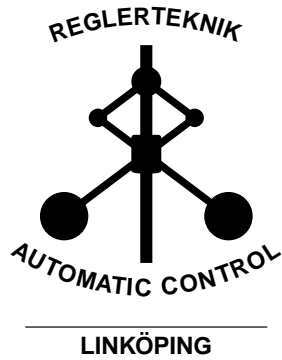


Linköping studies in science and technology. Thesis.  
No. 1511

# On Sensor Fusion Applied to Industrial Manipulators

**Patrik Axelsson**



Division of Automatic Control  
Department of Electrical Engineering  
Linköping University, SE-581 83 Linköping, Sweden  
<http://www.control.isy.liu.se>  
[axelsson@isy.liu.se](mailto:axelsson@isy.liu.se)

Linköping 2011

This is a Swedish Licentiate's Thesis.

Swedish postgraduate education leads to a Doctor's degree and/or a Licentiate's degree.

A Doctor's Degree comprises 240 ECTS credits (4 years of full-time studies).

A Licentiate's degree comprises 120 ECTS credits,  
of which at least 60 ECTS credits constitute a Licentiate's thesis.

Linköping studies in science and technology. Thesis.

No. 1511

**On Sensor Fusion Applied to Industrial Manipulators**

Patrik Axelsson

*axelsson@isy.liu.se*

*www.control.isy.liu.se*

*Department of Electrical Engineering*

*Linköping University*

*SE-581 83 Linköping*

*Sweden*

ISBN 978-91-7393-031-4

ISSN 0280-7971

LiU-TEK-LIC-2011:50

Copyright © 2011 Patrik Axelsson

Printed by LiU-Tryck, Linköping, Sweden 2011

*Till min familj*





## Abstract

One of the main tasks for an industrial robot is to move the end-effector in a predefined path with a specified velocity and acceleration. Different applications have different requirements of the performance. For some applications it is essential that the tracking error is extremely low, whereas other applications require a time optimal tracking. Independent of the application, the controller is a crucial part of the robot system. The most common controller configuration uses only measurements of the motor angular positions and velocities, instead of the position and velocity of the end-effector.

The development of new cost optimised robots have introduced unwanted flexibilities in the joints and the links. It is no longer possible to get the desired performance and robustness by only measuring the motor angular positions. This thesis investigates if it is possible to estimate the end-effector position when an accelerometer is mounted at the end-effector. The main focus is to investigate Bayesian estimation methods for state estimation, here represented by the *extended Kalman filter* (EKF) and the *particle filter* (PF).

A simulation study is performed on a two degrees of freedom industrial robot model using an EKF. The study emphasises three important problems to take care of in order to get a good performance. The first one is related to model errors which in general requires better identification methods. The second problem is about tuning of the EKF, i.e., the choice of covariance matrices for the measurement and process noise. It is desirable to have an automatic tuning procedure which minimises the estimation error and is robust to initial conditions of the tuned parameters. A variant of the *expectation maximisation* (EM) algorithm is proposed for estimation of the process noise covariance matrix  $Q$ . The EM algorithm iteratively estimates the unobserved state sequence and the matrix  $Q$  based on the observations of the process, where the *extended Kalman smoother* (EKS) is the instrument to find the unobserved state sequence.

The third problem considers the orientation and position of the accelerometer mounted to the end-effector. A novel method to find the orientation and position of the triaxial accelerometer is proposed and evaluated on experimental data. The method consists of two consecutive steps, where the first is to estimate the orientation of the sensor from static experiments. In the second step the sensor position relative to the robot base is identified using sensor readings when the sensor moves in a circular path and where the sensor orientation is kept constant in a path fixed coordinate system.

Finally, experimental evaluations are performed on an ABB IRB4600 robot. Different observers using the EKF, EKS and PF with different estimation models are proposed. The estimated paths are compared to the true path measured by a laser tracking system. There is no significant difference in performance between the six observers. Instead, execution time, model complexities and implementation issues have to be considered when choosing the method.



## Populärvetenskaplig sammanfattning

En av de viktigaste uppgifterna för en industrirobot är att förflytta verktyget i en fördefinierad bana med en specificerad hastighet och acceleration. Olika applikationer har olika krav på prestanda, för vissa applikationer är det viktigt att banföljningsfelet är extremt lågt, medan andra program kräver en tidsoptimal banföljning. Oberoende av applikationen är regulatorn en avgörande del av robotsystemet. Den vanligaste regulatorkonfigurationen använder bara mätningar av motorernas vinkelpositioner och -hastigheter, istället för positionen och hastigheten för verktyget.

Utvecklingen av nya kostnadsoptimerade robotar har infört önskade flexibiliteter i leder och länkar. Det är inte längre möjligt att få den önskade prestandan och robustheten genom att bara mäta motorns vinkelpositioner. Denna avhandling undersöker om det är möjligt att skatta verktygspositionen när en accelerometer är monterad på verktyget. Huvudfokus är att undersöka Bayesianska skattningsmetoder för tillståndsskattning, som här representeras av *extended Kalman-filtret* (EKF) och *partikelfiltret* (PF).

En simuleringsstudie utförs på en modell av en två-axlig industrirobot med ett EKF. Studien framhåller tre viktiga problem att ta hand om för att få ett bra resultat. Det första är relaterat till modellfel, som i allmänhet kräver bättre identifieringsmetoder. Det andra problemet handlar om trimning av EKF, d.v.s. valet av kovariansmatriser för mät- och processbruset. Det är önskvärt att ha en automatisk trimningsprocedur som minimerar skattningsfelet och samtidigt är robust mot startvärdena på de parametrar som ska trimmas. En variant av *expectation maximisation* (EM) algoritmen föreslås för skattning av processbrusets kovariansmatris  $Q$ . EM-algoritmen skattar iterativt den icke-observerade tillståndssekvensen och matrisen  $Q$  från observationer av processen, där *extended Kalman smoother* (EKS) är verktyget för att hitta den icke-observerade tillståndssekvensen.

Det tredje problemet handlar om orienteringen och positionen för accelerometern som är monterad på verktyget. En ny metod för att hitta orienteringen och positionen för en tre-axlig accelerometer föreslås och utvärderas på experimentella data. Metoden består av två på varandra följande steg, där det första är att skatta orienteringen av sensorn från statistiska experiment. I det andra steget identifieras positionen i förhållande till robotens bas med hjälp av sensoravläsningar när sensorn rör sig i en cirkulär bana och där sensorns orientering hålls konstant i ett banfixt koordinatsystem.

Slutligen utförs experimentella utvärderingar på en ABB IRB4600 robot. Olika observatörer som använder EKF, EKS och PF med olika skattningsmodeller föreslås, och den skattade banan jämförs med den sanna banan som är uppmätt med ett lasersystem. Det är ingen signifikant skillnad i prestanda mellan de sex observatörerna, utan istället måste beräkningstiden, modellkomplexiteten och implementeringen beaktas när man väljer metod.



## Acknowledgments

First I would like to thank my supervisors Professor Mikael Norrlöf and Professor Fredrik Gustafsson for their help and guidance with my research.

I am also very thankful to Professor Lennart Ljung, our former head, for letting me join the Automatic Control group at Linköping University, and to our current head, Professor Svante Gunnarsson. All the administrative work had been so much harder without help from our administrator Ninna Stengård, and before her Åsa Karmelind, and before her Ulla Salaneck, thanks to all of you.

This work has been supported by The Vinnova Industry Excellence Center LINKSIC and in particularly the partner ABB Robotics. The thesis would not be possible without their financial support, thanks a lot.

The robot models had not been that accurate if it had not been for Dr. Stig Moberg at ABB Robotics, thanks a lot. My gratitude also goes to Dr. Umut Orguner and Docent Rickard Karlsson for helping me with anything concerning probability. The thesis layout had not been this good if not the previous L<sup>A</sup>T<sub>E</sub>X gurus Dr. Henrik Tidefelt and Dr. Gustaf Hendeby had spent a lot of their time creating the thesis template, many thanks to you.

The content of the thesis had been much harder to read without all the good comments from M.Sc. André Carvalho Bittencourt, M.Sc. Peter Rosander, Docent Rickard Karlsson, M.Sc. Daniel Simon, M.Sc. Ylva Jung and my supervisors Professor Mikael Norrlöf and Professor Fredrik Gustafsson.

The time from I started at the Automatic Control group until now has been full of nice co-workers and activities, thank you all for all the fun we have had. I would give special thanks to my own computer support Lic. Daniel Petersson for all the help with CVS, SVN, Maple etcetera. Also many thanks to my room mate M.Sc. Ylva Jung for answering all my silly questions.

Thanks to all my friends outside the RT-corridor; my former classmates in the Y04A class, and also my frisbee golf fellows, it is always funny when you hit the trees and I am not.

Finally, many thanks to my family for their love and support. Last but by no means least also many thanks to Louise for her love and support, furthermore many thanks for her patience when I am at work instead of home with her.

*Linköping, November 2011  
Patrik Axelsson*



---

# Contents

Notation	xv
<b>I Background</b>	
<b>1 Introduction</b>	<b>3</b>
1.1 Background and Motivation . . . . .	3
1.2 Contributions . . . . .	7
1.3 Thesis Outline . . . . .	9
<b>2 Industrial Robots</b>	<b>11</b>
2.1 Introduction . . . . .	11
2.2 Kinematic Models . . . . .	14
2.2.1 Coordinate Transformation . . . . .	14
2.2.2 Forward Kinematics . . . . .	15
2.2.3 Inverse Kinematics . . . . .	16
2.3 Dynamic Models . . . . .	17
2.3.1 Rigid Link Model . . . . .	17
2.3.2 Flexible Joint Model . . . . .	18
<b>3 Estimation Theory</b>	<b>21</b>
3.1 The Filtering Problem . . . . .	22
3.1.1 The Extended Kalman Filter (EKF) . . . . .	22
3.1.2 The Particle Filter (PF) . . . . .	23
3.2 The Smoothing Problem . . . . .	25
3.3 The Expectation Maximisation Algorithm . . . . .	25
3.4 Estimation of Transformed PDFs . . . . .	27
<b>4 Models</b>	<b>29</b>
4.1 Two DOF Rigid Link Flexible Joint Robot Model . . . . .	29
4.1.1 Forward Kinematic Model . . . . .	29
4.1.2 Inverse Kinematic Model . . . . .	30
4.1.3 Dynamic Model . . . . .	31

4.2	Accelerometer Model . . . . .	33
4.3	Modelling of Bias . . . . .	34
4.4	Estimation Models . . . . .	35
4.4.1	Nonlinear Estimation Model . . . . .	35
4.4.2	Estimation Model with Linear Dynamic . . . . .	37
4.4.3	Linear Estimation Model with Acceleration as Input . . . . .	38
4.4.4	Nonlinear Estimation Model with Acceleration as Input . . . . .	38
4.4.5	Summary of the Estimation Models . . . . .	39
<b>II</b>	<b>Results</b>	
<b>5</b>	<b>Simulation Study</b>	<b>43</b>
5.1	Overview of the Simulation Model . . . . .	43
5.2	Simulation Setup . . . . .	45
5.2.1	Tuning of Covariance Matrices . . . . .	47
5.3	Investigation of the Sample Time . . . . .	49
5.4	Results . . . . .	51
5.5	Conclusions . . . . .	59
<b>6</b>	<b>The EM Algorithm for Covariance Estimation</b>	<b>61</b>
6.1	Derivation of the EM Solution . . . . .	61
6.1.1	Expectation Step . . . . .	62
6.1.2	Maximisation Step . . . . .	67
6.1.3	Stop Criterion . . . . .	67
6.2	Alternative Ways to Find the Covariance Matrix of the Process Noise . . . . .	68
6.3	Simulation Results . . . . .	69
6.4	Conclusions . . . . .	71
<b>7</b>	<b>Pose Estimation of an Accelerometer</b>	<b>73</b>
7.1	Problem Formulation . . . . .	74
7.2	Identification of Orientation, Sensitivity and Bias . . . . .	75
7.3	Estimation of the Position of the Accelerometer . . . . .	77
7.4	Experimental Results . . . . .	84
7.4.1	Evaluation of the Estimated Orientation, Sensitivity and Bias . . . . .	85
7.4.2	Evaluation of the Estimated Position . . . . .	88
7.5	Conclusions . . . . .	90
<b>8</b>	<b>Experimental Evaluations</b>	<b>93</b>
8.1	Observers . . . . .	93
8.2	Experimental Setup . . . . .	94
8.3	Results . . . . .	94
8.4	Conclusions . . . . .	100
<b>9</b>	<b>Conclusions and Future Work</b>	<b>103</b>
9.1	Conclusions . . . . .	103
9.2	Future Work . . . . .	104



**Bibliography**



---

# Notation

## ESTIMATION

Notation	Meaning
$x_k$	State vector at time $k$
$u_k$	Input vector at time $k$
$v_k$	Process noise vector at time $k$
$y_k$	Measurement vector at time $k$
$e_k$	Measurement noise vector at time $k$
$p(x y)$	Conditional density function for $x$ given $y$
$y_{1:k}$	Sequence of measurements from time 1 to time $k$
$\hat{x}_{k k'}$	Estimated state vector at time $k$ given measurements up to and including time $k'$
$P_{k k'}$	Covariance of the estimated state vector at time $k$ given measurements up to and including time $k'$
$\hat{x}_{k N}^s$	Smoothed state vector at time $k$ given measurements up to time $N$
$P_{k N}^s$	Covariance of the smoothed stated vector at time $k$ given measurements up to time $N$
$x_k^i$	Particle $i$ at time $k$
$w_k^i$	Weight for particle $i$ at time $k$
$\mathcal{N}(\cdot; \mu, \Sigma)$	Gaussian distribution with mean $\mu$ and covariance $\Sigma$
$\mathcal{U}(a, b)$	Uniform distribution on the interval $[a, b]$
$Q/R$	Covariance for the process/measurement noise

## ROBOTICS

Notation	Meaning
$Q_{j/i}, \mathcal{R}_{j/i}$	Rotation matrix for system $j$ with respect to system $i$
$H_{j/i}$	Homogeneous transformation matrix for system $j$ with respect to system $i$
$X$	Position and orientation of the end-effector
$\dot{X}/\ddot{X}$	Linear and angular velocity/acceleration of the end-effector
$q_a/\dot{q}_a/\ddot{q}_a$	Arm angular positions/velocities/accelerations
$q_m/\dot{q}_m/\ddot{q}_m$	Motor angular positions/velocities/accelerations
$\tau_m$	Motor torque
$\Upsilon(\cdot)$	Forward kinematic model
$\mathcal{J}(\cdot)$	Jacobian matrix of the forward kinematic model
$M(\cdot)$	Inertia matrix
$C(\cdot)$	Coriolis- and centrifugal terms
$G(\cdot)$	Gravitational torque
$F(\cdot)$	Friction torque
$T(\cdot)$	Stiffness torque
$\tilde{D}(\cdot)$	Damping torque
$\eta$	Gear ratio
$l_i$	Arm lengths

## MISCELLANEOUS

Notation	Meaning
$\mathbf{I}$	Identity matrix
$\mathbf{0}$	Null matrix
$\dagger$	Pseudo inverse
$\top$	Transpose
$T_s$	Sample time
$\text{tr}$	Trace operator
$E[\cdot]$	Expectation value
$\text{Cov}(\cdot)$	Covariance
$q$	Unit quaternion
$\text{atan2}$	Four quadrant arc tangent
$g$	Gravity constant
$x, z(\hat{x}, \hat{z})$	(Estimated) Cartesian coordinates
$e_k$	Path error at time $k$
$Ox_i y_i z_i$	Cartesian coordinate frame named $i$
$\ddot{\rho}_s$	Acceleration due to the motion in the accelerometer frame
$\mathbf{b}$	Bias vector
$\mathbb{R}$	Set of real numbers
$\mathbb{S}_{++}(\mathbb{S}_+)$	Set of symmetric positive definite (semidefinite) matrices

**ABBREVIATIONS**

---

<b>Abbreviation</b>	<b>Meaning</b>
CDF	Cumulative density function
DOF	Degree of freedom
EM	Expectation maximisation
EKF	Extended Kalman filter
EKS	Extended Kalman smoother
ILC	Iterative learning control
ML	Maximum likelihood
MC	Monte Carlo
PF	Particle filter
PDF	Probability density function
RMSE	Root mean square error
SNR	Signal to noise ratio
TCP	Tool centre point

---



**Part I**

**Background**





# 1

---

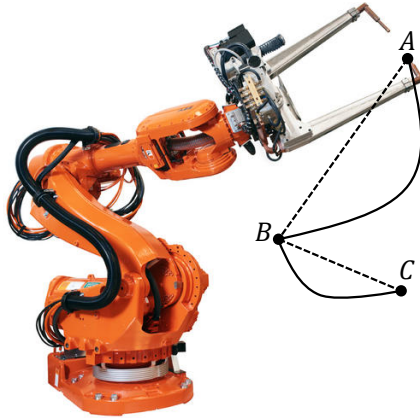
## Introduction

In this thesis, the problem of state estimation for industrial robots is covered. The estimated states, such as arm angular positions and velocities, are supposed to be used in the control loop to improve the accuracy of the robot.

The background and motivation of why estimates of the arm angular positions and velocities are needed are presented in Section 1.1. The main contributions of the thesis are listed in Section 1.2 and the outline of the thesis is given in Section 1.3.

### 1.1 Background and Motivation

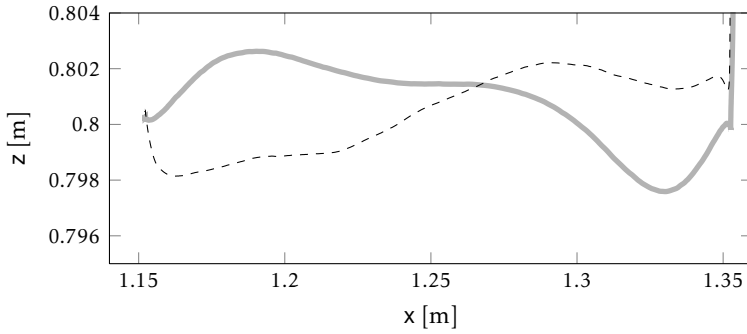
The first industrial robots were big and heavy with rigid links and joints. The development of new robot models has been focused on increasing the performance along with cost reduction, safety improvement and introduction of new functionalities as described in Brogårdh (2007). One way to reduce the cost is to lower the weight of the robot which conduces to lower mechanical stiffness in the links. Also, the components of the robot are changed such that the cost is reduced, which can infer larger individual variations and unwanted nonlinearities. The most crucial component, when it comes to flexibilities, is the gearbox. The gearbox has changed more and more to a flexible component described by nonlinear relations, which cannot be neglected in the motion control loop. The friction in the gearbox is also an increasing problem that is described by nonlinear relations. The available measurements for control are the motor angular positions, but since the end-effector, which is the desired control object, is on the other side of the gearbox it cannot be controlled in a satisfactory way. Instead, extensive use of mathematical models describing the nonlinear flexibilities are needed in



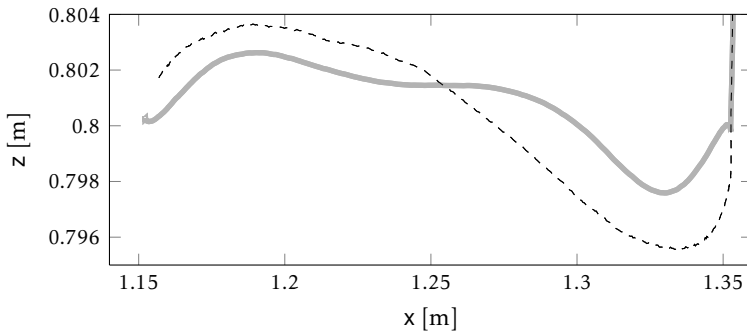
**Figure 1.1:** ABB IRB6600 with a spot welding gun (ABB Robotics, 2011) and a desired path. The desired path can be to go direct from point A to point C through point B without considering the path or follow a predefined path (solid line).

order to control the weight optimised robot. In practice, the static volumetric accuracy is approximately 2-15 mm due to the gravity deflection which is caused by the flexibilities. One solution to reduce the error is to model an extended kinematic model and an elasto-static model by conducting an offline identification procedure. The static accuracy can, in this way, be reduced to 0.5 mm. For the dynamic accuracy a new model-based motion control is presented in Björkman et al. (2008) where the maximum path error is one-fifth of the maximum path error from a typical controller. However, the more reduction of the material cost the more complex models are needed. There is therefore a demand of new approaches for motion control where less accurate models can be sufficient.

One solution can be to estimate the position and orientation of the end-effector along the path and then use the estimated position and orientation in the feedback loop of the motion controller. Figure 1.1 shows a robot with a spot welding gun. The desired path can be to go direct from point A to point C through point B without considering the path or follow a predefined path (solid line). This thesis deals only with the estimation problem of the end-effector and not the control problem. The most simple observer is to use the measured motor angular positions in the forward kinematic model to get the position and orientation of the end-effector. In Figure 1.2a it is shown that the estimated position of the end-effector does not track the true measured position very good. The reason for the poor result is that the oscillations on the arm side do not influence the motor side of the gearbox that much due to the flexibilities. The flexibilities can also distort the oscillations of the arm side, which means that the estimated path oscillates in a different way than the true path. The observer can consequently not track the true position and another observer is therefore needed. The observer requires a dynamic model of the robot in order to capture the oscillations on the arm side of



(a) Estimated position using the forward kinematic model with the measured motor angular positions.



(b) Estimated position using a particle filter.

**Figure 1.2:** True path (solid) and estimated path (dashed) of the end-effector using (a) the forward kinematic model with the measured motor angular positions, and (b) a particle filter using the acceleration of the end-effector as a measurement. The dynamical performance of the estimated path in (a) is insufficient due to the flexible gearboxes in between the measured motor angular positions and the end-effector. The estimated path from a particle filter in (b) is much better.

the gearbox as well as more measurements than only the motor angular positions. Figure 1.2b shows one of the results in this thesis where a particle filter has been used. The measured position is given by an external laser tracking system from Leica Geosystems (2011), which is only used for evaluation of the motion control system and not for feedback of the tool position. The laser system in Figure 1.3a tracks a crystal, see Figure 1.3b, attached to the robot. Note that it is only possible to measure the position of the end-effector and not the orientation.

Different types of observers for flexible joint robots have been proposed in the literature. In Jankovic (1995) a high gain observer is proposed using only the motor angular positions and velocities as measurements. In Nicosia et al. (1988); Tomei (1990), and Nicosia and Tomei (1992) different observers are proposed where it is assumed that the arm angular positions and/or the arm angular velocities are measured. The drawback is that this is not the case for a commercial robot. The solution is obviously to install rotational encoders on the arm side of the gearbox and use them in the forward kinematic model. However, the encoders on the arm side do not exactly give the desired angles. Take joint one in Figure 2.2 as an example. The system from the motor side of the gearbox to the end-effector can be seen as a three-mass system and not a two-mass system. The motor encoder measures the position of the first mass and the arm encoder measures the position of the second mass. The flexibility between the second and third mass is due to flexibilities in joints two and three. These flexibilities are in the same direction as joint one and cannot be measured with encoders in joints two and three. Hence, there is still a need of estimating the end-effector path. One way to obtain information about the oscillations on the arm side can be to attach an accelerometer on the robot, e.g. at the end-effector. The accelerometer used in this thesis is a triaxial accelerometer from Crossbow Technology (Crossbow Technology, 2004), see Figure 1.3c. A natural question is, how to estimate the arm angular positions from the measured acceleration as well as the measured motor angular positions. A common solution for this kind of problems is to apply sensor fusion methods for state estimation. The acceleration of the end-effector as well as the measured motor angular positions can be used as measurements in e.g. an *extended Kalman filter* (EKF) or *particle filter* (PF). In Karlsson and Norrlöf (2004, 2005), and Rigatos (2009) the EKF and PF are evaluated on a flexible joint model using simulated data only. The estimates from the EKF and PF are also compared with the theoretical Cramér-Rao lower bound in Karlsson and Norrlöf (2005) to see how good the filters are. An evaluation of the EKF using experimental data is presented in Henriksson et al. (2009), and in Jassemi-Zargani and Neculescu (2002) with different types of estimation models. A method using the measured acceleration of the end-effector as input instead of using it as measurements is described in De Luca et al. (2007). The observer, in this case, is a linear dynamic observer using pole placement, which has been evaluated on experimental data. In Lertpiriyasawat et al. (2000), and Li and Chen (2001) the case with flexible link models, where the acceleration or the position of the end-effector are measured, is presented.

In an online control perspective, it is important that the estimation method per-



**Figure 1.3:** The laser tracking system and the accelerometer used in this thesis.

forms in real-time. However, the estimated position of the end-effector can still be used if the estimation method performs slower than real-time. One offline application is the *iterative learning control* (ILC) as described in Wallén et al. (2009). In Wallén et al. (2008) it is shown that motor side learning is insufficient if the mechanical resonances are excited by the robot trajectory. Other applications that can improve if the estimated position of the end-effector is available, not necessarily online, are system identification, supervision, diagnostics, and automatic controller tuning.

## 1.2 Contributions

The main contribution in this thesis is how to estimate the position of the end-effector, i.e., the robot link positions, using an accelerometer attached to the end-effector. Also the robot link velocities and in some cases the accelerations are estimated as well. The thesis is focused on:

- experimental validation of the EKF and PF using different types of estimation models,
- a method to estimate the process noise covariance matrix to be used in the EKF, and
- a method to determine the orientation and position of the accelerometer mounted to the end-effector.

The results presented in this thesis have been published before in the following publications, where the author is the main contributor:

Patrik Axelsson, Umut Orguner, Fredrik Gustafsson, and Mikael Norrlöf. ML estimation of process noise variance in dynamic systems. In *Proceedings of the 18th IFAC World Congress*, pages 5609–5614, Milano, Italy, August 2011e.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Submitted to Control Engineering Practice*, 2011b.

The submitted version is available as a technical report, Axelsson et al. (2011d).

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Tool position estimation of a flexible industrial robot using recursive Bayesian methods. *Submitted to the IEEE Conference on Robotics and Automation 2012*, 2011a.

The submitted version is available as a technical report, Axelsson et al. (2011c).

Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. *Submitted to the IEEE Conference on Robotics and Automation 2012*, 2011a.

The submitted version is available as a technical report, Axelsson and Norrlöf (2011b).

Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. *Submitted to the IFAC Symposium on Robot Control 2012*, 2011b.

The submitted version is available as a technical report, Axelsson (2011c).

Patrik Axelsson, Mikael Norrlöf, Erik Wernholt, and Fredrik Gustafsson. Extended Kalman filter applied to industrial manipulators. In *Proceedings of Reglermötet 2010*, Lund, Sweden, June 2010.

Patrik Axelsson. A simulation study on the arm estimation of a joint flexible 2 DOF robot arm. Technical Report LiTH-ISY-R-2926, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, December 2009.

Patrik Axelsson. Simulation model of a 2 degrees of freedom industrial manipulator. Technical Report LiTH-ISY-R-3020, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, June 2011a.

Relevant material not included in this thesis, where the author has contributed, is,

---

André Carvalho Bittencourt, Patrik Axelsson, Ylva Jung, and Torgny Brogårdh. Modeling and identification of wear in a robot joint under temperature uncertainties. In *Proceedings of the 18th IFAC World Congress*, pages 10293–10299, Milano, Italy, August 2011.

## 1.3 Thesis Outline

The thesis is organised as follows; Chapter 2 gives an introduction to the field of industrial robots as well as the models needed for simulation and control. The estimation problem is summarised in Chapter 3 and different methods for estimation are presented. All the models needed for simulation and estimation are presented in Chapter 4 and the simulation study is given in Chapter 5. From the simulation study, two issues that are important for the estimation performance show up. The first issue is how to tune the extended Kalman filter and the result is presented in Chapter 6. The second issue, presented in Chapter 7, is about determining the orientation and position of the accelerometer mounted to the end-effector. Finally, the experimental evaluations are presented in Chapter 8, and Chapter 9 summarises the thesis and discusses possible future work.





# 2

---

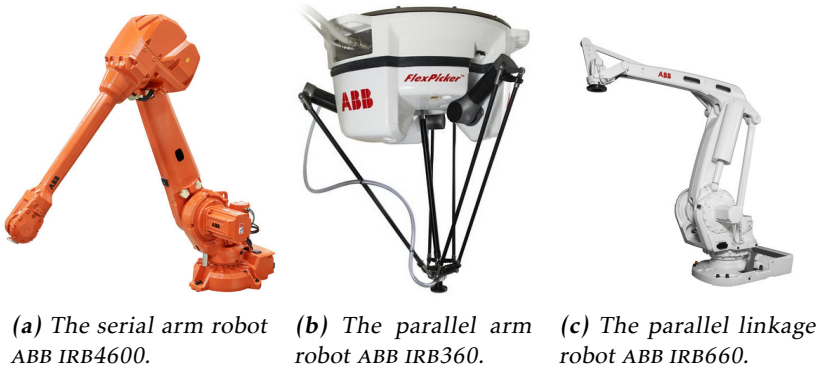
## Industrial Robots

Industrial robots are used in tasks where high precision and high speed are needed, or in tasks where the environment is harmful for humans. The robot needs therefore to be serviceable, have high precision, operate at high speeds and be robust to disturbances. Good models and controllers are necessary for all of these requirements. Common applications for an industrial robot are spot welding, arc welding, material handling and cutting. There are three types of robot structures for industrial robots. The most common is the serial arm robot in Figure 2.1a, whereas the other two robot structures have parallel arms, see Figure 2.1b and parallel links, see Figure 2.1c. In this thesis, the focus is on serial arm robots.

The chapter starts with an introduction to the concept of industrial robots in Section 2.1. Sections 2.2 and 2.3 present a short overview of the kinematic and dynamic models needed for control of an industrial robot, respectively.

### 2.1 Introduction

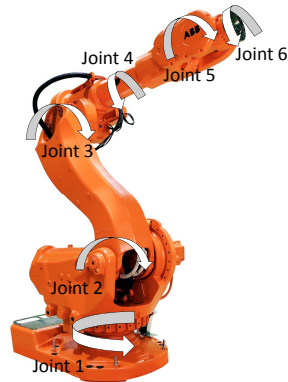
In 1954, the American inventor George C. Devol applied for the first patents for industrial robots, called the Programmed Article Transfer. Seven years later, in 1961, the patents were granted. Devol and Joseph Engelberger started the first robot manufacturing company Unimation Inc. in 1956. The first operating industrial robot Unimate was launched in 1959 and the first robot installation was performed in 1961 at General Motors plant in Trenton, New Jersey. The first robot installation in Europe was performed 1967 in Sweden. The first micro-processor controlled electrical robot IRB6 was launched by ASEA (current ABB) 1973. (Nof, 1999; Westerlund, 2000) Since then, ABB has evolved to one of the biggest manufactures of industrial robots and robot systems. ABB has over



**Figure 2.1:** Three types of robots from ABB (ABB Robotics, 2011).

190 000 robots installed world wide and the company was the first with over 100 000 sold robots. In 1998 the fastest pick and place robot FlexPicker, IRB360 in Figure 2.1b, was launched. (ABB Robotics, 2011) Other large manufactures are KUKA, who launched the first industrial robot with six electro mechanically driven axes in 1973 (KUKA, 2011), FANUC Robotics with over 220 000 installed robots (FANUC Robotics, 2011), and Motoman, with over 220 000 installed robots world wide (Motoman, 2011). Motoman launched the first robot controller where it was possible to control two robots in 1994, and a 13 axis dual arm robot in 2006.

The overall robot system includes the manipulator, computers and control electronics. The desired motion of the robot is given in the user program. The program is composed by motion commands, such as a linear or circular trajectory between two points for the end-effector in the three dimensional space. Also the three dimensional orientation of the end-effector can be affected. In particular, the *tool centre point* (TCP), defined somewhere on the end-effector, is of interest. For example, in arc welding applications the TCP is defined as the tip of the welding gun. The position and orientation, also known as the pose, are thus described in a six dimensional space. The robot needs therefore at least six *degrees of freedom* (DOF) to be able to manoeuvre the end-effector to a desired position and orientation. The total volume the robot end-effector can reach is called the workspace. The workspace is divided in the reachable and the dexterous workspace. The reachable workspace includes all points the end-effector can reach with some orientation. Whereas the dexterous workspace includes all points the end-effector can reach with an arbitrary orientation. The dexterous workspace is of course a subset of the reachable workspace. The motion can also be defined in the joint space, where each joint corresponds to one DOF. That means that the robot moves between two sets of joint angles where the path of the end-effector is implicit, meanwhile the velocity and acceleration are considered. The robot is said to have  $n$  DOF if it has  $n$  joints. Figure 2.2 shows how the six joints for a six DOF robot can be defined. A desired velocity and accel-



**Figure 2.2:** The six DOF serial arm robot ABB IRB6600 where the arrows indicate the joints (ABB Robotics, 2011).

ation of the end-effector or the joints can also be specified in the user program. It is also possible to manoeuvre the robot using a joystick, either the position or orientation of the end-effector are controlled or the joint angles.

The commands specified in the user program are sent to the path generator where a geometrical path for the end-effector pose is calculated. The geometrical path for the end-effector is then converted to a geometrical path for the  $n$  joints. Next step is to take the geometrical path and make a trajectory over time. Here, the desired velocity and acceleration given in the user program are used. Also physical limitations of the velocities, accelerations and motor torques are used. The problem to get the trajectory over time given the limitations can be formulated as an optimisation problem, see e.g. Verscheure et al. (2009) and Ardeshiri et al. (2011). A controller is needed to be able to follow the desired trajectory calculated in the path planner. The controller generates the control signals for the actuators, i.e., electrical motors, and consists of both feed-forward and feedback. The feed-forward part takes the desired joint angles and calculates the desired actuator torques, but also the corresponding actuator positions. The feedback loop corrects for model errors and disturbances by comparing the measured and desired actuator positions. The output from the controller is the desired actuator torque for each actuator. Actually, the control signals to the actuators are electrical currents. A torque controller is therefore needed which takes the actuator torques from the controller and calculates the corresponding current. The torque controller is assumed to be significantly faster than the robot dynamic and therefore omitted in this work.

The path planner and controller make an extensive use of models. The models can be divided into kinematic and dynamic models, where the kinematic models describe the relation between the pose of the end-effector and the joint angles, see Section 2.2 for details. The dynamic models describe the motion of the robot given the forces and torques acting on the robot, see Section 2.3.

The main references used in the remaining of this chapter are Spong et al. (2005); Sciavicco and Siciliano (2000); Siciliano and Khatib (2008); Craig (1989), and Kozłowski (1998).

## 2.2 Kinematic Models

The kinematics describe the motion of bodies relative to each other. The position, velocity, acceleration and higher order time derivatives of the pose are studied regardless the forces and torques acting upon the bodies. The kinematic relations contain therefore only the geometrical properties of the motion over time. The kinematic relations can be derived from simple coordinate transformations between different coordinate systems.

The kinematics for an industrial robot can be divided in to two different problems. The first problem consists of the relations between the known joint positions and the unknown pose of the end-effector. The second problem is the opposite, consisting of the relations between the known pose of the end-effector and unknown joint positions. These are called forward and inverse kinematics, respectively.

### 2.2.1 Coordinate Transformation

Let the vector  $r_j$  be fixed in frame  $j$ . The transformation to frame  $i$  can be written as

$$r_i = r_{j/i} + Q_{j/i}r_j, \quad (2.1)$$

where  $r_{j/i}$  is the vector from the origin in system  $i$  to the origin in system  $j$  and  $Q_{j/i}$  is the rotation matrix representing the orientation of system  $j$  with respect to frame  $i$ . A three dimensional rotation can be seen as a rotation of system  $j$  about the  $z$ -axis of system  $i$  by  $\phi$  rad followed by a rotation about the current  $y_j$ -axis by  $\theta$  rad, ending with a rotation about the current  $z_j$ -axis by  $\psi$  rad. The angles  $\phi$ ,  $\theta$  and  $\psi$  are known as the Euler angles. Rotation using Euler angles are intuitive but can cause singularities. Unit quaternions are more reliable but they are not as intuitive as Euler angles. Another representation of a rotation is the axis/angle representation.

A serial industrial robot with  $n$  DOF consists of  $n + 1$  rigid links (bodies) attached to each other in series. Let the links be numerated 0 to  $n$ , where link  $n$  is the end-effector and link 0 is the world, and let coordinate frame  $i$  be fixed in link  $i - 1$ . The pose of frame  $i$  relative to frame  $i - 1$  is assumed to be known, i.e.,  $r_{i/i-1}$  and  $Q_{i/i-1}$ . The transformation between two connected links can therefore be written as

$$r_{i-1} = r_{i/i-1} + Q_{i/i-1}r_i, \quad (2.2)$$

using (2.1). Iterating (2.2) over all links will give a relation of the pose of link  $n$  expressed in frame 0, which can be seen as the pose of the end-effector expressed in the world frame, for a robot application. Equation (2.2) is described by a sum and a matrix multiplication which can be rewritten as one matrix multiplication

if homogeneous coordinates are introduced according to

$$r_i^h = \begin{pmatrix} r_i \\ 1 \end{pmatrix}. \quad (2.3)$$

Equation (2.2) can now be reformulated as

$$r_{i-1}^h = \begin{pmatrix} \mathcal{Q}_{i/i-1} & r_{i/i-1} \\ 0 & 1 \end{pmatrix} r_i^h = H_{i/i-1} r_i^h. \quad (2.4)$$

The advantage with this representation is that the relation of the pose of link  $n$  expressed in link 0 can be written as only matrix multiplications according to

$$r_0^h = H_{1/0} \cdot \dots \cdot H_{n/n-1} r_n^h = \left( \prod_{i=1}^n H_{i/i-1} \right) r_n^h = \begin{pmatrix} \mathcal{Q}_{n/0} & r_{n/0} \\ 0 & 1 \end{pmatrix} r_n^h = H_{n/0} r_n^h, \quad (2.5)$$

where  $r_{n/0}$  represents the position and  $\mathcal{Q}_{n/0}$  the orientation of the end-effector frame with respect to frame 0,  $r_n^h$  is an arbitrary vector expressed in the end-effector frame, and  $H_{i/i-1}$  is given by (2.4).

## 2.2.2 Forward Kinematics

The forward kinematics for a  $n$  DOF industrial robot is the problem of determining the position and orientation of the end-effector frame relative the world frame given the joint angles  $q_a = (q_{a1} \ \dots \ q_{an})^\top$ . The world frame is a user defined frame where the robot is located, e.g. the industrial floor. The position  $p \in \mathbb{R}^3$  is given in Cartesian coordinates and the orientation  $\phi \in \mathbb{R}^3$  is given in Euler angles, or quaternions if desirable. The pose is said to be defined in the task space whereas the joint angles are said to be in the joint space. The forward kinematics has a unique solution for a serial robot whereas there exist several solutions for a parallel arm robot such as ABB IRB360 in Figure 2.1b.

The kinematic relations can be written as a nonlinear function according to

$$X = \begin{pmatrix} p \\ \phi \end{pmatrix} = \Upsilon(q_a), \quad (2.6)$$

where  $\Upsilon(q_a) : \mathbb{R}^n \rightarrow \mathbb{R}^6$  is a nonlinear function given by the homogeneous transformation matrix

$$H_{n/0} = \begin{pmatrix} \mathcal{Q}_{n/0} & r_{n/0} \\ 0 & 1 \end{pmatrix} \quad (2.7)$$

in (2.5). The position of the end-effector frame, i.e., the first three rows in  $\Upsilon(q_a)$ , is given by  $r_{n/0}$  and the orientation of the end-effector frame, i.e., the last three rows in  $\Upsilon(q_a)$ , is given by  $\mathcal{Q}_{n/0}$ . The construction of  $H_{n/0}$ , i.e., determination of all the  $H_{i/i-1}$  matrices in (2.4), can be difficult for complex robot structures. A systematic way to assign coordinate frames to simplify the derivation of  $H_{n/0}$  is the so-called Denavit-Hartenberg (DH) convention (Denavit and Hartenberg, 1955).

Taking the derivative of (2.6) with respect to time gives a relation between the joint velocities  $\dot{q}_a$ , the linear velocity  $v$ , and angular velocity  $\omega$  of the end-effector according to

$$\dot{X} = \begin{pmatrix} \dot{p} \\ \dot{\phi} \end{pmatrix} = \begin{pmatrix} v \\ \omega \end{pmatrix} = \frac{\partial \Upsilon(q_a)}{\partial q_a} \dot{q}_a = \mathcal{J}(q_a) \dot{q}_a, \quad (2.8)$$

where  $\mathcal{J}(q_a)$  is the Jacobian matrix of  $\Upsilon(q_a)$ . The linear acceleration  $a$  and angular acceleration  $\psi$  of the end-effector are given by the second time derivative of (2.6) according to

$$\ddot{X} = \begin{pmatrix} \dot{v} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} a \\ \psi \end{pmatrix} = \mathcal{J}(q_a) \ddot{q}_a + \left( \frac{d}{dt} \mathcal{J}(q_a) \right) \dot{q}_a, \quad (2.9)$$

where  $\dot{q}_a$  and  $\ddot{q}_a$  are the joint velocities and accelerations, respectively. The time derivative of the Jacobian can be written as

$$\frac{d}{dt} \mathcal{J}(q_a) = \sum_{i=1}^n \frac{\partial \mathcal{J}(q_{ai})}{\partial q_{ai}} \dot{q}_{ai}. \quad (2.10)$$

Except from calculating velocities, the Jacobian  $\mathcal{J}(q_a)$  can also be used for

- identification of singular configurations,
- trajectory planning, and
- transformation of forces and torques acting on the end-effector to the corresponding joint torques.

Another possible use of the Jacobian, which is of interest in this thesis, is when calculating an approximation of the distribution of the pose given the distribution of the joint positions, as described in Section 3.4.

The Jacobian in (2.8) is known as the analytical Jacobian. Another Jacobian is the geometrical Jacobian. The difference between the analytical and geometrical Jacobian affects only the angular velocity and acceleration. The geometrical Jacobian is not considered in this work.

### 2.2.3 Inverse Kinematics

In practice, often the position  $p$  and orientation  $\phi$  of the end-effector are given by the operating program and the corresponding joint angles  $q_a$  are required for the control loop. An inverse kinematic model is needed in order to get the corresponding joint angles. The inverse kinematics is a substantial harder problem which can have several solutions or no solutions at all for a serial robot, as opposed to the forward kinematics. For a parallel arm robot the inverse kinematics is much easier and gives a unique solution. In principle, the nonlinear system of equations in (2.6) must be inverted, i.e.,

$$q_a = \Upsilon^{-1}(X). \quad (2.11)$$

If an analytical solution does not exist, a numerical solver must be used in every time step.

Given the joint angles  $q_a$ , the linear velocity  $v$  and angular velocity  $\omega$ , i.e.,  $\dot{X}$ , then the angular velocities  $\dot{q}_a$  can be calculated from (2.8) according to

$$\dot{q}_a = \mathcal{J}^{-1}(q_a)\dot{X}, \quad (2.12)$$

if the Jacobian is square and nonsingular. The Jacobian is a square matrix when  $n = 6$  since  $\Upsilon(q_a)$  has six rows, three for the position and three for the orientation. The singularity depends on the joint angles  $q_a$ . The Jacobian is singular if the robot is at the boundary of the work space, i.e., outstretched or retracted, or if one or more axes are aligned. The intuitive explanation is that the robot has lost one or more degrees of freedom when the Jacobian is singular. It is then not possible to move the end-effector in all directions regardless of how large velocities the controller applies.

If the robot has less than six joints, i.e., the Jacobian has less than six columns, then the inverse velocity kinematics has a solution if and only if

$$\text{rank } \mathcal{J}(q_a) = \text{rank} \begin{pmatrix} \mathcal{J}(q_a) & \dot{X} \end{pmatrix}, \quad (2.13)$$

that is,  $\dot{X}$  lies in the range space of the Jacobian. If instead the robot has more than six joints, then the inverse velocity kinematics is given by

$$\dot{q}_a = \mathcal{J}^\dagger(q_a)\dot{X} + (\mathbf{I} - \mathcal{J}^\dagger(q_a)\mathcal{J}(q_a))b, \quad (2.14)$$

where  $\mathcal{J}^\dagger(q_a)$  is the pseudo inverse (Mitra and Rao, 1971) of  $\mathcal{J}(q_a)$  and  $b \in \mathbb{R}^n$  is an arbitrary vector. See Spong et al. (2005) for more details.

The angular acceleration  $\ddot{q}_a$  can be calculated in a similar way from (2.9), when  $\ddot{X}$ ,  $q_a$ , and  $\dot{q}_a$  are known and if the Jacobian is invertible, according to

$$\ddot{q}_a = \mathcal{J}^{-1}(q_a) \left( \ddot{X} - \frac{d}{dt} (\mathcal{J}(q_a)) \dot{q}_a \right). \quad (2.15)$$

## 2.3 Dynamic Models

The dynamics describes the motion of a body considering the forces and torques causing the motion. The dynamic equations can be derived from the Newton-Euler formulation or Lagrange's equation, see e.g. Goldstein et al. (2002). Here, only Lagrange's equation will be covered.

### 2.3.1 Rigid Link Model

For Lagrange's equation the Lagrangian  $L(q, \dot{q})$  is defined as the difference between the kinetic and potential energies. Let the kinetic and potential energies be denoted by  $K(q, \dot{q})$  and  $P(q)$ , respectively. The argument  $q$  to the Lagrangian is a set of generalised coordinates. A system with  $n$  DOF can be described by  $n$  generalised coordinates  $q = (q_1 \ \dots \ q_n)^\top$ , e.g. position, velocity, angle or angular

velocity that describe the system. The dynamic equations are given by Lagrange's equation

$$\frac{d}{dt} \frac{\partial L(q, \dot{q})}{\partial \dot{q}_i} - \frac{\partial L(q, \dot{q})}{\partial q_i} = \tau_i, \quad (2.16)$$

where  $L(q, \dot{q}) = K(q, \dot{q}) - P(q)$  and  $\tau_i$  is the generalised force associated with  $q_i$ . For an industrial robot, the generalised coordinates are the joint angles  $q_a$ , and the generalised forces are the corresponding motor torques. The equations given by Lagrange's equation can be summarised by

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = \tau_m^a, \quad (2.17)$$

where  $M(q)$  is the inertia matrix,  $C(q, \dot{q})$  is the Coriolis- and centrifugal terms,  $G(q)$  is the gravitational torque and  $\tau_m^a = (\tau_{m1}^a \ \dots \ \tau_{mn}^a)^T$ . Note that the equation is expressed on the arm side of the gearbox, that is, the applied motor torque  $\tau_{mi}$  must be converted from the motor side to the arm side of the gearbox. This is done by multiplication by the gear ratio  $\eta_i > 1$ , according to

$$\tau_{mi}^a = \tau_{mi} \eta_i. \quad (2.18)$$

Each link in the rigid body dynamics in (2.17) is described by a mass, three lengths describing the geometry, three lengths describing the centre of mass and six inertia parameters. The centre of gravity and the inertia are described in the local coordinate system. Each link is thus described by 13 parameters that have to be determined, see e.g. Kozłowski (1998).

The model can also be extended with a linear or nonlinear friction torque  $F(\dot{q})$ . A classical model is

$$F(\dot{q}) = f_v \dot{q} + f_c \text{sign}(\dot{q}), \quad (2.19)$$

where  $f_v$  is the viscous friction and  $f_c$  is the Coulomb friction. More advanced models are the LuGre model (Åström and Canudas de Wit, 2008) and the Dahl model, see Dupont et al. (2002) for an overview. In this work, a smooth static friction model, suggested in Feeny and Moon (1994), given by

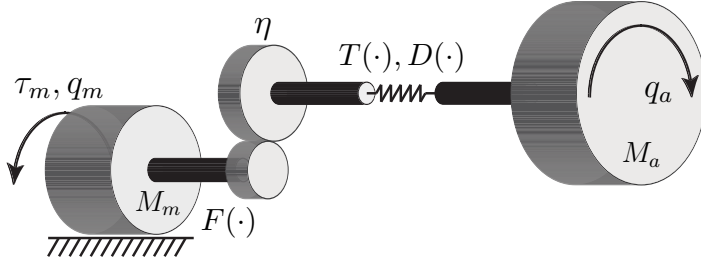
$$F(\dot{q}) = f_v \dot{q} + f_c (\mu_k + (1 - \mu_k) \cosh^{-1}(\beta \dot{q})) \tanh(\alpha \dot{q}), \quad (2.20)$$

is used. Here, the friction is only dependent on the velocity of the generalised coordinates. In practice, the measured friction curve on a real robot shows a dependency on the temperature and the dynamical load of the end-effector, as described in Carvalho Bittencourt et al. (2010).

### 2.3.2 Flexible Joint Model

In practice, the joints, specially the gearboxes, are flexible. Each joint can therefore be modelled as a torsional spring and damper pair between the motor and arm side of the gearbox, see Figure 2.3. The system now has  $2n$  DOF and can be





**Figure 2.3:** Flexible joint model where the arm angular position  $q_a$  is related to the motor angle  $q_m$  and motor torque  $\tau_m$  via the gear ratio  $\eta$  and the spring-damper pair modelled by  $T(\cdot)$  and  $D(\cdot)$ . The motor friction is modelled by  $F(\cdot)$ .

described by the simplified flexible joint model

$$M_a(q_a)\ddot{q}_a + C(q_a, \dot{q}_a) + G(q_a) = T(q_m^a - q_a) + D(\dot{q}_m^a - \dot{q}_a), \quad (2.21a)$$

$$M_m\ddot{q}_m^a + F(\dot{q}_m^a) = \tau_m^a - T(q_m^a - q_a) - D(\dot{q}_m^a - \dot{q}_a), \quad (2.21b)$$

where  $q_a \in \mathbb{R}^n$  are the arm angles,  $q_m^a \in \mathbb{R}^n$  are the motor angles. The superscript  $a$  indicates that the motor angles are expressed on the arm side of the gearbox, i.e.,  $q_{mi}^a = q_{mi}/\eta_i$  where  $q_{mi}$  is the motor angle on the motor side of the gear box for joint  $i$ . The same applies for the motor torque  $\tau_m^a$  according to (2.18). Further,  $M_a(q_a)$  is the inertia matrix for the arms,  $M_m$  is the inertia for the motors,  $C(q_a, \dot{q}_a)$  is the Coriolis- and centrifugal terms,  $G(q_a)$  is the gravitational torque and  $F(\dot{q}_m^a)$  is the friction torque. Moreover,  $T(q_m^a - q_a)$  is the stiffness torque and  $D(\dot{q}_m^a - \dot{q}_a)$  is the damping torque. Both the stiffness and damping torque can be modelled as linear or nonlinear. The simplified flexible joint model assumes that the couplings between the arms and motors are neglected, which is valid if the gear ratio is high (Spong, 1987). In the complete flexible link model the term  $S(q_a)\ddot{q}_m^a$  is added to (2.21a) and the term  $S^T(q_a)\ddot{q}_a$  as well as a Coriolis- and centrifugal term are added to (2.21b), where  $S(q_a)$  is a strictly upper triangular matrix. A complete description of the simplified and complete flexible link model can be found in De Luca and Book (2008).

The flexible joint model described above assumes that the spring and damper pairs are in the rotational direction. Another extension is to introduce multidimensional spring and damper pairs in the joints to deal with flexibilities in other directions than the rotational direction, where each dimension of the spring and damper pairs corresponds to two DOF. If the links are flexible, then it can be modelled by dividing each flexible link into several parts connected by multidimensional spring and damper pairs. This leads to extra non-actuated joints, hence more DOFs. This is known as the extended flexible joint model and a thorough description can be found in Moberg and Hanssen (2007).



# 3

---

## Estimation Theory

Different techniques for nonlinear estimation are presented in this chapter. The estimation problem for the discrete time nonlinear state space model

$$x_{k+1} = f(x_k, u_k, v_k; \theta), \quad (3.1a)$$

$$y_k = h(x_k, u_k, e_k; \theta), \quad (3.1b)$$

is to find the state vector  $x_k \in \mathbb{R}^{n_x}$  at time  $k$  and the unknown model parameters  $\theta$  given the measurements  $y_k \in \mathbb{R}^{n_y}$   $k = 1, \dots, N$ . In this work the focus is on nonlinear models with additive process noise  $v_k$  and measurement noise  $e_k$  given by

$$x_{k+1} = f(x_k, u_k) + g(x_k)v_k, \quad (3.2a)$$

$$y_k = h(x_k, u_k) + e_k, \quad (3.2b)$$

where the *probability density functions* (PDFs) for the process noise,  $p_v(v)$ , and measurement noise,  $p_e(e)$ , are known.

The estimation problem can be divided into the filtering problem where only previous measurements up to the present time are available, see Section 3.1 and the smoothing problem where both previous and future measurements are available, see Section 3.2. For the case with estimation of unknown parameters, the expectation maximisation algorithm can be used as described in Section 3.3. Section 3.4 presents how to obtain the distribution of a parameter after a nonlinear transformation if the distribution is known before the transformation.

### 3.1 The Filtering Problem

The filtering problem can be seen as calculation/approximation of the posterior density  $p(x_k|y_{1:k})$  using all measurements up to time  $k$ , where

$$y_{1:k} = \{y_1, \dots, y_k\}, \quad (3.3)$$

and the known conditional densities for the state transition and measurements, i.e.,

$$x_{k+1} \sim p(x_{k+1}|x_k), \quad (3.4a)$$

$$y_k \sim p(y_k|x_k), \quad (3.4b)$$

which are given by the model (3.2). Using Bayes' law,

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}, \quad (3.5)$$

and the Markov property for the state space model,

$$p(x_n|x_1, \dots, x_{n-1}) = p(x_n|x_{n-1}), \quad (3.6)$$

repeatedly, the optimal solution for the Bayesian inference (Jazwinski, 1970) can be obtained according to

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})}, \quad (3.7a)$$

$$p(x_{k+1}|y_{1:k}) = \int_{\mathbb{R}^{n_x}} p(x_{k+1}|x_k)p(x_k|y_{1:k}) dx_k, \quad (3.7b)$$

where  $k = 1, 2, \dots, N$  and

$$p(y_k|y_{1:k-1}) = \int_{\mathbb{R}^{n_x}} p(y_k|x_k)p(x_k|y_{1:k-1}) dx_k. \quad (3.7c)$$

The solution to (3.7) can in most cases not be given by an analytical expression. For the special case of linear dynamics, linear measurements and additive Gaussian noise the Bayesian recursions in (3.7) have an analytical solution, which is known as the *Kalman filter* (KF) (Kalman, 1960). For nonlinear and non-Gaussian systems, the posterior density can not in general be expressed with a finite number of parameters. Instead approximative methods must be used. Here, two approximative solutions are considered; the *extended Kalman filter* (EKF) and the *particle filter* (PF). Another approximative solution not considered here is the *unscented Kalman filter* (UKF) (Julier et al., 1995).

#### 3.1.1 The Extended Kalman Filter (EKF)

The *extended Kalman filter* (EKF) (Anderson and Moore, 1979; Kailath et al., 2000) solves the Bayesian recursions in (3.7) using a first order Taylor expansion of the nonlinear system equations around the previous estimate. The approximation is acceptable if the nonlinearity is almost linear or if the *signal to noise ratio*

(SNR) is high. The Taylor expansion requires derivatives of the nonlinear system equations, which can be obtained by symbolic or numeric differentiation.

The process noise  $v_k$  and measurement noise  $e_k$  are assumed to be Gaussian with zero means and covariance matrices  $Q_k$  and  $R_k$ , respectively. The time update,  $\hat{x}_{k|k-1}$  and  $P_{k|k-1}$ , and the measurement update,  $\hat{x}_{k|k}$  and  $P_{k|k}$ , for the EKF with the nonlinear model (3.2) can be derived relatively easy using the first order Taylor approximation and the KF equations. The time and measurement updates are presented in Algorithm 1, where the notation  $\hat{x}_{k|k}$ ,  $P_{k|k}$ ,  $\hat{x}_{k|k-1}$  and  $P_{k|k-1}$  denotes estimates of the state vector  $x$  and covariance matrix  $P$  at time  $k$  using measurements up to time  $k$  and  $k-1$ , respectively. It is also possible to use a second order Taylor approximation when the EKF equations are derived, see e.g. (Gustafsson, 2010).

---

**Algorithm 1** The Extended Kalman Filter (EKF)
 

---

Initialisation

$$\hat{x}_{0|0} = x_0 \quad (3.8a)$$

$$P_{0|0} = P_0 \quad (3.8b)$$

Time update

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k) \quad (3.9a)$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + G_k Q_k G_k^T \quad (3.9b)$$

$$F_k = \left. \frac{\partial f(x, u_k)}{\partial x} \right|_{x=\hat{x}_{k-1|k-1}} \quad (3.9c)$$

$$G_k = g(\hat{x}_{k-1|k-1}) \quad (3.9d)$$

Measurement update

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \quad (3.10a)$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (y_k - h(\hat{x}_{k|k-1}, u_k)) \quad (3.10b)$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \quad (3.10c)$$

$$H_k = \left. \frac{\partial h(x, u_k)}{\partial x} \right|_{x=\hat{x}_{k|k-1}} \quad (3.10d)$$


---

### 3.1.2 The Particle Filter (PF)

The *particle filter* (PF) (Doucet et al., 2001; Gordon et al., 1993; Arulampalam et al., 2002) solves the Bayesian recursions using stochastic integration. The PF approximates the posterior density  $p(x_k|y_{1:k})$  by a large set of  $N$  particles  $\{x_k^i\}_{i=1}^N$ , where each particle has an assigned relative weight  $w_k^i$ , chosen such that  $\sum_{i=1}^N w_k^i = 1$ . The position and weight of each particle approximate the posterior density in such a way that a high weight corresponds to a high probability

at the point given by the particle. The PF updates the particle location and the corresponding weights recursively with each new observed measurement. The particle method for solving the Bayesian recursion in (3.7) has been known for long but the PF filter has in practice been inapplicable due to degeneracy of the particles. The problem was solved in Gordon et al. (1993) by introducing a re-sampling step.

Compared to the EKF, the PF does not suffer from linearisation errors and can handle non-Gaussian noise models. Hard constraints on the state variables can also be incorporated into the estimation problem. Theoretical results show that the approximated posterior density converges to the true density when the number of particles tends to infinity, see e.g. Doucet et al. (2001). The PF is summarised in Algorithm 2, where the proposal density  $p^{\text{prop}}(x_{k+1}^i|x_k^i, y_{k+1})$  can be chosen arbitrary as long as it is possible to draw samples from it. For small SNR the conditional prior of the state vector, i.e.,  $p(x_{k+1}^i|x_k^i)$ , is a good choice (Gordon et al., 1993). Using the conditional prior, the weight update can be written as  $w_k^i = w_{k-1}^i p(y_k|x_k^i)$ . The optimal proposal should be to use the conditional density  $p(x_k|x_{k-1}^i, y_k)$  (Doucet et al., 2000). The problem is that it is difficult to sample from it and also to calculate the weights. In this work the optimal proposal density, approximated by an EKF (Doucet et al., 2000; Gustafsson, 2010), is used. The approximated proposal density can be written as

$$p^{\text{prop}}(x_k|x_{k-1}^i, y_k) \approx \mathcal{N}\left(x_k; f(x_{k-1}^i) + K_k^i(y_k - \hat{y}_k^i), (H_k^{i,\text{T}} R_k^{\text{T}} H_k^i + Q_{k-1}^{\text{T}})^{\dagger}\right), \quad (3.11)$$

where  $\dagger$  denotes the pseudo-inverse, and where

$$K_k^i = Q_{k-1} H_k^{i,\text{T}} (H_k^i Q_{k-1} H_k^{i,\text{T}} + R_k)^{-1}, \quad (3.12a)$$

$$H_k^i = \left. \frac{\partial h(x_k)}{\partial x_k} \right|_{x_k=f(x_{k-1}^i)}, \quad (3.12b)$$

$$\hat{y}_k^i = h(f(x_{k-1}^i)), \quad (3.12c)$$

The matrices in (3.12) are assumed to be evaluated for each particle state. The approximated optimal proposal density gives a weight update according to

$$w_k^i = w_{k-1}^i p(y_k|x_{k-1}^i), \quad (3.13a)$$

where

$$p(y_k|x_{k-1}^i) = \mathcal{N}(y_k; \hat{y}_k^i, H_k^i Q_{k-1} H_k^{i,\text{T}} + R_k). \quad (3.13b)$$

The state estimate for each sample  $k$  is often chosen as the minimum mean square estimate

$$\hat{x}_{k|k} = \arg \min_{x_k} \text{E} \left[ (\hat{x}_k - x_k)^2 | y_{1:k} \right] \quad (3.14)$$

which has the solution

$$\hat{x}_{k|k} = \mathbb{E} [x_k | y_{1:k}] = \int_{\mathbb{R}^{n_x}} x_k p(x_k | y_{1:k}) dx_k \approx \sum_{i=1}^N w_k^i x_k^i. \quad (3.15)$$

---

**Algorithm 2** The Particle Filter (PF)
 

---

- 1: Generate  $N$  samples  $\{x_0^i\}_{i=1}^N$  from  $p(x_0)$ .
- 2: Compute

$$w_k^i = w_{k-1}^i \cdot \frac{p(y_k | x_k^i) p(x_k^i | x_{k-1}^i)}{p^{\text{prop}}(x_k^i | x_{k-1}^i, y_k)}$$

and normalise, i.e.,  $\bar{w}_k^i = w_k^i / \sum_{j=1}^N w_k^j$ ,  $i = 1, \dots, N$ .

- 3: [Optional]. Generate a new set  $\{x_k^{i*}\}_{i=1}^N$  by resampling with replacement  $N$  times from  $\{x_k^i\}_{i=1}^N$ , with probability  $\bar{w}_k^i = \Pr\{x_k^{i*} = x_k^i\}$  and reset the weights to  $1/N$ .
- 4: Generate predictions from the proposal density

$$x_{k+1}^i \sim p^{\text{prop}}(x_{k+1}^i | x_k^{i*}, y_{k+1}), \quad i = 1, \dots, N.$$

- 5: Increase  $k$  and continue to step 2.
- 

## 3.2 The Smoothing Problem

The smoothing problem is essentially the same as the filtering problem except that future measurements are used instead of only measurements up to present time  $k$ . In other words the smoothing problem can be seen as computation/approximation of the density  $p(x_k | y_{1:l})$ , where  $l > k$ . The smoothing problem solves the same equations as the filter problem except that future measurements are available. Approximative solutions must be used here as well when the problem is nonlinear and non-Gaussian. Different types of smoothing problems are possible, e.g. fixed-lag, fixed-point and fixed-interval smoothing (Gustafsson, 2010). Here the fixed-interval smoothing problem is considered and the *extended Kalman smoother* (EKS) (Yu et al., 2004) is used. The fixed-interval smoothing problem is an offline method that use all available measurements  $y_{1:N}$ . The EKS, using the Rauch-Tung-Striebel formulas, is presented in Algorithm 3.

## 3.3 The Expectation Maximisation Algorithm

The *maximum likelihood* (ML) method (Fisher, 1912, 1922) is a well known tool for estimating unknown model parameters. The idea with the ML method is to find the unknown parameters  $\theta$  such that the measurements  $y_{1:N}$  become as likely

**Algorithm 3** The Extended Kalman Smoother (EKS)

1: Run the EKF and store the time and measurements updates,  $\hat{x}_{k|k-1}$ ,  $\hat{x}_{k|k}$ ,  $P_{k|k-1}$  and  $P_{k|k}$ .

2: Initiate the backward time recursion,

$$\hat{x}_{N|N}^s = \hat{x}_{N|N}, \quad (3.16a)$$

$$P_{N|N}^s = P_{N|N}. \quad (3.16b)$$

3: Apply the backward time recursion for  $k = N - 1, \dots, 1$ ,

$$\hat{x}_{k|N}^s = \hat{x}_{k|k} + P_{k|k} F_k^\top P_{k+1|k}^{-1} (\hat{x}_{k+1|N}^s - \hat{x}_{k+1|k}), \quad (3.17a)$$

$$P_{k|N}^s = P_{k|k} + P_{k|k} F_k^\top P_{k+1|k}^{-1} (P_{k+1|N}^s - P_{k+1|k}) P_{k+1|k}^{-1} F_k P_{k|k}, \quad (3.17b)$$

$$F_k = \left. \frac{\partial f(x, u_k)}{\partial x} \right|_{x=\hat{x}_{k|k}}. \quad (3.17c)$$

as possible. In other words,

$$\hat{\theta}^{\text{ML}} = \arg \max_{\theta \in \Theta} p_\theta(y_{1:N}), \quad (3.18)$$

where  $p_\theta(y_{1:N})$  is the PDF of the observations, i.e., the likelihood, parametrised with the parameter  $\theta$ . It is common to take the logarithm of the PDF,

$$L_\theta(y_{1:N}) = \log p_\theta(y_{1:N}), \quad (3.19)$$

and find the parameter  $\theta$  that maximises (3.19), i.e.,

$$\hat{\theta}^{\text{ML}} = \arg \max_{\theta \in \Theta} L_\theta(y_{1:N}). \quad (3.20)$$

These two problems are equivalent since the logarithm is a monotonic function. The ML problem can be solved using a standard optimisation method, see e.g. Nocedal and Wright (2006). The solution can however be hard to find which has led to the development of the *expectation maximisation* (EM) algorithm.

The EM algorithm was originally invented by Dempster et al. (1977). See McLachlan and Krishnan (2008) for theoretical results and examples. The principal idea with the EM algorithm is to introduce variables  $x_{1:N}$  which are not observed directly. The variables  $x_{1:N}$  can instead be observed indirectly from  $y_{1:N}$ . Take now the joint log-likelihood function

$$L_\theta(y_{1:N}, x_{1:N}) = \log p_\theta(y_{1:N}, x_{1:N}) \quad (3.21)$$

of the observed variables  $y_{1:N}$  and the variables  $x_{1:N}$ . Equation (3.21) cannot be used directly since  $x_{1:N}$  are unknown. Instead, calculate

$$\Gamma(\theta; \theta_l) = E_{\theta_l} [\log p_\theta(y_{1:N}, x_{1:N}) | y_{1:N}], \quad (3.22)$$

where  $E_{\theta_l}[\cdot | \cdot]$  is the conditional mean with respect to a PDF defined by the parameter  $\theta_l$  and  $p_\theta(\cdot)$  means that the PDF is parametrised by  $\theta$ . It can now be



shown (Dempster et al., 1977) that any  $\theta$ , such that

$$\Gamma(\theta; \theta_l) > \Gamma(\theta_l; \theta_l), \quad (3.23)$$

implies that

$$L_\theta(y_{1:N}) > L_{\theta_l}(y_{1:N}). \quad (3.24)$$

Hence, maximising  $\Gamma(\theta; \theta_l)$  provides a sequence  $\theta_l$ ,  $l = 1, 2, \dots$ , which approximates  $\hat{\theta}^{\text{ML}}$  better and better for every iteration. A possible stop criterion for the EM algorithm could be when the change in  $\theta$ , between two iterations, is small enough. The EM algorithm is summarised in Algorithm 4.

---

**Algorithm 4** The Expectation Maximisation (EM) Algorithm

---

1: Select an initial value  $\theta_0$  and set  $l = 0$ .

2: Expectation Step (E-step): Calculate

$$\Gamma(\theta; \theta_l) = \mathbb{E}_{\theta_l} [\log p_\theta(y_{1:N}, x_{1:N}) | y_{1:N}].$$

3: Maximisation Step (M-step): Compute

$$\theta_{l+1} = \arg \max_{\theta \in \Theta} \Gamma(\theta; \theta_l).$$

4: If converged, stop. If not, set  $l = l + 1$  and go to step 2.

---

## 3.4 Estimation of Transformed PDFs

Let the nonlinear transformation  $z = f(x)$  be given and assume  $x$  to be Gaussian distributed according to

$$x \sim \mathcal{N}(\mu_x, P_x). \quad (3.25)$$

The distribution of  $z$  can then be approximated by

$$z \sim \mathcal{N}(\mu_z, P_z), \quad (3.26)$$

where  $\mu_z$  and  $P_z$  can be calculated using different methods. A first order Taylor approximation of  $z = f(x)$  gives the mean and covariance matrix of  $z$  according to

$$\mu_z = f(\mu_x), \quad (3.27a)$$

$$P_z = f'(\mu_x) P_x (f'(\mu_x))^T. \quad (3.27b)$$

This method is known as Gauss' approximation formula. A more accurate method is the *Monte Carlo* (MC) transformation where  $N$  samples  $x^i$ ,  $i = 1, \dots, N$ , are generated from the distribution of  $x$ . The  $x^i$  values give the corresponding values of  $z$  using the nonlinear transformation, i.e.,  $z^i = f(x^i)$ . When the  $N$  samples of

$z$  are given, the mean and covariance matrix of  $z$  can be calculated according to

$$\mu_z = \frac{1}{N} \sum_{i=1}^N z^i, \quad (3.28a)$$

$$P_z = \frac{1}{N-1} \sum_{i=1}^N (z^i - \mu_z)(z^i - \mu_z)^\top. \quad (3.28b)$$

The MC approach can also be used when  $x$  has another distribution than the Gaussian distribution, see Section 7.4.1. Another method, not included here, is the unscented transformation, see e.g. Julier and Uhlmann (2004).

In this thesis, the approximation formulas are used to obtain the distribution of the pose of the end-effector when the distribution of the arm angular positions is given. In other words, Gauss' approximation formula gives the mean and covariance matrix of  $X$  in (2.6) according to

$$\hat{X} = \Upsilon(\hat{q}_a), \quad (3.29a)$$

$$P_{\hat{X}} = \mathcal{J}(\hat{q}_a) P_{\hat{q}_a} \mathcal{J}^\top(\hat{q}_a), \quad (3.29b)$$

where  $\mathcal{J}(\cdot)$  is the Jacobian matrix in (2.8). The mean and covariance matrix for the arm angular positions, i.e.,  $\hat{q}_a$  and  $P_{\hat{q}_a}$ , are given from e.g. the EKF or PF.

# 4

---

## Models

The models used for simulation and estimation will be presented in this chapter. The robot model is presented in Section 4.1 and the accelerometer model in Section 4.2. Section 4.3 presents how the bias component in e.g. an accelerometer can be modelled and Section 4.4 presents four different state space models for estimation.

### 4.1 Two DOF Rigid Link Flexible Joint Robot Model

This section describes the kinematic and dynamic models for a two-link nonlinear flexible robot, corresponding to joint two and three for a serial six DOF industrial robot. Figure 2.2 shows how the joints are numbered. The robot model is used for simulation and estimation of the TCP.

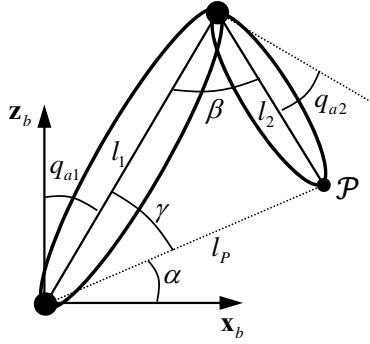
Here, the forward kinematic equations and dynamic equations are given without any derivations, see Section 2.2.2 and 2.3 for methods to derive them. The inverse kinematic model is explained in more details.

#### 4.1.1 Forward Kinematic Model

Only the position  $x$  and  $z$  of the forward kinematic model for the robot in Figure 4.1 is considered since the robot only contains two DOF. The kinematic model can be expressed as

$$\mathcal{P} = \begin{pmatrix} x \\ z \end{pmatrix} = \Upsilon(q_a), \quad (4.1)$$

where  $\mathcal{P}$  is the Cartesian coordinates for the TCP, expressed in the base coordinate frame  $(x_b \ z_b)^\top$ , denoted by  $Ox_bz_b$ . Note that the robot motion only takes



**Figure 4.1:** Two DOF robot model showing the kinematic properties of the structure.

place in the  $x_b z_b$ -plane, i.e.,  $y_b = 0$ . The function  $\Upsilon$  is the nonlinear forward kinematic function and  $q_a = (q_{a1} \quad q_{a2})^\top$  are the joint angles. The forward kinematics can be derived using the methods described in Section 2.2.1. However, the kinematic equation for this simple robot structure can be derived by considering the geometry in Figure 4.1 directly according to

$$\begin{aligned} \mathcal{P} = \Upsilon(q_a) &= \begin{pmatrix} l_1 \sin q_{a1} + l_2 \sin\left(\frac{\pi}{2} + q_{a1} + q_{a2}\right) \\ l_1 \cos q_{a1} + l_2 \cos\left(\frac{\pi}{2} + q_{a1} + q_{a2}\right) \end{pmatrix} \\ &= \begin{pmatrix} l_1 \sin q_{a1} + l_2 \cos(q_{a1} + q_{a2}) \\ l_1 \cos q_{a1} - l_2 \sin(q_{a1} + q_{a2}) \end{pmatrix}. \end{aligned} \quad (4.2)$$

The Jacobian matrix used for calculation of the velocity in (2.8) is given by

$$\mathcal{J}(q_a) = \begin{pmatrix} l_1 \cos q_{a1} - l_2 \sin(q_{a1} + q_{a2}) & -l_2 \sin(q_{a1} + q_{a2}) \\ -l_2 \cos(q_{a1} + q_{a2}) - l_1 \sin q_{a1} & -l_2 \cos(q_{a1} + q_{a2}) \end{pmatrix}. \quad (4.3)$$

### 4.1.2 Inverse Kinematic Model

An analytical solution to the inverse problem for the robot in Figure 4.1 can be derived using trigonometric identities. The solution is however not unique. Two different sets of joint angles will give the same position. These solutions are known as elbow-up and elbow-down.

The law of cosine gives

$$\cos \gamma = \frac{l_1^2 + l_p^2 - l_2^2}{2l_1 l_p}, \quad (4.4a)$$

$$\cos \beta = \frac{l_1^2 + l_2^2 - l_p^2}{2l_1 l_2} = \sin q_{a2}, \quad (4.4b)$$

where  $l_p^2 = x^2 + z^2$ . The joint angle  $q_{a2}$  can now be obtained directly from (4.4b).

However, the function  $\text{atan2}^1$  will be used to be able to get the solution in the correct quadrant. The angle for joint two can therefore be calculated according to

$$q_{a2} = \text{atan2} \left( \sin q_{a2}, \pm \sqrt{1 - \sin^2 q_{a2}} \right). \quad (4.5)$$

The angle  $\gamma$  can be obtained in the same way,

$$\gamma = \text{atan2} \left( \pm \sqrt{1 - \cos^2 \gamma}, \cos \gamma \right). \quad (4.6)$$

The angle for joint one can now be calculated according to

$$q_{a1} = \frac{\pi}{2} - \gamma - \alpha, \quad (4.7)$$

where  $\alpha = \text{atan2}(z, x)$ . The elbow-up solution is obtained if the plus sign in (4.5) and (4.6) is chosen, otherwise the solution will correspond to the elbow-down solution.

### 4.1.3 Dynamic Model

The dynamic robot model is a joint flexible two-axes model from Moberg et al. (2008), see Figure 4.2. Each link is modelled as a rigid-body and described by mass  $m_i$ , length  $l_i$ , centre of mass  $\xi_i$  and inertia  $J_i$  with respect to the centre of mass. The joints are modelled as a spring damping pair with nonlinear spring torque  $\tau_{si}$  and linear damping  $d_i$ . The deflection in each joint is given by the arm angle  $q_{ai}$  and the motor angle  $q_{mi}$ . The motor characteristics are given by the inertia  $J_{mi}$  and a nonlinear friction torque  $f_i$ . Let

$$q_a = (q_{a1} \quad q_{a2})^T, \quad (4.8a)$$

$$q_m^a = (q_{m1}/\eta_1 \quad q_{m2}/\eta_2)^T, \quad (4.8b)$$

$$\tau_m^a = (\tau_{m1}\eta_1 \quad \tau_{m2}\eta_2)^T, \quad (4.8c)$$

where  $\tau_{mi}$  is the motor torque and  $\eta_i = q_{mi}/q_{ai} > 1$  is the gear ratio. The gear ratio is used to transform the motor angles and motor torques from the motor side of the gear box to the arm side. A dynamic model can be derived as

$$M_a(q_a)\ddot{q}_a + C(q_a, \dot{q}_a) + G(q_a) + T(q_a - q_m^a) + \widetilde{D}(\dot{q}_a - \dot{q}_m^a) = 0, \quad (4.9a)$$

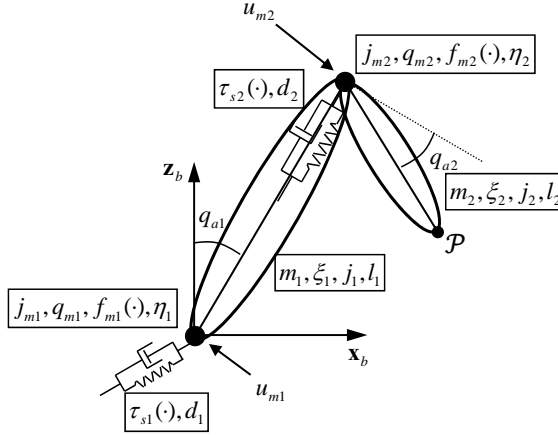
$$M_m\ddot{q}_m^a + F(\dot{q}_m^a) - T(q_a - q_m^a) - \widetilde{D}(\dot{q}_a - \dot{q}_m^a) = \tau_m^a, \quad (4.9b)$$

using Lagrange's equation, as presented in Section 2.3.

The inertia matrix  $M_a(q)$  is partitioned as

$$M_a(q_a) = \begin{pmatrix} M_{11}(q_a) & M_{12}(q_a) \\ M_{21}(q_a) & M_{22}(q_a) \end{pmatrix}, \quad (4.10)$$

<sup>1</sup>  $\text{atan2}(x, y) = \arctan(y/x)$  where the signs of  $x$  and  $y$  are used to determine the correct quadrant. Sometimes the name four quadrant arc tangent is used.



**Figure 4.2:** Serial robot with two DOF. The joints are modelled by nonlinear springs and linear dampers. Nonlinear friction torques are also included.

where

$$M_{11}(q_a) = J_1 + m_1 \xi_1^2 + J_2 + m_2 (l_1^2 + \xi_2^2 - 2l_1 \xi_2 \sin q_{a2}), \quad (4.11a)$$

$$M_{12}(q_a) = M_{21}(q_a) = J_2 + m_2 (\xi_2^2 - l_1 \xi_2 \sin q_{a2}), \quad (4.11b)$$

$$M_{22}(q_a) = J_2 + m_2 \xi_2^2. \quad (4.11c)$$

The motor inertia matrix is

$$M_m = \begin{pmatrix} J_{m1} \eta_1^2 & 0 \\ 0 & J_{m2} \eta_2^2 \end{pmatrix}. \quad (4.12)$$

The Coriolis and centripetal terms are described by

$$C(q_a, \dot{q}_a) = \begin{pmatrix} -m_2 l_1 \xi_2 (2\dot{q}_{a1} \dot{q}_{a2} + \dot{q}_{a2}^2) \cos q_{a2} \\ m_2 l_1 \xi_2 \dot{q}_{a1}^2 \cos q_{a2} \end{pmatrix}, \quad (4.13)$$

the gravity component is

$$G(q_a) = \begin{pmatrix} -g (m_1 \xi_1 \sin q_{a1} + m_2 (l_1 \sin q_{a1} + \xi_2 \cos(q_{a1} + q_{a2}))) \\ -m_2 \xi_2 g \cos(q_{a1} + q_{a2}) \end{pmatrix}, \quad (4.14)$$

and the linear damping term is given by

$$\bar{D}(\dot{q}_a - \dot{q}_m^a) = (D \quad -D) \begin{pmatrix} \dot{q}_a \\ \dot{q}_m^a \end{pmatrix}, \quad (4.15)$$

where

$$D = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix}. \quad (4.16)$$

The nonlinear spring torque is described by

$$T(q_a - q_m^a) = \begin{pmatrix} \tau_{s1}(\Delta_{q1}) \\ \tau_{s2}(\Delta_{q2}) \end{pmatrix}, \quad (4.17)$$

where

$$\Delta_{q_i} = q_{ai} - q_{mi}^a, \quad (4.18a)$$

$$\tau_{si} = \begin{cases} k_i^{low} \Delta_{q_i} + k_{i3} \Delta_{q_i}^3, & |\Delta_{q_i}| \leq \psi_i, \\ \text{sign}(\Delta_{q_i}) (m_{i0} + k_i^{high} (|\Delta_{q_i}| - \psi_i)), & |\Delta_{q_i}| > \psi_i, \end{cases} \quad (4.18b)$$

$$k_{i3} = \frac{k_i^{high} - k_i^{low}}{3\psi_i^2}, \quad (4.18c)$$

$$m_{i0} = k_i^{low} \psi_i + k_{i3} \psi_i^3, \quad (4.18d)$$

where  $\psi_i$  is the breakpoint deflection. Finally, the nonlinear friction torque is given by

$$F(\dot{q}_m^a) = \begin{pmatrix} f_{m1}(\dot{q}_{m1}^a) \\ f_{m2}(\dot{q}_{m2}^a) \end{pmatrix}, \quad (4.19)$$

where

$$f_{mi}(\dot{q}_{mi}^a) = \eta_i (f_{di} \dot{q}_{mi}^a \eta_i + f_{ci} (\mu_{ki} + (1 - \mu_{ki}) \cosh^{-1}(\beta_i \dot{q}_{mi}^a \eta_i)) \tanh(\alpha_i \dot{q}_{mi}^a \eta_i)). \quad (4.20)$$

Numerical values for the parameters can be found in Moberg et al. (2008).

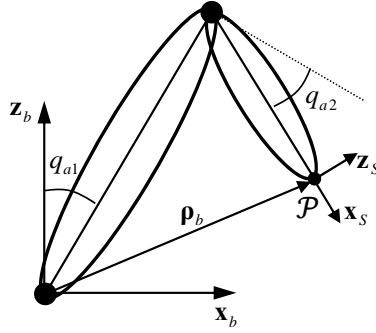
## 4.2 Accelerometer Model

The accelerometer attached to the robot measures the acceleration due to the motion the robot performs, the gravity component and in addition some measurement noise is introduced. When modelling the accelerometer it is also important to include a bias term. The acceleration is measured in a frame  $Ox_s z_s$  fixed to the accelerometer relative an inertial frame. The inertial frame is here chosen to be the world fixed base frame  $Ox_b z_b$ . The acceleration in  $Ox_s z_s$  can thus be expressed as

$$\ddot{\rho}_s(q_a) = \mathcal{R}_{b/s}(q_a) (\ddot{\rho}_b(q_a) + G_b) + \mathfrak{b}^{\text{ACC}}, \quad (4.21)$$

where  $\ddot{\rho}_b(q_a)$  is the acceleration due to the motion and  $G_b = \begin{pmatrix} 0 & g \end{pmatrix}^\top$  models the gravitation, both expressed in the base frame  $Ox_b z_b$ . The bias term is denoted by  $\mathfrak{b}^{\text{ACC}}$  and is expressed in  $Ox_s z_s$ .  $\mathcal{R}_{b/s}(q_a)$  is a rotation matrix that represents the rotation from frame  $Ox_b z_b$  to frame  $Ox_s z_s$ . The rotation matrix  $\mathcal{R}_{b/s}(q_a)$  can be obtained from Figure 4.3 according to

$$\mathcal{R}_{b/s}(q_a) = \begin{pmatrix} \cos(q_{a1} + q_{a2}) & -\sin(q_{a1} + q_{a2}) \\ \sin(q_{a1} + q_{a2}) & \cos(q_{a1} + q_{a2}) \end{pmatrix}. \quad (4.22)$$



**Figure 4.3:** The vector  $\rho_b$  from the origin of frame  $Ox_bz_b$  to the origin of frame  $Ox_s z_s$  is used to calculate the acceleration of the point  $\mathcal{P}$ , i.e., the acceleration measured by the accelerometer which originates from the motion.

The vector  $\ddot{\rho}_b(q_a)$  can be calculated as the second derivative of  $\rho_b(q_a)$  which is shown in Figure 4.3. Using the forward kinematic relation (2.6) for the position only, the vector  $\rho_b$  can be written as

$$\rho_b(q_a) = \Upsilon_{\text{ACC}}(q_a), \quad (4.23)$$

where the index ACC distinguish this model from the model describing the TCP. Taking the derivative of (4.23) with respect to time twice gives

$$\dot{\rho}_b(q_a) = \frac{d}{dt} \Upsilon_{\text{ACC}}(q_a) = \mathcal{J}_{\text{ACC}}(q_a) \dot{q}_a, \quad (4.24a)$$

$$\ddot{\rho}_b(q_a) = \frac{d^2}{dt^2} \Upsilon_{\text{ACC}}(q_a) = \mathcal{J}_{\text{ACC}}(q_a) \ddot{q}_a + \left( \frac{d}{dt} \mathcal{J}_{\text{ACC}}(q_a) \right) \dot{q}_a, \quad (4.24b)$$

where  $\mathcal{J}_{\text{ACC}}(q_a) = \frac{\partial \Upsilon_{\text{ACC}}}{\partial q_a}$  is the Jacobian matrix and the time derivative of the Jacobian matrix is given by

$$\frac{d}{dt} \mathcal{J}_{\text{ACC}}(q_a) = \sum_{i=1}^2 \frac{\partial \mathcal{J}_{\text{ACC}}(q_a)}{\partial q_{ai}} \dot{q}_{ai}. \quad (4.25)$$

The final expression for the acceleration given by the accelerometer can now be written as

$$\ddot{\rho}_s^M(q_a) = \mathcal{R}_{b/s}(q_a) \left( \mathcal{J}_{\text{ACC}}(q_a) \ddot{q}_a + \left( \sum_{i=1}^2 \frac{\partial \mathcal{J}_{\text{ACC}}(q_a)}{\partial q_{ai}} \dot{q}_{ai} \right) \dot{q}_a + G_b \right) + \mathbf{b}^{\text{ACC}}. \quad (4.26)$$

### 4.3 Modelling of Bias

The measurements from an accelerometer include more than the actual acceleration. A common problem is bias in the measurements which can be taken care of in the models. In (4.21) a bias component is included in the accelerom-



eter model, which is unknown and may vary over time. The bias component  $\mathbf{b}_k = (\mathbf{b}_k^1 \ \dots \ \mathbf{b}_k^{n_b})^\top$  can be modelled as a random walk, i.e.,

$$\mathbf{b}_{k+1} = \mathbf{b}_k + \mathbf{v}_k^b, \quad (4.27)$$

where  $\mathbf{v}_k^b = (v_k^{b,1} \ \dots \ v_k^{b,n_b})^\top$  is process noise and  $n_b$  is the number of bias terms. The random walk model is then included in the estimation problem and the bias terms are estimated simultaneously as the other states. Let a state space model without any bias states be given according to

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) + g(\mathbf{x}_k)\mathbf{v}_k, \quad (4.28)$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{e}_k. \quad (4.29)$$

The augmented model with the bias states can then be written as

$$\begin{pmatrix} \mathbf{x}_{k+1} \\ \mathbf{b}_{k+1} \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{b}_k \end{pmatrix} + \begin{pmatrix} g(\mathbf{x}_k) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{v}_k \\ \mathbf{v}_k^b \end{pmatrix}, \quad (4.30)$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) + \mathcal{C}\mathbf{b}_k + \mathbf{e}_k, \quad (4.31)$$

where  $\mathbf{I}$  and  $\mathbf{0}$  are the identity and null matrices, respectively, and  $\mathcal{C} \in \mathbb{R}^{n_y \times n_b}$  is a constant matrix.

This type of model can also handle model errors in the measurement equation as will be described in Section 4.4.2.

## 4.4 Estimation Models

Four different estimation models are presented using the robot and acceleration models described previously in this chapter.

### 4.4.1 Nonlinear Estimation Model

In this section a nonlinear discrete state space model is derived based on the dynamic model in Section 4.1.3. Let the state vector be

$$\mathbf{x} = (x_1^\top \ x_2^\top \ x_3^\top \ x_4^\top)^\top = (q_a^\top \ q_m^{a,\top} \ \dot{q}_a^\top \ \dot{q}_m^{a,\top})^\top, \quad (4.32)$$

where  $q_a = (q_{a1} \ q_{a2})^\top$  are the arm angular positions and  $q_m^a = (q_{m1}^a \ q_{m2}^a)^\top$  are the motor angular positions on the arm side of the gearbox. Let also the input vector  $\mathbf{u} = \tau_m^a$ . Taking the derivative of  $\mathbf{x}$  with respect to time and using (4.9) gives

$$\dot{\mathbf{x}} = \begin{pmatrix} x_3 \\ x_4 \\ M_a^{-1}(\mathbf{x}_1)(-C(\mathbf{x}_1, \mathbf{x}_3) - G(\mathbf{x}_1) - T(\mathbf{x}_1 - \mathbf{x}_2) - D(\mathbf{x}_3 - \mathbf{x}_4)) \\ M_m^{-1}(\mathbf{u} - F(\mathbf{x}_4) + T(\mathbf{x}_1 - \mathbf{x}_2) + D(\mathbf{x}_3 - \mathbf{x}_4)) \end{pmatrix}, \quad (4.33)$$

where the model equations are given from (4.10) to (4.20). The time dependence on the state vector and input vector is here omitted for notational convenience.

In order to use the estimation methods described in Chapter 3, the continuous state space model (4.33) has to be discretised. The time derivative of the state vector can be approximated using Euler forward according to

$$\dot{x} = \frac{x_{k+1} - x_k}{T_s}, \quad (4.34)$$

where  $T_s$  is the sample time. In Section 5.3, an investigation of the influence of the sample time is made, since the sample time is crucial when Euler forward is used. Taking the right-hand side in (4.33) and (4.34) equal to each other give the nonlinear discrete state space model

$$x_{k+1} = \begin{pmatrix} x_{1,k} + T_s x_{3,k} \\ x_{2,k} + T_s x_{4,k} \\ x_{3,k} + T_s M_a^{-1}(x_{1,k}) \left( -C(x_{1,k}, x_{3,k}) - G(x_{1,k}) - T(x_{1,k} - x_{2,k}) - D(x_{3,k} - x_{4,k}) \right) \\ x_{4,k} + T_s M_m^{-1} \left( u_k - F(x_{4,k}) + T(x_{1,k} - x_{2,k}) + D(x_{3,k} - x_{4,k}) \right) \end{pmatrix}. \quad (4.35)$$

The process noise is modelled as a torque disturbance on the arms and motors, giving a model according to (3.2a) where  $f(x_k, u_k)$  is given by the right-hand side in (4.35) and

$$g(x_k) = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ T_s M_a^{-1}(x_{1,k}) & \mathbf{0} \\ \mathbf{0} & T_s M_m^{-1} \end{pmatrix}, \quad (4.36)$$

where  $\mathbf{0}$  is a two by two null matrix and the process noise  $v_k \in \mathbb{R}^4$  is Gaussian with zero mean and covariance matrix  $Q_k$ .

The measurements are the motor positions  $q_m$  and the end-effector acceleration  $\ddot{\rho}_s^M(q_a)$ . The measured motor positions are transformed to the arm side of the gearboxes before they are used, i.e., the measurements can be seen as  $q_m^a$ . The measurement model (3.2b) can therefore be written as

$$y_k = \left( \mathcal{R}_{b/s}(x_{1,k}) \left( J_{\text{ACC}}(x_{1,k}) \ddot{q}_{a,k} + \left( \frac{d}{dt} J_{\text{ACC}}(x_{1,k}) \right) x_{3,k} + G_b \right) \right) + e_k, \quad (4.37)$$

using (4.26). Here,  $\ddot{q}_{a,k}$  is given by the third row in (4.33) and  $e_k \in \mathbb{R}^4$  is the measurement noise. The measurement model for the end-effector acceleration is a big and complex expression due to the fact that the arm acceleration  $\ddot{q}_a$  is not a state. Instead it has to be calculated from the dynamic equations. The measurement noise  $e_k$  is Gaussian with zero mean and covariance matrix  $R_k$ .

The accelerometer bias  $b_k^{\text{ACC}} = \left( b_k^{\text{ACC},x} \quad b_k^{\text{ACC},z} \right)^T$  is modelled as it is described in Section 4.3 with

$$C = \begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix}, \quad (4.38)$$

where  $\mathbf{I}$  and  $\mathbf{0}$  are two by two identity and null matrices, respectively.

### 4.4.2 Estimation Model with Linear Dynamic

A linear dynamic model with arm angular positions, velocities and accelerations as state variables is suggested. Let the state vector be

$$x = \begin{pmatrix} x_1^\top & x_2^\top & x_3^\top \end{pmatrix}^\top = \begin{pmatrix} q_a^\top & \dot{q}_a^\top & \ddot{q}_a^\top \end{pmatrix}^\top, \quad (4.39)$$

This yields the following state space model in discrete time

$$x_{k+1} = Fx_k + G_u u_k + G_v v_k, \quad (4.40)$$

where  $u_k$  is the input vector and  $v_k$  is the process noise vector. The constant matrices are given by

$$F = \begin{pmatrix} \mathbf{I} & T_s \mathbf{I} & \frac{T_s^2}{2} \mathbf{I} \\ \mathbf{0} & \mathbf{I} & T_s \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad G_u = \begin{pmatrix} \frac{T_s^3}{6} \mathbf{I} \\ \frac{T_s^2}{2} \mathbf{I} \\ T_s \mathbf{I} \end{pmatrix}, \quad G_v = \begin{pmatrix} \frac{T_s^3}{6} \mathbf{I} \\ \frac{T_s^2}{2} \mathbf{I} \\ T_s \mathbf{I} \end{pmatrix}, \quad (4.41)$$

where  $\mathbf{I}$  and  $\mathbf{0}$  are two by two identity and null matrices, respectively. The input  $u_k$  is the arm jerk reference, i.e., the differentiated arm angular acceleration reference. The control performance is essential with this approach since the estimation model is driven by the arm jerk reference. If the controller is badly tuned, then the arm jerk reference does not follow the true arm jerk, hence the arm jerk reference will give an incorrect update of the states.

The measurement model for the motor positions is linear in the state vector in (4.37). Here, on the other hand, the motor positions are calculated from (4.9a) where the spring is linear, i.e.,

$$T(q_m^a - q_a) = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix} (q_m^a - q_a) = K(q_m^a - q_a). \quad (4.42)$$

The damping term is small compared to the other terms (Karlsson and Norrlöf, 2005) and is therefore neglected for simplicity. The damping term can be included by considering  $\dot{q}_m^a$  as an input to the measurement model. The measurement model for the accelerometer is the same as in (4.37) where  $\ddot{q}_{a,k}$  is a state in this case. The measurement model can now be written as

$$y_k = \begin{pmatrix} x_{1,k} + K^{-1} (M_a(x_{1,k})x_{3,k} + C(x_{1,k}, x_{2,k}) + G(x_{1,k})) \\ \mathcal{R}_{b/s}(x_{1,k}) (J_{\text{ACC}}(x_{1,k})x_{3,k} + \left(\frac{d}{dt} J_{\text{ACC}}(x_{1,k})\right) x_{2,k} + G_b) \end{pmatrix} + e_k. \quad (4.43)$$

The process noise,  $v_k$  and measurement noise  $e_k$  are considered Gaussian with zero mean and covariance matrices  $Q_k$  and  $R_k$ , respectively.

Once again, the accelerometer bias  $\mathfrak{b}_k^{\text{ACC}}$  is modelled according to Section 4.3. However, the estimation result is improved if bias components for the motor measurements are also included. The explanation is that there are model errors in the measurement equation. The total bias component is  $\mathfrak{b}_k = \left( \mathfrak{b}_k^{q_m, \top} \quad \mathfrak{b}_k^{\text{ACC}, \top} \right)^\top$ , where  $\mathfrak{b}_k^{q_m} = \left( \mathfrak{b}_k^{q_{m1}} \quad \mathfrak{b}_k^{q_{m2}} \right)^\top$  and  $\mathfrak{b}_k^{\text{ACC}} = \left( \mathfrak{b}_k^{\text{ACC}, x} \quad \mathfrak{b}_k^{\text{ACC}, z} \right)^\top$ . The matrix  $\mathcal{C}$  in (4.31) is for this model a four by four identity matrix.

### 4.4.3 Linear Estimation Model with Acceleration as Input

In De Luca et al. (2007) a model using the arm angular acceleration as input is presented. Identifying the third row in (4.33) as the arm angular acceleration and using that as an input signal with (4.32) as state vector gives the following model,

$$\dot{x} = \begin{pmatrix} x_3 \\ x_4 \\ \ddot{q}_a^{\text{IN}} \\ M_m^{-1} (u - F(x_4) + T(x_1 - x_2) + D(x_3 - x_4)) \end{pmatrix}, \quad (4.44)$$

where  $\ddot{q}_a^{\text{IN}}$  is the new input signal and  $u = \tau_m^a$  is the input torque. If the friction, spring stiffness and damping are modelled with linear relations, then

$$\dot{x} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ M_m^{-1}K & -M_m^{-1}K & M_m^{-1}D & -M_m^{-1}(D + F_d) \end{pmatrix} x + \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \\ \mathbf{0} & M_m^{-1} \end{pmatrix} \begin{pmatrix} \ddot{q}_a^{\text{IN}} \\ u \end{pmatrix}, \quad (4.45)$$

where

$$K = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix}, \quad D = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix}, \quad F_d = \begin{pmatrix} \eta_1^2 f_{d1} & 0 \\ 0 & \eta_1^2 f_{d2} \end{pmatrix},$$

and where  $\mathbf{I}$  and  $\mathbf{0}$  are two by two identity and null matrices, respectively. The parameters are the same as in Section 4.1.3, where  $k_i = k_i^{\text{high}}$ . The linear state space model is discretised using *zero order hold* (ZOH) (Rugh, 1996). The only remaining measurements are the motor positions, which give a linear measurement model according to

$$y_k = \begin{pmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{pmatrix} x_k + e_k, \quad (4.46)$$

where  $e_k \in \mathbb{R}^2$  is Gaussian with zero mean and covariance matrix  $R_k$ .

The observer used for this system is a linear dynamic observer using pole placement. In De Luca et al. (2007) it is proven that the observer can be made globally exponential stable with arbitrary decay rate if the true arm angular acceleration is used. However, the arm angular acceleration  $\ddot{q}_a^{\text{IN}}$  is not measured directly, instead, it has to be calculated from the accelerometer signal using (4.26), which is possible as long as the Jacobian  $J_{\text{ACC}}(x_{1,k})$  has full rank. The error dynamic contains in that case a residual vector and the convergence of the observer is explained in De Luca et al. (2007).

### 4.4.4 Nonlinear Estimation Model with Acceleration as Input

The linear model presented in Section 4.4.3 is reformulated as a nonlinear model. Instead of having linear models for the friction and spring, the nonlinear models in Section 4.1.3 are used. Assuming the arm angular acceleration as input gives

the following nonlinear continuous state space model

$$\dot{x} = \begin{pmatrix} x_3 \\ x_4 \\ \ddot{q}_a^{\text{IN}} \\ M_m^{-1} (u - F(x_4) + T(x_1 - x_2) + D(x_3 - x_4)) \end{pmatrix}. \quad (4.47)$$

The state space model is discretised in the same way as in Section 4.4.1 giving

$$x_{k+1} = \begin{pmatrix} x_{1,k} + T_s x_{3,k} \\ x_{2,k} + T_s x_{4,k} \\ x_{3,k} + T_s \ddot{q}_{a,k}^{\text{IN}} \\ x_{4,k} + T_s M_m^{-1} (u_k - F(x_{4,k}) + T(x_{1,k} - x_{2,k}) + D(x_{3,k} - x_{4,k})) \end{pmatrix}, \quad (4.48)$$

and the noise model is assumed to be the same as in Section 4.4.1, i.e.,  $g(x_k)$  is given by (4.36). The measurement is of course the same as in Section 4.4.3, hence

$$y_k = \begin{pmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{pmatrix} x_k + e_k. \quad (4.49)$$

The process noise,  $v_k$  and measurement noise  $e_k$  are considered Gaussian with zero mean and covariance matrices  $Q_k$  and  $R_k$ , respectively.

#### 4.4.5 Summary of the Estimation Models

The estimation models can be summarised according to:

1. Nonlinear state space model with
  - States:** arm and motor angular positions and velocities
  - Input:** motor torque
  - Output:** motor angular positions and acceleration of the end-effector
2. Linear dynamic and nonlinear measurement model with
  - States:** arm angular positions, velocities, and accelerations
  - Input:** arm jerk reference
  - Output:** motor angular positions and acceleration of the end-effector
3. Linear state space model with
  - States:** arm and motor angular positions and velocities
  - Input:** motor torque and acceleration of the end-effector
  - Output:** motor angular positions
4. Nonlinear state space model with
  - States:** arm and motor angular positions and velocities
  - Input:** motor torque and acceleration of the end-effector
  - Output:** motor angular positions



# **Part II**

# **Results**





# 5

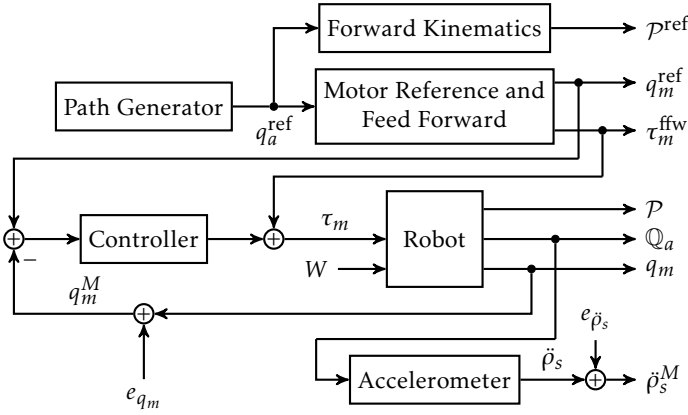
---

## Simulation Study

The simulation results for the estimation of the position of the end-effector, here denoted as TCP, are presented in this chapter. First, the simulation model is described in Section 5.1. The simulation model is composed of the robot models described in Section 4.1 and the accelerometer model in Section 4.2. The simulation setup is then presented in Section 5.2 and an investigation of the impact of the sample time is given in Section 5.3. The results presented in Section 5.4 are based on the nonlinear state space model in Section 4.4.1 using the EKF in Algorithm 1 with or without bias compensation. Finally, the conclusions of the simulation results are given in Section 5.5.

### 5.1 Overview of the Simulation Model

The simulation model is implemented in MATLAB Simulink and a block diagram of the model can be seen in Figure 5.1. The block *Path Generator* generates the desired arm angles  $q_a^{\text{ref}}$  from a set of points in the joint space or in the Cartesian space. It is also possible to use a predefined path for TCP. The desired arm angles are then used for calculating the reference trajectory for TCP, i.e.,  $\mathcal{P}^{\text{ref}}$ , using the *Forward Kinematics* in (4.2). The desired arm angles are also used to calculate references for the motor angles  $q_m^{\text{ref}}$  and a feed forward torque  $\tau_m^{\text{ffw}}$ . This is done in the block *Motor Reference and Feed Forward* using an inverse of the dynamical model in (4.9). The feedback controller is a diagonal PID controller which uses the measured motor position  $q_{mi}^M$  and the reference  $q_{mi}^{\text{ref}}$  to calculate a torque for motor  $i$ . The feed forward torque  $\tau_m^{\text{ffw}}$  is added to the calculated torque from the PID controllers before it enters the robot. The block *Robot* only simulates the robot models, described in Section 4.1.3, with or without disturbances  $W$ . The outputs from the *Robot* block are the true position  $\mathcal{P}$  of the TCP, true motor



**Figure 5.1:** Overview of the simulation model.

angles  $q_m$  and true arm angular positions, velocities and accelerations, i.e.,  $\mathbb{Q}_a = (q_a^T \quad \dot{q}_a^T \quad \ddot{q}_a^T)^T$ . The block *Accelerometer* uses  $\mathbb{Q}_a$  to calculate the acceleration of TCP according to (4.26).

Different options, listed below, are available for adjusting the realism and complexity of the simulation model.

**Mounting Errors** The position and orientation of the accelerometer can be set to deviate from the nominal values.

**Accelerometer Bias** The value of the parameter  $b^{\text{ACC}}$  in (4.21).

**Model complexity** The complexity of the dynamic equation (4.9) can be changed.

It is possible to turn on or off the gravity component  $G(q_a)$  and the friction component  $F(\dot{q}_m^a)$ . It is also possible to choose between linear and nonlinear spring torque  $T(q_a - q_m^a)$ . For the linear case  $k_i^{\text{low}} = k_i^{\text{high}}$  in (4.18).

**Disturbances** Two types of disturbances can be used. The first one is disturbances on the motor torques, which are modelled as chirp signals. The disturbance  $\tau_{md}$  is simply added to the applied torque  $\tau_m$ , i.e.,  $u_m = \tau_m + \tau_{md}$ . If no motor disturbance is present, then  $u_m = \tau_m$ . The second type is a force acting on TCP. The force is described by an amplitude and an angle relative the base frame  $Ox_bz_b$ . An angle of zero degrees corresponds to a force in the same direction as the  $x$ -axis of  $Ox_bz_b$  and an angle of 90 degrees corresponds to a force in the same direction as the  $z$ -axis.

**Ripple** Torque ripple  $\tau_{ri}$ , for joint  $i$ , can be added to the applied motor torque  $\tau_{mi}$ , i.e.,  $u_{mi} = \tau_{mi} + \tau_{ri}$ . The torque ripple is dependent of the motor angle

$q_{mi}$  and the applied torque  $\tau_{mi}$ . The model is given by

$$\tau_{ri} = A_{c1} \sin(C_1 q_{mi} + \phi_{c1}) \tau_{mi} + \sum_{j=1}^3 A_{tj} \sin(T_j q_{mi} + \phi_{tj}),$$

where the first term depends on the applied torque and the second term is the cogging torque. If no torque ripple is present, then  $u_m = \tau_m$ . Ripple can also be added to the resolvers measuring the motor angles, i.e.,  $q_{mi}^M = q_{mi} + R_{ri}$ , where the resolver ripple  $R_{ri}$  is modelled as

$$R_{ri} = A_{r1} \sin q_{mi} + A_{r2} \sin(2q_{mi} + \phi_{r2}).$$

If no resolver ripple is present, then  $q_{mi}^M = q_{mi}$ .

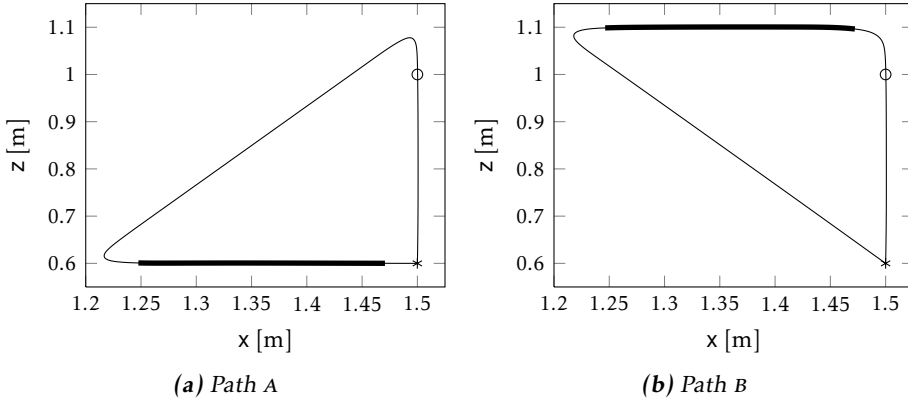
**Model Errors** The model parameters in (4.15), (4.18) and (4.20) and the masses  $m_1$  and  $m_2$  can be given uncertainty values. In this case, the feed forward block in Figure 5.1 does not use the same values as the robot model uses.

**Path** The path can be set either as interpolation in joint space between two sets of arm angles, or linear interpolation in the Cartesian space. For more complicated paths for the TCP it is possible to create them offline before the simulation starts.

**Measurement Noise** Normal distributed white measurement noise with zero mean and variance  $\sigma_{q_m}^2$  and  $\sigma_{\rho_s}^2$  can be added to the motor angles and/or the acceleration of TCP, respectively.

## 5.2 Simulation Setup

The simulations are performed using two different paths as can be seen in Figure 5.2. The paths start at the star and go clockwise and the circle indicates the TCP when  $q_a = (0 \ 0)^\top$ . The thick line indicates the segment of the paths that is used in Section 5.4 to show the result of the EKF. These two paths are generated using a standard ABB controller. Three different combinations from the list in Section 5.1 are suggested for each one of the two paths. The combinations are presented in Table 5.1 and are chosen such that model errors in the dynamical model parameters, as well as bias and mounting errors for the accelerometer are covered. All three configurations use the full model complexity, i.e., gravity, friction and nonlinear spring torque are present. Motor torque ripple, resolver ripple and measurement noise on the motor angles and the acceleration of the TCP are also present, but no external forces or motor disturbances. The model errors can be seen as the worst case and they are chosen based on suggestions from the authors of Moberg et al. (2008). Note that the true trajectory for the TCP will be the same for SIM1 and SIM3 and different for SIM2. The difference can be seen by comparing the grey line in Figure 5.6a and Figure 5.7a. The reason is that the feed forward controller changes, i.e., the control performance is changed, when model errors are present.



**Figure 5.2:** Paths used for simulation. Each path starts at the star and goes clockwise. The circle indicates the tool position when  $q_a = (0 \ 0)^T$ , and the thick line indicates the segment of the paths that is used later on in the chapter to show the result of the EKF.

**Table 5.1:** Four different simulation scenarios.

Name	Description
SIM1	Without mounting errors, bias and model errors.
SIM2	With mounting errors (4 mm in x-direction, -5 mm in z-direction and $2^\circ$ in orientation), bias ( $b^{\text{ACC},x} = b^{\text{ACC},z} = 0.1 \text{ m/s}^2$ ) and model errors (20% in stiffness parameters and 50% in friction parameters).
SIM3	With mounting errors (4 mm in x-direction, -5 mm in z-direction and $2^\circ$ in orientation), bias ( $b^{\text{ACC},x} = b^{\text{ACC},z} = 0.1 \text{ m/s}^2$ ) and without model errors.

### 5.2.1 Tuning of Covariance Matrices

In order to use the EKF one must choose good estimates of the covariance matrices  $Q$  and  $R$ , see Algorithm 1, which are assumed to be constants. An automatic approach for estimating the covariance matrices is here presented where the norm of the path error is minimised. The path error is calculated as

$$e_k = \sqrt{(x_k - \hat{x}_k)^2 + (z_k - \hat{z}_k)^2}, \quad (5.1)$$

where  $x_k$ ,  $\hat{x}_k$ ,  $z_k$  and  $\hat{z}_k$  are the true and estimated position for the tool in the x- and z-direction at time  $k$ . In practice, it is not possible to simultaneously estimate both the covariance matrix  $Q$  for the process noise and the covariance matrix  $R$  for the measurement noise. The main reason is problem with observability. Another thing is that it is the scaling between them that affects the estimate when an EKF is used. Therefore, estimate first  $R$  with dedicated experiments according to Algorithm 5.

---

**Algorithm 5** Estimation of the covariance matrix for the measurement noise.

---

- 1: Perform experiments and select a constant segment  $\bar{y}$  of the measured signal  $y$ .
- 2: Calculate  $e_k = \tilde{y}_k - \bar{y}$ , where  $\bar{y}$  is the mean of  $\tilde{y}$ .
- 3: Finally,

$$R = \frac{1}{N} \sum_{k=1}^N e_k e_k^T. \quad (5.2)$$


---

The matrix  $R$  can now be used in the minimisation problem to obtain an estimate of  $Q$ . To simplify the problem, the covariance matrix  $Q$  is parametrised as a diagonal matrix. The method is summarised in Algorithm 6, where  $(\hat{x}, \hat{z}) = \text{EKF}(Q)$  denotes that the estimated position is a function of  $Q$ . A standard optimisation method can be used to solve the second step in Algorithm 6, see e.g. Nocedal and Wright (2006). Here, `fmincon` in MATLAB is used, which is an Active Set method. The method is straightforward and a similar method has been used before, see Henriksson et al. (2009). One disadvantage with the method is that the true tool position is required.

The choice of the initial matrix  $Q_0$  is arbitrary as long as it has positive diagonal elements. A straightforward choice is to let the diagonal of  $Q_0$  be the diagonal of the covariance matrix calculated from an estimate of the noise sequence. The noise sequence  $v_k$  can be estimated as

$$\hat{v}_k = g(x_k)^\dagger (x_{k+1} - f(x_k, u_k)), \quad (5.4)$$

using the nonlinear state space model (3.2), where  $^\dagger$  is the pseudo inverse. It is possible to estimate the noise like this since this work is performed on simulated data where the true states are available. The process noise includes e.g. discretisation errors, model errors, and torque ripple.

Some of the simulations are used to optimise the covariance matrices for the EKF

---

**Algorithm 6** Estimation of the covariance matrix for the process noise.

---

- 1: Select an initial diagonal matrix  $Q_0 \in \mathbb{R}^{4 \times 4}$ .
- 2: Solve the optimisation problem

$$\begin{aligned} & \text{Minimise}_{\lambda} \quad \sqrt{\sum_{k=1}^N |e_k|^2} \\ & \text{subject to} \quad \lambda_j > 0 \quad j = 1, \dots, 4 \\ & \quad \quad \quad Q_{\lambda} = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & \lambda_4 \end{pmatrix} Q_0 \\ & \quad \quad \quad (\hat{x}, \hat{z}) = \text{EKF}(Q_{\lambda}) \end{aligned}$$

- 3: The optimal covariance matrix is given by

$$Q = \begin{pmatrix} \lambda_1^* & 0 & 0 & 0 \\ 0 & \lambda_2^* & 0 & 0 \\ 0 & 0 & \lambda_3^* & 0 \\ 0 & 0 & 0 & \lambda_4^* \end{pmatrix} Q_0, \quad (5.3)$$

where  $\lambda_j^*$  is the solution from step 2.

---

**Table 5.2:** Five different covariance matrices used to evaluate the EKF with or without bias compensation.

Name	Description
COV1	Covariance matrix for the EKF, without bias compensation, tuned on path A for SIM1
COV2	Covariance matrix for the EKF, without bias compensation, tuned on path A for SIM2
COV3	Covariance matrix for the EKF, without bias compensation, tuned on path A for SIM3
COV4	Covariance matrix for the EKF, with bias compensation, tuned on path A for SIM2
COV5	Covariance matrix for the EKF, with bias compensation, tuned on path A for SIM3

---

with or without bias compensation, which can be seen in Table 5.2. Different combinations of the simulations and the matrices are then used to estimate the tool position for the two paths, see Section 5.4. The different combinations of covariance matrices and simulation setups are chosen such that it can be possible to evaluate the EKF with respect to model errors, mounting errors and impact of the bias. More over, the EKF is evaluated on both path A and path B but the covariance matrices are optimised for path A only. This can give an insight in how the result is affected when the path changes.

### 5.3 Investigation of the Sample Time

In this section the magnitude of the sample time is investigated for the time update in (3.9). The simulation model is simulated in 2 kHz and it is interesting to see if it is possible to run the EKF with a longer sample time, leading to a filter for real-time applications. Only the time update in the EKF is used here, since it is only that part that is affected by the sample time. A longer sample time with a factor  $\Lambda$  is investigated, where  $\Lambda = \{2 \ 3 \ 4 \ 5\}$ . It means that the sample time is  $T_s = \Lambda \cdot 0.5 \text{ ms}$ , hence the control signal is decimated with a factor  $1/\Lambda$ . Figure 5.3 shows the first state after the time update. It can be seen that a longer sample time makes the system unstable. The cases  $\Lambda = 4$  and  $\Lambda = 5$  are not shown in the figure due to numerical instability in the time update equations. The measurement update is able to correct for the errors caused by the sample time but that will give an oscillatory behaviour.

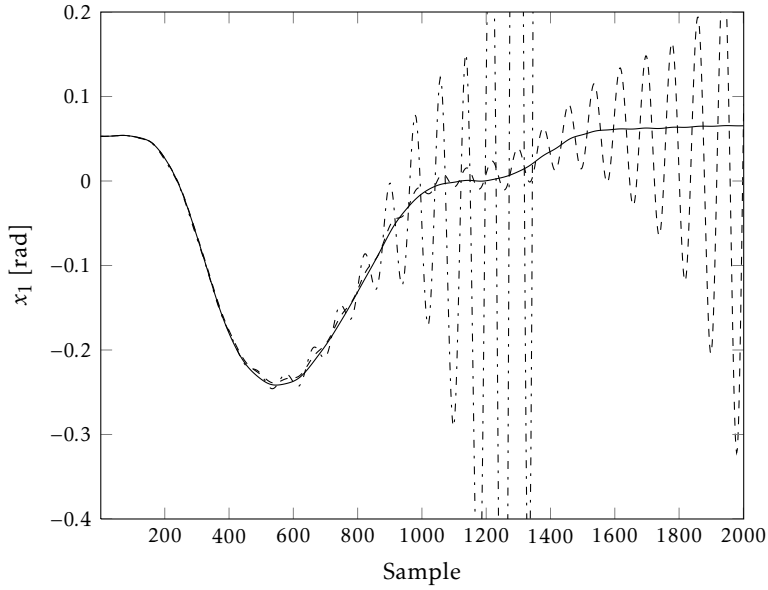
A shorter sample time is also investigated to see if the EKF can perform better. The shorter sample times that are investigated are  $0.5/\Delta \text{ ms}$ , where

$$\Delta = \{5 \ 10 \ 50 \ 100 \ 500 \ 1000 \ 5000\}. \quad (5.5)$$

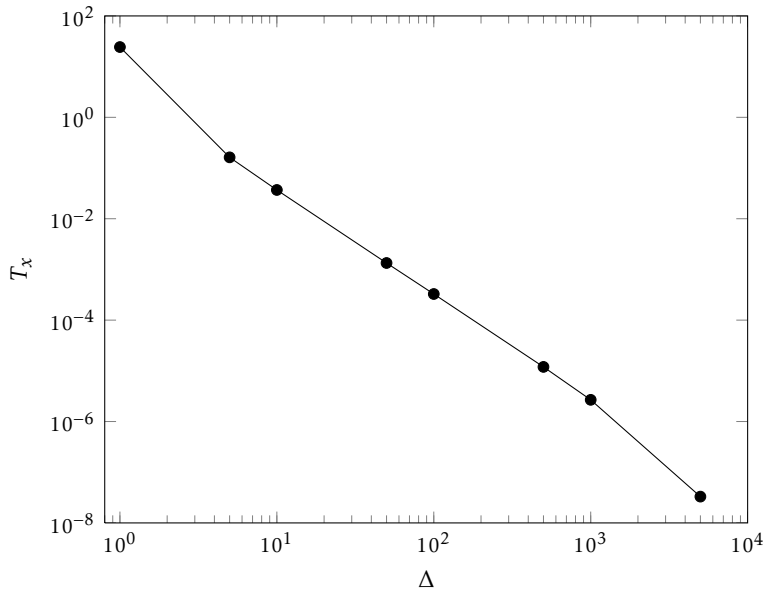
Each sample of the control signal is here repeated  $\Delta$  times to get the correct sample time. A measure of the filter performance is to calculate

$$T_x = \sum_{k=1}^N \|x_k^{\text{TRUE}} - \hat{x}_k^{\text{TU}}\|_2^2, \quad (5.6)$$

where  $x_k^{\text{TRUE}}$  is the true states and  $\hat{x}_k^{\text{TU}}$  is the estimated states from the time update only. A decrease in  $T_x$  between two values of  $\Delta$  implies a better filter performance. Figure 5.4 shows  $T_x$  in (5.6) for the different values of  $\Delta$  in (5.5), where it can be seen that the performance is increased. A better performance is obtained to the cost of a slower filter. It is a trade off between computational capacity and filter performance that has to be considered in order to choose a sample time. In the remainder of this thesis, the sample time is kept to 0.5 ms.

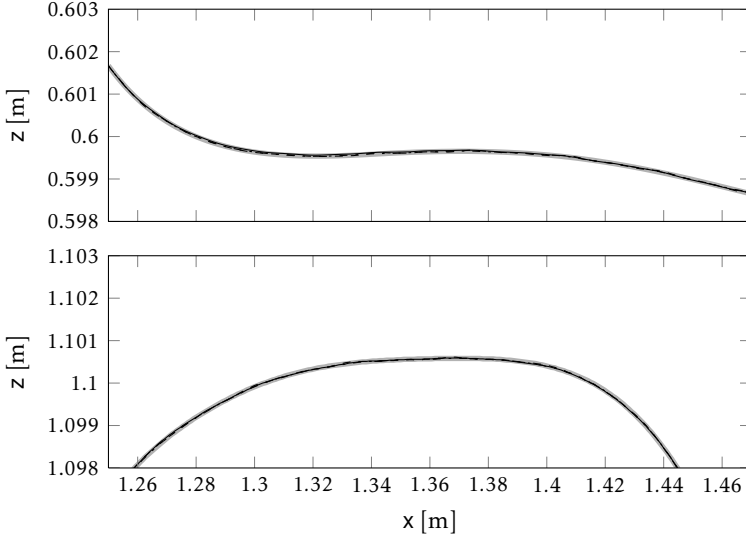


**Figure 5.3:** The first state after the time update for the original sample time (solid),  $\Lambda = 2$  (dashed) and  $\Lambda = 3$  (dash-dot).



**Figure 5.4:**  $T_x$  in (5.6) for different values of  $\Delta$ , i.e., a shorter sample time.





**Figure 5.5:** Estimation of path A (top) and path B (bottom) on SIM1 using the EKF without bias compensation with COV1 (solid), COV2 (dashed), and COV3 (dash-dot). The grey line is the true path

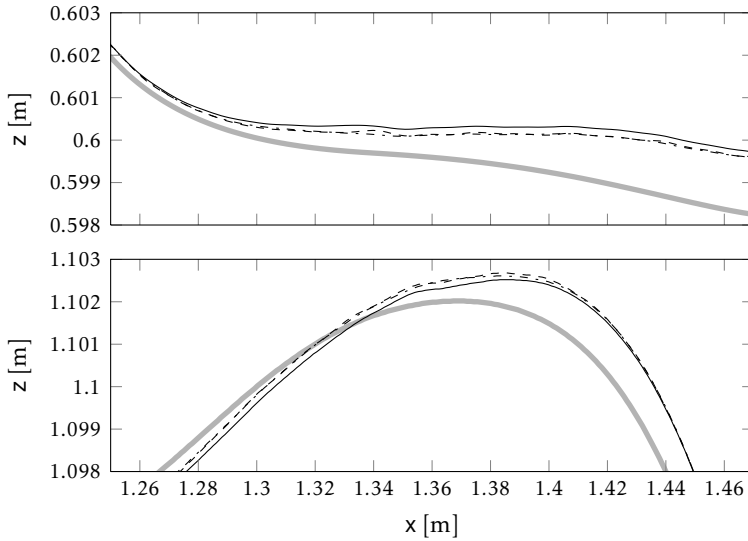
## 5.4 Results

The result is evaluated using the three simulation scenarios in Table 5.1 and the five covariance matrices in Table 5.2 using the EKF with or without bias compensation for path A and path B. Only the EKF without bias compensation (COV1, COV2, and COV3) is used for SIM1 since no bias in the measured acceleration is present. For SIM2 and SIM3, both EKF with and without bias compensation (COV1 to COV5) are used. The evaluation is focused on comparing the true path of the end-effector with the estimated path. Since the EKF does not give the path directly, the estimated arm angular positions have to be used in the forward kinematic model (4.2), i.e.,

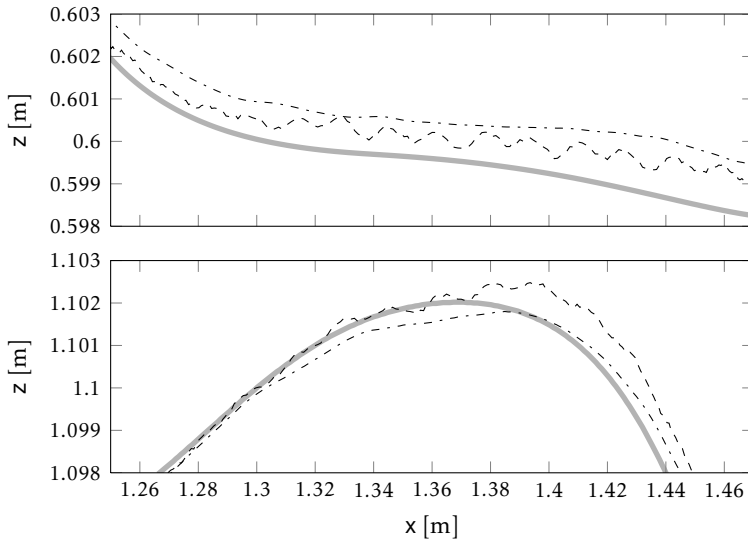
$$\begin{pmatrix} \hat{x}_k \\ \hat{z}_k \end{pmatrix} = \Upsilon(\hat{q}_{a,k}) = \Upsilon(\hat{x}_{1:2,k}), \quad (5.7)$$

to obtain the estimated path, where  $\hat{x}_{1:2,k}$  denotes the first two states at time  $k$ . To be able to see how the estimated path behaves, the bottom horizontal part for path A and the top horizontal part of path B are magnified as can be seen in Figure 5.5 for SIM1. The figure shows that the estimated paths follow the true path very well independent of the covariance matrices. Moreover, it does not matter if the path has changed, i.e., the covariance matrices optimised for path A also give a good result for path B. It is not surprising that the three estimated paths follow the true paths well since SIM1 is without any errors.

The result for SIM2 in Figure 5.6 is more relevant. The EKF without bias com-



(a) EKF without bias compensation with COV1 (solid), COV2 (dashed), and COV3 (dash-dot). Path A (top) and path B (bottom).



(b) EKF with bias compensation with COV4 (dashed) and COV5 (dash-dot). Path A (top) and path B (bottom).

**Figure 5.6:** Estimation of path A and path B on SIM2 using the EKF without bias compensation (a) and with bias compensation (b). The grey line is the true path

**Table 5.3:** RMSE values of the path error  $\epsilon$  for the end-effector position given in mm for path A.

	COV1	COV2	COV3	COV4	COV5
SIM1	0.0221	0.0143	0.0143	—	—
SIM2	1.0967	1.0816	1.0913	1.1636	1.2755
SIM3	0.3053	0.2904	0.2767	0.1681	0.1267

pensation in Figure 5.6a cannot follow the true path as good as for SIM1. The difference between the three covariance matrices is however not that evident and it seems that the path does not matter either. Using the EKF with bias compensation improves the result a bit as can be seen in Figure 5.6b. According to this it is possible to conclude that the model errors are the underlying problem. An accurate model is therefore needed in order to get a good estimate of the path.

In Figure 5.7, where only mounting errors of the accelerometer are present, it can be seen that the estimated path is better than when model errors are present but the path is still not as good as for SIM1. Using an EKF with bias compensation improves the estimated path.

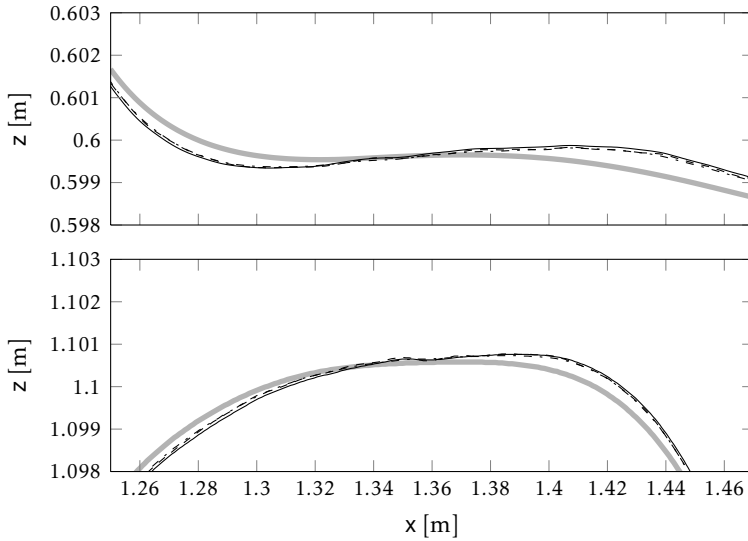
To get a better understanding of how the estimated paths behave, the path errors  $\epsilon_k$  of path A for SIM1, SIM2, and SIM3 are shown in Figure 5.8, where the first 0.05 s are removed to eliminate the effects of the transients. It can be seen that the path error more or less has the same behaviour and an accuracy of around 0.05 mm when no errors are present. An accuracy of less than 3 mm is achieved when model errors and mounting errors are present and less than 1 mm when only mounting errors are present. The figure only shows the result for path A using the EKF without bias compensation but the results for path B and the EKF with bias compensation are similar. To get a measure of how large the path error is, the *root mean square error* (RMSE)  $\epsilon$  of each path error signal is calculated according to

$$\epsilon = \sqrt{\frac{1}{N} \sum_{k=1}^N \epsilon_k^2}, \quad (5.8)$$

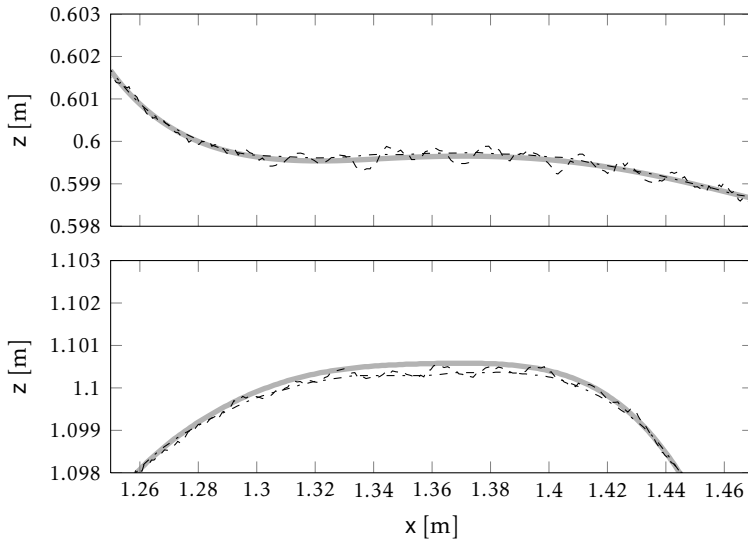
where  $N$  is the number of samples. The result is presented in Table 5.3.

The table shows that the covariance matrix COV1 estimated for SIM1 does not give the lowest error, here both COV2 and COV3 give a lower error. This means that COV1 is not a global solution to the optimisation problem. Also, if a new optimisation of the covariance matrix is performed with a different initial value, then the covariance matrix will give another error. Figure 5.9 shows the path errors for ten different solutions to the optimisation problem, with and without model errors, where the initial value  $Q_0$  has changed. The conclusion here is that the optimisation problem ends up in different solutions.

The evaluation has so far been focused on the position of the end-effector. It

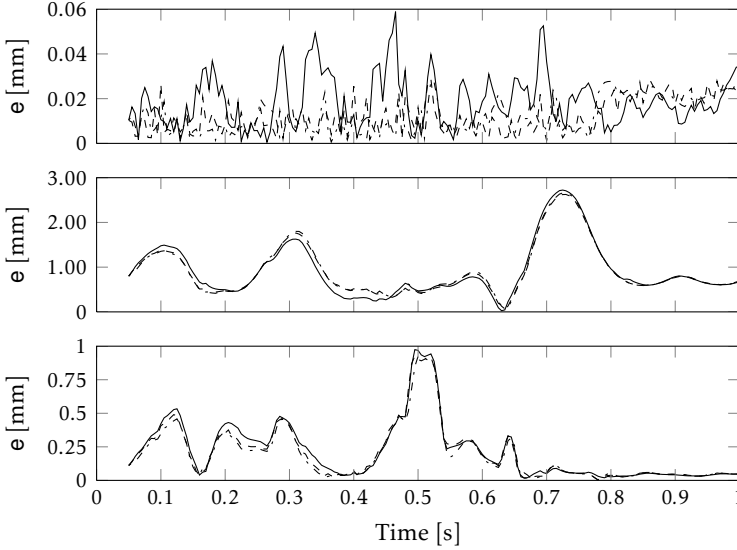


(a) EKF without bias compensation with COV1 (solid), COV2 (dashed), and COV3 (dash-dot). Path A (top) and path B (bottom).



(b) EKF with bias compensation with COV4 (dashed) and COV5 (dash-dot). Path A (top) and path B (bottom).

**Figure 5.7:** Estimation of path A and path B on SIM3 using the EKF without bias compensation (a) and with bias compensation (b). The grey line is the true path



**Figure 5.8:** The path error  $e$  for the end-effector position using the EKF without bias compensation on path A for SIM1 (top), SIM2 (middle), and SIM3 (bottom) with COV1 (solid), COV2 (dashed), and COV3 (dash-dot).

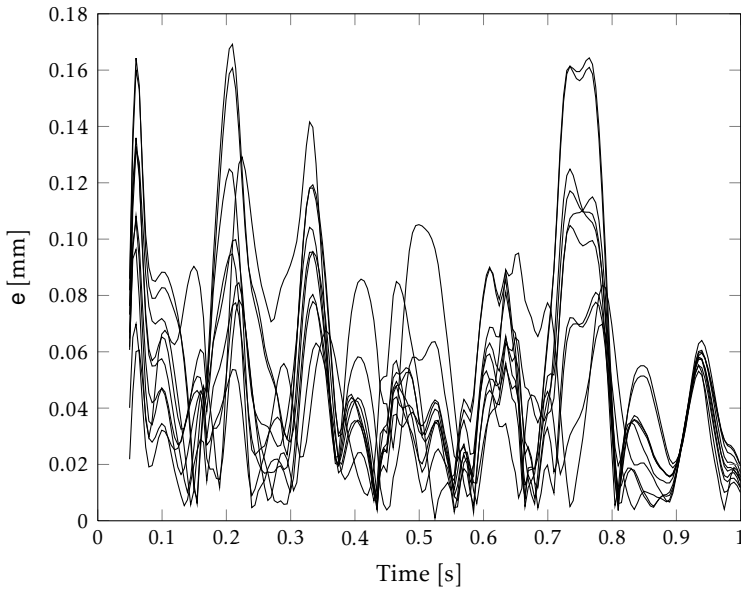
is however possible to compare the estimated arm angular positions and arm angular velocities with the true counterparts. The error is calculated as

$$\mathbf{e}_{q_a}(k) = \left\| q_a^0(k) - \hat{q}_a(k) \right\|_2, \quad (5.9a)$$

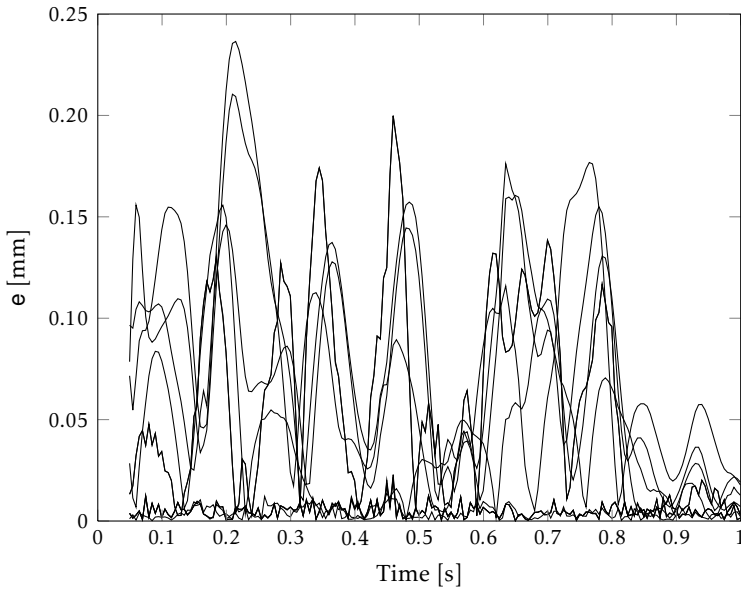
$$\mathbf{e}_{\dot{q}_a}(k) = \left\| \dot{q}_a^0(k) - \hat{\dot{q}}_a(k) \right\|_2, \quad (5.9b)$$

where  $q_a^0(k)$  and  $\dot{q}_a^0(k)$  are the true arm angular positions and velocities for the two joints at time  $k$  given from the simulation model, and  $\hat{q}_a(k)$  and  $\hat{\dot{q}}_a(k)$  are the estimated counterparts from the EKF. Figure 5.10 shows the error  $\mathbf{e}_{q_a}(k)$  for the arm angular positions, Figure 5.11 shows the error  $\mathbf{e}_{\dot{q}_a}(k)$  for the arm angular velocities, and Tables 5.4 and 5.5 show the RMSE values calculated according to (5.8) with the path errors in (5.9). The arm angular positions show similar behaviour as the end-effector position since there is a static relation between them, i.e., the forward kinematic model in (4.2). Note that the difference in the magnitude of the velocity errors are in the same order for all three simulations, whereas the errors for SIM1 are much lower than for the other two simulations when the TCP errors or the arm angular position errors are considered.

The uncertainty of the estimated path can be calculated using Gauss' approximation formula as described in Section 3.4. The topmost plot in Figure 5.12 shows the true path, the estimated path and the covariance ellipses for each time index  $k$ . It can be seen that the true path is not covered by the covariance ellipses. One drawback with Gauss' approximation formula is that the estimated covariance matrix after the nonlinear transformation can be smaller than the true one. To

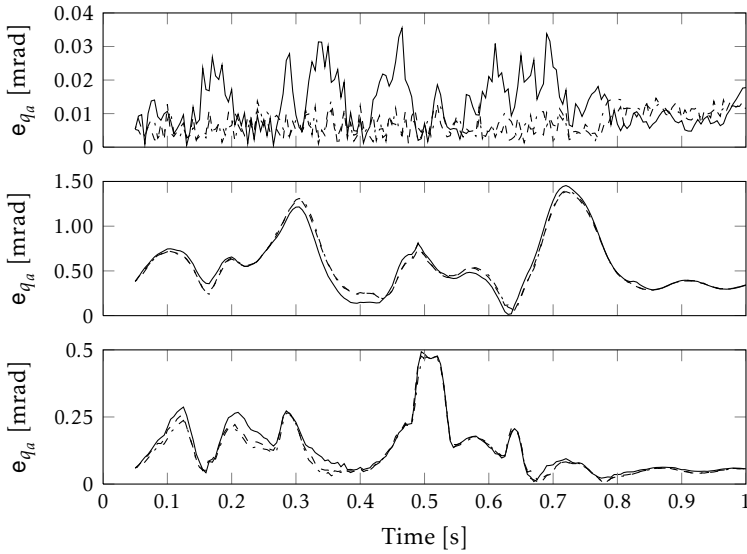


(a) SIM1.

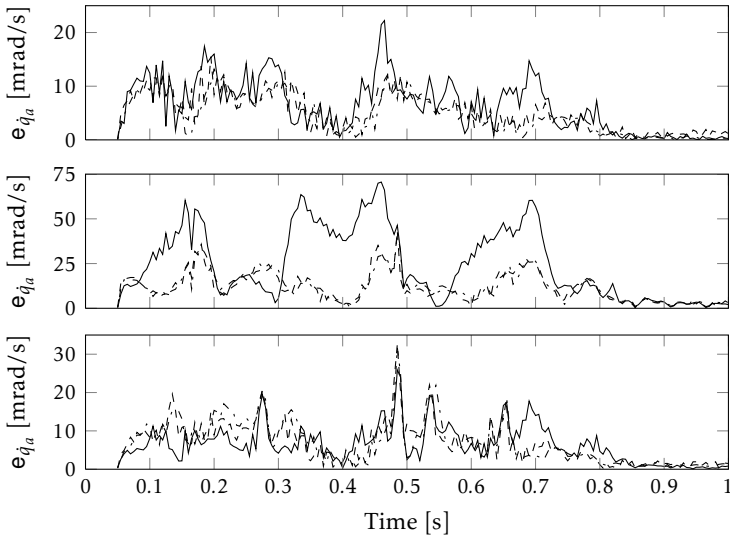


(b) SIM2.

**Figure 5.9:** The path error  $e$  for 10 MC simulations of Algorithm 6 with different starting matrices  $Q_0$  for SIM1 and SIM2. The different solutions give different path errors, which indicates that the problem is non convex.



**Figure 5.10:** The error  $e_{q_a}$  for the arm angular positions using the EKF without bias compensation on path A for SIM1 (top), SIM2 (middle), and SIM3 (bottom) with COV1 (solid), COV2 (dashed), and COV3 (dash-dot).



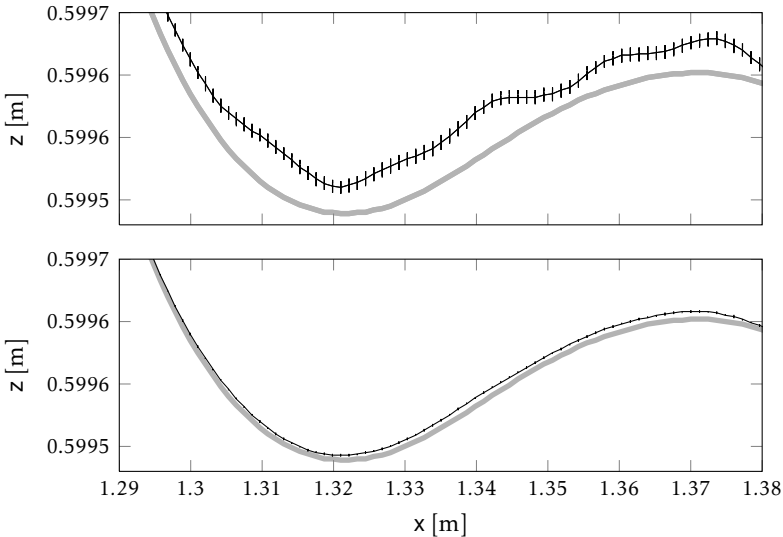
**Figure 5.11:** The error  $e_{\dot{q}_a}$  for the arm angular velocities using the EKF without bias compensation on path A for SIM1 (top), SIM2 (middle), and SIM3 (bottom) with COV1 (solid), COV2 (dashed), and COV3 (dash-dot).

**Table 5.4:** RMSE values of the error  $e_{q_a}$  for the arm angular positions given in mrad for path A.

	COV1	COV2	COV3	COV4	COV5
SIM1	0.0148	0.0078	0.0081	—	—
SIM2	0.6529	0.6466	0.6532	0.6985	0.7195
SIM3	0.1675	0.1583	0.1532	0.1349	0.0752

**Table 5.5:** RMSE values of the error  $e_{\dot{q}_a}$  for the arm angular velocities given in mrad/s for path A.

	COV1	COV2	COV3	COV4	COV5
SIM1	8.1598	6.2601	5.9184	—	—
SIM2	31.6386	14.4535	14.4860	28.9059	34.4696
SIM3	8.0256	8.7230	8.9496	21.2968	10.1817



**Figure 5.12:** True path (grey) and estimated path (black) with the covariance ellipses (vertical black lines). The topmost plot shows the estimated path with  $T_s = 0.5$  ms and the bottom plot is for  $T_s = 50$  ns.



exclude this as the reason for too small covariance matrices is to do the MC transformation instead. However, the MC transformation gives similar results and is not showed here. The problem is instead that the EKF underestimates the covariance matrices for the arm angles. If the sample time is decreased by a factor 10 000 to eliminate discretisation errors, then the estimated path follows the true path better as can be seen in the bottom plot in Figure 5.12, which is in accordance with Section 5.3. However, the covariance ellipses are decreased and the true path is still not covered by the ellipses.

The reason for underestimating the covariance matrices is probably model errors in combination with high SNR, i.e., small values in the covariance matrix  $R$ . If data are generated with the same models as are used in the EKF, then the covariance matrices will cover the true path, however, the ellipses are still very small. The magnitude of the elements in the covariance matrices have been verified using the stationary solution for the parametric Cramér-Rao lower bound, evaluated around the true state trajectory. See Bergman (1999, Theorem 4.4) for details.

## 5.5 Conclusions

The main conclusion is that model errors in the dynamic models are troublesome. One solution is better identification methods which is not a part of this work and is therefore left out. Another possibility can be to introduce a bias term for each model parameter which is estimated at the same time as the states. This is the same approach as for the accelerometer bias described in Section 4.3.

The choice of the covariance matrices is a hard work and the accuracy of the estimated path can be better with good choices. The optimisation problem in Section 5.2.1 works but it is not robust for different initial values. A better method is therefore preferable.

The bias compensation in the EKF improves the results as expected, but it is still necessary to have good knowledge of the mounting position as well as the orientation of the accelerometer.

In this small study, the path does not seem to affect the result of different covariance matrices that much. A more detailed study, with more paths in a wider part of the workspace, is however necessary. The indication that the result may not be affected of the path is that the time update of the covariance matrix  $P$  depends on the term  $G_k Q_k G_k^T$  which is dependent of the state  $\hat{x}_{k-1|k-1}$  as can be seen in Algorithm 1. The estimated covariance matrix can therefore be seen as independent of the path.

The uncertainties of the TCP estimates, calculated using Gauss' approximation formula and the uncertainties of the estimated arm angles, are very low. The EKF underestimates the covariance matrices  $P_k$  due to model errors and high SNR.



# 6

---

## The EM Algorithm for Covariance Estimation

The performance of the EKF depends in the end of the covariance matrices for the process noise and the measurement noise. The simulation study in Chapter 5 has shown that it is hard to get good estimates of, in particular, the covariance matrix for the process noise. This chapter presents a new method, using the *expectation maximisation* (EM) algorithm and the *extended Kalman smoother* (EKS), for estimation of the covariance matrix for the process noise. The material on which this chapter is based has previously been published in Axelsson et al. (2011e).

The EM algorithm has been explored for linear Gaussian models in Cappé et al. (2005), where the system matrices ( $A, C, Q, R$ ) are estimated using the Kalman smoother as the state estimator. For nonlinear models, the particle smoother is used to estimate the parameters in a nonlinear dynamic model, see Schön et al. (2011). However, the particle smoother is not applicable for models with high state dimension which is the case for industrial robots.

First, the derivation of the EM solution is presented in Section 6.1. After that, two alternative methods are presented in Section 6.2. Finally, the results and conclusions are given in Sections 6.3 and 6.4, respectively.

### 6.1 Derivation of the EM Solution

This section describes how the covariance matrices for the process and measurement noise in

$$x_{k+1} = F_1(x_k, u_k) + F_2(x_k)v_k, \quad (6.1a)$$

$$y_k = h(x_k, u_k) + e_k, \quad (6.1b)$$

can be estimated using the EM algorithm described in Section 3.3. The process noise  $v_k$  and measurement noise  $e_k$  are Gaussian with zero mean and covariance matrices  $Q$  and  $R$ , respectively. All model parameters are assumed to be known except for  $Q \in \mathbb{S}_+^{n_y}$  and  $R \in \mathbb{S}_{++}^{n_y}$ <sup>1</sup>. First, the solution is derived using a general matrix  $F_2(x_k)$ , then the solution is given when it is assumed that  $F_2(x_k)$  has the following structure

$$F_2(x_k) = \begin{pmatrix} 0 \\ \widetilde{F}_2(x_k) \end{pmatrix}. \quad (6.2)$$

The structure in (6.2) is common for mechanical systems derived by Newton's law or Lagrange's equation when Euler forward is used for discretisation, cf. Section 4.4.1 and in particular (4.36).

It is not possible to simultaneously estimate both the covariance matrix  $Q$  for the process noise and the covariance matrix  $R$  for the measurement noise, as described in Section 5.2.1. The covariance matrix  $R$  is therefore estimated using Algorithm 5, and then used in the EM algorithm to estimate  $Q$ .

Equation (6.1) can also be expressed in the more general conditional densities as

$$x_{k+1} \sim p(x_{k+1}|x_k) = \mathcal{N}(x_{k+1}; F_1(x_k), F_2(x_k)QF_2^\top(x_k)) \quad (6.3a)$$

$$y_k \sim p(y_k|x_k) = \mathcal{N}(y_k; h(x_k), R) \quad (6.3b)$$

where  $\mathcal{N}(\cdot)$  is the multivariate Gaussian distribution function. The multivariate Gaussian distribution for the  $n$ -dimensional variable  $\mu$  with mean  $\bar{\mu}$  and covariance  $\Sigma$  is defined according to

$$\mathcal{N}(\mu; \bar{\mu}, \Sigma) \triangleq \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mu-\bar{\mu})^\top \Sigma^{-1}(\mu-\bar{\mu})}. \quad (6.4)$$

The control signal  $u_k$  is omitted in (6.3) and in the sequel of this chapter to simplify the notation.

Proceed now with the derivation of the expectation and maximisation steps in Algorithm 4 where  $\theta = Q$  is the sought parameter.

### 6.1.1 Expectation Step

The joint likelihood can easily be written as

$$p_Q(y_{1:N}, x_{1:N}) = p(x_1, y_1) \prod_{i=2}^N p(y_i|x_i)p_Q(x_i|x_{i-1}), \quad (6.5)$$

where

$$p(x_k, y_k|x_{1:k-1}, y_{1:k-1}) = p(x_k, y_k|x_{k-1}) = p(y_k|x_k)p(x_k|x_{k-1}) \quad (6.6)$$

---

<sup>1</sup> $\mathbb{S}_{++}^p$  ( $\mathbb{S}_+^p$ ) is the set of all symmetric positive definite (semidefinite)  $p \times p$  matrices

has been used repeatedly. The notation  $p_Q(\cdot)$  means that the PDF is parametrised by  $Q$ . Taking the logarithm<sup>2</sup> of (6.5) and using (6.3) together with (6.4) give

$$L_Q(y_{1:N}, x_{1:N}) = \log p_Q(y_{1:N}, x_{1:N}) = \tilde{L} + \frac{1}{2} \sum_{i=2}^N \log \left( \left| F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right|^{-1} \right) - \frac{1}{2} \sum_{i=2}^N (x_i - F_1(x_{i-1}))^\top \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})), \quad (6.7)$$

where the terms from the likelihood  $p(y_i|x_i)$ , which are independent of  $Q$ , are collected in  $\tilde{L}$ .

Next step is to calculate the expectation of  $L_Q(y_{1:N}, x_{1:N})$  to obtain  $\Gamma(Q; Q_I)$ .

$$\begin{aligned} \Gamma(Q; Q_I) &= E_{Q_I} \left[ L_Q(y_{1:N}, x_{1:N}) | y_{1:N} \right] = \tilde{L} \\ &+ \frac{1}{2} \sum_{i=2}^N E_{Q_I} \left[ \log \left( \left| F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right|^{-1} \right) \middle| y_{1:N} \right] \\ &- \frac{1}{2} \text{tr} \sum_{i=2}^N E_{Q_I} \left[ \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top \middle| y_{1:N} \right], \end{aligned} \quad (6.8)$$

where  $\tilde{L}$  is independent of  $Q$ . The trace operator comes from the fact that the trace of a scalar is the same as the scalar itself, i.e.,

$$\begin{aligned} (x_i - F_1(x_{i-1}))^\top \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) \\ &= \text{tr} (x_i - F_1(x_{i-1}))^\top \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) \\ &= \text{tr} \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top. \end{aligned} \quad (6.9)$$

Start with the calculations of the first expectation in (6.8),

$$\begin{aligned} E_{Q_I} \left[ \log \left( \left| F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right|^{-1} \right) \middle| y_{1:N} \right] \\ &= \int \log \left( \left| F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right|^{-1} \right) p_{Q_I}(x_{i-1} | y_{1:N}) dx_{i-1}. \end{aligned} \quad (6.10)$$

The integral cannot be solved analytically. Instead, an approximation has to be made. The smoothed density of  $x_{i-1}$  has a high peak around the smoothed estimate if the sampling frequency and the SNR are high. Here, the EKS in Algorithm 3 is used. The state  $x_{i-1}$  can therefore be approximated with the smoothed state  $\hat{x}_{i-1|N}^s$ , in other words,

$$E_{Q_I} \left[ \log \left( \left| F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right|^{-1} \right) \middle| y_{1:N} \right] \approx \log \left( \left| F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right|^{-1} \right). \quad (6.11)$$

<sup>2</sup>The natural logarithm is used.

The second expectation in (6.8) can be written as

$$\begin{aligned} & E_{Q_l} \left[ \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top \middle| y_{1:N} \right] \\ &= \int \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top \\ &\quad \times p_{Q_l}(x_i, x_{i-1} | y_{1:N}) dx_i dx_{i-1}. \end{aligned} \quad (6.12)$$

Now use the smoothed density again and let

$$F_2(x_{i-1}) \approx F_2(\hat{x}_{i-1|N}^s). \quad (6.13)$$

The term  $\left( F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right)^{-1}$  is now a constant and the expectation value can be written as

$$\begin{aligned} & E_{Q_l} \left[ \left( F_2(x_{i-1}) Q F_2^\top(x_{i-1}) \right)^{-1} (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top \middle| y_{1:N} \right] \\ &\approx \left( F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right)^{-1} \int (x_i - F_1(x_{i-1})) (x_i - F_1(x_{i-1}))^\top \\ &\quad \times p_{Q_l}(x_i, x_{i-1} | y_{1:N}) dx_i dx_{i-1}. \end{aligned} \quad (6.14)$$

The PDF  $p_{Q_l}(x_i, x_{i-1} | y_{1:N})$  can be seen as the smoothed density of the augmented state vector  $\xi_i = \begin{pmatrix} x_{i-1}^\top & x_i^\top \end{pmatrix}^\top$ , i.e.,

$$p_{Q_l}(x_i, x_{i-1} | y_{1:N}) = p_{Q_l}(\xi_i | y_{1:N}) = \mathcal{N}(\xi_i; \hat{\xi}_{i|N}^s, P_{i|N}^{\xi, s}). \quad (6.15)$$

The first and second order moments of the smoothed  $\xi_i$  can be expressed as

$$\hat{\xi}_{i|N}^s = \begin{pmatrix} \hat{x}_{i-1|N}^s \\ \hat{x}_{i|N}^s \end{pmatrix}, \quad P_{i|N}^{\xi, s} = \begin{pmatrix} P_{i-1|N}^s & P_{i-1, i|N}^s \\ \left( P_{i-1, i|N}^s \right)^\top & P_{i|N}^s \end{pmatrix}, \quad (6.16)$$

where  $\hat{x}_{i-1|N}^s$ ,  $\hat{x}_{i|N}^s$ ,  $P_{i-1|N}^s$  and  $P_{i|N}^s$  are the first and second order moments of the smoothed  $\hat{x}_{i-1}$  and  $\hat{x}_i$ , respectively. These are obtained if the augmented model

$$\xi_{k+1} = \begin{pmatrix} x_k \\ F_1(x_k, u_k) \end{pmatrix} \quad (6.17)$$

is used in the EKS. The integral in (6.14) cannot be solved analytically. Instead, a first order Taylor expansion of  $F_1(x_{i-1})$  around  $\hat{x}_{i-1|N}^s$  is used. The quadratic term

in (6.14) can now be written as

$$\begin{aligned}
(x_i - F_1(x_{i-1}))(x_i - F_1(x_{i-1}))^\top &\approx (x_i - F_1(\hat{x}_{i-1|N}^s) - J_{1,i-1}(x_{i-1} - \hat{x}_{i-1|N}^s)) \\
&\quad \times (x_i - F_1(\hat{x}_{i-1|N}^s) - J_{1,i-1}(x_{i-1} - \hat{x}_{i-1|N}^s))^\top \\
&= (-J_{1,i-1} \quad I) (\xi_i - \hat{\xi}_{i|N}^s) (\xi_i - \hat{\xi}_{i|N}^s)^\top (-J_{1,i-1} \quad I)^\top \\
&\quad + (-J_{1,i-1} \quad I) (\xi_i - \hat{\xi}_{i|N}^s) (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s))^\top \\
&\quad + (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s)) (\xi_i - \hat{\xi}_{i|N}^s)^\top (-J_{1,i-1} \quad I)^\top \\
&\quad + (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s)) (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s))^\top, \tag{6.18}
\end{aligned}$$

where the Taylor expansion

$$F_1(x_{i-1}) \approx F_1(\hat{x}_{i-1|N}^s) + J_{1,i-1}(x_{i-1} - \hat{x}_{i-1|N}^s), \tag{6.19a}$$

$$J_{1,i-1} = \left. \frac{\partial F_1(x)}{\partial x} \right|_{x=\hat{x}_{i-1|N}^s}, \tag{6.19b}$$

has been used.

The integral in (6.14) now becomes

$$\begin{aligned}
M_i &\triangleq \int (x_i - F_1(x_{i-1}))(x_i - F_1(x_{i-1}))^\top p_{Q_i}(x_i, x_{i-1}|y_{1:N}) dx_i dx_{i-1} \\
&= (-J_{1,i} \quad I) P_{i|N}^{\xi,s} (-J_{1,i} \quad I)^\top + (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s)) (\hat{x}_{i|N}^s - F_1(\hat{x}_{i-1|N}^s))^\top. \tag{6.20}
\end{aligned}$$

It is thus possible to calculate  $\Gamma(Q; Q_i)$  according to

$$\begin{aligned}
\Gamma(Q; Q_i) &= \bar{L} + \frac{1}{2} \sum_{i=2}^N \log \left( \left| F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right|^{-1} \right) \\
&\quad - \frac{1}{2} \text{tr} \sum_{i=2}^N \left( F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right)^{-1} M_i. \tag{6.21}
\end{aligned}$$

If it is assumed that  $F_2^\top(x_{i-1})$  and  $Q$  have full row rank, and  $F_2(x_{i-1})$  and  $F_2(x_{i-1})Q$  have full column rank, then (Mitra and Rao, 1971)

$$\left( F_2(\hat{x}_{i-1|N}^s) Q F_2^\top(\hat{x}_{i-1|N}^s) \right)^{-1} = \left( F_2^\dagger(\hat{x}_{i-1|N}^s) \right)^\top Q^{-1} F_2^\dagger(\hat{x}_{i-1|N}^s). \tag{6.22}$$

Using this together with the trace rule  $\text{tr} ABC = \text{tr} BCA$  give

$$\begin{aligned}
\Gamma(Q; Q_i) &= \bar{L} + \frac{1}{2} \sum_{i=2}^N \log \left( \left| \left( F_2^\dagger(\hat{x}_{i-1|N}^s) \right)^\top Q^{-1} F_2^\dagger(\hat{x}_{i-1|N}^s) \right| \right) \\
&\quad - \frac{1}{2} \text{tr} Q^{-1} \sum_{i=2}^N F_2^\dagger(\hat{x}_{i-1|N}^s) M_i \left( F_2^\dagger(\hat{x}_{i-1|N}^s) \right)^\top. \tag{6.23}
\end{aligned}$$

### Special Structure of the Noise Model

Using the noise model in Section 4.4.1, i.e., the matrix (4.36), gives a singular covariance matrix in the conditional density for the state transition, i.e., the matrix  $F_2(x_i)QF_2^\top(x_i)$  in (6.3a) is singular, hence the pseudo inverse of the covariance matrix has to be used instead of the regular inverse. This trick solves not all the problems since

$$\left|F_2(x_{i-1})QF_2^\top(x_{i-1})\right| = 0 \quad (6.24)$$

and when taking the logarithm in (6.7) it becomes

$$\log\left(\left|F_2(x_{i-1})QF_2^\top(x_{i-1})\right|^\dagger\right) = \infty. \quad (6.25)$$

A common trick is to approximate  $\left|F_2(x_{i-1})QF_2^\top(x_{i-1})\right|$  by the product of all non-singular values. Given the structure of  $F_2(x_k)$  in (6.2) the covariance matrix can be written as

$$F_2(x_{i-1})QF_2^\top(x_{i-1}) = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \tilde{F}_2(x_{i-1})Q\tilde{F}_2^\top(x_{i-1}) \end{pmatrix}, \quad (6.26)$$

where  $\mathbf{0}$  is a null matrix. The product of all nonsingular values is therefore the same as  $\left|\tilde{F}_2(x_{i-1})Q\tilde{F}_2^\top(x_{i-1})\right|$ , where it is assumed that  $\tilde{F}_2(x_{i-1})Q\tilde{F}_2^\top(x_{i-1})$  is nonsingular. The joint log likelihood function in (6.7) is now given by

$$\begin{aligned} L_Q(y_{1:N}, x_{1:N}) &= \tilde{L} + \frac{1}{2} \sum_{i=2}^N \log\left(\left|\tilde{F}_2(x_{i-1})Q\tilde{F}_2^\top(x_{i-1})\right|^{-1}\right) \\ &\quad - \frac{1}{2} \sum_{i=2}^N (x_i - F_1(x_{i-1}))^\top \left(F_2(x_{i-1})QF_2^\top(x_{i-1})\right)^\dagger (x_i - F_1(x_{i-1})). \end{aligned} \quad (6.27)$$

Using this expression during the calculation of the expectation of  $L_Q(y_{1:N}, x_{1:N})$  gives

$$\begin{aligned} \Gamma(Q; Q_l) &= \tilde{L} + \frac{1}{2} \sum_{i=2}^N \left[ \log\left(\left|\left(\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right)^\top\right|\right) + \log|Q^{-1}| + \log\left(\left|\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right|\right) \right] \\ &\quad - \frac{1}{2} \text{tr} Q^{-1} \sum_{i=2}^N F_2^\dagger(\hat{x}_{i-1|N}^s) M_i \left(F_2^\dagger(\hat{x}_{i-1|N}^s)\right)^\top. \end{aligned} \quad (6.28)$$

Here it has been used that

$$\begin{aligned} \log\left(\left|\tilde{F}_2(\hat{x}_{i-1|N}^s)Q\tilde{F}_2^\top(\hat{x}_{i-1|N}^s)\right|^{-1}\right) &= \log\left(\left|\left(\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right)^\top Q^{-1} \tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right|\right) \\ &= \log\left(\left|\left(\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right)^\top\right| |Q^{-1}| \left|\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right|\right) \\ &= \log\left(\left|\left(\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right)^\top\right|\right) + \log|Q^{-1}| + \log\left(\left|\tilde{F}_2^\dagger(\hat{x}_{i-1|N}^s)\right|\right), \end{aligned} \quad (6.29)$$

which is possible since  $\tilde{F}_2^\top(x_{i-1})$  and  $Q$  have full row rank, and  $\tilde{F}_2(x_{i-1})$  and  $\tilde{F}_2(x_{i-1})Q$  have full column rank, and  $\tilde{F}_2(x_{i-1})$  is a square matrix.



### 6.1.2 Maximisation Step

Here, the function  $\Gamma(Q; Q_l)$  in (6.23) will be maximised with respect to  $Q$ . Note that maximisation with respect to  $Q$  is the same as maximisation with respect to  $Q^{-1}$ , which can be seen as the argument in (6.23). The maximisation of  $\Gamma(Q; Q_l)$  is obtained if the derivative of  $\Gamma(Q; Q_l)$  with respect to  $Q^{-1}$  is equal to zero. The derivative of  $\Gamma(Q; Q_l)$  can be derived analytical, which results in large complex expressions. However, for the case with  $F_2(x_k)$  given by (6.2) there is a much simpler expression. The following matrix derivatives are useful for the derivation (Lütkepohl, 1996),

$$\frac{\partial \log |X|}{\partial X} = (X^T)^{-1}, \quad (6.30a)$$

$$\frac{\partial \text{tr} XB}{\partial X} = B^T. \quad (6.30b)$$

The derivative of (6.28) can now be written as

$$\frac{\partial \Gamma(Q; Q_l)}{\partial Q^{-1}} = \frac{N-1}{2} Q - \frac{1}{2} \sum_{i=2}^N F_2^\dagger(\hat{x}_{i-1|N}^s) M_i (F_2^\dagger(\hat{x}_{i-1|N}^s))^T = 0, \quad (6.31)$$

since  $M_i$  is a symmetric matrix. The solution of the maximisation step is finally obtained as

$$Q_{l+1} = \frac{1}{N-1} \sum_{i=2}^N F_2^\dagger(\hat{x}_{i-1|N}^s) M_i (F_2^\dagger(\hat{x}_{i-1|N}^s))^T. \quad (6.32)$$

### 6.1.3 Stop Criterion

The stop criterion can be chosen in different ways. Section 3.3 suggests that the EM algorithm stops when the difference in the new and previous estimate is less than a threshold. Another way is to use  $L_Q(y_{1:N})$  in (3.19). The main problem is to maximise  $L_Q(y_{1:N})$ , therefore stop the algorithm when no increase in  $L_Q(y_{1:N})$  can be observed. Equation (3.19) can be written as

$$\begin{aligned} L_Q(y_{1:N}) &= \log p_Q(y_{1:N}) = \log \left( p(y_1) \prod_{i=1}^{N-1} p_Q(y_{i+1}|y_{1:i}) \right) \\ &= \log p(y_1) + \sum_{i=1}^{N-1} \log p_Q(y_{i+1}|y_{1:i}), \end{aligned} \quad (6.33)$$

where Bayes' rule has been used repeatedly. Here,  $\log p(y_1)$  is a constant and can be omitted in the sequel for simplicity. The PDF  $p_Q(y_{i+1}|y_{1:i})$  is identified as the PDF for the innovations which can be calculated as

$$p_Q(y_{i+1}|y_{1:i}) = \mathcal{N}(y_{i+1}; h(\hat{x}_{i+1|i}), H_{i+1} P_{i+1|i} H_{i+1}^T + R), \quad (6.34)$$

$$H_{i+1} = \left. \frac{\partial h(x)}{\partial x} \right|_{x=\hat{x}_{i+1|i}}, \quad (6.35)$$

where  $\hat{x}_{i+1|i}$  and  $P_{i+1|i}$  are calculated in the EKF during the measurement update. The algorithm can now be stopped when

$$\left| L_{Q_l}(y_{1:N}) - L_{Q_{l-m}}(y_{1:N}) \right| \leq \gamma, \quad (6.36)$$

where  $m$  and  $\gamma$  are positive parameters to choose. The steps for calculating the matrix  $Q$  are summarised in Algorithm 7.

---

**Algorithm 7** Calculation of  $Q$  using the EM algorithm.

---

- 1: Select an initial value  $Q_0 \in \mathbb{S}_+^{n_v}$  and set  $l = 0$ .
- 2: Calculate  $M_i$  according to (6.20) using  $Q_l$ .
- 3: Compute the update of  $Q$  according to

$$Q_{l+1} = \frac{1}{N-1} \sum_{i=2}^N F_2^\dagger(\hat{x}_{i-1|N}^s) M_i \left( F_2^\dagger(\hat{x}_{i-1|N}^s) \right)^\top.$$

- 4: If  $\left| L_{Q_l}(y_{1:N}) - L_{Q_{l-m}}(y_{1:N}) \right| \leq \gamma$ , stop. If not, set  $l = l + 1$  and go to step 2.
- 

## 6.2 Alternative Ways to Find the Covariance Matrix of the Process Noise

Two alternative ways to estimate the covariance matrix for the process noise are presented here. These two alternatives, which are less complicated than the EM algorithm, will be compared to the result of the EM algorithm in Section 6.3. The first alternative is the same as in Section 5.2.1, i.e., minimisation of the path error, see algorithm 6.

The second method starts with an initial guess  $Q_0$ . The smoothed states are then calculated using  $Q_0$ . After that, equation (6.1a) and the smoothed states are used in order to derive the noise  $v_k$ ,  $k = 1, \dots, N$ . The covariance matrix is finally obtained from the sequence  $v_{1:N} = \{v_1 \dots v_N\}$ . The method is repeated with the new  $Q$ -matrix until convergence is obtained. The method is summarised in Algorithm 8.

---

**Algorithm 8** Iterative covariance estimation with the EKS

---

- 1: Select an initial value  $Q_0$  and set  $l = 0$ .
- 2: Use the EKS with  $Q_l$ .
- 3: Calculate the noise according to

$$v_k = F_2^\dagger(\hat{x}_{k|N}^s) \left( \hat{x}_{k+1|N}^s - F_1(\hat{x}_{k|N}^s, u_k) \right).$$

- 4: Let  $Q_{l+1}$  be the covariance matrix for  $v_k$  according to

$$Q_{l+1} = \frac{1}{N} \sum_{k=1}^N v_k v_k^\top.$$

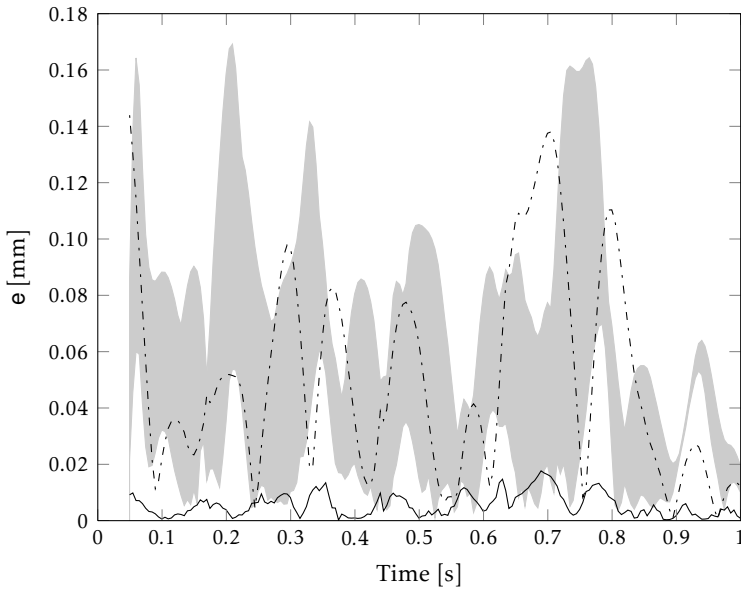
- 5: If converged, stop. If not, set  $l = l + 1$  and go to step 2.
-

## 6.3 Simulation Results

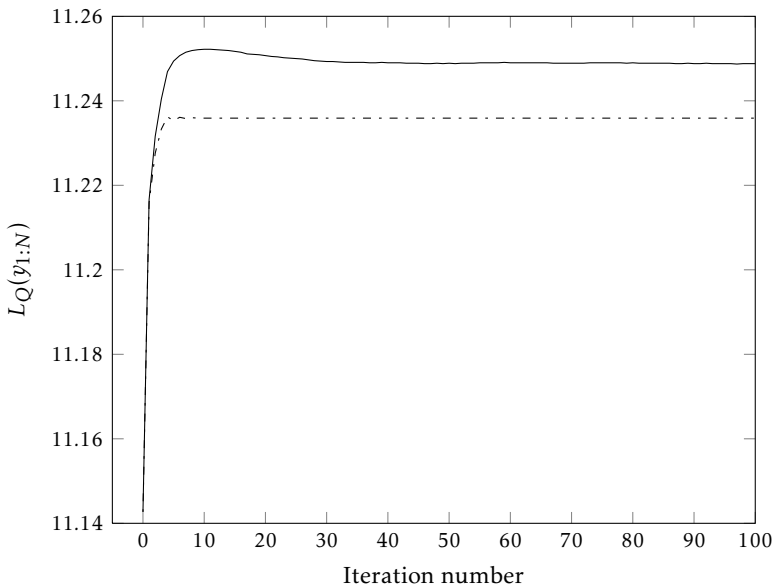
The method in Section 6.1 is evaluated and compared to the two alternative methods described in Section 6.2. The model given in Section 4.4.1, where  $f(x_k, u_k) = F_1(x_k, u_k)$  and  $g(x_k) = F_2(x_k)$ , is used, and the simulation model described in Section 5.1 with SIM1 and SIM2, see Table 5.1, is simulated to get all the required quantities, i.e.,  $u_k$ ,  $y_k$ ,  $x_k$  and  $z_k$ . The rank conditions in order to use (6.22) are satisfied for the model. In system identification, it is common to estimate a certain parameter or parameters starting at different initial values and see if the true one is obtained. Here, on the other hand, there is no information about the true covariance matrix, even for simulated data. The process noise is here implicit in the simulation model and originates from e.g. discretisation errors, model errors, and torque ripple. Instead, the estimated covariance matrices, for different initial values, are used to calculate the path error according to (5.1). When the path error differs a lot with different initial values it means that the method converges to different solutions, which is the case for Algorithm 6 as described in Section 5.4. There is however no guarantee that a solution is in a global optimum although the path errors do not differ. Here, the maximum and minimum of the RMSE values in (5.8) for all MC simulation are used to see how much the solutions differ with different initial values. It is preferred to have a method that results in small and similar path errors for different initial values.

It has already been concluded in Section 5.4 that Algorithm 6 gives different solutions for different initial values. Table 6.1 shows that the maximal and minimal path errors for the EM algorithm are more or less the same for SIM1. The same concerns Algorithm 8. The EM algorithm gives however a lower path error. A comparison between the path errors for the EM algorithm and Algorithm 8 is shown in Figure 6.1a, where the grey area covers all the path errors from Algorithm 6, i.e., the edges of the grey area are the maximum and minimum path error for each sample for all MC simulations in Figure 5.9a. The EM algorithm is clearly much better than the two alternatives. Compare also with Figure 5.8.

It is also interesting to see how (6.33) looks like for  $Q_l$ ,  $l = 0, \dots$ , both for the EM algorithm and Algorithm 8. The EM algorithm and Algorithm 8 were therefore forced to take more iterations than necessary. The log-likelihood function (6.33) can be seen in Figure 6.1b for 100 iterations. It can be seen that the curve for the EM algorithm flattens out somewhere around 50 iterations and stays constant after that. It means that it is unnecessary to continue to more than about 60 iterations. One thing to comment is the peak around ten iterations in the curve. This contradicts the property of the EM algorithm that the sequence  $Q_l$ ,  $l = 0, \dots$ , approximates  $\hat{Q}^{\text{ML}}$  better and better. This can be explained by the approximations that have been made during the expectation step and that the calculation of (6.33) in the EKF is approximative. The curve for Algorithm 8 flattens out after ten iterations and stays constant after that. Algorithm 8 is also without any peak and the stationary value is lower than for the EM algorithm. That means that the estimated covariance matrix from the EM algorithm reflects the true noise covariance better. For Algorithm 6, the solution gives a value of (6.33) that is lower



(a) The resulting path error for the EM algorithm (solid) and Algorithm 8 (dash-dot). The grey area covers all the resulting path errors from Algorithm 6.



(b) The log-likelihood function for the first 100 iterations in the EM algorithm (solid) and Algorithm 8 (dash-dot).

**Figure 6.1:** Path errors and the log-likelihood function for SIM1.

**Table 6.1:** Max and min of the RMSE values of the path error in mm for the EM algorithm and Algorithm 8 for SIM1.

	Max	Min
EM	0.0069	0.0069
Alg. 8	0.0615	0.0615

**Table 6.2:** Max and min of the RMSE values of the path error in mm for the EM algorithm and Algorithm 8 for SIM2.

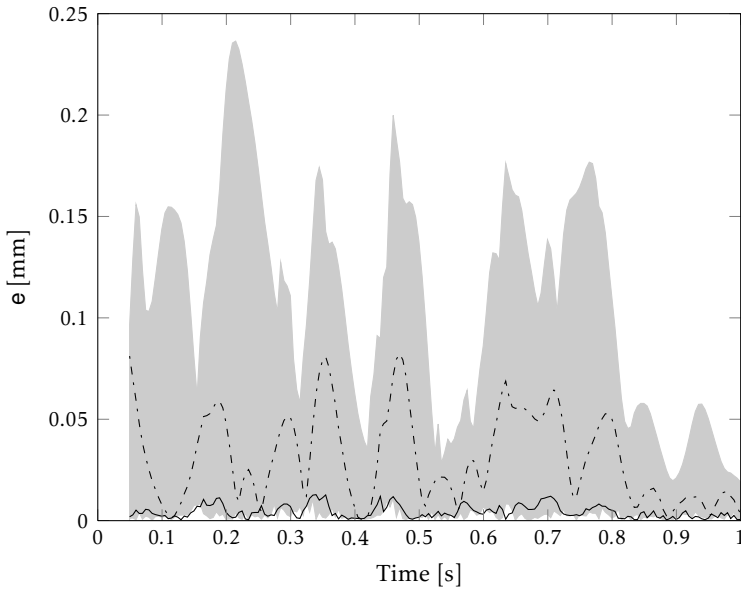
	Max	Min
EM	0.0054	0.0054
Alg. 8	0.0377	0.0377

than for the other two methods.

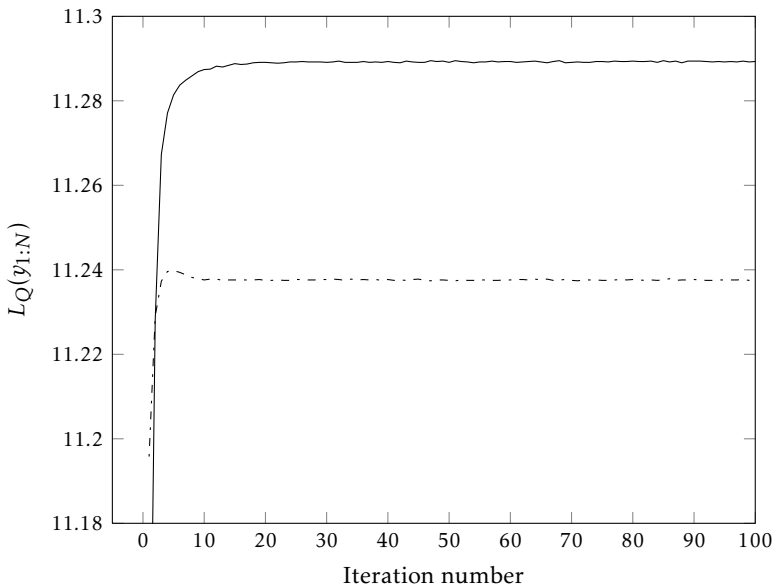
For SIM2 both the EM algorithm and Algorithm 8 give consistent values for different initial values, see Table 6.2, whereas different path errors are obtained for Algorithm 6, as has been described in Section 5.4. Figure 6.2a shows the path errors for the EM algorithm and Algorithm 8, where the grey area is the same as described above but for the path errors in Figure 5.9b. What is interesting is that Algorithm 6 sometimes gives lower path error than the EM algorithm for some simulation, whereas the path error is much worse for other simulations. The EM algorithm can be seen to handle model errors better than Algorithm 6. Compare also to the result in Figure 5.8. In Figure 6.2b it can be seen that the EM algorithm and Algorithm 8 converges in the same number of iterations and that the EM algorithm gives a higher likelihood. Once again, Algorithm 6 gives a lower value on (6.33) for the optimal solution, than the other two methods.

## 6.4 Conclusions

Three different methods to estimate the covariance matrix for the process noise have been compared. The EM algorithm derived in Section 6.1 gives a lower path error, considering the true path and the estimated path from an EKF. The EM algorithm is also robust to changes in the initial value. One advantage with the EM algorithm is that no true tool position is needed, which is the case for Algorithm 6.



(a) The resulting path error for the EM algorithm (solid) and Algorithm 8 (dash-dot). The grey area covers all the resulting path errors from Algorithm 6.



(b) The log-likelihood function for the first 100 iterations in the EM algorithm (solid) and Algorithm 8 (dash-dot).

**Figure 6.2:** Path errors and the log-likelihood function for SIM2.

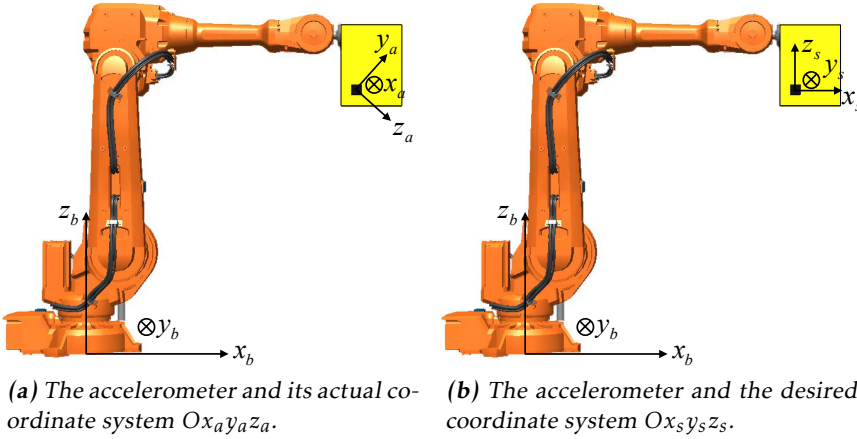
# 7

---

## Pose Estimation of an Accelerometer

In this chapter a novel method to estimate the position and orientation of a tri-axial accelerometer mounted on an industrial robot is presented. Previous work, e.g. Renk et al. (2005) and Won and Golnaraghi (2010), are only focused on the accelerometer calibration, i.e., the internal parameters of the accelerometer, such as sensitivity and bias are identified, but also alignment of each one of the three accelerometer measurement channels is considered. The estimation method presented here uses a two step procedure where the first step is to identify the orientation of the sensor using a number of static experiments. It is assumed that the sensor is mounted in such a way that it can be arbitrarily oriented using the six DOF robot arm. The desired orientation of the sensor is hence known while the actual orientation is unknown. In the second step the position of the accelerometer is estimated using dynamic experiments. The orientation is obtained from an optimisation problem with a closed form solution and the position is obtained from an overdetermined system of equations, hence a least square solution is required. Most parts of this chapter have previously been published in Axelsson and Norrlöf (2011b).

The chapter starts with the problem formulation in Section 7.1. The solution to the orientation estimation is presented in Section 7.2 and the estimation of the mounting position is presented in Section 7.3. Finally, the results from experimental evaluations are given in Section 7.4 and Section 7.5 concludes the chapter.



**Figure 7.1:** The accelerometer mounted on the robot. The yellow rectangle represents the tool or a weight and the black square on the yellow rectangle is the accelerometer. The base coordinate system  $Ox_b y_b z_b$  of the robot is also shown.

## 7.1 Problem Formulation

Assume that the accelerometer is mounted on the robot according to Figure 7.1a where the sensor is assumed to be rigidly attached to the robot tool. Given a definition of the tool coordinate system the estimation method presented in this chapter finds the relative orientation and position of the triaxial sensor. The orientation of the desired coordinate system can be seen in Figure 7.1b. Let  $\rho_a$  be an accelerometer measurement vector in the sensor coordinate system  $Ox_a y_a z_a$  of the accelerometer and  $\rho_s$  an acceleration vector in the desired coordinate system  $Ox_s y_s z_s$ , describing the acceleration in  $\text{m/s}^2$ . The relation between  $\rho_a$  and  $\rho_s$  is given by,

$$\rho_s = \kappa \mathcal{R}_{a/s} \rho_a + \rho_0, \quad (7.1)$$

where  $\mathcal{R}_{a/s}$  is the rotation matrix from  $Ox_a y_a z_a$  to  $Ox_s y_s z_s$ ,  $\kappa$  is the accelerometer sensitivity and  $\rho_0$  is the bias. It is assumed that the same sensitivity value  $\kappa$  can be used for all three sensors in the triaxial accelerometer. The sensitivity and bias are chosen such that the unit in  $Ox_s y_s z_s$  is  $\text{m/s}^2$ . When the unknown parameters in (7.1) have been found the position of the accelerometer is identified, where the position is expressed relative to the wrist coordinate system. To solve for the unknown parameters, the vector  $\rho_a$  is measured by the accelerometer while the vector  $\rho_s$  is computed from a model. In the static case  $\rho_s$  is simply the gravity vector, while in the dynamic case when the sensor is moved the acceleration will depend on the speed and orientation of the sensor. To be able to divide the estimation problem in two distinct problems the orientation is estimated using static measurements only while the position of the sensor is found by moving the ac-



celerometer along a known path with known speed. Using the known orientation of the accelerometer it is possible to numerically cancel the effect of gravity and only measure the dynamic acceleration, with constant speed in a circular path, perpendicular to the gravity field. The orientation of the accelerometer is kept fixed with respect to the path coordinates during the motion. This means that the acceleration originating from the movement can be isolated from the gravity component. A special case is when  $Ox_s y_s z_s$  is rotated such that the coordinate system of the accelerometer is directed to give gravity measurements along one coordinate axis only. The two other axes of the accelerometer directly gives the dynamic acceleration component which can be used to estimate the position.

## 7.2 Identification of Orientation, Sensitivity and Bias

The relation in (7.1) cannot be satisfied exactly for all times due to noisy measurements and other disturbances acting on the system. To solve for the parameters  $\mathcal{R}_{a/s}$ ,  $\kappa$  and  $\rho_0$  in (7.1), the residual has to be defined according to

$$e_k = \rho_{s,k} - \kappa \mathcal{R}_{a/s} \rho_{a,k} - \rho_0, \quad (7.2)$$

where  $k$  indicates the sample number. The problem is now to minimise a cost function which depends on the residual. Here the cost function is chosen as the sum of the squared norm of the residuals. Moreover, to ensure  $\mathcal{R}_{a/s} \in SO(3)$ , i.e., a rotation matrix, the two constraints  $\det(\mathcal{R}_{a/s}) = 1$  and  $\mathcal{R}_{a/s}^T = \mathcal{R}_{a/s}^{-1}$  have to be included in the optimisation problem. The optimisation problem can now be written as

$$\begin{aligned} & \text{minimise} && \sum_{k=1}^N \|e_k\|^2 \\ & \text{subject to} && \det(\mathcal{R}_{a/s}) = 1 \\ & && \mathcal{R}_{a/s}^T = \mathcal{R}_{a/s}^{-1} \end{aligned} \quad (7.3)$$

By introducing the centroids for the measurements in  $Ox_a y_a z_a$  and  $Ox_s y_s z_s$ ,

$$\bar{\rho}_s = \frac{1}{N} \sum_{k=1}^N \rho_{s,k}, \quad \bar{\rho}_a = \frac{1}{N} \sum_{k=1}^N \rho_{a,k}, \quad (7.4)$$

and defining new coordinates,

$$\rho'_{s,k} = \rho_{s,k} - \bar{\rho}_s, \quad \rho'_{a,k} = \rho_{a,k} - \bar{\rho}_a, \quad (7.5)$$

the optimisation problem has the closed form solution (Horn et al., 1988),

$$\kappa = \sqrt{\frac{\sum_{k=1}^N \|\rho'_{s,k}\|^2}{\sum_{k=1}^N \|\rho'_{a,k}\|^2}}, \quad (7.6a)$$

$$\mathcal{R}_{a/s} = M (M^T M)^{-1/2}, \quad (7.6b)$$

$$\rho_0 = \bar{\rho}_s - \kappa \mathcal{R} \bar{\rho}_a, \quad (7.6c)$$

where

$$M = \sum_{k=1}^N \rho'_{s,k} (\rho'_{a,k})^T. \quad (7.7)$$

$N$  is the total number of measurements and it has to be assumed that  $N \geq 3$ . In addition a condition of sufficient excitement has to be fulfilled, such that  $M^T M$  has full rank. As an alternative to the formulation above where the rotation is parameterised by the orthonormal matrix  $\mathcal{R}_{a/s}$  it is also possible to find a closed-form solution to (7.1) using unit quaternions, see e.g. Horn (1987). With respect to the number of operations, the matrix formulation is, however, computationally more efficient than the quaternion formulation.

As indicated in Section 7.1 the orientation and the sensor parameters are found using static measurements, i.e., moving the tool into a number,  $N_C$ , of different configurations. The gravity vector is measured by the accelerometer in each of the  $N_C$  configurations, which gives  $N_{M,j}$ ,  $j = 1, \dots, N_C$  measurements for each configuration. Let

$$\{\rho_a\} = \left\{ \left\{ \rho_{a,i}^1 \right\}_{i=1}^{N_{M,1}}, \dots, \left\{ \rho_{a,i}^{N_C} \right\}_{i=1}^{N_{M,N_C}} \right\} \quad (7.8)$$

denote the set of all the  $N = \sum_{j=1}^{N_C} N_{M,j}$  measurements in all  $N_C$  configurations, and let

$$\{\rho_s\} = \left\{ \left\{ \rho_s^1 \right\}_{i=1}^{N_{M,1}}, \dots, \left\{ \rho_s^{N_C} \right\}_{i=1}^{N_{M,N_C}} \right\} \quad (7.9)$$

be the gravity vector from the model in the desired coordinate system  $Ox_s y_s z_s$  for each configuration, where  $\rho_s^j$ ,  $j = 1, \dots, N_C$  is a constant. Using the measured accelerations and the model values to solve the optimisation problem in (7.3) according to (7.4) to (7.7) the transformation parameters can be computed.

The  $N_C$  different configurations can be chosen arbitrary as long as the matrix  $M^T M$  has full rank<sup>1</sup>. Here six different configurations, as shown in Figure 7.2, are suggested. From Figure 7.2 it can be seen that

$$\begin{aligned} \rho_s^1 &= \begin{pmatrix} 0 & 0 & g \end{pmatrix}^T, & \rho_s^2 &= \begin{pmatrix} 0 & g & 0 \end{pmatrix}^T, & \rho_s^3 &= \begin{pmatrix} 0 & 0 & -g \end{pmatrix}^T, \\ \rho_s^4 &= \begin{pmatrix} 0 & -g & 0 \end{pmatrix}^T, & \rho_s^5 &= \begin{pmatrix} -g & 0 & 0 \end{pmatrix}^T, & \rho_s^6 &= \begin{pmatrix} g & 0 & 0 \end{pmatrix}^T, \end{aligned} \quad (7.10)$$

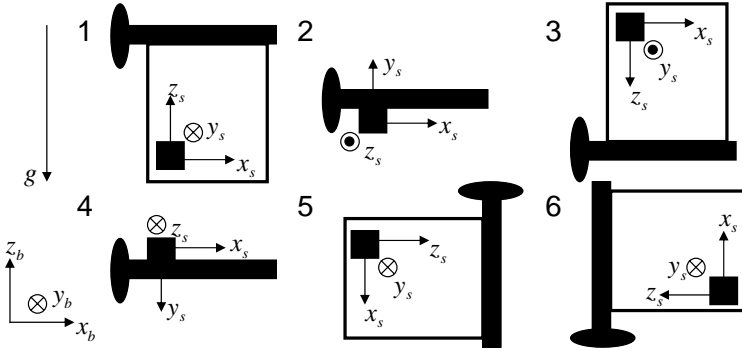
where  $g = 9.81 \text{ m/s}^2$ . Note that the vectors in (7.10) are directed towards the gravity vector in Figure 7.2. The explanation for this is that an accelerometer measures the normal force which is directed towards the gravity vector.

The six configurations in Figure 7.2 are straightforward to obtain for a six degree of freedom industrial manipulator, see Chapter 2. The procedure to estimate the triaxial accelerometer sensor parameters is summarised in Algorithm 9.

<sup>1</sup>The matrix  $M^T M$  has always full rank if none of the two sets  $\{\rho_a\}$  and  $\{\rho_s\}$  are coplanar.

**Algorithm 9** Estimation of the sensor parameters

- 1: Measure the acceleration for the different configurations in Figure 7.2 to obtain  $\{\rho_a\}$  according to (7.8).
- 2: Construct  $\{\rho_s\}$  in (7.9) from (7.10).
- 3: Calculate  $\mathcal{R}_{a/s}$ ,  $\kappa$  and  $\rho_0$  from (7.4) to (7.7).



**Figure 7.2:** Six different configurations of the robot tool used in Algorithm 9. The orientation of the desired coordinate system  $Ox_s y_s z_s$  is shown for each configuration. The base coordinate system  $Ox_b y_b z_b$  and the gravity vector are also shown.

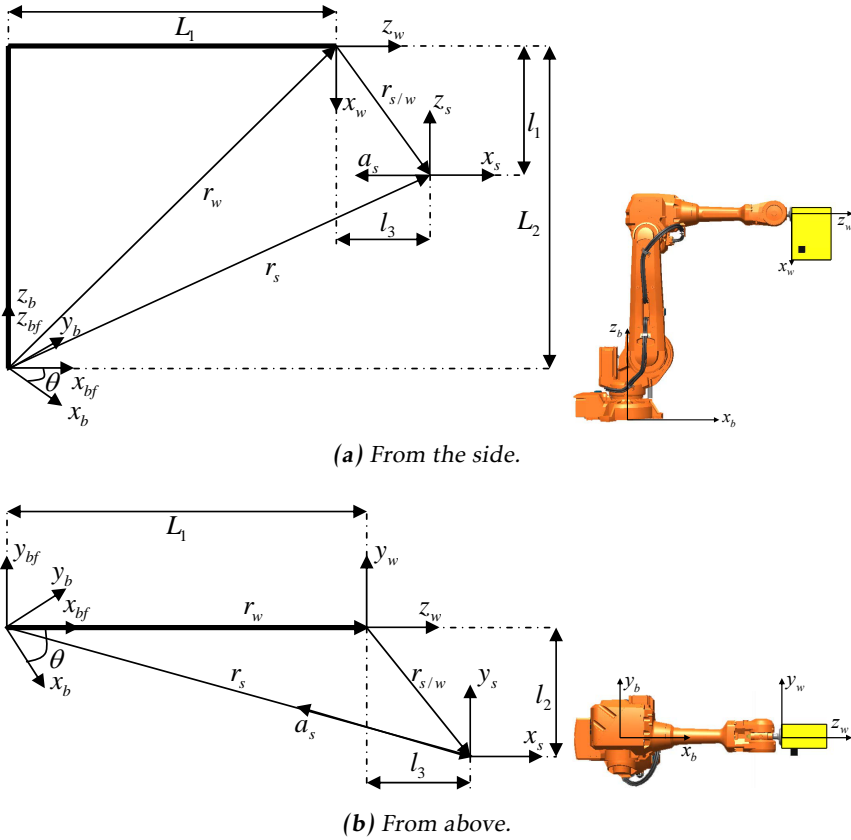
### 7.3 Estimation of the Position of the Accelerometer

In the second step of the proposed orientation and position estimation process a method for the position estimation is explained for the accelerometer's coordinate system  $Ox_s y_s z_s$ , expressed in a coordinate system  $Ox_{bf} y_{bf} z_{bf}$  fixed to the robot. From Section 7.2 the orientation and sensor parameters are known, hence the acceleration measured by the accelerometer has a known orientation.

Using a mathematical model of the robot motion it is possible to compute the acceleration, parameterised in some unknown parameters. To simplify the mathematical model for the acceleration and to make it possible to parameterise the unknown parameters, consider the case when the robot is in the configuration shown in Figure 7.3. The figure shows the vector  $r_s$ , the two coordinate systems  $Ox_{bf} y_{bf} z_{bf}$  and  $Ox_s y_s z_s$ , the world fixed coordinate system  $Ox_b y_b z_b$  attached to the base of the robot, the coordinate system  $Ox_w y_w z_w$  fixed to the end of the robot arm, and the vector

$$a_s \triangleq \frac{d^2}{dt^2} r_s, \quad (7.11)$$

which describes the acceleration of  $Ox_s y_s z_s$ . The mathematical expression for  $a_s$  together with the measured acceleration are used in order to estimate the unknown parameters. The figure also shows a parameter  $\theta$  describing the rotation



**Figure 7.3:** The first robot configuration for estimation of the mounting position. The black cube on the yellow box indicates the sensor, i.e., the origin of  $Ox_s y_s z_s$ . The yellow box is attached to the robot in the point  $(L_1 \ 0 \ L_2)^T$  expressed in  $Ox_{bf} y_{bf} z_{bf}$ .

between  $Ox_{bf} y_{bf} z_{bf}$  and  $Ox_b y_b z_b$ , two known parameters  $L_1$  and  $L_2$  describing the lengths of the arms and the three unknown parameters  $l_i$ ,  $i = 1, 2, 3$  describing the vector  $r_{s/w}$  in  $Ox_w y_w z_w$ .

All the calculations are done in the world fixed coordinate system in order to obtain an expression for  $\frac{d^2}{dt^2} r_s$ . In a body fixed coordinate system  $Ox_{bf} y_{bf} z_{bf}$   $\frac{d^2}{dt^2} r_s = 0$ . In the sequel, the notation  $[r_s]_i$  is used to emphasise that  $r_s$  is expressed in coordinate system  $i$ .

Figure 7.3 shows that  $r_s$  can be written as a sum of two vectors,

$$[r_s]_{bf} = [r_w]_{bf} + [r_{s/w}]_{bf}, \quad (7.12)$$

where

$$[r_{s/w}]_{bf} = (l_3 \quad -l_2 \quad -l_1)^\top, \quad (7.13)$$

$$[r_w]_{bf} = (L_1 \quad 0 \quad L_2)^\top. \quad (7.14)$$

The transformation of  $r_s$  from  $Ox_{bf}y_{bf}z_{bf}$  to  $Ox_b y_b z_b$  can be expressed as

$$[r_s]_b = [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}), \quad (7.15)$$

where

$$[\mathcal{Q}_{bf/b}]_b = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (7.16)$$

is the rotation matrix that relates the coordinate system  $Ox_{bf}y_{bf}z_{bf}$  to  $Ox_b y_b z_b$ , and  $\theta = \theta(t)$  is the angle relating  $Ox_b y_b z_b$  and  $Ox_{bf}y_{bf}z_{bf}$  according to Figure 7.3. Taking the derivative of  $[r_s]_b$  with respect to time gives

$$\frac{d}{dt} [r_s]_b = \left( \frac{d}{dt} [\mathcal{Q}_{bf/b}]_b \right) ([r_w]_{bf} + [r_{s/w}]_{bf}). \quad (7.17)$$

The time derivative of the rotation matrix is given by (Spong et al., 2005)

$$\frac{d}{dt} [\mathcal{Q}_{bf/b}]_b = S(\omega) [\mathcal{Q}_{bf/b}]_b, \quad (7.18)$$

where  $\omega = (0 \quad 0 \quad \dot{\theta})^\top$  and

$$S(\omega) = \begin{pmatrix} 0 & -\dot{\theta} & 0 \\ \dot{\theta} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (7.19)$$

is a skew symmetric matrix. Hence, the time derivative of  $[r_s]_b$  can be written

$$\frac{d}{dt} [r_s]_b = S(\omega) [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}). \quad (7.20)$$

The second time derivative of  $[r_s]_b$  becomes

$$\begin{aligned} [a_s]_b &= \frac{d^2}{dt^2} [r_s]_b = \left( \frac{d}{dt} S(\omega) \right) [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}) \\ &+ S(\omega) \left( \frac{d}{dt} [\mathcal{Q}_{bf/b}]_b \right) ([r_w]_{bf} + [r_{s/w}]_{bf}) = S(\dot{\omega}) [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}) \\ &+ S(\omega) S(\omega) [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}) = S(\omega) S(\omega) [\mathcal{Q}_{bf/b}]_b ([r_w]_{bf} + [r_{s/w}]_{bf}), \end{aligned} \quad (7.21)$$

where  $\dot{\omega} = (0 \quad 0 \quad 0)^\top$  follows from the assumption of constant angular velocity.

It now remains to transform the measured acceleration  $a_s^M$  from the sensor frame

$Ox_s y_s z_s$  to the base frame  $Ox_b y_b z_b$ . From Figure 7.3 it can be seen directly that

$$\begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf} = \begin{pmatrix} a_{s,x}^M & a_{s,y}^M & 0 \end{pmatrix}^T, \quad (7.22)$$

hence

$$\begin{bmatrix} a_s^M \\ \end{bmatrix}_b = \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b \begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf}. \quad (7.23)$$

Equations (7.21) and (7.23) give

$$\begin{aligned} \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b \begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf} &= S(\omega)S(\omega) \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b ([r_w]_{bf} + [r_{s/w}]_{bf}) \\ &\Leftrightarrow \\ \begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf} &= \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b^T S(\omega)S(\omega) \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b ([r_w]_{bf} + [r_{s/w}]_{bf}) \end{aligned} \quad (7.24)$$

since  $\begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b^T = \begin{bmatrix} \mathcal{Q}_{bf/b} \\ \end{bmatrix}_b^{-1}$ . Carrying out the matrix multiplication for the right-hand side of (7.24) gives

$$\begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf} = \begin{pmatrix} -\dot{\theta}^2(L_1 + l_3) \\ \dot{\theta}^2 l_2 \\ 0 \end{pmatrix}, \quad (7.25)$$

where (7.13), (7.14), (7.16) and (7.19) have been used. Equations (7.22) and (7.25) can now be written as a system of equations where  $l_2$  and  $l_3$  are unknown,

$$\begin{pmatrix} 0 & -\dot{\theta}^2 \\ \dot{\theta}^2 & 0 \end{pmatrix} \begin{pmatrix} l_2 \\ l_3 \end{pmatrix} = \begin{pmatrix} a_{s,x}^M + \dot{\theta}^2 L_1 \\ a_{s,y}^M \end{pmatrix}. \quad (7.26)$$

It is thus possible to find  $l_2$  and  $l_3$  from (7.26) but unfortunately not  $l_1$ . Rotating the sensor according to Figure 7.4 will give information about  $l_1$ . The same calculations as before with

$$[r_{s/w}]_{bf} = \begin{pmatrix} -l_1 & -l_2 & -l_3 \end{pmatrix}^T, \quad (7.27)$$

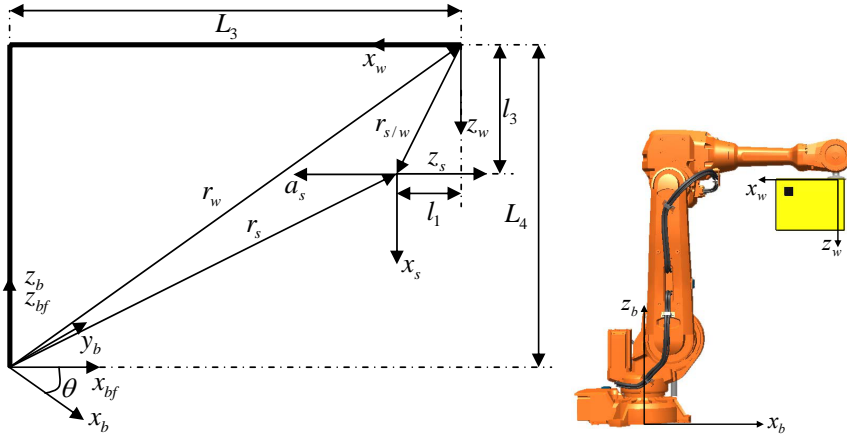
$$[r_w]_{bf} = \begin{pmatrix} L_3 & 0 & L_4 \end{pmatrix}^T, \quad (7.28)$$

$$\begin{bmatrix} a_s^M \\ \end{bmatrix}_{bf} = \begin{pmatrix} a_{s,z}^M & a_{s,y}^M & 0 \end{pmatrix}^T, \quad (7.29)$$

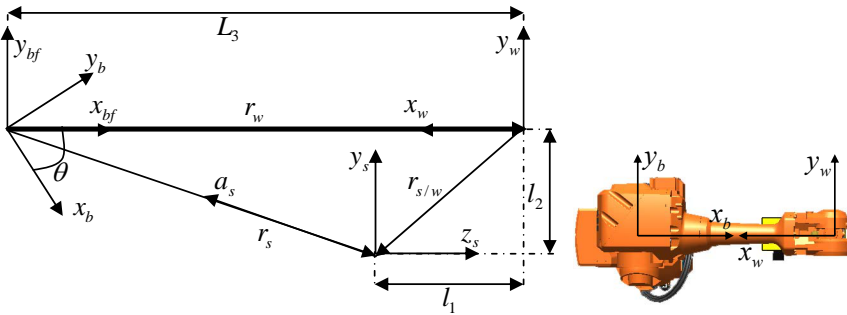
see Figure 7.4, give

$$\begin{pmatrix} \dot{\theta}^2 & 0 \\ 0 & \dot{\theta}^2 \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \end{pmatrix} = \begin{pmatrix} a_{s,z}^M + \dot{\theta}^2 L_3 \\ a_{s,y}^M \end{pmatrix}. \quad (7.30)$$

Equations (7.26) and (7.30) can now be used to estimate the unknown parameters. Using (7.26) and (7.30) the estimation of  $l_2$  uses approximately twice as much data than the estimation of  $l_1$  and  $l_3$ . To get equal amount of data for each parameter, which gives a more accurate estimation, the robot configuration in

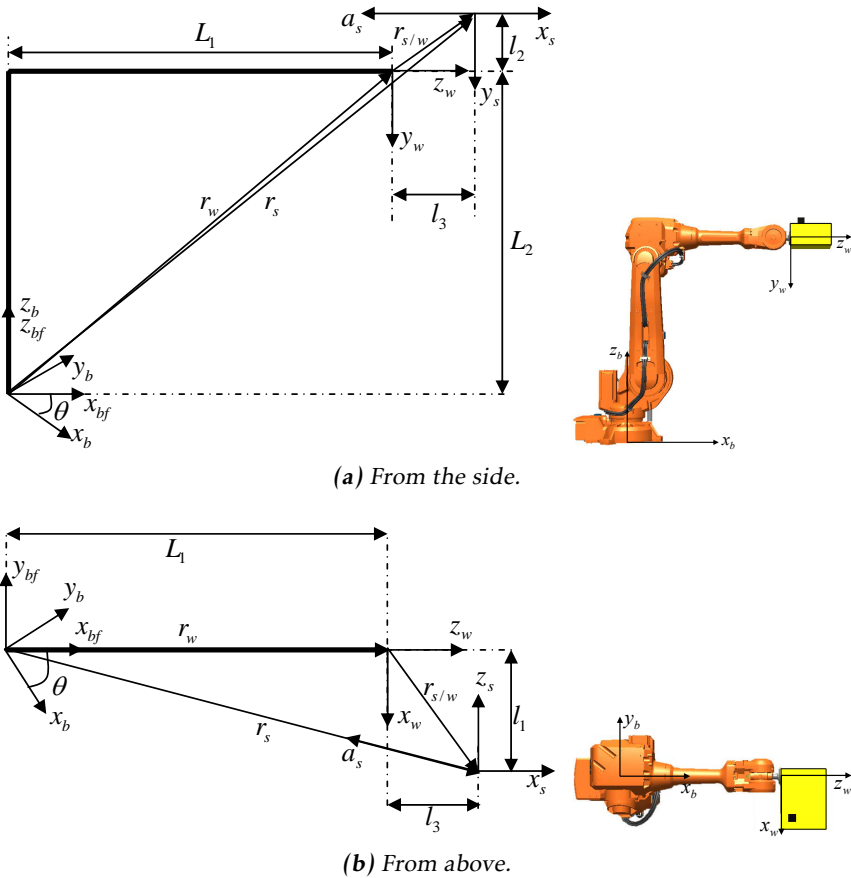


(a) From the side.



(b) From above.

**Figure 7.4:** The second robot configuration for estimation of the mounting position. The black cube on the yellow box indicates the sensor, i.e., the origin of  $Ox_s y_s z_s$ . The yellow box is attached to the robot in the point  $(L_3 \ 0 \ L_4)^T$  expressed in  $Ox_{bf} y_{bf} z_{bf}$ .



**Figure 7.5:** The third robot configuration for estimation of the mounting position. The black cube on the yellow box indicates the sensor, i.e., the origin of  $Ox_s y_s z_s$ . The yellow box is attached to the robot in the point  $(L_1 \ 0 \ L_2)^T$  expressed in  $Ox_b f y_b f z_b f$ .



Figure 7.5 is used, which gives

$$[r_{s/w}]_{bf} = (l_3 \quad -l_1 \quad l_2)^\top, \quad (7.31)$$

$$[r_w]_{bf} = (L_1 \quad 0 \quad L_2)^\top, \quad (7.32)$$

$$[a_s^M]_{bf} = (a_{s,x}^M \quad a_{s,z}^M \quad 0)^\top. \quad (7.33)$$

From (7.24) the following equation is obtained,

$$\begin{pmatrix} 0 & -\dot{\theta}^2 \\ \dot{\theta}^2 & 0 \end{pmatrix} \begin{pmatrix} l_1 \\ l_3 \end{pmatrix} = \begin{pmatrix} a_{s,x}^M + \dot{\theta}^2 L_1 \\ a_{s,z}^M \end{pmatrix}. \quad (7.34)$$

Equations (7.26), (7.30) and (7.34) can now be written as one system of equations according to

$$\underbrace{\begin{pmatrix} 0 & 0 & -\dot{\theta}_{c1}^2 \\ 0 & \dot{\theta}_{c1}^2 & 0 \\ \dot{\theta}_{c2}^2 & 0 & 0 \\ 0 & \dot{\theta}_{c2}^2 & 0 \\ 0 & 0 & -\dot{\theta}_{c3}^2 \\ \dot{\theta}_{c3}^2 & 0 & 0 \end{pmatrix}}_A \underbrace{\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix}}_l = \underbrace{\begin{pmatrix} a_{s,x,c1}^M + \dot{\theta}_{c1}^2 L_1 \\ a_{s,y,c1}^M \\ a_{s,z,c2}^M + \dot{\theta}_{c2}^2 L_3 \\ a_{s,y,c2}^M \\ a_{s,x,c3}^M + \dot{\theta}_{c3}^2 L_1 \\ a_{s,z,c3}^M \end{pmatrix}}_b, \quad (7.35)$$

where index  $ci$ ,  $i = 1, 2, 3$  indicates from which robot configuration the measurements come from. Equation (7.35) has more rows than unknowns, hence the solution to (7.35) is the solution to the optimisation problem

$$\arg \min_l \|b - Al\|_2^2, \quad (7.36)$$

which has the analytical solution

$$\hat{l} = (A^\top A)^{-1} A^\top b. \quad (7.37)$$

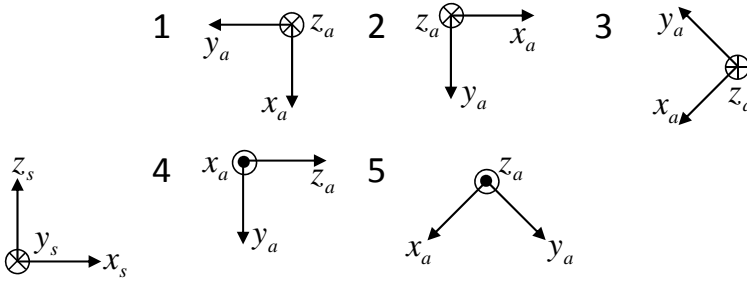
Note that there exist better numerical solutions to (7.35) than (7.37), e.g.  $l=A \setminus b$  in MATLAB. The procedure to estimate the position of the accelerometer is summarised in Algorithm 10.

---

**Algorithm 10** Estimation of the mounting position

---

- 1: Measure the acceleration of the tool  $[a_s^M]_s$  and the angular velocity  $\dot{\theta}$  for the three different configurations in Figures 7.3, 7.4 and 7.5 when  $\theta$  varies from  $\theta_{min}$  to  $\theta_{max}$  with constant angular velocity.
  - 2: Construct  $A$  and  $b$  in (7.35).
  - 3: Solve (7.35) with respect to  $l$ , for example according to (7.37).
-



**Figure 7.6:** Orientation for the five mounting positions that were used to evaluate the two algorithms. The orientation of the desired coordinate system is also shown.

## 7.4 Experimental Results

In this section the proposed orientation and position estimation method described in the two algorithms in Sections 7.2 and 7.3 is evaluated using experimental data. For Algorithm 9, the data, i.e., the acceleration values, are collected during 4s for each one of the six configurations in Figure 7.2 using a sample rate of 2 kHz. For Algorithm 10, the arm angular velocity  $\dot{\theta}$  for joint one and the acceleration measurements are collected when the robot is in the three different configurations according to Figures 7.3, 7.4 and 7.5. In the position estimation experiments data are collected during 4s in each one of the three configurations, but it is only the constant angular velocity part of the data that is used. The arm angular velocity of joint one is computed from the motor angular velocity  $\dot{\theta}_m$  using,

$$\dot{\theta} = \frac{\dot{\theta}_m}{\eta}, \quad (7.38)$$

where  $\eta$  is the gear ratio. The same sample rate as before is used, i.e., 2 kHz. The accelerometer used in the experiments is a triaxial accelerometer from Crossbow Technology, with a range of  $\pm 2g$ , and a sensitivity of approximately 1 V/g (Crossbow Technology, 2004). The accelerometer is connected to the measurement system of the robot controller, and hence the acceleration and motor angular velocity can be synchronised and measured with the same sampling rate.

Five different mounting positions and different orientations of the accelerometer have been used for evaluation of Algorithms 9 and 10. The actual physical orientation of the sensor was measured using a protractor, see Figure 7.6, where the orientation of the desired sensor coordinate system also is shown.

**Table 7.1:** Estimated parameters in (7.1) using Algorithm 9 for five different test cases.

Test	$\hat{\kappa}$	$\hat{\rho}_0$	$\hat{\mathcal{R}}_{a/s}$
1	9.91	$\begin{pmatrix} 25.05 \\ -23.75 \\ 24.26 \end{pmatrix}$	$\begin{pmatrix} -0.0138 & -0.9998 & -0.0170 \\ -0.0094 & -0.0169 & 0.9998 \\ -0.9999 & 0.0140 & -0.0092 \end{pmatrix}$
2	9.91	$\begin{pmatrix} -23.89 \\ -24.03 \\ 25.11 \end{pmatrix}$	$\begin{pmatrix} 0.9999 & -0.0070 & -0.0131 \\ 0.0129 & -0.0276 & 0.9995 \\ -0.0073 & -0.9996 & -0.0275 \end{pmatrix}$
3	9.91	$\begin{pmatrix} 34.80 \\ -23.73 \\ 3.07 \end{pmatrix}$	$\begin{pmatrix} -0.6348 & -0.7724 & -0.0208 \\ -0.0027 & -0.0247 & 0.9997 \\ -0.7727 & 0.6347 & 0.0135 \end{pmatrix}$
4	9.91	$\begin{pmatrix} -24.46 \\ 24.86 \\ 23.74 \end{pmatrix}$	$\begin{pmatrix} 0.0169 & -0.0139 & 0.9998 \\ -0.9992 & -0.0355 & 0.0164 \\ 0.0353 & -0.9993 & -0.0145 \end{pmatrix}$
5	9.92	$\begin{pmatrix} -3.91 \\ 24.95 \\ 33.81 \end{pmatrix}$	$\begin{pmatrix} -0.6314 & 0.7751 & 0.0209 \\ -0.0269 & 0.0050 & -0.9996 \\ -0.7750 & -0.6318 & 0.0177 \end{pmatrix}$

### 7.4.1 Evaluation of the Estimated Orientation, Sensitivity and Bias

Algorithm 9 was applied to the five test cases presented above and the result  $\hat{\mathcal{R}}_{a/s}$ ,  $\hat{\kappa}$  and  $\hat{\rho}_0$  can be seen in Table 7.1. From Figure 7.6 it can be seen that the rotation matrix  $R$  in (7.1) should resemble

$$\begin{aligned} \mathcal{R}_{a/s}^1 &= \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{pmatrix}, \mathcal{R}_{a/s}^2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}, \mathcal{R}_{a/s}^3 = \begin{pmatrix} -a^3 & -b^3 & 0 \\ 0 & 0 & 1 \\ -c^3 & d^3 & 0 \end{pmatrix} \\ \mathcal{R}_{a/s}^4 &= \begin{pmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}, \mathcal{R}_{a/s}^5 = \begin{pmatrix} -a^5 & b^5 & 0 \\ 0 & 0 & -1 \\ -c^5 & -d^5 & 0 \end{pmatrix}, \end{aligned}$$

where  $a, b, c$  and  $d$  are positive numbers that should be close to  $\cos(45^\circ) \approx 0.7071$ . The superscript indicates the test number. A rotational difference between the measured rotation matrix  $\mathcal{R}_{a/s}^i$  and the estimated matrix  $\hat{\mathcal{R}}_{a/s}^i$  can be computed using the corresponding unit quaternions  $q^i$  and  $\hat{q}^i$ . The rotation angle  $\vartheta^i$  of  $q_{\Delta}^i$ , where  $q_{\Delta}^i = (q^i)^{-1} * \hat{q}^i$ , which should be small, is a good measure of the difference between  $\mathcal{R}_{a/s}^i$  and  $\hat{\mathcal{R}}_{a/s}^i$ .

Given a rotation matrix  $\mathcal{R}$  the corresponding unit quaternion  $q = \{\eta, \epsilon\}$  can be

**Table 7.2:** The rotation angle  $\vartheta$  indicates how close the estimated and measured rotation matrices are to each other. The matrices are identical if  $\vartheta = 0^\circ$

Test	1	2	3	4	5
$\vartheta$	$1.4^\circ$	$1.8^\circ$	$5.8^\circ$	$2.4^\circ$	$6.0^\circ$

calculated according to (Sciavicco and Siciliano, 2000)

$$\eta = \frac{1}{2} \sqrt{r_{11} + r_{22} + r_{33} + 1}, \quad (7.39a)$$

$$\epsilon = \frac{1}{2} \begin{pmatrix} \text{sign}(r_{32} - r_{23}) \sqrt{r_{11} - r_{22} - r_{33} + 1} \\ \text{sign}(r_{13} - r_{31}) \sqrt{r_{22} - r_{33} - r_{11} + 1} \\ \text{sign}(r_{21} - r_{12}) \sqrt{r_{33} - r_{11} - r_{22} + 1} \end{pmatrix}, \quad (7.39b)$$

where  $r_{ij}$  is the element in row  $i$  and column  $j$  in  $\mathcal{R}$ . The inverse of a quaternion is given by  $q^{-1} = \{\eta, -\epsilon\}$  and the product of two quaternions is given by

$$q_1 * q_2 = \{\eta_1 \eta_2 - \epsilon_1^\top \epsilon_2, \eta_1 \epsilon_2 + \eta_2 \epsilon_1 + \epsilon_1 \times \epsilon_2\}. \quad (7.40)$$

Finally, the rotation angle  $\vartheta$  is given by

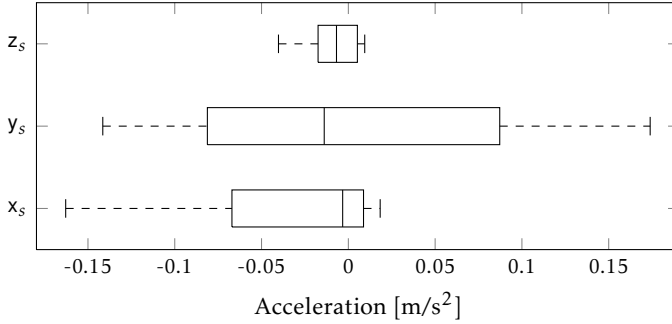
$$\vartheta = 2 \arccos(\eta). \quad (7.41)$$

The Robotics toolbox (Corke, 1996) in MATLAB can be used for computations involving quaternions.

The resulting rotation angle  $\vartheta^i$  for the five test cases can be seen in Table 7.2. The difference is small in all cases, but for tests 3 and 5 a larger deviation can be seen. One explanation for this is that it is more difficult to mount the accelerometer in a configuration not aligned with the robot tool, as seen in Figure 7.1.

It is more difficult to obtain true values for the parameters  $\kappa$  and  $\rho_0$ . To verify them, the measured acceleration vector for all five test cases in configuration 1, in Figure 7.2, is transformed from  $Ox_a y_a z_a$  to  $Ox_s y_s z_s$ , which results in three constant signals  $a_{s,x}^M$ ,  $a_{s,y}^M$  and  $a_{s,z}^M$  for the three axes of the accelerometer. Figure 7.2 shows that the measured acceleration in frame  $Ox_s y_s z_s$  should resemble  $a_{s,x} = 0$ ,  $a_{s,y} = 0$  and  $a_{s,z} = g$ . Subtracting  $a_{s,j}$  from the mean of  $a_{s,j}^M$ ,  $j = x, y, z$ , gives an error for the transformed acceleration. A diagram of the errors for each coordinate axis in  $Ox_s y_s z_s$  is shown in Figure 7.7. The diagram shows the median as the central mark, the edges of the box are the 25th and 75th percentiles and the dashed lines extend to the most extreme error. The errors are small and, as expected, the errors are larger in  $x$  and  $y$  due to the higher sensitivity to orientation errors in these axis when measuring gravity along the  $z$ -axis. The bias in  $x$  can be explained by a systematic error in orientation due to the robot elasticity and gravitational force acting on the robot in the evaluation position, see Figure 7.1.

The results presented in Tables 7.1 and 7.2 are based on one experiment for each test case. In order to get an estimation of the distribution of the estimated parameters, more precisely the mean and variance of the distribution for each pa-



**Figure 7.7:** Diagram of the transformation errors in the  $x$ -,  $y$ - and  $z$ -direction for (7.1) in configuration 1 (Figure 7.2) for all five test cases. The central mark is the median, the edges of the box are the 25th and 75th percentiles and the dashed lines extend to the most extreme error.

parameter, several experiments have to be performed. Repeating the experiments will give measurements with the same mean, that is, only the noise has changed in the new measurements. Instead of performing expensive experiments, the MC transformation described in Section 3.4 can be used to obtain an estimate of the variance of the estimated parameters. Firstly, new signals are generated from the distribution of the measurements in each robot configuration in Figure 7.2. Secondly, the transformation parameters  $\mathcal{R}_{a/s}$ ,  $\kappa$ , and  $\rho_0$ , as well as the rotation angle  $\vartheta$  are estimated with the new signals. This is repeated  $N_{MC}$  times for each test case. The mean and variance of the distribution for each parameter can now be calculated according to (3.28). Figure 7.8a shows a typical PDF for one coordinate axis in one of the six robot configurations. New samples are drawn from the empirical PDF  $p(x)$  in Figure 7.8a using the inverse transform sampling method described in Algorithm 11. The *cumulative distribution function* (CDF) can here be calculated as a cumulative sum resulting in the stair in Figure 7.8b.

---

#### Algorithm 11 Inverse Transform Sampling

---

- 1: Calculate the CDF  $P(x)$  from the PDF  $p(x)$ ,

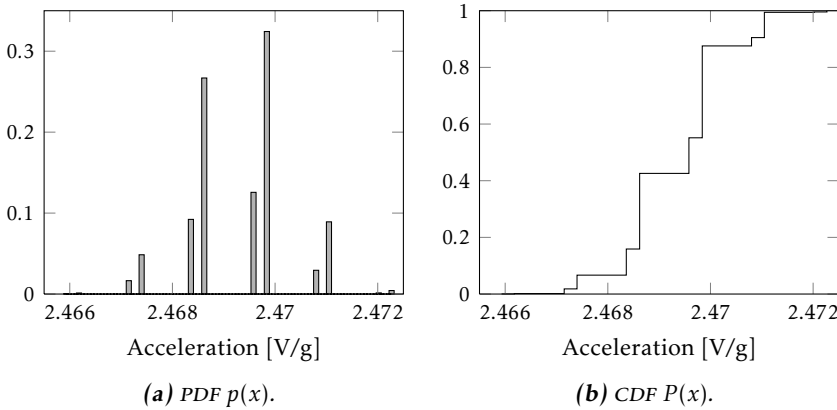
$$P(x) = \int_{-\infty}^x p(x') dx'. \quad (7.42)$$

- 2: Generate a number  $u$  from the uniform distribution  $\mathcal{U}(0, 1)$ .
- 3: Compute  $x$  from

$$P(x) = u. \quad (7.43)$$


---

Table 7.3 shows the mean and variance for each parameter, where  $\mathcal{R}_{a/s}$  is represented by each individual element, over  $N_{MC} = 1\,000$  MC simulations. The mean of the parameters coincide with the estimated parameters in Table 7.1. Table 7.4b also shows that the elements in  $\mathcal{R}_{a/s}$  that are close to  $\pm 1$  are more accurate than



**Figure 7.8:** Empirical PDF and CDF for the measured data for the  $y_a$  axis for test 1 in configuration 1.

the ones that are close to zero, compare e.g.  $r_{11}$  and  $r_{12}$  for test 1. Since the matrix  $\mathcal{R}_{a/s}$  has low variances for each element, the variance of the rotation angle  $\vartheta$  should be small, which is confirmed in Table 7.3b.

## 7.4.2 Evaluation of the Estimated Position

Algorithm 10 was also applied for the five test cases. Figure 7.9 shows how the measured data, i.e., the acceleration in  $Ox_s y_s z_s$  and the arm angular velocity, can look like when the robot is in the configuration according to Figure 7.3. Note that it is only the sequence where the angular velocity is constant, in this case around 3 rad/s, that is used. Figure 7.3 shows that the acceleration in the z-direction only originate from the gravity which is verified by Figure 7.9a. It can also be seen that the acceleration due to the circular motion should be in the negative x-direction and in the positive y-direction which is the case in Figure 7.9a. Hence, the transformation from  $Ox_a y_a z_a$  to  $Ox_s y_s z_s$ , given by the identified parameters in (7.1), is correct.

The estimated position  $\hat{l}$  for the five test cases can be seen in Table 7.4. Note that  $\hat{l}_2$  for test 5 is negative which comes from the fact that the sensor is placed on the other side of the weight than was used in the derivation in Section 7.3. The table also shows the error  $\Delta$  between  $\hat{l}$  and the measured position  $l^M$ . The position was always measured using a tape measure to the centre of the accelerometer, since the position of the origin of the accelerometer's coordinate system inside the sensor is unspecified.

If the measured  $\dot{\theta}$  is assumed to be without noise, which is a reasonable assumption for the robot system used during the evaluation, then  $\hat{l}$  is linear dependent of the noise, originating from the measured acceleration, according to (7.37). The

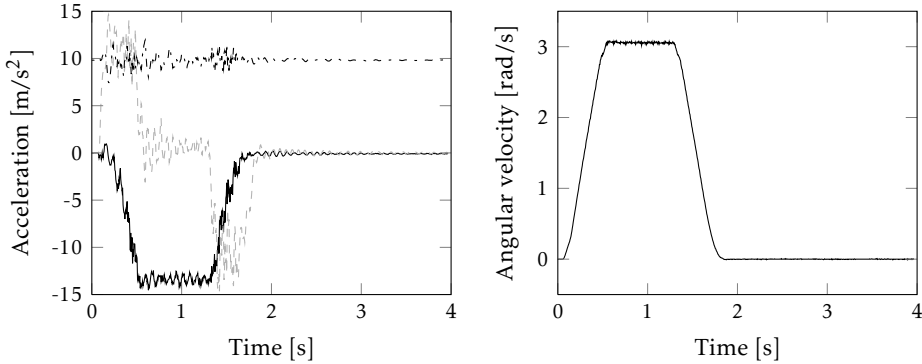
**Table 7.3:** Mean and variance of the estimated transformation parameters  $\mathcal{R}_{a/s}$ ,  $\kappa$  and  $\rho_0$  as well as the mean for the rotational difference  $\vartheta$  between the measured and estimated rotation matrix  $\mathcal{R}_{a/s}$ , for  $N_{MC} = 1\,000$  MC simulations.

Test	1	2	3	4	5
$r_{11}$	-0.0138	0.9999	-0.6348	0.0169	-0.6314
$r_{12}$	-0.9998	-0.0070	-0.7724	-0.0139	0.7751
$r_{13}$	-0.0170	-0.0131	-0.0208	0.9998	0.0209
$r_{21}$	-0.0094	0.0129	-0.0028	-0.9992	-0.0269
$r_{22}$	-0.0169	-0.0276	-0.0247	-0.0355	0.0050
$r_{23}$	0.9998	0.9995	0.9997	0.0164	-0.9996
$r_{31}$	-0.9999	-0.0073	-0.7727	0.0353	-0.7750
$r_{32}$	0.0140	-0.9996	0.6347	-0.9993	-0.6318
$r_{33}$	-0.0092	-0.0275	0.0135	-0.0145	0.0177
$\kappa$	9.9116	9.9121	9.9146	9.9124	9.9153
$\rho_{0,1}$	25.0472	-23.8879	34.7958	-24.4614	-3.9071
$\rho_{0,2}$	-23.7469	-24.0299	-23.7326	24.8569	24.9526
$\rho_{0,3}$	24.2601	25.1132	3.0750	23.7401	33.8097
$\vartheta$	1.3641	1.7924	5.7692	2.3844	6.0309

(a) Mean.

Test	1	2	3	4	5
$r_{11}$	4.55 e-11	6.45 e-15	2.49 e-11	7.61 e-11	2.22 e-11
$r_{12}$	2.13 e-14	2.05 e-11	1.69 e-11	3.51 e-11	1.48 e-11
$r_{13}$	4.34 e-11	3.16 e-11	1.00 e-10	2.86 e-14	3.44 e-11
$r_{21}$	2.45 e-11	3.16 e-11	4.20 e-11	5.13 e-14	2.43 e-11
$r_{22}$	4.33 e-11	1.73 e-11	1.05 e-10	2.42 e-11	3.54 e-11
$r_{23}$	1.49 e-14	1.77 e-14	6.70 e-14	7.60 e-11	1.95 e-14
$r_{31}$	1.11 e-14	2.05 e-11	1.68 e-11	2.42 e-11	1.48 e-11
$r_{32}$	4.55 e-11	1.41 e-14	2.50 e-11	3.79 e-14	2.23 e-11
$r_{33}$	2.45 e-11	1.73 e-11	4.68 e-11	3.52 e-11	2.55 e-11
$\kappa$	3.28 e-9	2.77 e-9	2.92 e-9	2.22 e-9	2.57 e-9
$\rho_{0,1}$	8.15 e-8	4.60 e-8	8.65 e-8	7.82 e-8	6.24 e-8
$\rho_{0,2}$	6.33 e-8	4.42 e-8	1.43 e-7	7.24 e-8	4.91 e-8
$\rho_{0,3}$	6.77 e-8	3.86 e-8	7.99 e-8	4.98 e-8	5.09 e-8
$\vartheta$	1.39 e-7	6.27 e-8	1.49 e-7	1.12 e-7	1.19 e-7

(b) Variance.



(a) Measured acceleration in  $x_s$  (solid),  $y_s$  (dashed) and  $z_s$  (dash-dot). (b) Measured arm angular velocity.

**Figure 7.9:** Measured data, to be used to estimate the position  $l$ , for test 1 when the robot is in the configuration according to Figure 7.3.

covariance matrix of  $\hat{l}$  can therefore be calculated as

$$\begin{aligned} \text{Cov}(\hat{l}) &= (A^T A)^{-1} A^T \text{Cov}(b) \left( (A^T A)^{-1} A^T \right)^T \\ &= (A^T A)^{-1} A^T A (A^T A)^{-T} \text{Cov}(b) = (A^T A)^{-T} \text{Cov}(b), \end{aligned} \quad (7.44)$$

where  $^{-T}$  means both transpose and inverse. In (7.44) it has been used that  $\text{Cov}(b)$  is a scalar. The structure of  $A$  in (7.35) results in a diagonal matrix  $(A^T A)^{-T}$ , hence  $\text{Cov}(\hat{l})$  is a diagonal matrix. The standard deviation of the estimated position is now given by the square root of the diagonal elements in  $\text{Cov}(\hat{l})$ , which can be found in Table 7.4. Considering the accuracy of the measurements and the uncertainty of the origin of the accelerometer coordinate system the result in Table 7.4 is considered as acceptable. The actual requirement of the result, in terms of position and orientation accuracy, will depend on the application where the accelerometer is used. The accuracy is sufficient in experiments with dynamic position and orientation estimation of the tool position using sensor fusion methods, such as EKF and PF, see Chapter 8.

## 7.5 Conclusions

A method to find the position and orientation of a triaxial accelerometer mounted on a six DOF robot is presented. The method is divided into two main steps, where in the first step, the orientation is estimated by finding the transformation from the actual coordinate system of the accelerometer, with unknown orientation, to a new coordinate system with known orientation. It is also possible to find the sensitivity and the bias parameters. The estimation of the orientation is based on



**Table 7.4:** Estimated positions  $\hat{l}$  of the accelerometer in the coordinate system  $Ox_w y_w z_w$  for five different mounting positions, the error  $\Delta$  relative the measured position  $l^M$ , and the standard deviation for  $\hat{l}$ .

Test	Est. pos. ( $\hat{l}$ ) [cm]	$\Delta = \hat{l} - l^M$ [cm]	Std. for $\hat{l}$ [cm]
1	$(35.2 \ 6.3 \ 15.5)^T$	$(0.2 \ 2.3 \ -1.0)^T$	$(0.4 \ 0.5 \ 0.5)^T$
2	$(14.2 \ 5.8 \ 16.9)^T$	$(-0.3 \ -1.2 \ 1.8)^T$	$(0.3 \ 0.3 \ 0.3)^T$
3	$(36.3 \ 6.3 \ 21.4)^T$	$(-1.7 \ 2.3 \ -1.6)^T$	$(0.5 \ 0.6 \ 0.6)^T$
4	$(29.2 \ 1.6 \ 5.9)^T$	$(2.2 \ 1.6 \ 0.4)^T$	$(0.4 \ 0.4 \ 0.4)^T$
5	$(34.8 \ -3.9 \ 16.5)^T$	$(-0.7 \ 0.1 \ 1.0)^T$	$(0.5 \ 0.6 \ 0.6)^T$

static measurements of the gravity vector when the accelerometer is placed in different orientations using the six DOF robot arm. In the second step of the method, the mounting position of the accelerometer in a robot fixed coordinate system is computed using several experiments where the robot is moving with constant speed. Finally, the method is evaluated on experimental data. The resulting position and orientation accuracy are evaluated using measurements on the physical system. The orientation error is in the range one to six degrees and the position error up to two centimeters. The accuracy is sufficient in experiments with dynamic position and orientation estimation of the tool position using sensor fusion methods, see Chapter 8.



# 8

---

## Experimental Evaluations

The experimental evaluation presented in this chapter is performed on the ABB IRB4600 industrial robot in Figure 2.1a, and has previously been published in Axelsson et al. (2011d,c) and Axelsson (2011c).

Six different observers are proposed in Section 8.1 with different estimation models and different filters. The system setup and data acquisition are described in Section 8.2. The performance of the six observers are compared in Section 8.3 and Section 8.4 concludes the experimental results.

### 8.1 Observers

Six observers using the four different estimation models described in Section 4.4 are evaluated. The observers are based on the EKF, EKS, or PF from Chapter 3 or a linear dynamic observer using pole placement (Franklin et al., 2002).

**OBS1:** EKF with the nonlinear model in Section 4.4.1.

**OBS2:** EKS with the nonlinear model in Section 4.4.1.

**OBS3:** EKF with the linear state model and nonlinear measurement model in Section 4.4.2.

**OBS4:** PF with the linear state model and nonlinear measurement model in Section 4.4.2.

**OBS5:** EKF with the nonlinear model where the acceleration of the end-effector is input, see Section 4.4.4 .

**OBS6:** Linear dynamic observer using pole placement with the linear model where the acceleration of the end-effector is input, see Section 4.4.3. (De Luca et al., 2007)

## 8.2 Experimental Setup

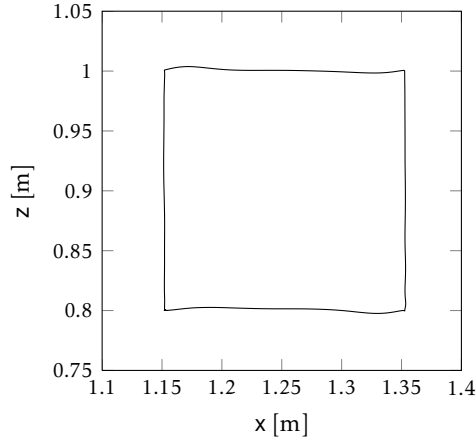
The accelerometer used in the experiments is a triaxial accelerometer from Crossbow Technology, with a range of  $\pm 2g$ , and a sensitivity of 1 V/g (Crossbow Technology, 2004). The orientation and position of the accelerometer were estimated using the method described in Chapter 7. All measured signals, i.e., acceleration, motor angles and arm angular acceleration references, are synchronous and sampled with a rate of 2 kHz. The accelerometer measurements are filtered with an LP-filter before any estimation method is applied to better reflect the tool movement. The path used in the evaluation is illustrated in Figure 8.1 and it is programmed such that only joint two and three of the six DOF robot are moved. Moreover, the wrist is configured such that the couplings to joint two and three are minimised. The dynamic model parameters are obtained using a grey-box identification method described in Wernholt and Moberg (2011). Note that the robot stands on a flexible foundation, which has been taken care of in the model parameters describing the spring and damper pair for joint two.

It is not possible to get measurements of the true state variables, as is the case for the simulation, instead, only the true trajectory of the end-effector, more precise the TCP,  $x$  and  $z$ , is used for evaluation. The true trajectory is measured using a laser tracking system from Leica Geosystems. The tracking system has an accuracy of 0.01 mm per meter and a sample rate of 1 kHz (Leica Geosystems, 2008). However, the measured tool position is not synchronised with the other measured signals. Resampling of the measured signal and a manual synchronisation is therefore needed, which can introduce small errors. Another source of error is the accuracy of the programmed TCP in the control system of the robot. It means that if the programmed TCP differ from the one assumed in (4.1), then the measured position will differ from the estimated position even if the estimated arm angular positions are correct. The estimated data are therefore aligned with the measured position to avoid any static errors. The alignment is performed using a least square fit between the estimated position and the measured position.

## 8.3 Results

The only measured quantity, to compare the estimates with, is the measured tool position, as was mentioned in Section 8.2. Therefore, the estimated arm angles are used to compute an estimate of the TCP using the kinematic relation, i.e.,

$$\begin{pmatrix} \hat{x}_k \\ \hat{z}_k \end{pmatrix} = \Upsilon(\hat{q}_{a,k}), \quad (8.1)$$



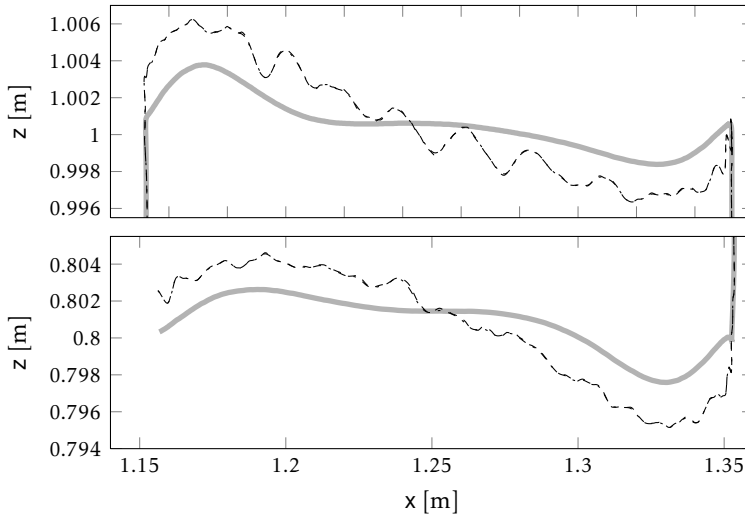
**Figure 8.1:** Measured path for the end-effector used for experimental evaluations.

where  $\hat{q}_{a,k}$  is the result from one of the six observers in Section 8.1 at time  $k$ . The result is presented with diagrams where the estimated paths and the true path in Figure 8.1 are compared. To better see the result, each side of the square is shown separately, where the size of the magnified axes are the same, it is therefore possible to compare the result between different sides without having trouble with the size of the axes. The path error in (5.1) is also used as well as the RMSE values calculated according to (5.8) in the evaluation. Moreover, the first 250 samples are always removed because of transients. The execution time for the observers is also examined. Note that the execution times are with respect to the current MATLAB implementation. The execution time may be faster after some optimisation of the MATLAB code or by using another programming language, e.g. C++. The observers are first paired such that the same estimation model is used, hence OBS1–OBS2, OBS3–OBS4, and OBS5–OBS6 are compared. After that, the best observers from each pair are compared to each other.

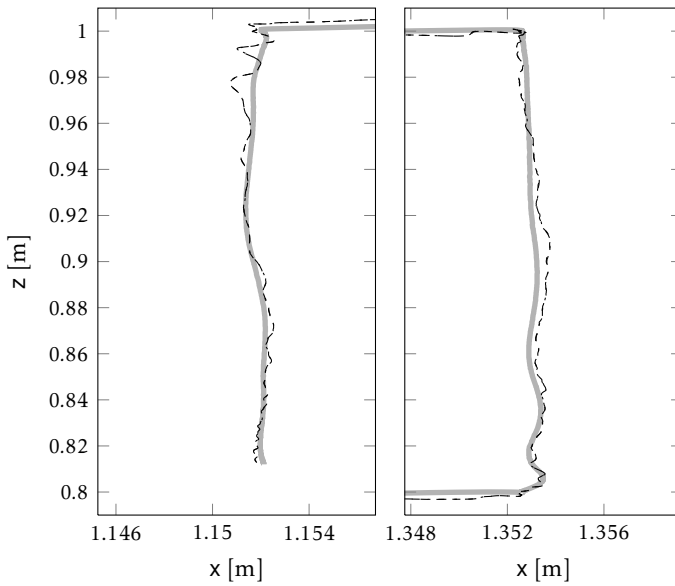
**OBS1 and OBS2.** It is expected that OBS2 (EKS with nonlinear model) will give a better result than OBS1 (EKF with nonlinear model) since the EKS uses both previous and future measurements. This is not the case as can be seen in Figures 8.2, and 8.3. The reason for this can be the nonlinearities.

One interesting observation is the higher orders of oscillations in the estimated paths which can be seen in Figure 8.2a. The oscillations can be reduced if the covariance matrix  $Q$  for the process noise is decreased. However, this leads to a larger path error. The RMSE values can be found in Table 8.1. The table shows that OBS2 is slightly better than OBS1.

If the bias components are removed the filter performance is unchanged. In Chapter 5 it was showed that the bias components improved the estimated position, at least when no model errors are present. The model errors are therefore a bigger

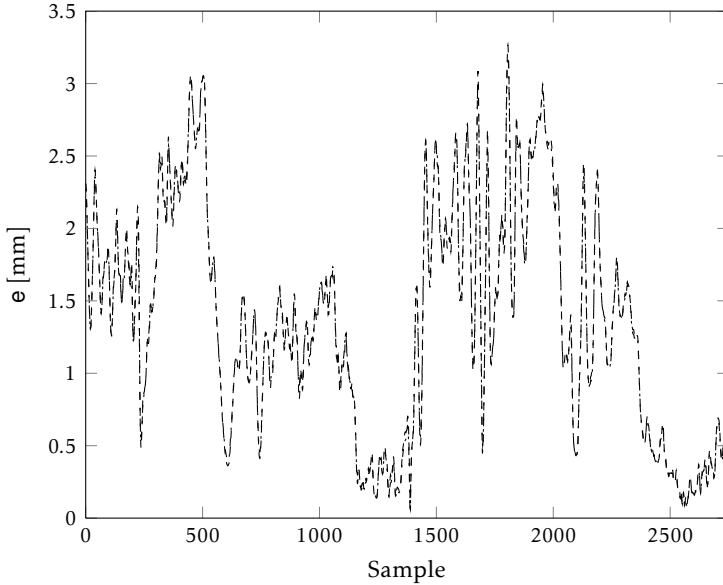


(a) Bottom and top.



(b) Left and right.

**Figure 8.2:** The four sides of the true path (solid), and the estimated path using OBS1 (dashed), and OBS2 (dash-dot).



**Figure 8.3:** The path error for the estimated path using OBS1 (dashed), and OBS2 (dash-dot).

problem than the bias in the acceleration measurements. One reason that there is no difference between the EKF with or without bias compensation can be that the trajectory is performed under a short time, hence a small bias error has no time to affect the result that much.

With the current MATLAB implementation the execution times are around five and seven seconds, respectively, and the total length of the measured path is four seconds, hence none of the observers are real-time. Most of the time is spent in evaluating the Jacobian  $H_k$  in Algorithm 1 and it is probably possible to decrease that time with a more efficient implementation. Another possibility can be to run the filter with a lower sample rate. However, in Section 5.3 it is showed that a too low sample rate can cause instability in the time update. There is a possibility to save time if the time update is used with the original sample rate and the measurement update with a lower sample rate, since most of the time is spent in evaluating the Jacobian  $H_k$ . OBS2 is slower since an EKF is used first and then the backward time recursion, see Algorithm 3. However, most of the time in the EKS is spent in the EKF. As a matter of fact, the execution time is irrelevant for OBS2 since the EKS uses future measurements and has to be implemented offline.

None of the two observers can be said to be better than the other in terms of estimation performance and execution time. The decision is whether future measurement can be used or not. OBS1 is chosen as the one that will be compared with the other observers in Section 8.4.

**Table 8.1:** RMSE values of the path error  $e$  for the end-effector position given in mm for the six observers.

	OBS1	OBS2	OBS3	OBS4	OBS5	OBS6
$\epsilon$	1.5704	1.5664	2.3752	1.5606	1.6973	1.7624

**OBS3 and OBS4.** Figure 8.4 shows that the estimated paths follow the true path for both observers. It can be noticed that the estimate for OBS3 (EKF with linear dynamic model) goes somewhat past the corners before it changes direction and that OBS4 (PF with linear dynamic model) performs better in the corners. The estimate for OBS4 is also closer to the true path, at least at the vertical sides. The path error is presented in Figure 8.5, where the error for OBS4 is below the error for OBS3 most of the time. The RMSE values of the path error for OBS3 and OBS4 are presented in Table 8.1. The RMSE for OBS4 is approximately two-thirds of the RMSE for OBS3.

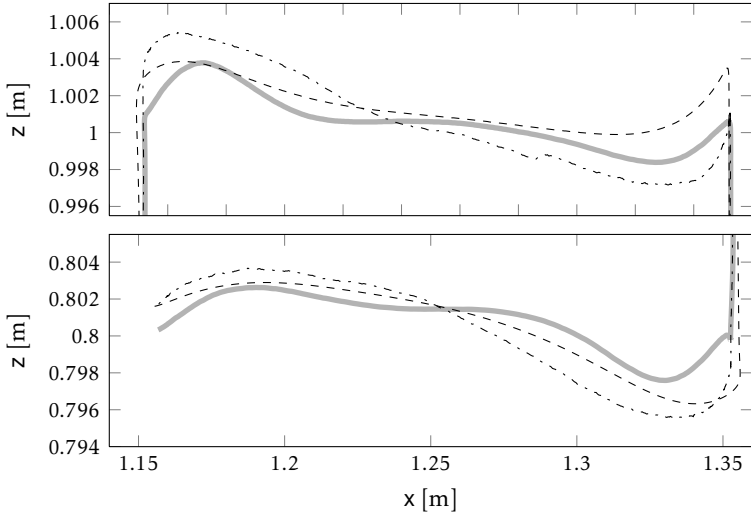
OBS4 has also been executed without the bias states included in the measurement equation for the motor positions, see Section 4.4.2. The result is that the path error for OBS4 will be increased without the bias states. The explanation for this is the model errors in the dynamic model. The MATLAB implementation of OBS3 is almost real-time, just above four seconds, and the execution time for OBS4 is in the order of hours. The execution time for OBS3 can be reduced to real-time without losing performance if the measurements are decimated to approximately 200 Hz. Note that the problem with instability during the time update is of no importance since OBS3 uses the linear dynamic model and not the nonlinear model.

The best observer in terms of the path error is obviously OBS4 but if the execution time is of importance, OBS3 is preferable. OBS4 will be used in Section 8.4 to compare with the other observers since the path error is of more interest in this thesis.

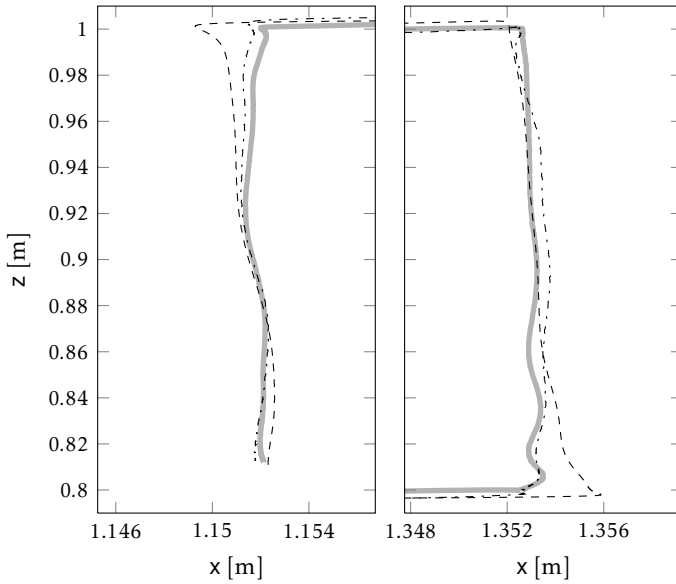
**OBS5 and OBS6.** OBS6 (linear model with acceleration as input using pole placement) performs surprisingly good although a linear time invariant model is used, see Figure 8.6. It can also be seen that OBS5 (linear model with acceleration as input using EKF) performs a bit better, at least in the vertical sides. OBS5 also has a higher order oscillation as was the case with OBS1 and OBS2. This is a matter of tuning where less oscillations induce higher path error. Figure 8.7 shows the path error, where it can be seen that OBS5 has oscillations with higher frequencies than OBS6 but OBS6 has oscillations with lower frequencies where the amplitude is much higher than for OBS5. The RMSE values of the path error are showed in Table 8.1.

No bias compensation has been performed in these two observers. Since the bias compensation does not improve the results for OBS1, it can be assumed that it will not improve the result for OBS5 and OBS6 either. Both observers execute in real-time. The execution times are just below one second and around one-fifth of a second, respectively. OBS6 is clearly the fastest one of the six proposed observers. OBS5 is the one that will be compared to the other observers in Section 8.4



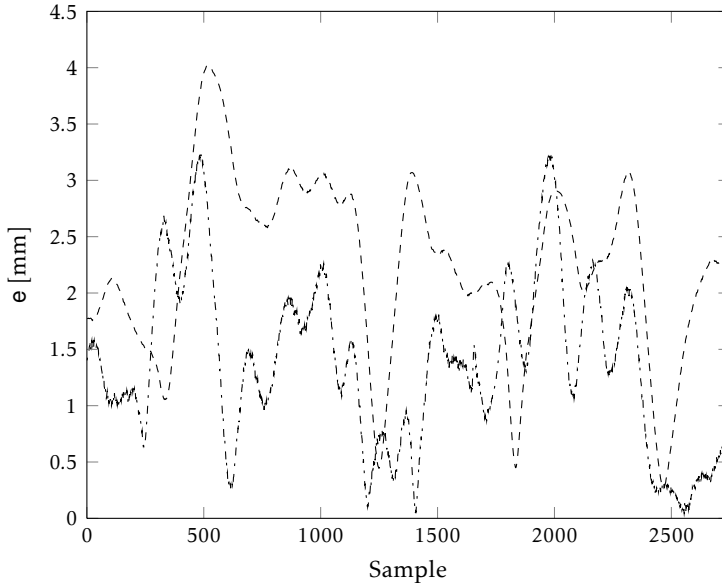


(a) Bottom and top.



(b) Left and right.

**Figure 8.4:** The four sides of the true path (solid), and the estimated path using OBS3 (dashed), and OBS4 (dash-dot).



*Figure 8.5: The path error for the estimated path using OBS3 (dashed), and OBS4 (dash-dot).*

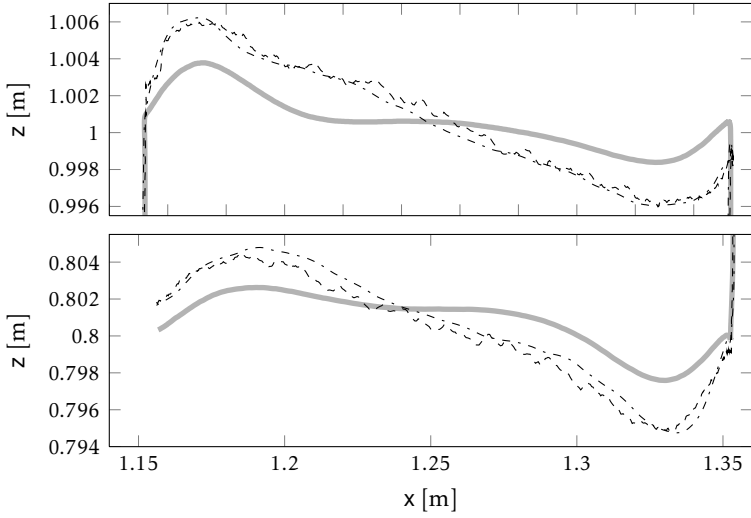
## 8.4 Conclusions

The three observers OBS1, OBS4, and OBS5 are the best ones from each pair, according to Section 8.3. From Table 8.1 it can be seen that OBS1 and OBS4 have the same performance and that OBS5 is a bit worse. The differences are small so it is difficult to say which one that is the best. Instead of filter performance, other things have to be considered, such as complexity, computation time, and robustness.

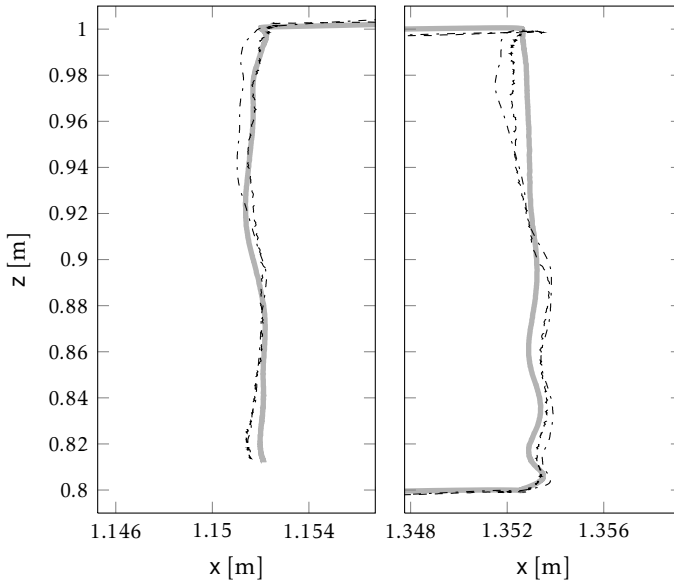
**Complexity.** The complexity of the filters can be divided into model complexity and implementation complexity. The implementation of OBS1 is straightforward and no particular tuning has to be performed in order to get an estimate. The tuning is of course important to get a good estimate. Instead, most of the time has to be spent on a rigorous modelling work and identification of the parameters to minimise model errors.

For OBS4 the opposite is true. The model is simple and requires not that much work. Most of the time has to be spent on implementing the PF. The standard choices of a proposal distribution did not work due to high SNR and non-invertible measurement model. Instead, an approximation of the optimal proposal, using an EKF, was required. The consequence is more equations to implement and more tuning knobs to adjust.

The model complexity for OBS5 is in between OBS1 and OBS2. No model for the

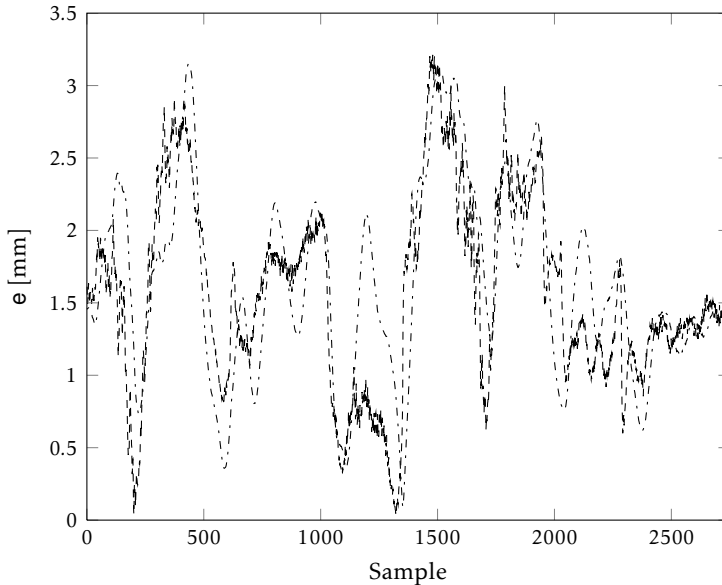


(a) Bottom and top.



(b) Left and right.

**Figure 8.6:** The four sides of the true path (solid), and the estimated path using OBS5 (dashed), and OBS6 (dash-dot).



**Figure 8.7:** The path error for the estimated path using OBS5 (dashed), and OBS6 (dash-dot).

rigid body motion on the arm side is needed which is a difference from the other two. However, a nonlinear model for the flexibilities and friction is still required, which is not the case for OBS4.

**Computation time.** The computation time differs a lot for the three observers. OBS5 is in real-time with the current MATLAB implementation and OBS1 can probably be executed in real-time after some optimisation of the MATLAB code or with another programming language. The computation time for OBS4 is in the order of hours and is therefore far from real-time.

**Robustness.** For this path, the effects of the acceleration bias seem to be minor. If instead the bias is bigger or the path is longer, then the bias effects can be more severe. OBS5 can then have some problems since no compensation for the bias is included in the observer. An advantage with OBS5, compared to the other two, is that the equations describing the arm dynamics are removed, hence no robustness issues concerning the model parameters describing the arm, such as inertia, masses, centre of gravity, etcetera. However, the model parameters describing the flexibilities remain.

**Other advantages.** An advantage with OBS4 is that the PF provides the entire distribution of the states, which is approximated as a Gaussian distribution in the EKF. The information about the distribution can be used in e.g. control and diagnosis.

# 9

---

## Conclusions and Future Work

This chapter concludes the work in this thesis and discusses possible directions for future work.

### 9.1 Conclusions

A sensor fusion approach to estimate the end-effector position and velocity by combining a triaxial accelerometer at the end-effector and the motor angular positions of an industrial robot is presented. The estimation is formulated as a Bayesian problem where the *extended Kalman filter* (EKF) has been used frequently. The estimation problem has been evaluated on simulated data from a realistic robot model, and it is also evaluated with experimental data from a state of the art industrial robot.

The simulation study was performed with the EKF and it showed that the estimation accuracy degrades with parameter errors in the dynamical model. The only solution is to develop a better dynamical model, which could be considered as future work. The simulation study also showed that uncertainties in the parameters describing the accelerometer model affected the performance. A two-step method to find the position and orientation of a triaxial accelerometer mounted on a six *degrees of freedom* (DOF) robot was therefore presented. In the first step, the orientation was estimated using static measurements of the gravity vector. In the second step, the mounting position was computed as the solution to a linear system of equations, from measurements where the robot moves with constant angular speed. The method was evaluated on experimental data, where the resulting orientation error was in the range of one to six degrees and the position error less than two centimeters.

The simulation study also showed that the performance of the EKF changed very much with different settings of the filter, i.e., with different covariance matrices for the process and measurement noise. The proposed method for tuning the EKF worked but different results were obtained for different initial values. A drawback is that the true position of the end-effector is needed. Another method was therefore proposed where the EM algorithm was used to estimate the process noise covariance matrix. The EM method gave a lower path error than the first method and there were no indications of different solutions when the initial values of  $Q$  changed. A great advantage with the EM method is that the true position of the end-effector is not needed.

Moreover, experimental data have been used to evaluate the estimation performance. The EKF from the simulation study was compared to other types of observers where both the estimation model and the filter were changed. The three observers with the best performance were

- a) an EKF using a nonlinear dynamic model,
- b) a particle filter using a linear dynamic model, and
- c) an EKF with a nonlinear model, where the acceleration of the end-effector is used as an input instead of a measurement.

The performance of these three observers was very similar when considering the path error. The execution time for a) was just above the real-time limit, for c) just below the limit, and for b) in the order of hours. The time required for modelling and implementation is also different for the three different observers. For b), most of the time was spent to implement the filter and get it to work, whereas most of the time for a) was spent on modelling the dynamics.

Although most of the observers in this thesis are not running in real-time it is possible to use the estimates in offline methods such as *iterative learning control* (ILC), system identification, and diagnosis. However, the computation time can be decreased by optimising the MATLAB code or by using another programming language, e.g. C++. The estimation methods in this thesis are general and can be extended to higher degrees of freedom robots and additional sensors, such as gyros and camera systems. The main effect is more complex state space descriptions and longer computation time.

## 9.2 Future Work

In this thesis several problems have come up that need to be investigated more thoroughly. One such problem is discretisation, using Euler forward, of the continuous state space model derived from Lagrange's equation. The estimation errors decrease when the sample rate is increased. However, higher sample rate gives longer computation time. Other discretisation methods than Euler forward can be unpractical when analytical integration is not possible. A possibility can be to perform the time update in the EKF in continuous time using an ODE-solver,

instead of discretisation of the model.

Another problem that has been encountered is model errors in the dynamical parameters. A sensitivity analysis is needed to be able to find out to which parameters the observers are sensitive. It is interesting to see if the parameters that are crucial for the performance can be adapted at the same time as the states are estimated. One way can be to use the same approach as for bias in the measured signals. That is, introduce bias states for the parameters and estimate them simultaneously as the states.

For the linear dynamic model, the estimated path goes somewhat past the corners. It is interesting to see what happens if the noise covariance matrices are time dependent, e.g. the covariance matrix for the process noise increases when the path changes drastically. This can be done in several ways, e.g. find out when the path changes from the measured data or from the programmed path.

A natural continuation is to extend the estimation problem to cover the complete six DOF robot. The sensor system could be extended and a first step would be to include a gyro to get measurements of the rotation of the end-effector and not only the translation. Another measurement to consider is the arm angular position, i.e., a measurement on the arm side of the gearbox. Including more DOFs and measurements will increase the computational complexity of the observers even more.

Finally, the control performance has to be investigated when the estimated states are available. However, a closed loop real-time system can be difficult to achieve due to the computational complexity for most of the presented methods. Instead, an offline method like ILC has to be considered.





---

## Bibliography

- ABB Robotics. Company homepage. URL: <http://www.abb.com>, accessed September 2011. Cited on pages 4, 12, and 13.
- Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Information and System Sciences Series. Prentice Hall Inc., Englewood Cliffs, New Jersey, USA, 1979. Cited on page 22.
- Tohid Ardehshiri, Mikael Norrlöf, Johan Löfberg, and Anders Hansson. Convex optimization approach for time-optimal path tracking of robots with speed dependent constraints. In *Proceedings of the 18th IFAC World Congress*, pages 14648–14653, Milano, Italy, August 2011. Cited on page 13.
- M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002. Cited on page 23.
- Karl-Johan Åström and Carlos Canudas de Wit. Revisiting the LuGre friction model. *IEEE Control Systems Magazine*, 28(6):101–114, December 2008. Cited on page 18.
- Patrik Axelsson. A simulation study on the arm estimation of a joint flexible 2 DOF robot arm. Technical Report LiTH-ISY-R-2926, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, December 2009. Not cited.
- Patrik Axelsson. Simulation model of a 2 degrees of freedom industrial manipulator. Technical Report LiTH-ISY-R-3020, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, June 2011a. Not cited.
- Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. *Submitted to the IFAC Symposium on Robot Control 2012*, 2011b. Not cited.
- Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. Technical Report LiTH-ISY-R-3035, De-

- partment of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, November 2011c. Cited on pages 8 and 93.
- Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. *Submitted to the IEEE Conference on Robotics and Automation 2012*, 2011a. Not cited.
- Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. Technical Report LiTH-ISY-R-3025, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, September 2011b. Cited on pages 8 and 73.
- Patrik Axelsson, Mikael Norrlöf, Erik Wernholt, and Fredrik Gustafsson. Extended Kalman filter applied to industrial manipulators. In *Proceedings of Reglermötet 2010*, Lund, Sweden, June 2010. Not cited.
- Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Tool position estimation of a flexible industrial robot using recursive Bayesian methods. *Submitted to the IEEE Conference on Robotics and Automation 2012*, 2011a. Not cited.
- Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Submitted to Control Engineering Practice*, 2011b. Not cited.
- Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Tool position estimation of a flexible industrial robot using recursive Bayesian methods. Technical Report LiTH-ISY-R-3024, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, September 2011c. Cited on pages 8 and 93.
- Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. Technical Report LiTH-ISY-R-3027, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, October 2011d. Cited on pages 8 and 93.
- Patrik Axelsson, Umut Orguner, Fredrik Gustafsson, and Mikael Norrlöf. ML estimation of process noise variance in dynamic systems. In *Proceedings of the 18th IFAC World Congress*, pages 5609–5614, Milano, Italy, August 2011e. Cited on page 61.
- Niclas Bergman. *Recursive Bayesian Estimation: Navigation and Tracking Applications*. Linköping Studies in Science and Technology. Dissertations. No. 579, Linköping University, Linköping, Sweden, SE-581 83 Linköping, Sweden, May 1999. Cited on page 59.
- Mattias Björkman, Torgny Brogårdh, Sven Hanssen, Sven-Erik Lindström, Stig Moberg, and Mikael Norrlöf. A new concept for motion control of industrial robots. In *Proceedings of the 17th IFAC World Congress*, pages 15714–15715, Seoul, Korea, July 2008. Cited on page 4.

- Torgny Brogårdh. Present and future robot control development—an industrial perspective. *Annual Reviews in Control*, 31(1):69–79, 2007. Cited on page 3.
- Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer, New York, USA, 2005. Cited on page 61.
- André Carvalho Bittencourt, Erik Wernholt, Shiva Sander-Tavallaey, and Torgny Brogårdh. An extended friction model to capture load and temperature effects in robot joints. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6161–6167, Taipei, Taiwan, October 2010. Cited on page 18.
- André Carvalho Bittencourt, Patrik Axelsson, Ylva Jung, and Torgny Brogårdh. Modeling and identification of wear in a robot joint under temperature uncertainties. In *Proceedings of the 18th IFAC World Congress*, pages 10293–10299, Milano, Italy, August 2011. Not cited.
- Peter I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, March 1996. Cited on page 86.
- John J. Craig. *Introduction to Robotics Mechanics and Control*. Addison Wesley, Menlo Park, California, USA, second edition, 1989. Cited on page 14.
- Crossbow Technology. Accelerometers, High Sensitivity, LF Series, CXL02LF3, January 2004. Available at <http://www.xbow.com>. Cited on pages 6, 7, 84, and 94.
- Alessandro De Luca and Wayne Book. *Springer Handbook of Robotics*, chapter Robots with Flexible Elements, pages 287–319. Springer-Verlag, Berlin Heidelberg, 2008. Cited on page 19.
- Alessandro De Luca, Dierk Schröder, and Michael Thümmel. An acceleration-based state observer for robot manipulators with elastic joints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3817–3823, Roma, Italy, April 2007. Cited on pages 6, 38, and 94.
- Arthur Dempster, Nan Laird, and Donald Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977. Cited on pages 26 and 27.
- Jaques Denavit and Richard S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Journal of Applied Mechanics*, 22:215–221, 1955. Cited on page 15.
- Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000. Cited on page 24.
- Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, USA, 2001. Cited on pages 23 and 24.

- Pierre Dupont, Vincent Hayward, Brian Armstrong, and Friedhelm Altpeter. Single state elastoplastic friction models. *IEEE Transactions on Automatic Control*, 47(5):787–792, May 2002. Cited on page 18.
- FANUC Robotics. Company homepage. URL: <http://www.fanucrobotics.com>, accessed October 2011. Cited on page 12.
- B. Feeny and F. C. Moon. Chaos in a forced dry-friction oscillator: Experiments and numerical modelling. *Journal of Sound and Vibration*, 170(3):303–323, 1994. Cited on page 18.
- Ronald A. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Mathematics*, 41:155–160, 1912. Cited on page 25.
- Ronald A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society, Series A*, 222:309–368, 1922. Cited on page 25.
- Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall Inc., Upper Saddle River, New Jersey, USA, fourth edition, 2002. Cited on page 93.
- Herbert Goldstein, Charles Poole, and John Safko. *Classical Mechanics*. Addison Wesley, San Francisco, USA, third edition, 2002. Cited on page 17.
- Neil J. Gordon, David J. Salmond, and Adrian F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, 140(2):107–113, April 1993. Cited on pages 23 and 24.
- Fredrik Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur, Lund, Sweden, first edition, 2010. Cited on pages 23, 24, and 25.
- Robert Henriksson, Mikael Norröf, Stig Moberg, Erik Wernholt, and Thomas B. Schön. Experimental comparison of observers for tool position estimation of industrial robots. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 8065–8070, Shanghai, China, December 2009. Cited on pages 6 and 47.
- Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, April 1987. Cited on page 76.
- Berthold K. P. Horn, Hugh M. Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America*, 5(7):1127–1135, July 1988. Cited on page 75.
- Mrdjan Jankovic. Observer based control for elastic joint robots. *IEEE Transactions on Robotics and Automation*, 11(4):618–623, August 1995. Cited on page 6.
- Rahim Jassemi-Zargani and Dan Neacsulescu. Extended Kalman filter-based sensor fusion for operational space control of a robot arm. *IEEE Transactions on*

- Instrumentation and Measurement*, 51(6):1279–1282, December 2002. Cited on page 6.
- Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, USA, 1970. Cited on page 22.
- Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004. Cited on page 28.
- Simon J. Julier, Jeffrey K. Uhlmann, and Hugh F. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the American Control Conference*, volume 3, pages 1628–1632, Seattle, Washington, USA, June 1995. Cited on page 22.
- Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, New Jersey, USA, 2000. Cited on page 22.
- Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the AMSE—Journal of Basic Engineering*, 82(Series D):35–45, 1960. Cited on page 22.
- Rickard Karlsson and Mikael Norrlöf. Bayesian position estimation of an industrial robot using multiple sensors. In *Proceedings of the IEEE Conference on Control Applications*, pages 303–308, Taipei, Taiwan, September 2004. Cited on page 6.
- Rickard Karlsson and Mikael Norrlöf. Position estimation and modeling of a flexible industrial robot. In *Proceedings of the 16th IFAC World Congress*, Prague, Czech Republic, July 2005. Cited on pages 6 and 37.
- Krzysztof Kozłowski. *Modelling and Identification in Robotics*. Advances in Industrial Control. Springer, London, Great Britain, 1998. Cited on pages 14 and 18.
- KUKA. Company homepage. URL: <http://www.kuka-ag.de/en/>, accessed October 2011. Cited on page 12.
- Leica Geosystems. Case study ABB robotics - Västerås, 2008. Available at <http://metrology.leica-geosystems.com/en/index.htm>. Cited on page 94.
- Leica Geosystems. Company homepage. URL: <http://metrology.leica-geosystems.com/en/index.htm>, accessed November 2011. Cited on pages 6 and 7.
- Vatchara Lertpiriyasuwat, Martin C. Berg, and Keith W. Buffinton. Extended Kalman filtering applied to a two-axis robotic arm with flexible links. *The International Journal of Robotics Research*, 19(3):254–270, March 2000. Cited on page 6.

- Y. F. Li and X. B. Chen. End-point sensing and state observation of a flexible-link robot. *IEEE/ASME Transactions on Mechatronics*, 6(3):351–356, September 2001. Cited on page 6.
- Helmut Lütkepohl. *Handbook of Matrices*. John Wiley & Sons, Chichester, West Sussex, England, 1996. Cited on page 67.
- Geoffrey J. McLachlan and Thiriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, New Jersey, USA, second edition, 2008. Cited on page 26.
- Sujit Kumar Mitra and C. Radhakrishna Rao. *Generalized Inverse of Matrices and its Applications*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 1971. Cited on pages 17 and 65.
- Stig Moberg and Sven Hanssen. A DAE approach to feedforward control of flexible manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3439–3444, Roma, Italy, April 2007. Cited on page 19.
- Stig Moberg, Jonas Öhr, and Svante Gunnarsson. A benchmark problem for robust control of a multivariable nonlinear flexible manipulator. In *Proceedings of the 17th IFAC World Congress*, pages 1206–1211, Seoul, Korea, July 2008. URL: <http://www.robustcontrol.org>. Cited on pages 31, 33, and 45.
- Motoman. Company homepage. URL: <http://www.motoman.eu/>, accessed October 2011. Cited on page 12.
- Salvatore Nicosia and Patrizio Tomei. State observers for rigid and elastic joint robots. *Robotics and Computer-Integrated Manufacturing*, 9(2):113–120, 1992. Cited on page 6.
- Salvatore Nicosia, Patrizio Tomei, and Antonio Tornambé. A nonlinear observer for elastic robots. *IEEE Journal of Robotics and Automation*, 4(1):45–52, February 1988. Cited on page 6.
- Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, USA, second edition, 2006. Cited on pages 26 and 47.
- Shimon Y. Nof, editor. *Handbook of Industrial Robotics*. John Wiley & Sons, Hoboken, NJ, USA, second edition, 1999. Cited on page 11.
- Erin L. Renk, Walter Collins, Matthew Rizzo, Fujun Lee, and Dennis S. Bernstein. Calibrating a triaxial accelerometer-magnetometer—using robotic actuation for sensor reorientation during data collection. *IEEE Control Systems Magazine*, 25(6):86–95, December 2005. Cited on page 73.
- Gerasimos G. Rigatos. Particle filtering for state estimation in nonlinear industrial systems. *IEEE Transactions on Instrumentation and Measurement*, 58(11):3885–3900, November 2009. Cited on page 6.

- Wilson J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, New Jersey, USA, second edition, 1996. Cited on page 38.
- Thomas B. Schön, Adrian Wills, and Brett Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, January 2011. Cited on page 61.
- Lorenzo Sciavicco and Bruno Siciliano. *Modelling and Control of Robot Manipulators*. Springer, London, UK, second edition, 2000. Cited on pages 14 and 86.
- Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer-Verlag, Berlin Heidelberg, 2008. Cited on page 14.
- Mark W. Spong. Modeling and control of elastic joint robots. *Journal of Dynamic Systems, Measurement, and Control*, 109:310–319, December 1987. Cited on page 19.
- Mark W. Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons, 2005. Cited on pages 14, 17, and 79.
- Patrizio Tomei. An observer for flexible joint robots. *IEEE Transactions on Automatic Control*, 35(6):739–743, June 1990. Cited on page 6.
- Diederik Verscheure, Bram Demeulenaere, Jan Swevers, Joris De Schutter, and Moritz Diehl. Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control*, 54(10):2318–2327, October 2009. Cited on page 13.
- Johanna Wallén, Mikael Norrlöf, and Svante Gunnarsson. Arm-side evaluation of ILC applied to a six-degrees-of-freedom industrial robot. In *Proceedings of the 17th IFAC World Congress*, pages 13450–13455, Seoul, Korea, July 2008. Cited on page 7.
- Johanna Wallén, Svante Gunnarsson, Robert Henriksson, Stig Moberg, and Mikael Norrlöf. ILC applied to a flexible two-link robot model using sensor-fusion-based estimates. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 458–463, Shanghai, China, December 2009. Cited on page 7.
- Erik Wernholt and Stig Moberg. Nonlinear gray-box identification using local models applied to industrial robots. *Automatica*, 47(4):650–660, April 2011. Cited on page 94.
- Lars Westerlund. *The Extended Arm of Man. A History of the Industrial Robot*. Informationsförlaget, Stockholm, Sweden, 2000. Cited on page 11.
- Seong-hoon Peter Won and Farid Golnaraghi. A triaxial accelerometer calibration method using a mathematical model. *IEEE Transactions on Instrumentation and Measurement*, 59(8):2144–2153, August 2010. Cited on page 73.

Byron M. Yu, Krishna V. Shenoy, and Maneesh Sahani. Derivation of extended Kalman filtering and smoothing equations. URL: [http://www-npl.stanford.edu/~byronyu/papers/derive\\_eks.pdf](http://www-npl.stanford.edu/~byronyu/papers/derive_eks.pdf), 19 October 2004. Cited on page 25.



**Licentiate Theses**  
**Division of Automatic Control**  
**Linköping University**

- P. Andersson:** Adaptive Forgetting through Multiple Models and Adaptive Control of Car Dynamics. Thesis No. 15, 1983.
- B. Wahlberg:** On Model Simplification in System Identification. Thesis No. 47, 1985.
- A. Isaksson:** Identification of Time Varying Systems and Applications of System Identification to Signal Processing. Thesis No. 75, 1986.
- G. Malmberg:** A Study of Adaptive Control Missiles. Thesis No. 76, 1986.
- S. Gunnarsson:** On the Mean Square Error of Transfer Function Estimates with Applications to Control. Thesis No. 90, 1986.
- M. Viberg:** On the Adaptive Array Problem. Thesis No. 117, 1987.
- K. Ståhl:** On the Frequency Domain Analysis of Nonlinear Systems. Thesis No. 137, 1988.
- A. Skeppstedt:** Construction of Composite Models from Large Data-Sets. Thesis No. 149, 1988.
- P. A. J. Nagy:** MaMiS: A Programming Environment for Numeric/Symbolic Data Processing. Thesis No. 153, 1988.
- K. Forsman:** Applications of Constructive Algebra to Control Problems. Thesis No. 231, 1990.
- I. Klein:** Planning for a Class of Sequential Control Problems. Thesis No. 234, 1990.
- F. Gustafsson:** Optimal Segmentation of Linear Regression Parameters. Thesis No. 246, 1990.
- H. Hjalmarsson:** On Estimation of Model Quality in System Identification. Thesis No. 251, 1990.
- S. Andersson:** Sensor Array Processing; Application to Mobile Communication Systems and Dimension Reduction. Thesis No. 255, 1990.
- K. Wang Chen:** Observability and Invertibility of Nonlinear Systems: A Differential Algebraic Approach. Thesis No. 282, 1991.
- J. Sjöberg:** Regularization Issues in Neural Network Models of Dynamical Systems. Thesis No. 366, 1993.
- P. Pucar:** Segmentation of Laser Range Radar Images Using Hidden Markov Field Models. Thesis No. 403, 1993.
- H. Fortell:** Volterra and Algebraic Approaches to the Zero Dynamics. Thesis No. 438, 1994.
- T. McKelvey:** On State-Space Models in System Identification. Thesis No. 447, 1994.
- T. Andersson:** Concepts and Algorithms for Non-Linear System Identifiability. Thesis No. 448, 1994.
- P. Lindskog:** Algorithms and Tools for System Identification Using Prior Knowledge. Thesis No. 456, 1994.
- J. Plantin:** Algebraic Methods for Verification and Control of Discrete Event Dynamic Systems. Thesis No. 501, 1995.
- J. Gunnarsson:** On Modeling of Discrete Event Dynamic Systems, Using Symbolic Algebraic Methods. Thesis No. 502, 1995.
- A. Ericsson:** Fast Power Control to Counteract Rayleigh Fading in Cellular Radio Systems. Thesis No. 527, 1995.
- M. Jirstrand:** Algebraic Methods for Modeling and Design in Control. Thesis No. 540, 1996.
- K. Edström:** Simulation of Mode Switching Systems Using Switched Bond Graphs. Thesis No. 586, 1996.

**J. Palmqvist:** On Integrity Monitoring of Integrated Navigation Systems. Thesis No. 600, 1997.

**A. Stenman:** Just-in-Time Models with Applications to Dynamical Systems. Thesis No. 601, 1997.

**M. Andersson:** Experimental Design and Updating of Finite Element Models. Thesis No. 611, 1997.

**U. Forssell:** Properties and Usage of Closed-Loop Identification Methods. Thesis No. 641, 1997.

**M. Larsson:** On Modeling and Diagnosis of Discrete Event Dynamic systems. Thesis No. 648, 1997.

**N. Bergman:** Bayesian Inference in Terrain Navigation. Thesis No. 649, 1997.

**V. Einarsson:** On Verification of Switched Systems Using Abstractions. Thesis No. 705, 1998.

**J. Blom, F. Gunnarsson:** Power Control in Cellular Radio Systems. Thesis No. 706, 1998.

**P. Spångéus:** Hybrid Control using LP and LMI methods – Some Applications. Thesis No. 724, 1998.

**M. Norrlöf:** On Analysis and Implementation of Iterative Learning Control. Thesis No. 727, 1998.

**A. Hagenblad:** Aspects of the Identification of Wiener Models. Thesis No. 793, 1999.

**E. Tjärnström:** Quality Estimation of Approximate Models. Thesis No. 810, 2000.

**C. Carlsson:** Vehicle Size and Orientation Estimation Using Geometric Fitting. Thesis No. 840, 2000.

**J. Löfberg:** Linear Model Predictive Control: Stability and Robustness. Thesis No. 866, 2001.

**O. Härkegård:** Flight Control Design Using Backstepping. Thesis No. 875, 2001.

**J. Elbornsson:** Equalization of Distortion in A/D Converters. Thesis No. 883, 2001.

**J. Roll:** Robust Verification and Identification of Piecewise Affine Systems. Thesis No. 899, 2001.

**I. Lind:** Regressor Selection in System Identification using ANOVA. Thesis No. 921, 2001.

**R. Karlsson:** Simulation Based Methods for Target Tracking. Thesis No. 930, 2002.

**P.-J. Nordlund:** Sequential Monte Carlo Filters and Integrated Navigation. Thesis No. 945, 2002.

**M. Östring:** Identification, Diagnosis, and Control of a Flexible Robot Arm. Thesis No. 948, 2002.

**C. Olsson:** Active Engine Vibration Isolation using Feedback Control. Thesis No. 968, 2002.

**J. Jansson:** Tracking and Decision Making for Automotive Collision Avoidance. Thesis No. 965, 2002.

**N. Persson:** Event Based Sampling with Application to Spectral Estimation. Thesis No. 981, 2002.

**D. Lindgren:** Subspace Selection Techniques for Classification Problems. Thesis No. 995, 2002.

**E. Geijer Lundin:** Uplink Load in CDMA Cellular Systems. Thesis No. 1045, 2003.

**M. Enqvist:** Some Results on Linear Models of Nonlinear Systems. Thesis No. 1046, 2003.

**T. Schön:** On Computational Methods for Nonlinear Estimation. Thesis No. 1047, 2003.

**F. Gunnarsson:** On Modeling and Control of Network Queue Dynamics. Thesis No. 1048, 2003.

**S. Björklund:** A Survey and Comparison of Time-Delay Estimation Methods in Linear Systems. Thesis No. 1061, 2003.

**M. Gerdin:** Parameter Estimation in Linear Descriptor Systems. Thesis No. 1085, 2004.

**A. Eidehall:** An Automotive Lane Guidance System. Thesis No. 1122, 2004.

**E. Wernholt:** On Multivariable and Nonlinear Identification of Industrial Robots. Thesis No. 1131, 2004.

**J. Gillberg:** Methods for Frequency Domain Estimation of Continuous-Time Models. Thesis No. 1133, 2004.

**G. Hendeby:** Fundamental Estimation and Detection Limits in Linear Non-Gaussian Systems. Thesis No. 1199, 2005.

**D. Axehill:** Applications of Integer Quadratic Programming in Control and Communication. Thesis No. 1218, 2005.

**J. Sjöberg:** Some Results On Optimal Control for Nonlinear Descriptor Systems. Thesis No. 1227, 2006.

**D. Törnqvist:** Statistical Fault Detection with Applications to IMU Disturbances. Thesis No. 1258, 2006.

**H. Tedefelt:** Structural algorithms and perturbations in differential-algebraic equations. Thesis No. 1318, 2007.

**S. Moberg:** On Modeling and Control of Flexible Manipulators. Thesis No. 1336, 2007.

**J. Wallén:** On Kinematic Modelling and Iterative Learning Control of Industrial Robots. Thesis No. 1343, 2008.

**J. Harju Johansson:** A Structure Utilizing Inexact Primal-Dual Interior-Point Method for Analysis of Linear Differential Inclusions. Thesis No. 1367, 2008.

**J. D. Hol:** Pose Estimation and Calibration Algorithms for Vision and Inertial Sensors. Thesis No. 1370, 2008.

**H. Ohlsson:** Regression on Manifolds with Implications for System Identification. Thesis No. 1382, 2008.

**D. Ankelhed:** On low order controller synthesis using rational constraints. Thesis No. 1398, 2009.

**P. Skoglar:** Planning Methods for Aerial Exploration and Ground Target Tracking. Thesis No. 1420, 2009.

**C. Lundquist:** Automotive Sensor Fusion for Situation Awareness. Thesis No. 1422, 2009.

**C. Lyzell:** Initialization Methods for System Identification. Thesis No. 1426, 2009.

**R. Falkeborn:** Structure exploitation in semidefinite programming for control. Thesis No. 1430, 2010.

**D. Petersson:** Nonlinear Optimization Approaches to  $\mathcal{H}_2$ -Norm Based LPV Modelling and Control. Thesis No. 1453, 2010.

**Z. Sjanic:** Navigation and SAR Auto-focusing in a Sensor Fusion Framework. Thesis No. 1464, 2011.

**K. Granström:** Loop detection and extended target tracking using laser data. Thesis No. 1465, 2011.

**J. Callmer:** Topics in Localization and Mapping. Thesis No. 1489, 2011.

**F. Lindsten:** Rao-Blackwellised particle methods for inference and identification. Thesis No. 1480, 2011.

**M. Skoglund:** Visual Inertial Navigation and Calibration. Thesis No. 1500, 2011.

**S. Khoshfetrat Pakazad:** Topics in Robustness Analysis. Thesis No. 1512, 2011.