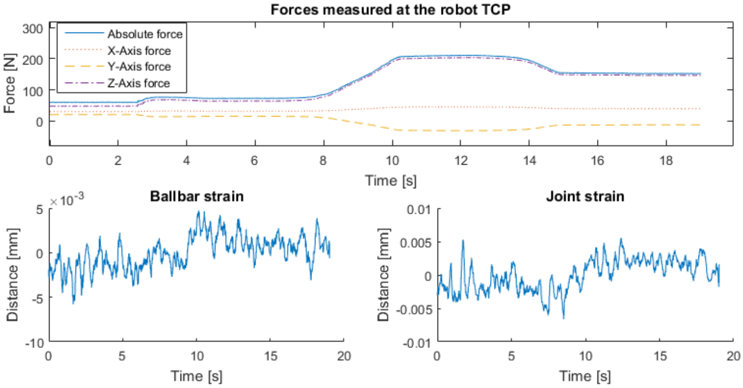


Figure 5.8 Ball-bar base movement, joint strain and rod strain when applying force to the ball-bar.

identification performance, depending on the size of other unmodeled errors.

5.4 Parameter identification

The parameter identification experiments were done to test various different cost functions against each other. To simplify comparison the same parameters were set as identifiable for all cost functions, with the motivation that most parameters that were identifiable using C_P and $C_{P\theta}$ but not using C_ℓ were ball-bar parameters that were assumed to be known, due to the tolerances used in ball-bar production. Since the angle for the third ball-bar angle was not measured in the experiments

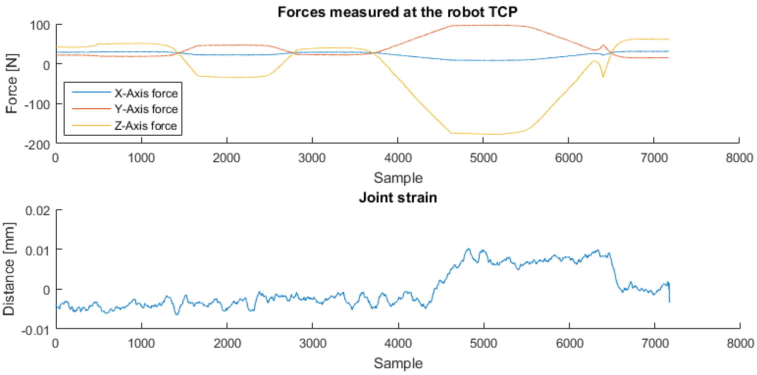


Figure 5.9 Ball-bar joint backlash when changing ball-bar force direction.

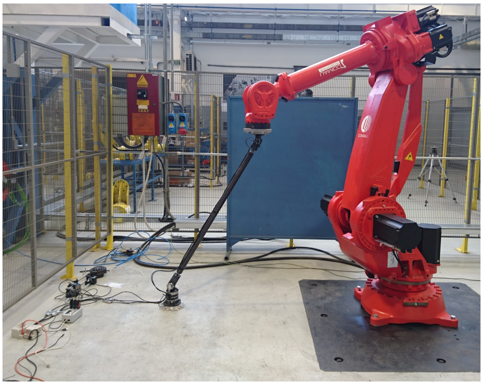


Figure 5.10 The experimental setup for experiments using Comau NJ220-2.7 in the Comau factory in Turin, Italy.

only parameter identification using the cost functions C_ℓ, C_P and $C_{P\theta}$ could be performed. The logged angle values were for the experiments not compensated for joint backlash or robot deformation under load, which therefore becomes an an additional varying error source that might influence the parameter identification result.

Since the actual DH parameters for the robot probably differ from the nominal DH parameters and the error between them were not completely known there was some uncertainty when comparing two parameter identification results against each other. The parameter errors were assumed to be less than 2 mm|mrad, and identified errors much larger than that was assumed to be due to bad performance in the identification. To be able to compare the size of the identified errors the Euclidean

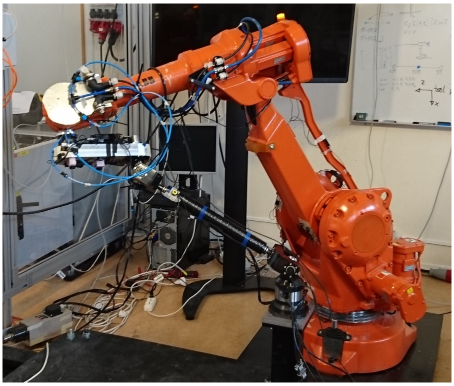


Figure 5.11 The experimental setup for experiments using ABB IRB2400/16 in the Robot Lab, Lund.

distance and maximum error compared to the nominal DH parameters were also calculated for the identified errors for the experiments. Since the initial ball-bar origin and the ball-bar joint angle offsets were rough estimates, and the number of identifiable ball-bar joint angles depend on the cost functions used, these were not included when calculating the Euclidean distance or the maximum error.

An overview of the setup can be seen in Figure 5.10 for the Comau NJ220-2.7 experiments and in Figure 5.11 for the ABB IRB2400/16 experiments.

Comau NJ220-2.7

For two of the paths the identified parameter errors using the cost functions C_ℓ, C_P and $C_{P\theta}$ can be seen in Figures 5.12, 5.13 and 5.14, respectively. It can be seen that the identified errors are large for all cost functions, and that the result for two different paths are not consistent. Especially the ball-bar length errors are large which can be seen as a sign that the minimum found, if found, is not the correct solution. Comparing the result from two experiments using the same path the identified errors are consistent, which can be seen as some indication that the solutions are fairly robust, even if the minimum found is not correct.

Any improvement by using ball-bar angles in the cost that was shown in the simulations can not be seen in these experiments, while it is possible to see the problem with unmodeled errors affecting the parameter identification using only ball-bar length in the cost function. The problems using the C_P and $C_{P\theta}$ cost functions could also be seen by the slow convergence for the solver, which was stopped after 500 solver iterations. For comparison all simulations except simulations with synchronization errors converged in less than 50 solver iterations.

However during and after the tests several problems with the setup were discovered. The offset plate used was not designed for low deflection under force; with a calculated value of 60 µm for a force of 100 N. This would, considering

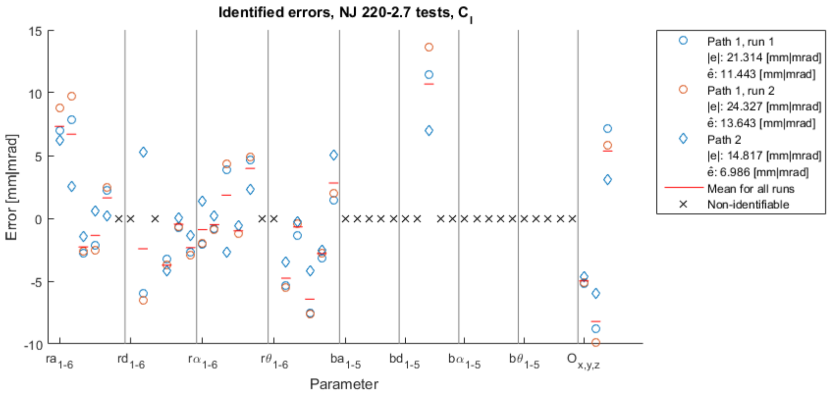


Figure 5.12 Identified errors using C_ℓ for different robot paths, of which two are identical but captured during two different experiments.

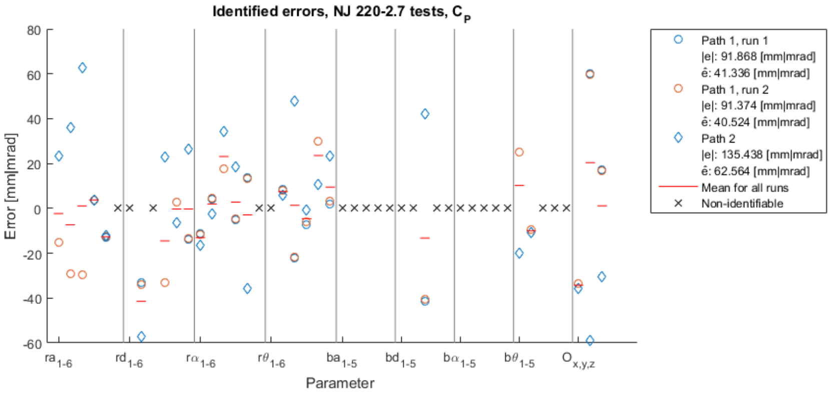


Figure 5.13 Identified errors using C_P for different robot paths, of which two are identical but captured during two different experiments.

the simulations for ball-bar backlash, cause significant performance loss in the parameter identification, especially for the C_ℓ and C_P cost functions. Considering the precision $\pm 10\,\text{ms}$ for the log synchronization, any improvement by using C_P or $C_{P\theta}$ instead of C_ℓ as cost function might be very small or non-existing. Along with a robot previously used for endurance testing and the last ball-bar sensor that can not be completely trusted due to unsatisfactory fastening, the parameter identification would have been significantly affected.

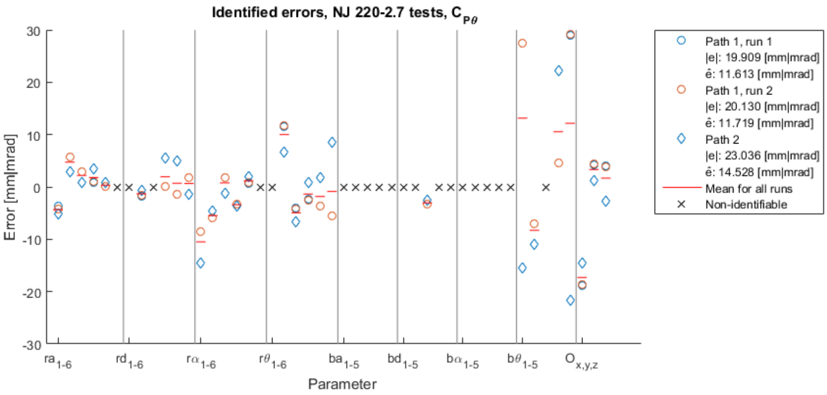


Figure 5.14 Identified errors using $C_{P\theta}$ for different robot paths, of which two are identical but captured during two different experiments.

ABB IRB2400/16

For two of the paths used in the ABB-experiments the identified parameter errors can be seen for the cost functions C_ℓ, C_P and $C_{P\theta}$ in Figures 5.15, 5.16 and 5.17, respectively. Compared to the Comau experiments the errors identified by the C_ℓ cost function are larger. One possible explanation to this is the slower movement speed used for the Comau experiments, and that the excitation of the robot joints were somewhat larger for the Comau experiments. The errors identified using the cost functions C_P and $C_{P\theta}$ are however smaller, both compared to the C_ℓ result and the Comau experiments, which can be seen as an improvement in the parameter identification similar to the improvement seen in the simulations. The errors are however still larger than expected and are not consistent between two paths, even using the same path in two experiments. One possible reason to this is that even if the same path was used for two experiments they were run with different robot speeds, which might have influenced the unmodeled errors coming from robot deformation.

Compared to the Comau experiments the ABB experiments did not have as many known problems with the setup and the parameter identification using C_P and $C_{P\theta}$ did show some improvement in the parameter identification compared to the C_ℓ cost function.

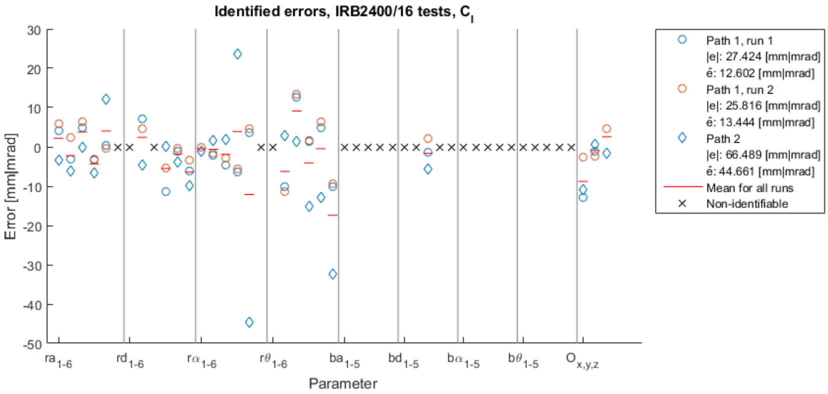


Figure 5.15 Identified errors using C_ℓ for different robot paths, of which two are identical but run with different speeds.

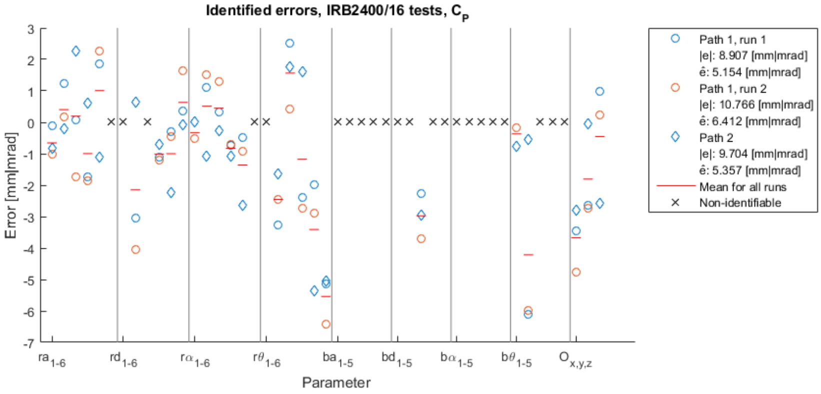


Figure 5.16 Identified errors using C_P for different robot paths, of which two are identical but run with different speeds.

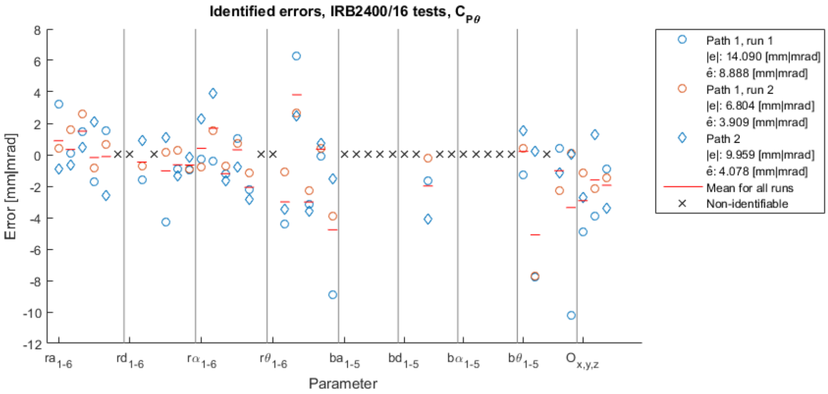


Figure 5.17 Identified errors using $C_{P\theta}$ for different robot paths, of which two are identical but run with different speeds.

6

Discussion and Conclusion

Assuming that the robot and ball-bar logging systems can be synchronized satisfactorily it is possible to improve the robustness of double ball-bar based robot calibration by using fairly cheap ball-bar joint encoders. This is however only possible when using whole or part of the ball-bar rotation in the cost function, where the accuracy needed is estimated to be better than $75\mu rad$ to be able to get acceptable results. In comparison sensors used for the experiments are considered fairly cheap, and they had a specified accuracy of $87\mu rad$, which can be improved by calibrating the sensors. However, for identification using only the two base ball-bar encoders the accuracy needed is much higher to be able to get an acceptable result, as well as using only two ball-bar angles also does not perform as well for certain types of errors such as as ball-bar sphere deviations.

One error source that was found to influence the parameter identification using C_ℓ, which also is a plausible error that influences the robot path, is ball-bar sphere deviations. Regarding the large forces needed even for small ball-bar deformations and that the sign of the force might be enough to improve the parameter identification significantly, the requirements on an acceptable force sensor is not very tough. It might even be enough to estimate the ball-bar force using the robot's joint torques. The estimated improvement by using a force sensor is however much lower when using the cost functions $C_{P\theta}$ and C_T which are less sensitive for ball-bar deviations.

The hardware developed for experiments worked as intended while being relatively simple to produce, and requiring only small changes to existing ball-bar joints. The sensor on the third ball-bar joint is however considered more important than initially thought which along with the mounting calibration needed for the lock-nut adapter is considered the main drawbacks with the hardware designed.

When evaluating the solver and the cost functions, it is quite clear that four or five ball-bar encoders are preferred. Even if parameter identification using four and five ball-bar encoders show similar results in the simulations, it is of interest to use five encoders for experiments to avoid the poor robustness near ball-bar singularities for the $C_{P\theta}$ cost function. The C_T cost function also showed an increase in the number of identifiable robot parameters.

The slow convergence in the Comau experiments and poor results for even small synchronization errors seen in the simulations however shows that the used cost functions together with the Levenberg-Marquardt algorithm might not be good for parameter identification on a real robot. And even if the ABB experiments showed an decrease in the errors identified when using the $C_{P\theta}$ over the C_{ℓ} cost function the identified errors were still large compared to the expected errors. The method in this report should therefore rather be taken as a proof of concept for using ball-bar sensors to lower the sensitivity for unmodeled errors, than a description of a working method.

7

Ongoing and Future work

During the completion of this report several areas has been worked on in order to improve the calibration results. The work has been focused on improving the log synchronization, changing the mathematical convention used to describe the robot, and improving the robot joint model. Future work includes further work on these areas as well as re-evaluating the minimization algorithm and improving setup for the experiments.

Log synchronization is one of the main problems that have to be solved to be able to use the ball-bar angles in the robot parameter identification. Ideally all logging of encoder and robot angles should be done on the same computer via direct access so that all values are logged at the same instant. This way the unknown time between the start of the encoder logging and the robot joint logging can be avoided. An alternative to this method could be to use a different robot path, with short breaks in the robot movement in the different positions that can be used as synchronisation points.

The mathematical convention used to describe the robot is of interest to re-evaluate. Partly since an error in a not identifiable parameter influences some identifiable parameters, and partly since the mathematical singularities present in the Denavit-Hartenberg convention might influence the convergence behaviour negatively. Non-identifiable parameters might, with a change to a different convention, become identifiable and result in a more complete calibration of the robot kinematic parameters. Another method to increase the total number of identifiable parameters would be to identify the motor offsets separately using other techniques, which also would solve the problem with them being difficult to identify.

To enable the use of the C_T cost function for experiments a fifth encoder should be added to the rod, which also might improve the convergence of the identified values. A calibration of the encoders to improve the angular accuracy could also be done. This calibration would increase the quality of the data that is collected and hopefully improve the results of the identification. Another step to increase the quality of the data could be to extend the robot joint models, to be able filter out joint deformation under load and joint gear backlash from the logged joint values.

Cost functions and the minimization algorithm are interesting areas to evaluate further, to improve the convergence speed and reduce the influence of outliers. It might also be of interest to evaluate algorithms that tries to find the global minima, where one example would be RANSAC [*Robust Estimation: RANSAC*]. The algorithm and cost functions should for improved simulation speed, which allows for more thorough simulations, also be written in another environment than Matlab.

For future experiments it is of interest to calibrate the robot with other methods to get a better reference of what result that is expected. Another interesting aspect that can be improved regarding the experiments is to evaluate if filtering out specific points, for example where the robot is standing still or moving with constant speed, improves the result with the identification algorithm. Specific for experiments on the Comau NJ 220-2.7 it is of interest to manufacture a new offset plate to improve the bending stiffness, change the mounting of the tool attachment to the correct side of the offset plate, and to create a mounting mechanism for the encoder. There is also a need to add guiding pins to both the offset plate mounting and the tool changer on the robot flange to correctly define the mounting position.

A

Robot descriptions

ABB IRB 2400/16

The ABB IRB 2400/16 is a medium-sized serial robot with a parallel rod and a three degree-of-freedom wrist joint. It has a maximum handling capacity of 16 kg and a maximum reach of 1.55 meters [*Product Specification IRB 2400*].

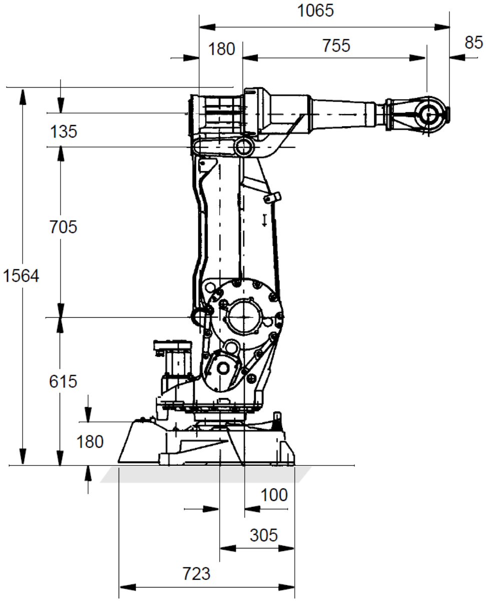


Figure A.1 Overview of the ABB IRB 2400/16 robot [*Product Specification IRB 2400*].

	a [m]	α [rad]	d [m]	θ [rad]
Link 1	0.100	$\pi/2$	0.615	θ_1
Link 2	0.705	0	0.0	$\theta_2 - \pi/2$
Link 3	0.135	$-\pi/2$	0.0	θ_3
Link 4	0.0	$\pi/2$	0.755	$\theta_4 + \pi$
Link 5	0.0	$-\pi/2$	0.0	θ_5
Link 6	0.0	0	0.085	θ_6

Table A.1 Denavit-Hartenberg parameters for the ABB IRB 2400/16 robot.

Transmission matrix

The transmission matrix for the IRB2400 can be seen below, and is to be multiplied according to $\theta_{joints} = \theta_{motors} \cdot T$. As can be seen, not only motor six but also motor four and five influence the actual angle of joint six. The influence of the parallel rod on joint three can also be seen in the matrix.

$$T = \begin{pmatrix} -5/648 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100/12999 & -100/12999 & 0 & 0 & 0 \\ 0 & 0 & -5/648 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1/55 & -1/759 & -62/41745 \\ 0 & 0 & 0 & 0 & -5/276 & -5/552 \\ 0 & 0 & 0 & 0 & 0 & -1/44 \end{pmatrix}$$

Comau NJ 220 - 2.7

The Comau NJ 220 - 2.7 is a large 6 DOF industrial robot with a parallel rod and a 3 DOF wrist joint. It has a maximum handling capacity of 220 kg and a maximum reach of 2.7 meters [*Working Areas NJ 200 - 2.7*]. It is designed for spot welding, material handling and assembly operations.

DH parameters

	a [m]	α [rad]	d [m]	θ [rad]
Link 1	0.400	$-\pi/2$	0.830	θ_1
Link 2	1.175	0	0.0	$\theta_2 - \pi/2$
Link 3	0.250	$-\pi/2$	0.0	θ_3
Link 4	0.0	$-\pi/2$	1.125	$\theta_4 + \pi$
Link 5	0.0	$\pi/2$	0.0	θ_5
Link 6	0.0	0	0.230	θ_6

Table A.2 Denavit-Hartenberg parameters for the Comau NJ 220 - 2.7 robot.

Logged angle modification

The program used for logging uses the transmission matrix for the Comau NJ-200 internally, but to be able to be able to use the logged joint angle values with the DH parameter representation chosen the logged values had to be modified according to the relations below. Note that the third joint on the robot is controlled with a parallel bar which makes it necessary to subtract the joint angle for the second joint. θ_i are the DH parameter compliant angles and the $\theta_{i,log}$ are the logged joint values.

$$\theta_1 = -\theta_{1,log} \cdot \pi/180$$
$$\theta_2 = \theta_{2,log} \cdot \pi/180$$
$$\theta_3 = -\theta_{3,log} \cdot \pi/180 - \theta_2$$
$$\theta_4 = -\theta_{4,log} \cdot \pi/180$$
$$\theta_5 = \theta_{5,log} \cdot \pi/180$$
$$\theta_6 = -\theta_{6,log} \cdot \pi/180$$

B

Additional simulation data

Figure B.1 shows the comparison between the identification error for noise applied to the robot joint angles and the identification error for noise applied to the ball-bar joint angles. Comparing the two shows no evident difference in size, which is the motivation for speeding up simulation by only applying varying errors to the robot joint angles.

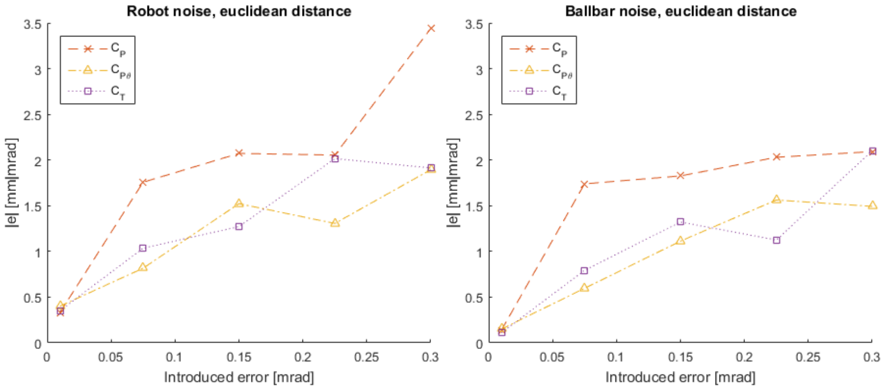


Figure B.1 Robot noise and ball-bar noise.

Table B.1 shows the five parameters with largest errors in the parameter identification using data from simulations with 0.01$mrad$ noise applied to both robot joints and ball-bar joints. Each row represents a simulation with different random seed, and the parameters are ordered with larger errors to the left. The five parameter errors are however somewhat equal in size, which is around 100 μrad.

Simulation					
1	$b\theta_5$	$r\theta_6$	o_y	ra_1	ra_5
2	o_y	bd_3	ra_2	rd_4	$r\theta_5$
3	o_y	bd_3	$b\theta_5$	$r\theta_6$	ra_5
4	$b\theta_5$	o_y	$r\theta_6$	bd_3	ra_1
5	o_y	$b\theta_5$	$r\theta_6$	ra_1	rd_3
6	o_y	$r\theta_6$	$b\theta_5$	bd_3	ra_2
7	$b\theta_5$	$r\theta_6$	o_y	ra_1	bd_3
8	o_y	bd_3	ra_1	ra_2	$b\theta_5$
9	$r\theta_6$	$b\theta_5$	o_y	ra_5	o_z
10	o_y	$b\theta_5$	$r\theta_6$	ra_1	bd_3
11	$r\theta_6$	$b\theta_5$	o_y	ra_1	bd_3
12	o_y	bd_3	ra_2	rd_4	ra_3
13	o_y	$b\theta_5$	$r\theta_6$	ra_1	bd_3
14	o_y	ra_1	$r\theta_6$	$b\theta_5$	bd_3
15	$r\theta_6$	$b\theta_5$	o_y	ra_1	ra_5
16	o_y	$r\theta_6$	$b\theta_5$	bd_3	ra_1
17	o_y	$b\theta_5$	$r\theta_6$	rd_4	bd_3
18	o_y	$r\theta_6$	$b\theta_5$	ra_1	bd_3
19	o_y	$b\theta_5$	$r\theta_6$	bd_3	ra_2
20	$r\theta_6$	$b\theta_5$	o_y	ra_1	bd_3

Table B.1 Five largest errors using C_T and the same simulation type, but different random seed.

Bibliography

ABB. *Product specification IRB 2400*. https://library.e.abb.com/public/fa8324850004a97fc125766d003e863c/Product%20specification%202400%20M98A%20BWOS3.2.pdf. [Online; accessed 2016-05-25].

ABB. *RobotStudio*. http://new.abb.com/products/robotics/robotstudio. [Online; accessed 2016-06-01].

Araneda, E. (2004). *A Variation of the Levenberg Marquardt Method: An Attempt to Improve Efficiency*. Thesis for Master of Science in Geosystem. Department of Earth, Atmospheric, and Planetary Sciences, Massachusetts Institute of Technology, http://hdl.handle.net/1721.1/28609 . [Online; accessed 2016-07-01].

Benjamin, W. M., S. R. Zvi, and R. D. Morris (1991). "Fundamentals of manipulator calibration". In: John Wiley & Sons Inc, New York, pp. v–vi. ISBN: 0-471-50864-0.

Bennett, D. J. and J. M. Hollerbach (1991). "Autonomous calibration of single-loop closed kinematic chains formed by manipulators with passive endpoint constraints". *IEEE Transactions on Robotics and Automation* **7**:5, pp. 597–606.

berkeley.edu. *QR-decomposition of a matrix*. https://inst.eecs.berkeley.edu/~ee127a/book/login/l_mats_qr.html. [Online; accessed 2016-06-01].

Collin, S. (2016). *Kinematic Robot Calibration Using a Double Ball-Bar*. Master's Thesis TFRT-6000. Department of Automatic Control, Lund University, Sweden.

Collins, R. *Robust estimation: RANSAC*. http://www.cse.psu.edu/~rtc12/CSE486/lecture15.pdf. [Online; accessed 2016-07-01].

Comau. *Working areas NJ 200 - 2.7*. http://www.comau.com/Download/robot/nj220-27/EN_workingareas_nj22027.pdf. [Online; accessed 2016-05-30].

Corke, P. I. (2011). *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer, Berlin Heidelberg. ISBN: 978-3-642-20143-1.

Gallager, R. G. (2013). “Stochastic processes, theory for applications”. In: Cambridge University Press, Cambridge, United Kingdom, pp. 139–140. ISBN: 978-1-107-03975-9.

Geng, H. (2004). “Manufacturing engineering handbook”. In: McGraw-Hill, New York. Chap. 7. ISBN: 978-0-071-39825-1.

Ginani, L. S. and J. M.S. T. Motta (2011). “Theoretical and practical aspects of robot calibration with experimental verification”. *Journal of the Brazilian Society of Mechanical Sciences and Engineering* **33**:1, pp. 15–21.

Goswami, A., A. Quaid, and M. Peshkin (1993). “Identifying robot paramters using partial pose information”. *IEEE Control Systems Magazine* **13**:5, pp. 6–4.

Hartenberg, R. S. and J. Denavit (1955). “A kinematic notation for lower pair mechanisms based on matrices”. *Journal of Applied Mechanics* **77**, pp. 215–221.

Hayati, S. and M Mirmirani (1985). “Improving the absolute positioning accuracy of robot manipulators”. *Journal of Robotic Systems* **2**:4, pp. 397–413.

Heidenhain (2015). *Rotary encoder catalog*. `http://www.heidenhain.com/fileadmin/pdb/media/img/350457-2I_General_Catalog_en.pdf`. [Online; accessed 2016-05-30].

IEC 61131-3:2013 (2013). *Programmable controllers - part 3: programming languages*. International Electrotechnical Commission, Geneva, Switzerland.

ISO 9283:1998(en) (1998). *Manipulating industrial robots - performance criteria and related test methods*. International Organization for Standardization, Geneva, Switzerland.

Joubair, A. *Accuracy and repeatability in industrial robots*. `http://blog.robotiq.com/bid/72766/What-are-Accuracy-and-Repeatability-in-Industrial-Robots`. [Online; accessed 2016-06-07].

Joubair, A. and I. A. Bonev (2015). “Non-kinematic calibration of a six-axis serial robot using planar constraints”. *Precision Engineering* **40**, pp. 325–333.

Low, K.-H. (2007). *Industrial Robotics: Programming, Simulation and Applications*. Pro-Literatur-Verlag, Mammendorf, Germany, p. 532. ISBN: 3-86611-286-6.

Nikon. *K-series optical cmm solutions*. `http://www.nikonmetrology.com/en_EU/content/download/11040/220148/version/4/file/Optical_CMM_EN.pdf`. [Online; accessed 2016-05-21].

Nubiola, A. and I. A. Bonev (2012). “Absolute calibration of an ABB IRB 1600 robot using a laser tracker”. *Robotics and Computer-Integrated Manufacturing* **29**:1, pp. 236–245.

Spong, M. W., S. Hutchinson, and M. Vidyasagar (2006). *Robot Modeling and Control*. John Wiley & Sons Inc, New York. ISBN: 978-0-471-64990-8.

Yang, D. C. H. and J. G. Blanche (1990). “Design and application guidelines for cycloid drives with machining tolerances”. *Mechanism and Machine Theory* **25**:5, pp. 487–501.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER´S THESIS

Date of issue
December 2016

Document Number
ISRN LUTFD2/TFRT--6017--SE

Author(s)
Sebastian Brand
Niklas Nilsson

Supervisor
Martin Holmstrand, Cognibotics
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden
Rolf Johansson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Calibration of robot kinematics using a double ball-bar with embedded sensing

Abstract

Today's industrial robots are highly repeatable, but need to be calibrated to improve their absolute accuracy. This calibration can be done on many of the robot properties such as kinematic parameters, joint friction or bending stiffness. This thesis explores a calibration procedure for the kinematic parameters, that use a specialized piece of hardware - the double ball-bar.

The double ball-bar restricts the motion of the robot to a spherical surface. Sensors were added to the ball-bar joints, which made it possible to use the forward kinematic homogeneous transformation matrix from both the robot side and the ball-bar side, the matrices can be compared to each other and the parameters of the robot can be identified using a non-linear least-squares minimization algorithm.

The calibration proved promising in simulations and showed an increased robustness to error sources such as white noise and fluctuations of the gear ratio found in cycloid drives. It also provided an improved identification of the robot parameters compared to the calibration done using the sensor-less double ball-bar.

In experiments the identification showed some improvement in the identification over using the sensor-less double ball-bar, but also that the method needs to be further improved to be able to produce satisfactory calibration results.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-71

Recipient's notes

Security classification