Linköping studies in science and technology. Dissertations.
No. 1585

Sensor Fusion and Control Applied to Industrial Manipulators

Patrik Axelsson

Department of Electrical Engineering
Linköping University, SE–581 83 Linköping, Sweden

Linköping 2014

Cover illustration: Filled contour plot for the upper bound of the sample time given by (20) in Paper C. The upper bound is derived such that a stable continuous-time linear system remains stable after discretisation using Euler sampling. The sample time decreases when going from dark to light blue.

Linköping studies in science and technology. Dissertations.
No. 1585

Sensor Fusion and Control Applied to Industrial Manipulators

Patrik Axelsson

axelsson@isy.liu.se
www.control.isy.liu.se
Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE–581 83 Linköping
Sweden

ISBN 978-91-7519-368-7 ISSN 0345-7524

Printed by LiU-Tryck, Linköping, Sweden 2014

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 small, 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 has introduced unwanted flexibilities in the joints and the links. The consequence is that 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 using Bayesian estimation methods for state estimation, here represented by the extended Kalman filter and the particle filter. The arm-side information is provided by an accelerometer mounted at the end-effector. The measurements consist of the motor angular positions and the acceleration of the end-effector. In a simulation study on a realistic flexible industrial robot, the angular position performance is shown to be close to the fundamental Cramér-Rao lower bound. The methods are also verified in experiments on an ABB IRB4600 robot, where the dynamic performance of the position for the end-effector is significantly improved. There is no significant difference in performance between the different methods. Instead, execution time, model complexities and implementation issues have to be considered when choosing the method. The estimation performance depends strongly on the tuning of the filters and the accuracy of the models that are used. Therefore, a method for estimating the process noise covariance matrix is proposed. Moreover, sampling methods are analysed and a low-complexity analytical solution for the continuous-time update in the Kalman filter, that does not involve oversampling, is proposed.

The thesis also investigates two types of control problems. First, the norm-optimal *iterative learning control* (ILC) algorithm for linear systems is extended to an estimation-based norm-optimal ILC algorithm where the controlled variables are not directly available as measurements. The algorithm can also be applied to non-linear systems. The objective function in the optimisation problem is modified to incorporate not only the mean value of the estimated variable, but also information about the uncertainty of the estimate. Second, $\mathcal{H}_\infty$ controllers are designed and analysed on a linear four-mass flexible joint model. It is shown that the control performance can be increased, without adding new measurements, compared to previous controllers. Measuring the end-effector acceleration increases the control performance even more. A non-linear model has to be used to describe the behaviour of a real flexible joint. An $\mathcal{H}_\infty$-synthesis method for control of a flexible joint, with non-linear spring characteristic, is therefore proposed.

Populärvetenskaplig sammanfattning

En av de viktigaste uppgifterna för en industrirobot är att förflytta verktyget i en fördefinierad bana med en specificerad hastighet och acceleration. Exempel på användningsområden för en industrirobot är bland annat bågsvetsning eller limning. För dessa typer av applikationer är det viktigt att banföljningsfelet är extremt litet, men även hastighetsprofilen måste följas så att det till exempel inte appliceras för mycket eller för lite lim. Andra användningsområden kan vara punktsvetsning av bilkarosser och paketering av olika varor. För dess applikationer är banföljningen inte det viktiga, istället kan till exempel en tidsoptimal banföljning krävas eller att svängningarna vid en inbromsning minimeras. 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, som är det man egentligen vill styra.

En del av utvecklingsarbetet för nya generationers robotar är att reducera kostnaden men samtidigt förbättra prestandan. Ett sätt att minska kostnaden kan till exempel vara att minska dimensionerna på länkarna eller köpa in billigare växellådor. Den här utvecklingen av kostnadsoptimerade robotar har infört oönskade vekheter i leder och länkar. Det är därför inte längre möjligt att få den önskade prestandan och robustheten genom att bara mäta motorernas vinkelpositioner och -hastigheter. Istället krävs det omfattande matematiska modeller som beskriver dessa oönskade vekheter. Dessa modeller kräver mycket arbete att dels ta fram men även för att identifiera parametrarna. Det finns automatiska metoder för att beräkna modellparametrarna men oftast krävs det en manuell justering för att få bra prestanda.

Den här avhandlingen undersöker möjligheterna att beräkna verktygspositionen med hjälp av bayesianska metoder för tillståndsskattning. De bayesianska skattningsmetoderna beräknar tillstånden för ett system iterativt. Med hjälp av en matematisk modell över systemet predikteras vad tillståndet ska vara vid nästa tidpunkt. Efter att mätningar av systemet vid den nya tidpunkten har genomförts justeras skattningen med hjälp av dessa mätningar. De metoder som har använts i avhandlingen är det så kallade *extended Kalman filtret* samt *partikelfiltret.*

Informationen på armsidan av växellådan ges av en accelerometer som är monterad på verktyget. Med hjälp av accelerationen för verktyget och motorernas vinkelpositioner kan en skattning av verktygspositionen beräknas. I en simuleringsstudie för en realistisk vek robot har det visats att skattningsprestandan ligger nära den teoretiska undre gränsen, känd som Cramér-Raos undre gräns, samt att införandet av en accelerometer förbättrar prestandan avsevärt. Metoderna har även utvärderats på experimentella mätningar för en ABB IRB4600 robot. Ett av resultaten i avhandlingen visar att prestandan mellan olika metoder inte skiljer sig markant. Istället måste exekveringstid, modellkomplexitet och implementeringskostnader tas med i valet av metod. Vidare beror skattningsprestandan till stor del på hur filtren har trimmats. Trimningsparametrarna är kopplade till process-

och mätstörningar som påverkar roboten. För att underlätta trimningen så har en metod för att skatta processbrusets kovariansmatris föreslagits. En annan viktig del som påverkar prestandan är modellerna som används i filtren. Modellerna för en industrirobot är vanligtvis framtagna i kontinuerlig tid medan filtren använder modeller i diskret tid. För att minska felen som uppkommer då de tidskontinuerliga modellerna överförs till diskret tid har olika samplingsmetoder studerats. Vanligtvis används enkla metoder för att diskretisera vilket innebär problem med prestanda och stabilitet. För att hantera dessa problem införs översampling vilket innebär att tidsuppdateringen sker med en mycket kortare sampeltid än vad mätuppdateringen gör. För att undvika översampling kan det motsvarande tidskontinuerliga filtret användas för att prediktera tillstånden vid nästa diskreta tidpunkt. En analytisk lösning med låg beräkningskomplexitet till detta problem har föreslagits.

Vidare innehåller avhandlingen två typer av reglerproblem relaterade till industrirobotar. För det första har den så kallade norm-optimala *iterative learning control* styrlagen utökats till att hantera fallet då en skattning av den önskade reglerstorheten används istället för en mätning. Med hjälp av skattningen av systemets tillståndsvektor kan metoden nu även användas till olinjära system vilket inte är fallet med standardformuleringen. Den föreslagna metoden utökar målfunktionen i optimeringsproblemet till att innehålla inte bara väntevärdet av den skattade reglerstorheten utan även skattningsfelets kovariansmatris. Det innebär att om skattningsfelet är stort vid en viss tidpunkt ska den skattade reglerstorheten vid den tidpunkten inte påverka resultatet mycket eftersom det finns en stor osäkerhet i var den sanna reglerstorheten befinner sig.

För det andra har design och analys av $\mathcal{H}_\infty$-regulatorer för en linjär modell av en vek robotled, som beskrivs med fyra massor, genomförts. Det visar sig att reglerprestandan kan förbättras, utan att lägga till fler mätningar än motorns vinkelposition, jämfört med tidigare utvärderade regulatorer. Genom att mäta verktygets acceleration kan prestandan förbättras ännu mer. Modellen över leden är i själva verket olinjär. För att hantera detta har en $\mathcal{H}_\infty$-syntesmetod föreslagits som kan hantera olinjäriteten i modellen.

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, to our current head, Professor Svante Gunnarsson, and our administrator Ninna Stengård for making the administrative work easier.

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

Many thanks to Dr. Stig Moberg at ABB Robotics for providing me with the robot models, both the mathematical equations and the models implemented in Matlab and Simulink. My gratitudes also goes to my co-authors, Docent Rickard Karlsson for helping me with anything concerning probability, and to Dr. Anders Helmersson for helping me endure the tough world of tuning $\mathcal{H}_\infty$ controllers. The thesis layout had not been this good if not the previous LaTeX gurus Dr. Henrik Tidefelt and Dr. Gustaf Hendeby (who will soon rule the world again as the LaTeX guru) had spent a lot of their time creating the thesis template, many thanks to you. Also many thanks to Dr. Daniel Petersson and Dr. Christian Lyzell for all the help when Ti*k*Z doesn't produce the figures I want it to produce. Daniel also deserves many thanks for being my own computer support and answering all my easy/hard questions about matrix algebra.

The content of the thesis had been much harder to read without all the good comments from Lic. André Carvalho Bittencourt, Lic. Daniel Simon, Lic. Ylva Jung, M.Sc. Clas Veibäck and my supervisors Professor Mikael Norrlöf and Professor Fredrik Gustafsson. Thanks a lot to all of you.

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. Everything from just taking a beer at the Friday-pub or at some nice place down town, to having a winter barbecue, or sharing an "Il Kraken". Thanks also to all my friends outside the RT-corridor, I hope you know who you are.

Finally, many thanks to my family for their support and to Louise for her patience and love. Louise, now it is finally time for me to decide what to do after this journey. I know where you want me, but also that everything can change if the price is right. Anyway, I hope that the future will be great for both of us.

Linköping, March 2014
Patrik Axelsson

Contents

II Publications

Notation

Estimation

Notation	Meaning
$\mathbf{x}_k$	State vector at time k
$\mathbf{u}_k$	Input vector at time k
$\mathbf{w}_k$	Process noise vector at time k
$\mathbf{y}_k$	Measurement vector at time k
$\mathbf{v}_k$	Measurement noise vector at time k
$p(\mathbf{x}\|\mathbf{y})$	Conditional density function for $\mathbf{x}$ given $\mathbf{y}$
$\mathbf{y}_{1:k}$	Sequence of measurements from time 1 to time k
$\hat{\mathbf{x}}_{k\|k'}$	Estimated state vector at time k given measurements up to and including time k'
$\mathbf{P}_{k\|k'}$	Covariance of the estimated state vector at time k given measurements up to and including time k'
$\hat{\mathbf{x}}^s_{k\|N}$	Smoothed state vector at time k given measurements up to time $N \geq k$
$\mathbf{P}^s_{k\|N}$	Covariance of the smoothed stated vector at time k given measurements up to time $N \geq k$
$\mathbf{x}^{(i)}_k$	Particle i at time k
$w^{(i)}_k$	Weight for particle i at time k
$\mathcal{N}\left(\,\cdot\,;\boldsymbol{\mu},\Sigma\right)$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance Σ
$\mathbf{Q}/\mathbf{R}$	Covariance matrix for the process/measurement noise
$\boldsymbol{\theta}$	Unknown parameter vector

Robotics

Notation	Meaning
$\boldsymbol{\mathcal{Q}}_{j/i}$, $\boldsymbol{\mathcal{R}}_{j/i}$	Rotation matrix for system j with respect to system i
$\mathbf{r}_{j/i}$	Vector from origin in frame i to origin in frame j
$\mathbf{H}_{j/i}$	Homogeneous transformation matrix for system j with respect to system i
$\mathbf{r}_i^h$	Homogeneous coordinate
$\boldsymbol{\Xi}$	Position and orientation of the end-effector
$\dot{\boldsymbol{\Xi}}/\ddot{\boldsymbol{\Xi}}$	Linear and angular velocity/acceleration of the end-effector
$\mathbf{q}_a/\dot{\mathbf{q}}_a/\ddot{\mathbf{q}}_a$	Arm angular positions/velocities/accelerations
$\mathbf{q}_m/\dot{\mathbf{q}}_m/\ddot{\mathbf{q}}_m$	Motor angular positions/velocities/accelerations
$\mathbf{q}_m^a/\dot{\mathbf{q}}_m^a/\ddot{\mathbf{q}}_m^a$	Motor angular positions/velocities/accelerations expressed on the arm side of the gearbox
$\boldsymbol{\tau}_m$ ($\boldsymbol{\tau}_m^a$)	Motor torque (expressed on the arm side of the gearbox)
$\Upsilon(\cdot)$	Forward kinematic model
$\mathbf{J}(\cdot)$	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
$D(\cdot)$	Damping torque
η	Gear ratio

Control

Notation	Meaning
$P(s)$	System from exogenous input signals and control signals to exogenous output signals and measurement signals
$K(s)$	Controller
$F_l(\cdot,\cdot)$	Linear fractional transformation
$\mathbf{K}_P$, $\mathbf{K}_D$	Parameters for the PD controller
$\Vert\cdot\Vert_\infty$	Infinity-norm
$\boldsymbol{\mathcal{Q}}$, $\boldsymbol{\mathcal{L}}$	ILC algorithm matrices
$W_{\mathbf{u}}$, W_S, W_P, M_S, W_T,	Weighting functions for $\mathcal{H}_\infty$-control synthesis
W_1, W_2	Weighting functions for loop shaping synthesis
$\overline{\mathbf{x}}$ ($\overline{\mathbf{z}}$, $\overline{\mathbf{y}}$, $\overline{\mathbf{e}}$, $\overline{\mathbf{r}}$, $\overline{\mathbf{u}}$)	The vector $\mathbf{x}(t)$ ($\mathbf{z}(t)$, $\mathbf{y}(t)$, $\mathbf{e}(t)$, $\mathbf{r}(t)$, $\mathbf{u}(t)$) stacked for $t = 0, \ldots, N$
$\mathscr{C}$ ($\mathscr{C}^o$)	(Output) Controllability matrix
$\mathscr{O}$	Observability matrix
$\boldsymbol{\mathcal{P}}$	Lyapunov matrix

Miscellaneous

Notation	Meaning
$\mathbf{I}$	Identity matrix
$\mathbf{0}$	Null matrix
†	Pseudo inverse
$^{\top}$	Transpose
T_s, h	Sample time
tr	Trace operator
rank	Rank of a matrix
$\mathrm{E}\left[\,\cdot\,\right]$	Expectation value
$\mathrm{Cov}\left(\,\cdot\,\right)$	Covariance
g	Gravity constant
x, z $(\hat{\mathsf{x}}, \hat{\mathsf{z}})$	(Estimated) Cartesian coordinates
$O\mathsf{x}_i\mathsf{y}_i\mathsf{z}_i$	Cartesian coordinate frame named i
$\ddot{\boldsymbol{\rho}}_s$	Acceleration due to the motion in the accelerometer frame
$\mathbf{b}$	Bias vector
$\mathbb{R}/\mathbb{R}_{+}/\mathbb{R}_{++}$	Real/Non-negative/Positive numbers
$\mathbb{R}^{n}$	n-dimensional Euclidian space
$\mathbb{R}^{n\times m}$	Space of real $n \times m$ matrices
$\mathbb{S}^{n}_{++}$ $(\mathbb{S}^{n}_{+})$	Set of symmetric positive definite (semi-definite) $n \times n$ matrices
$\rho(\,\cdot\,)$	Spectral radius
$\bar{\sigma}(\,\cdot\,)$	Maximal singular value
$\otimes$	Kronecker product
vec $(\mathrm{vech}\,)$	(Half) Vectorisation
$\mathbf{u}_k(t)$	Vector $\mathbf{u}$ at time t and ILC iteration k
$e_{p,m}(\,\cdot\,)$	pth order Taylor expansion of the matrix exponential with scaling of the argument with a factor m
$d\boldsymbol{\beta}(t)$	Vector of Wiener processes

Abbreviations

Abbreviation	Meaning
CDLE	Continuous-time differential Lyapunov equation
CRLB	Cramér-Rao lower bound
DDLE	Discrete-time difference Lyapunov equation
DH	Denavit-Hartenberg
DOF	Degrees of freedom
EM	Expectation maximisation
EKF	Extended Kalman filter
EKS	Extended Kalman smoother
ILC	Iterative learning control
IMM	interacting multiple model
IMU	Inertial measurement unit
KF	Kalman filter
KKF	Kinematic Kalman filter
KKT	Karush-Kuhn-Tucker
ML	Maximum likelihood
MC	Monte Carlo
ODE	Ordinary differential equation
PF	Particle filter
PDF	Probability density function
RGA	Relative gain array
RMSE	Root mean square error
SDE	Stochastic differential equation
SNR	Signal to noise ratio
SSV	Structured singular value
TCP	Tool centre point
TPC	Target path controllability
ZOH	Zero order hold

Part I

Background

1

Introduction

IN THIS THESIS, state estimation and control for industrial manipulators are studied. First, estimation of the unknown joint angles of the robot using motor angular position measurements together with acceleration measurements of the tool, is considered. Second, control of the manipulator using the estimated states, and the use of the acceleration measurement of the tool directly in the feedback loop, are investigated.

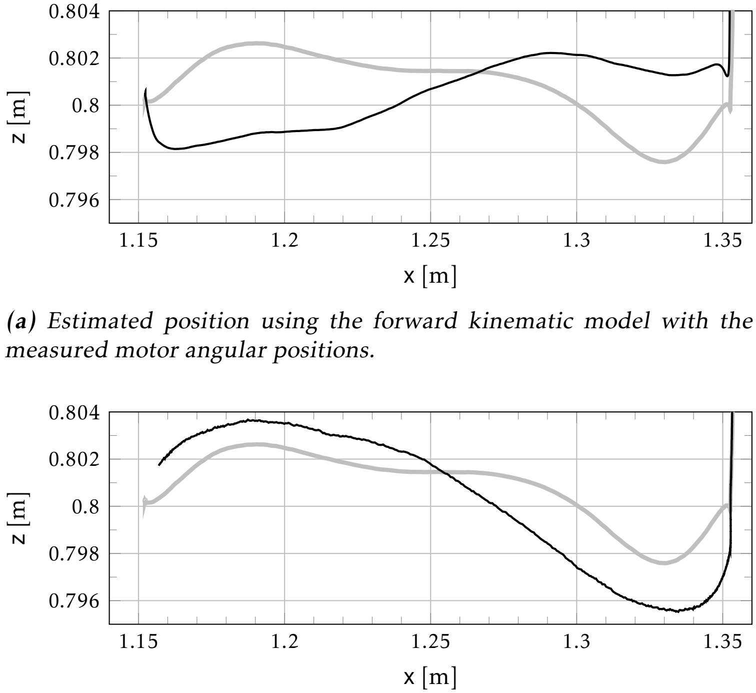
The background and motivation of the work are presented in Section 1.1. The contributions of the thesis are listed in Section 1.2 and the outline of the thesis is presented in Section 1.3.

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 leads 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 non-linearities. 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 non-linear relations, which cannot be neglected in the motion control loop. The friction in the gearbox is also an increasing problem that is described by non-linear 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 non-linear flexibilities are needed in order to control the weight optimised robot [Moberg, 2010]. 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 create an extended kinematic model and an elasto-static model by conducting an off-line identification procedure. The static error can, in this way, be reduced to 0.5 mm. For the dynamic accuracy a new model-based motion control scheme 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, reducing the material cost leads to more complex mathematical models that can explain the behaviour of the manipulator. Therefore, there is a demand for new approaches of motion control schemes, where new types of measurements are introduced and less complex 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. The simplest 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 1.1a it is shown that the estimated position of the end-effector does not track the true measured position very well. The reason for the poor result is that the oscillations on the arm side do not influence the motor side of the gearbox due to the flexibilities. The flexibilities can also distort the oscillations on the arm side, which means that the estimated path oscillates in a different way than the true path. The kinematic model 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 the gearbox and possibly also more measurements than only the motor angular positions. Figure 1.1b shows one of the experimental results in this thesis, presented in Papers A and B, where a particle filter has been used. The true position of the end-effector can, for evaluation purposes, be measured by an external laser tracking system from Leica Geosystems [2014], which tracks a crystal attached to the robot. The tracking system is very expensive and it requires line of sight and is therefore not an option to use for feedback control in practice. Note that measurements of the orientation of the end-effector cannot be obtained using this type of tracking system.

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, perfect measurements from the encoders on the arm side and the desired angles will differ. Take the first joint of the robot in Figure 2.2 as an example. The system from the motor



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

(b) *Estimated position using a particle filter.*

Figure 1.1: *True path (grey) and estimated path (black) of the end-effector using (a) the forward kinematic model with the measured motor angular positions, and (b) a particle filter using the motor angular positions and the acceleration of the end-effector as measurements. The dynamical performance of the estimated path in (a) is insufficient due to the flexible gearboxes between the measured motor angular positions and the end-effector. The estimated path from a particle filter in (b) is much better.*

side of the gearbox to the end-effector can be seen as a three-mass system, or more, 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 for estimating the end-effector path. One way to obtain information about the oscillations on the arm side can be to attach an accelerometer to the robot, e.g. at the end-effector, which is the approach used in this thesis. The accelerometer that has been used is a triaxial accelerometer from Crossbow Technology [Crossbow Technology, 2004].

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 Necsulescu [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. Estimating the joint angles using a combination of measurements from accelerometers, gyroscopes and vision is presented in Jeon et al. [2009]. The so called *kinematic Kalman filter* (KKF), which basically is a Kalman filter applied to a kinematic model, is used for estimation and the results are verified on a planar two-link robot. In Chen and Tomizuka [2014] the estimates are obtained using a two-step procedure. First, rough estimates of the joint angles are obtained using the dynamical model and numerical differentiation. Second, the rough estimates are used to decouple the complete system into smaller systems where the KKF is applied to improve the estimates. The method is verified experimentally on a six *degrees-of-freedom* (DOF) manipulator. In Lertpiriyasuwat 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.

From an on-line control perspective, it is important that the estimation method performs in real-time. The sample time for industrial manipulators is usually on the time scale of milliseconds, hence the real time requirements are very high, and e.g. the PF can have difficulties to achieve real-time performance. However, the estimated position of the end-effector can still be used in an off-line application, such as *iterative learning control* (ILC), if the estimation method performs slower than real-time. In Wallén et al. [2008] it is shown that motor side learning is insufficient if the mechanical resonances are excited by the robot trajectory. Instead estimation-based ILC has to be considered, as presented in Wallén et al. [2009], to improve the performance even more. Other applications that can improve if the estimated position of the end-effector is available, not necessarily online, are system identification, supervision, diagnosis, and automatic controller tuning.

Although the estimation method performs slower than real-time, it is interesting to investigate the direct use of an accelerometer, attached to the end-effector, in the feedback loop together with more accurate models. Model-based controllers for flexible joints, modelled as a two-mass model with linear flexibility, have been considered for many years [Sage et al., 1999; De Luca and Book, 2008]. Section 2.3 presents an overview of common controllers for industrial manipulators. As previously stated, a two mass model with linear flexibility does not represent the actual robot structure sufficiently well, hence more complex models are needed for control. Moberg et al. [2009] presents a benchmark model for a single joint intended to be used for evaluation of new controllers. The joint is modelled as a four mass model, where the first flexibility is given by a non-linear function, and the only available measurement is the motor position. Four model based solutions are presented. An interesting thing from Moberg et al. [2009] is that one of the best controllers can be realised as a parallel PID controller. To improve the control performance significantly, more measurements must be included. An encoder measuring the arm angular position will of course improve the performance. However, the position of all masses cannot be measured, hence all oscillations cannot be taken care of. Instead sensors attached to the end-effector, such as an accelerometer, must be used to give more information. Feedback of the end-effector acceleration has been considered in e.g. Kosuge et al. [1989] and Xu and Han [2000], where a rigid joint model has been used. In Kosuge et al. [1989] the non-linear robot model is linearised using feedback linearisation. For controller synthesis the $\mathcal{H}_\infty$ methodology can be used. Song et al. [1992] and Stout and Sawan [1992] uses a rigid joint model which is first linearised using feedback linearisation. Second, the linearised model is used for design of an $\mathcal{H}_\infty$ controller. A similar approach is used in Sage et al. [1997] where the model is a linear flexible joint model and the motor positions are measured. Non-linear $\mathcal{H}_\infty$ methods applied to rigid joint manipulators are considered in Yim and Park [1999]; Taveira et al. [2006]; Miyasato [2008] and Miyasato [2009]. For flexible joint models the non-linear $\mathcal{H}_\infty$ approach is presented in Yeon and Park [2008] and Lee et al. [2007].

1.2 Contributions

The main contributions in this thesis are i) how to estimate the position of the end-effector using an accelerometer, and ii) control strategies using either the estimated position or the accelerometer measurement directly.

1.2.1 Included Publications

Paper A: Bayesian State Estimation of a Flexible Industrial Robot

A sensor fusion method for state estimation of a flexible industrial robot is presented in

> Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012b.

By measuring the acceleration at the end-effector, the accuracy of the estimated arm angular position, as well as the estimated position of the end-effector are improved using the extended Kalman filter and the particle filter. In a simulation study the influence of the accelerometer is investigated and the two filters are compared to each other. The angular position performance is increased when the accelerometer is used and the performance for the two filters is shown to be close to the fundamental Cramér-Rao lower bound. The technique is also verified in experiments on an ABB IRB4600 robot. The experimental results have also been published in

> Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Tool position estimation of a flexible industrial robot using recursive Bayesian methods. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5234–5239, St. Paul, MN, USA, May 2012a.

Paper B: Evaluation of Six Different Sensor Fusion Methods for an Industrial Robot Using Experimental Data

Experimental evaluations for path estimation on an ABB IRB4600 robot are presented in

> Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 126–132, Dubrovnik, Croatia, September 2012.

Different observers using Bayesian techniques with different estimation models are investigated. It is shown that there is no significant difference in performance between the best observers. Instead, execution time, model complexities and implementation issues have to be considered when choosing the method.

Paper C: Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation with Applications to Kalman Filtering

Over-sampling strategies for filtering of continuous-time stochastic processes are analysed in

> Patrik Axelsson and Fredrik Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *Submitted to IEEE Transactions on Automatic Control*, 2012,

where a novel low-complexity analytical solution to the *continuous-time differential Lyapunov equation* (CDLE), that does not involve oversampling, is proposed. The results are illustrated on Kalman filtering problems in both linear and nonlinear systems. Another approach to the discretisation problem is presented in the related publication

> Niklas Wahlström, Patrik Axelsson, and Fredrik Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014.

A method to calculate the covariance matrix for the discretised noise is proposed, instead of solving the CDLE. The covariance matrix is given as the solution to an algebraic Lyapunov equation. The same Lyapunov equation is a part of the solution to the CDLE, which is presented in this thesis.

Paper D: ML Estimation of Process Noise Variance in Dynamic Systems

Parameter identification using the expectation maximisation algorithm, which iteratively estimates the unobserved state sequence and the process noise covariance matrix based on the observations of the process, is presented in

> 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/September 2011.

The extended Kalman smoother is the instrument to find the unobserved state sequence, and the proposed method is compared to two alternative methods on a simulated robot model.

Paper E: $\mathcal{H}_\infty$-Controller Design Methods Applied to one Joint of a Flexible Industrial Manipulator

Control of a flexible joint of an industrial manipulator using $\mathcal{H}_\infty$-design methods is presented in

> Patrik Axelsson, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-controller design methods applied to one joint of a flexible industrial manipulator. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014b.

The considered design methods are i) mixed-$\mathcal{H}_\infty$ design, and ii) $\mathcal{H}_\infty$ loop shaping design. Two different controller configurations are examined: one uses only the actuator position, while the other uses the actuator position and the acceleration of the end-effector. The four resulting controllers are compared to a standard PID controller where only the actuator position is measured. Model order reduction of the controllers is also briefly discussed.

Paper F: $\mathcal{H}_\infty$-Synthesis Method for Control of Non-linear Flexible Joint Models

An $\mathcal{H}_\infty$-synthesis method for control of a flexible joint, with non-linear spring characteristic, is proposed in

> Patrik Axelsson, Goele Pipeleers, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-synthesis method for control of non-linear flexible joint models. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014d.

The method is motivated by the assumption that the joint operates in a specific stiffness region of the non-linear spring most of the time, hence, the performance requirements are only valid in that region. However, the controller must stabilise the system in all stiffness regions. The method is validated in simulations on a two mass non-linear flexible joint model.

Paper G: Estimation-based Norm-optimal Iterative Learning Control

The norm-optimal *iterative learning control* (ILC) algorithm for linear systems is in

> Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based norm-optimal iterative learning control. *Submitted to Systems & Control Letters*, 2014c

extended to an estimation-based norm-optimal ILC algorithm, where the controlled variables are not directly available as measurements. For linear time-invariant systems with a stationary Kalman filter it is shown that the ILC design is independent of the design of the Kalman filter. The algorithm is also extended to non-linear state space models using linearisation techniques. Finally, stability and convergence properties are derived.

Paper H: Controllability Aspects for Iterative Learning Control

The aspects of controllability in the iteration domain, for systems that are controlled using ILC, are discussed in

> Patrik Axelsson, Daniel Axehill, Torkel Glad, and Mikael Norrlöf. Controllability aspects for iterative learning control. *Submitted to International Journal of Control*, 2014a.

A state space model in the iteration domain is proposed to support the discussion. It is shown that it is more suitable to investigate if a system can follow a trajectory instead of the ability to control the system to an arbitrary point in the state space. This is known as target path controllability. A simulation study is also performed to show how the ILC algorithm can be designed using the LQ-method. It is shown that the control error can be reduced significantly using the LQ-method compared to the norm-optimal approach.

1.2.2 Additional Publications

Related articles, which are not included in this thesis, are presented here with a short description of the contributions.

Simulation results for the estimation problem are presented in the following two publications

> Patrik Axelsson, Mikael Norrlöf, Erik Wernholt, and Fredrik Gustafsson. Extended Kalman filter applied to industrial manipulators. In *Proceedings of Reglermötet*, 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 Enginering, Linköping University, SE-581 83 Linköping, Sweden, December 2009,

where performance in case of uncertain dynamical model parameters as well as uncertainties in the position and orientation of the accelerometer are studied. A detailed description of the simulation model that has been used is given in

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

Parts of the material in this thesis have previously been published in

> Patrik Axelsson. *On Sensor Fusion Applied to Industrial Manipulators*. Linköping Studies in Science and Technology. Licentiate Thesis No. 1511, Linköping University, SE-581 83 Linköping, Sweden, December 2011b.

Method to Estimate the Position and Orientation of a Triaxial Accelerometer Mounted to an Industrial Robot

A method, used in Papers A and B, to find the orientation and position of a triaxial accelerometer mounted on a six DOF industrial robot is proposed in

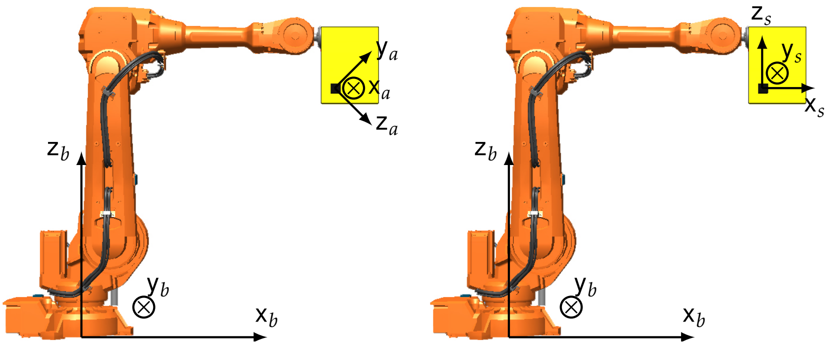
> Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 283–288, Dubrovnik, Croatia, September 2012.

Assume that the accelerometer is mounted on the robot according to Figure 1.2a. The problem is to find:

(i) The internal sensor parameters and the orientation of the sensor.

(ii) The position of the accelerometer with respect to the robot end-effector coordinate system.

The method consists of two consecutive steps, where the first is to estimate the orientation of the accelerometer from static experiments. In the second step the accelerometer position relative to the robot base is identified using accelerometer readings. Identification of the orientation can be seen as finding a transformation from the actual coordinate system $O\mathsf{x}_a\mathsf{y}_a\mathsf{z}_a$ to a desired coordinate system $O\mathsf{x}_s\mathsf{y}_s\mathsf{z}_s$, which can be seen in Figure 1.2b. The relation between $O\mathsf{x}_a\mathsf{y}_a\mathsf{z}_a$ and



*(**a**) The accelerometer and its actual coordinate system $Ox_ay_az_a$.*

*(**b**) The accelerometer and the desired coordinate system $Ox_sy_sz_s$.*

***Figure 1.2:** 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 $Ox_by_bz_b$ of the robot is also shown.*

$Ox_sy_sz_s$ is given by,

$$\boldsymbol{\rho}_s = \kappa \mathcal{R}_{a/s} \boldsymbol{\rho}_a + \boldsymbol{\rho}_0, \tag{1.1}$$

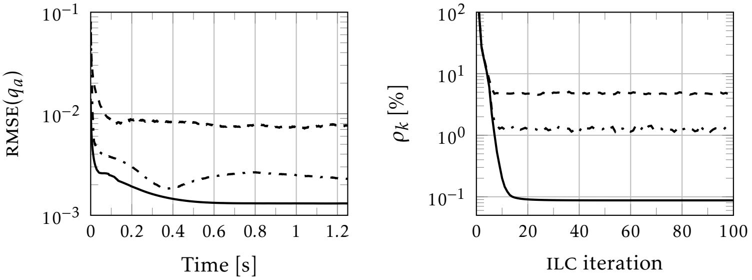
where $\boldsymbol{\rho}_a$ is a vector in $Ox_ay_az_a$, $\boldsymbol{\rho}_s$ is a vector in $Ox_sy_sz_s$, $\mathcal{R}_{a/s}$ is the rotation matrix from $Ox_ay_az_a$ to $Ox_sy_sz_s$, κ is the accelerometer sensitivity and $\boldsymbol{\rho}_0$ the bias. When the unknown parameters in (1.1) have been found the position of the accelerometer, expressed relative to the end-effector coordinate system, can be identified. The position is identified using accelerometer readings when the accelerometer moves in a circular path and where the accelerometer orientation is kept constant in a path fixed coordinate system.

Estimation-based ILC using Particle Filter with Application to Industrial Manipulators

In

> Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based ILC using particle filter with application to industrial manipulators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1740–1745, Tokyo, Japan, November 2013,

an estimation-based ILC algorithm is applied to a realistic industrial manipulator model. By measuring the acceleration of the end-effector, the arm angular position accuracy is improved when the measurements are fused with the motor angular position observations using the extended Kalman filter, the unscented Kalman filter, and the particle filter. In an extensive Monte Carlo simulation study it is shown that the particle filter outperforms the other methods, see Fig-



(a) Root mean square error for the EKF and UKF (dashed) and the PF (dash-dotted) compared to the Cramér-Rao lower bound (solid).

(b) Normalised control error when true arm position is used (solid), the EKF and UKF (dashed) and the PF (dash-dotted).

Figure 1.3: *Results for estimation-based ILC using Bayesian state estimation.*

ure 1.3a, and that the control error is substantially improved using the particle filter estimate in the ILC algorithm, see Figure 1.3b.

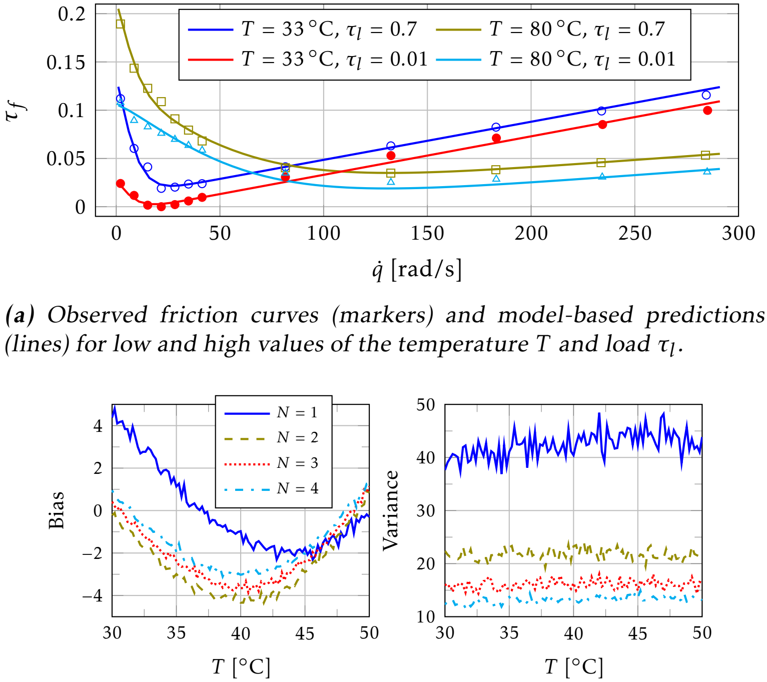
Identification of Wear in a Robot Joint

The effects of wear to friction based on constant-speed friction data are studied in the two papers

> André Carvalho Bittencourt and Patrik Axelsson. Modeling and experiment design for identification of wear in a robot joint under load and temperature uncertainties based on friction data. *IEEE/ASME Transactions on Mechatronics*, 2013. DOI: 10.1109/TMECH.2013.2293001,

> 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/September 2011.

It is shown how the effects of temperature and load uncertainties, see Figure 1.4a, produce larger changes to friction than those caused by wear. Based on empirical observations, an extended friction model is proposed to describe the effects of speed, load, temperature and wear. A maximum likelihood wear estimator is proposed, where it is assumed that an extended friction model and constant-speed friction data are available. The distribution of the load uncertainties can be estimated, hence the load can be marginalised away. The wear and temperature can then be jointly estimated. The proposed method is evaluated in both extensive simulations, see Figure 1.4b, and on experimental data. It is shown that reliable wear estimates can be achieved even under load and temperature uncertainties. Also, experiment design is considered in terms of an optimal selection of speed points, where the friction data should be collected.



(a) *Observed friction curves (markers) and model-based predictions (lines) for low and high values of the temperature T and load τ_l.*

(b) *Monte Carlo based estimates of bias and variance for different temperatures T, and N different number of speed points.*

Figure 1.4: *Friction model dependent of temperature and load used for identification of wear in (a), and the results of the identification method based on Monte Carlo simulations in (b).*

1.3 Thesis Outline

The thesis is divided into two parts. The first part contains background and motivation of the thesis as well as a theoretical background. Chapter 1 presents the problem and lists all included publications. Chapter 2 contains background material for industrial manipulators, both modelling and control. The general non-linear estimation problem is presented in Chapter 3 and several algorithms for state and parameter estimation are presented. Chapter 4 introduces different control strategies that have been used in the thesis. Finally, Chapter 5 concludes the work in the thesis and gives suggestions for future work. The second part of the thesis contains eight edited versions of selected papers.

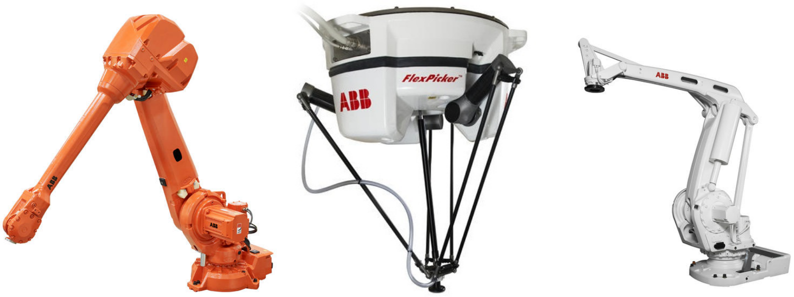
2
Industrial Robots

INDUSTRIAL MANIPULATORS are used in tasks where high precision and/or high speed are needed, such as spot welding, arc welding and laser cutting. Industrial manipulators are also important for tasks where the environment is harmful for humans, e.g. painting of car bodies. 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. 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. Section 2.2 presents a short overview of the kinematic and dynamic models needed for control, and Section 2.3 discusses the control problem and what types of control structures that are commonly used.

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. Europe had to wait until 1967 to get the first robot installation, which was carried out in Sweden. In 1973, the Swedish company ASEA (current ABB) launched the first



(a) *The serial arm robot ABB IRB4600.* ***(b)*** *The parallel arm robot ABB IRB360.* ***(c)*** *The parallel linkage robot ABB IRB660.*

Figure 2.1: *Three types of robots from ABB [ABB Robotics, 2014].*

micro-processor controlled electrical robot called IRB6. [Nof, 1999; Westerlund, 2000] Since then, ABB has evolved to one of the largest manufactures of industrial robots and robot systems. ABB has over 200,000 robots installed world wide and the company was the first with over 100,000 sold robots. ABB introduced the first paint robot in 1969 and in 1998 the fastest pick and place robot FlexPicker, IRB360 in Figure 2.1b, was launched [ABB Robotics, 2014]. Other big manufactures are the German company KUKA, FANUC Robotics with over 200,000 installed robots [FANUC Robotics, 2014], and Motoman owned by the Japanese company Yaskawa. KUKA built the first welding line with robots for Daimler-Benz in 1971 and launched the first industrial robot with six electro mechanically driven axes in 1973 [KUKA, 2014]. 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. Today Motoman has over 270,000 installed robots world wide and produces 20,000 robots per year [Motoman, 2014].

2.1.1 The Complete Robot System – An Overview

A general 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[1] in 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 pose is said to be defined in the task space whereas the joint

[1] The end-effector is an equipment mounted at the end of the robot arm to interact with the environment.

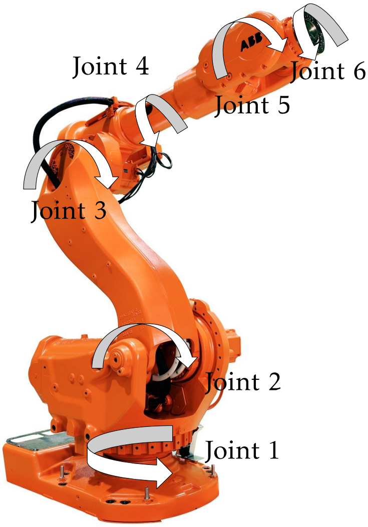


***Figure 2.2:** The six DOF serial arm robot ABB IRB6600 where the arrows indicate the joints [ABB Robotics, 2014].*

angles are said to be in the joint space. 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. A serial 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 acceleration 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 control system, depicted in Figure 2.3, can be divided up into three main parts; path planning, trajectory generation, and motion control. The three parts will be discussed briefly. A more thorough description of the motion controller will be addressed in Section 2.3.

The path planner defines, based on the user specifications, a geometrical path in the task space, which is then converted to a geometrical path in the joint space using the inverse kinematic model discussed in Section 2.2.1. If the user specifications are expressed for the joints then a geometrical path is directly defined in the joint space. Obstacle avoidance is also taken care of in the path planner. Note that the path only includes geometrical properties and nothing about time

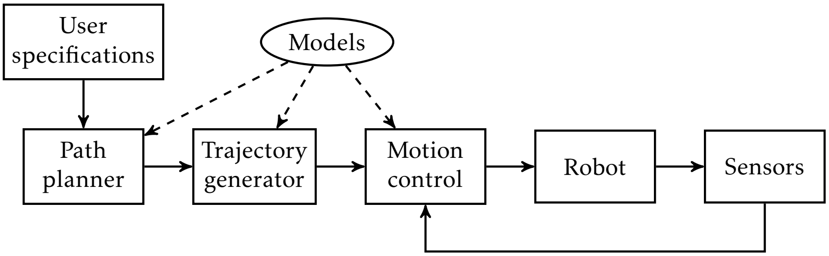


Figure 2.3: *Block diagram of the robot control system.*

dependencies such as velocity and acceleration.

The trajectory generator takes the geometrical path, which is defined in the path planner, and the time dependencies. The time dependencies are defined by the velocities and accelerations, which are specified by the user, but also limitations on the actuators, such as the joint torques, for the particular robot are used. To get a time dependent trajectory, that does not run into the physical limitations of the robot, requires a dynamical model of the manipulator. The dynamical model is discussed in Section 2.2.2. 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]. The output from the trajectory generator is position, velocity and acceleration of the robot joints as a function of time, which corresponds to the desired motion of the end-effector in the task space.

The motion controller uses the trajectories generated in the trajectory generator for control of the actuators, which often are electrical motors. Different types of control structures have been proposed in the literature. The most common ones are independent joint control, feed-forward control, and feedback linearisation. These control methods will be discussed in Section 2.3. 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 torque from the controller and calculates the corresponding current. The torque controller is usually assumed to be significantly faster than the robot dynamic and can therefore be omitted without influencing the control performance.

The path planner, trajectory generator, and motion 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 2.2.1 for details. The dynamic models describe the motion of the robot given the forces and torques acting on the robot, see Section 2.2.2.

The remaining of this chapter presents modelling and control for industrial manipulators and the main references are Spong et al. [2006]; Sciavicco and Siciliano [2000]; Siciliano and Khatib [2008]; Craig [1989], and Kozłowski [1998].

2.2 Modelling

2.2.1 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 into two different parts. The first consists of the relations between the known joint positions and the unknown pose of the end-effector. The second 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.

Coordinate Transformation

Let the vector $\mathbf{r}_j$ be fixed in frame j. The transformation to frame i can be written as

$$\mathbf{r}_i = \mathbf{r}_{j/i} + \mathcal{Q}_{j/i}\mathbf{r}_j, \tag{2.1}$$

where $\mathbf{r}_{j/i}$ is the vector from the origin in frame i to the origin in frame j and $\mathcal{Q}_{j/i}$ is the rotation matrix representing the orientation of frame j with respect to frame i. The rotation can be described using the so called Euler angles, which are intuitive but can cause singularities. Instead, unit quaternions, which do not suffer from singularities, can be used. They are however 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 numbered 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., $\mathbf{r}_{i/i-1}$ and $\mathcal{Q}_{i/i-1}$ are known. The transformation between two connected links can therefore be written as

$$\mathbf{r}_{i-1} = \mathbf{r}_{i/i-1} + \mathcal{Q}_{i/i-1}\mathbf{r}_i, \tag{2.2}$$

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

$$\mathbf{r}_i^h = \begin{pmatrix} \mathbf{r}_i \\ 1 \end{pmatrix}. \tag{2.3}$$

Equation (2.2) can now be reformulated as

$$\mathbf{r}_{i-1}^{h} = \begin{pmatrix} \mathcal{Q}_{i/i-1} & \mathbf{r}_{i/i-1} \\ \mathbf{0} & 1 \end{pmatrix} \mathbf{r}_{i}^{h} = \mathbf{H}_{i/i-1}\mathbf{r}_{i}^{h}. \tag{2.4}$$

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

$$\mathbf{r}_{0}^{h} = \mathbf{H}_{1/0} \cdot \ldots \cdot \mathbf{H}_{n/n-1}\mathbf{r}_{n}^{h} = \left(\prod_{i=1}^{n} \mathbf{H}_{i/i-1}\right)\mathbf{r}_{n}^{h} = \begin{pmatrix} \mathcal{Q}_{n/0} & \mathbf{r}_{n/0} \\ \mathbf{0} & 1 \end{pmatrix} \mathbf{r}_{n}^{h} = \mathbf{H}_{n/0}\mathbf{r}_{n}^{h}, \tag{2.5}$$

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

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 to the world frame given the joint angles $\mathbf{q}_a = \begin{pmatrix} q_{a1} & \ldots & q_{an} \end{pmatrix}^{\mathsf{T}}$. Here the subscript a is used to emphasis that the joint angles are described at the arm side of the gearbox, which will be convenient when flexible models are introduced later in this chapter. The world frame is a user defined frame where the robot is located, e.g. the industrial floor. The position $\mathbf{p} \in \mathbb{R}^3$ is given in Cartesian coordinates and the orientation $\boldsymbol{\phi} \in \mathbb{R}^3$ is given in Euler angles, or quaternions if desirable. The forward kinematics has a unique solution for a serial robot, e.g. ABB IRB4600 in Figure 2.1a, 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 non-linear mapping according to

$$\boldsymbol{\Xi} = \begin{pmatrix} \mathbf{p} \\ \boldsymbol{\phi} \end{pmatrix} = \Upsilon(\mathbf{q}_a), \tag{2.6}$$

where $\Upsilon(\mathbf{q}_a) : \mathbb{R}^n \to \mathbb{R}^6$ is a non-linear function given by the homogeneous transformation matrix

$$\mathbf{H}_{n/0} = \begin{pmatrix} \mathcal{Q}_{n/0} & \mathbf{r}_{n/0} \\ \mathbf{0} & 1 \end{pmatrix} \tag{2.7}$$

in (2.5), where the dependency of $\mathbf{q}_a$ has been omitted. The position of the end-effector frame, i.e., the first three rows in $\Upsilon(\mathbf{q}_a)$, is given by $\mathbf{r}_{n/0}$ and the orientation of the end-effector frame, i.e., the last three rows in $\Upsilon(\mathbf{q}_a)$, is given by $\mathcal{Q}_{n/0}$. The construction of $\mathbf{H}_{n/0}$, i.e., determination of all the $\mathbf{H}_{i/i-1}$ matrices in (2.4), can be difficult for complex robot structures. A systematic way to assign coordinate frames to simplify the derivation of $\mathbf{H}_{n/0}$ 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 $\dot{\mathbf{q}}_a$, the linear velocity $\mathbf{v}$, and angular velocity $\boldsymbol{\omega}$ of the end-effector according to

$$\dot{\boldsymbol{\Xi}} = \begin{pmatrix} \dot{\mathbf{p}} \\ \dot{\boldsymbol{\phi}} \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{pmatrix} = \frac{\partial \boldsymbol{\Upsilon}(\mathbf{q}_a)}{\partial \mathbf{q}_a} \dot{\mathbf{q}}_a = \mathbf{J}(\mathbf{q}_a)\dot{\mathbf{q}}_a, \tag{2.8}$$

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

$$\ddot{\boldsymbol{\Xi}} = \begin{pmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{pmatrix} = \begin{pmatrix} \mathbf{a} \\ \boldsymbol{\psi} \end{pmatrix} = \mathbf{J}(\mathbf{q}_a)\ddot{\mathbf{q}}_a + \left(\frac{d}{dt}\mathbf{J}(\mathbf{q}_a) \right) \dot{\mathbf{q}}_a, \tag{2.9}$$

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

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

The Jacobian matrix is fundamental in robotics. Besides being useful for the calculation of velocities and accelerations, it can also be used for

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

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.

Inverse Kinematics

In practice, often the position $\mathbf{p}$ and orientation $\boldsymbol{\phi}$ of the end-effector are given by the operating program and the corresponding joint angles $\mathbf{q}_a$ are required for the control loop. An inverse kinematic model is needed in order to get the corresponding joint angles. For a serial robot, the inverse kinematics is a substantially harder problem which can have several solutions or no solutions at all, 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 non-linear system of equations in (2.6) must be inverted, i.e.,

$$\mathbf{q}_a = \boldsymbol{\Upsilon}^{-1}(\boldsymbol{\Xi}). \tag{2.11}$$

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

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

$$\dot{\mathbf{q}}_a = \mathbf{J}^{-1}(\mathbf{q}_a)\dot{\boldsymbol{\Xi}}, \tag{2.12}$$

if the Jacobian is square and non-singular. The Jacobian is a square matrix when $n = 6$ since $\Upsilon(\mathbf{q}_a)$ has six rows, three for the position and three for the orientation. The singularity depends on the joint angles $\mathbf{q}_a$. 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 torques 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

$$\operatorname{rank} \mathbf{J}(\mathbf{q}_a) = \operatorname{rank} \begin{pmatrix} \mathbf{J}(\mathbf{q}_a) & \dot{\Xi} \end{pmatrix}, \tag{2.13}$$

that is, $\dot{\Xi}$ 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

$$\dot{\mathbf{q}}_a = \mathbf{J}^{\dagger}(\mathbf{q}_a)\dot{\Xi} + \left(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q}_a)\mathbf{J}(\mathbf{q}_a)\right)\mathbf{b}, \tag{2.14}$$

where $\mathbf{J}^{\dagger}(\mathbf{q}_a)$ is the pseudo inverse [Mitra and Rao, 1971] of $\mathbf{J}(\mathbf{q}_a)$ and $\mathbf{b} \in \mathbb{R}^n$ is an arbitrary vector. See Spong et al. [2006] for more details.

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

$$\ddot{\mathbf{q}}_a = \mathbf{J}^{-1}(\mathbf{q}_a)\left(\ddot{\Xi} - \frac{d}{dt}\left(\mathbf{J}(\mathbf{q}_a)\right)\dot{\mathbf{q}}_a\right). \tag{2.15}$$

2.2.2 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]. The methods may differ in computational efficiency and structure but the outcome is the same. Here, only Lagrange's equation will be covered.

Rigid Link Model

For Lagrange's equation the Lagrangian $L(\mathbf{q}, \dot{\mathbf{q}})$ is defined as the difference between the kinetic and potential energies. The argument $\mathbf{q}$ to the Lagrangian is a set of generalised coordinates. A system with n DOF can be described by n generalised coordinates $\mathbf{q} = \begin{pmatrix} q_1 & \dots & q_n \end{pmatrix}^{\mathsf{T}}$, e.g. position, velocity, angle or angular velocity that describe the system. The dynamic equations are given by Lagrange's equation

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

where the Lagrangian $L(\mathbf{q}, \dot{\mathbf{q}}) = K(\mathbf{q}, \dot{\mathbf{q}}) - P(\mathbf{q})$ is the difference between the kinetic and potential energies, and τ_i is the generalised force associated with q_i. For an industrial robot, the generalised coordinates are the joint angles $\mathbf{q}_a$, and the generalised forces are the corresponding motor torques. The dynamical equations for an industrial manipulator, given by Lagrange's equation, can be summarised by

$$M(\mathbf{q}_a)\ddot{\mathbf{q}}_a + C(\mathbf{q}_a, \dot{\mathbf{q}}_a) + G(\mathbf{q}_a) = \boldsymbol{\tau}_m^a, \tag{2.17}$$

where $M(\mathbf{q}_a)$ is the inertia matrix, $C(\mathbf{q}_a, \dot{\mathbf{q}}_a)$ is the Coriolis- and centrifugal terms, $G(\mathbf{q}_a)$ is the gravitational torque and $\boldsymbol{\tau}_m^a = \begin{pmatrix}\tau_{m1}^a & \dots & \tau_{mn}^a\end{pmatrix}^\mathsf{T}$. Note that the equation is expressed on the arm side of the gearbox, that is, the applied motor torque τ_{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

$$\tau_{mi}^a = \tau_{mi}\eta_i. \tag{2.18}$$

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 friction torque $F(\dot{q}_i)$ for each joint. A classical model is

$$F(\dot{q}_i) = f_{vi}\dot{q}_i + f_{ci}\,\mathrm{sign}(\dot{q}_i), \tag{2.19}$$

where f_{vi} is the viscous friction and f_{ci} is the Coulomb friction for joint i. 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

$$F(\dot{q}_i) = f_{vi}\dot{q}_i + f_{ci}\left(\mu_{ki} + (1-\mu_{ki})\cosh^{-1}(\beta\dot{q}_i)\right)\tanh(\alpha_i\dot{q}_i), \tag{2.20}$$

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]; Carvalho Bittencourt and Gunnarsson [2012].

Flexible Joint Model

In practice, the joints, specially the gearboxes, are flexible. These flexibilities are distributed in nature but this can be simplified by considering a finite number of flexible modes. With a reasonable accuracy, it is possible to model each joint as a torsional spring and damper pair between the motor and arm side of the gearbox, see Figure 2.4. The system now has $2n$ DOF and can be described by the simplified flexible joint model

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

$$\mathbf{M}_m\ddot{\mathbf{q}}_m^a + F(\dot{\mathbf{q}}_m^a) = \boldsymbol{\tau}_m^a - T(\mathbf{q}_m^a - \mathbf{q}_a) - D(\dot{\mathbf{q}}_m^a - \dot{\mathbf{q}}_a), \tag{2.21b}$$

where $\mathbf{q}_a \in \mathbb{R}^n$ are the arm angles, $\mathbf{q}_m^a \in \mathbb{R}^n$ are the motor angles. The superscript a indicates that the motor angles are expressed on the arm side of the gearbox,

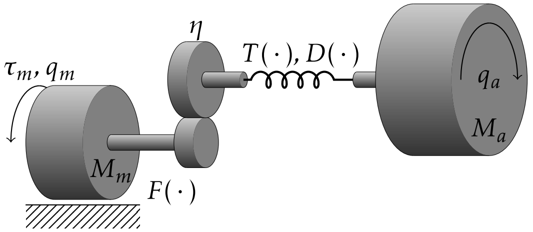


Figure 2.4: *Flexible joint model where the arm angular position q_a is related to the motor angular position q_m and motor torque τ_m via the gear ratio η and the spring-damper pair modelled by $T(\cdot)$ and $D(\cdot)$. The motor friction is modelled by $F(\cdot)$.*

i.e., $q^a_{mi} = q_{mi}/\eta_i$ where q_{mi} is the motor angle on the motor side of the gearbox for joint i. The same applies for the motor torque $\boldsymbol{\tau}^a_m$ according to (2.18). Furthermore, $M_a(\mathbf{q}_a)$ is the inertia matrix for the arms, $\mathbf{M}_m$ is the inertia for the motors, $C(\mathbf{q}_a, \dot{\mathbf{q}}_a)$ is the Coriolis- and centrifugal terms, $G(\mathbf{q}_a)$ is the gravitational torque and $F(\dot{\mathbf{q}}^a_m)$ is the friction torque. Moreover, $T(\mathbf{q}^a_m - \mathbf{q}_a)$ is the stiffness torque and $D(\dot{\mathbf{q}}^a_m - \dot{\mathbf{q}}_a)$ is the damping torque. Both the stiffness and damping torque can be modelled as linear or non-linear. 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 joint model the term $S(\mathbf{q}_a)\ddot{\mathbf{q}}^a_m$ is added to (2.21a) and the term $S^{\mathsf{T}}(\mathbf{q}_a)\ddot{\mathbf{q}}_a$ as well as a Coriolis- and centrifugal term are added to (2.21b), where $S(\mathbf{q}_a)$ is a strictly upper triangular matrix. A complete description of the simplified and complete flexible joint 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 et al. [2014].

2.3 Motion Control

Control of industrial manipulators is a challenging task. The robot is a strongly coupled multivariate flexible system with non-linear dynamics which changes all over the work space. In addition, other non-linearities such as hysteresis, backlash, friction, and joint flexibilities have to be taken care of. Furthermore, the controlled variable is not measured directly and available sensors only provide parts of the information important for control, e.g. the measured motor angu-

Table 2.1: *Performance for an ABB IRB6640 with a payload of 235 kg and a reach of 2.55 m at 1.6 m/s according to ISO 9283 [ABB Robotics, 2013].*

Pose accuracy	0.15 mm
Pose repeatability	0.05 mm
Settling time (within 0.4 mm)	0.19 s
Path accuracy	2.17 mm
Path repeatability	0.66 mm

lar positions do not contain information about the oscillation of the end-effector, hence good control of the TCP is difficult. The available measurements are also affected by uncertainties such as quantisation errors and measurement noise, as well as deterministic disturbances such as resolver ripple.

The requirements for controllers in modern industrial manipulators are that they should provide high performance, despite the flexible and non-linear mechanical structure, and at the same time, robustness to model uncertainties. Typical requirements for the motion controller are presented in Table 2.1. In the typical standard control configuration the actuator positions are the only measurements used in the higher level control loop. At a lower level, in the drive system, the currents and voltages in the motors are measured to provide torque control for the motors. In this thesis it is assumed that the lower level control loop is ideal, i.e., the control loop is significantly faster than the remaining system and the control performance is sufficiently good not to affect the performance of the higher level control loop. Adding extra sensors such as encoders measuring the joint angles and/or accelerometers measuring the acceleration of the end-effector are possible extensions to the control problem. The use of an accelerometer for a single flexible joint model, using $\mathcal{H}_\infty$-control methods, is investigated in Paper E.

In the literature, the three control structures i) independent joint control, ii) feed-forward control, and iii) feedback linearisation are common and they will be summarised in this section. Also, how the model complexity affects the control structure will be discussed. The survey Sage et al. [1999] and the references therein describe how various models and control structures are combined for robust control. The survey includes both rigid and flexible joint models with or without the actuator dynamics. The control structures are, among others, feedback linearisation, PID controllers, linear and non-linear $\mathcal{H}_\infty$ methods, passivity based controllers, and sliding mode controllers.

2.3.1 Independent Joint Control

In this control structure, each joint is, as the name suggests, controlled independently from the other joints using PID controllers. The influence from the other joints can be seen as disturbances. Usually the motor positions are measured and used in the feedback loop to calculate the motor torque for the corresponding joint. A common controller presented in the literature is the PD controller

$$\mathbf{u} = \mathbf{K}_P \cdot (\mathbf{q}_{d,m} - \mathbf{q}_m) + \mathbf{K}_D \cdot (\dot{\mathbf{q}}_{d,m} - \dot{\mathbf{q}}_m), \tag{2.22}$$

where $\mathbf{K}_P$ and $\mathbf{K}_D$ are diagonal matrices. Due to flexible joints, presented in Section 2.2.2, other sensors such as encoders, measuring the arm angles, can be used to improve the performance of the robot. However, using measurements from the arm side of the gearbox to control the motor torque on the motor side result in so called non-collocated control [Franklin et al., 2002]. Non-collocated control problems are difficult to stabilise as Example 2.1 shows.

Example 2.1: Non-collocated control

For a single flexible joint in a horizontal plane, i.e., no gravity, it is fairly simple to show that the motor velocity has to be present in the feedback loop in order to stabilise the system [De Luca and Book, 2008]. For simplicity, no damping and friction are present, which corresponds to the worst case. Let the dynamical model be described by

$$M_a\ddot{q}_a - K \cdot (q_m - q_a) = 0, \tag{2.23a}$$

$$M_m\ddot{q}_m + K \cdot (q_m - q_a) = \tau. \tag{2.23b}$$

Using Laplace transformation, gives the two transfer functions

$$Q_a(s) = \frac{K}{M_a M_m s^4 + (M_a + M_m)Ks^2}\mathcal{T}(s), \tag{2.24a}$$

$$Q_m(s) = \frac{M_a s^2 + K}{K} Q_a(s). \tag{2.24b}$$

Four feedback loops will be analysed using different combinations of the arm position and velocity and the motor position and velocity as measurements. It will be assumed, without loss of generality, that both the arm and motor velocity references are equal to zero. The controllers are:

1. feedback from arm position and arm velocity,

$$\tau = K_P \cdot (q_{a,d} - q_a) - K_D\dot{q}_a \tag{2.25a}$$

2. feedback from arm position and motor velocity,

$$\tau = K_P \cdot (q_{a,d} - q_a) - K_D\dot{q}_m \tag{2.25b}$$

3. feedback from motor position and arm velocity,

$$\tau = K_P \cdot (q_{m,d} - q_m) - K_D\dot{q}_a \tag{2.25c}$$

4. feedback from motor position and motor velocity,

$$\tau = K_P \cdot (q_{m,d} - q_m) - K_D\dot{q}_m \tag{2.25d}$$

The corresponding closed-loop transfer functions from $Q_{a,d}(s)$ to $Q_a(s)$ become

1.

$$\frac{KK_P}{M_aM_ms^4 + (M_a + M_m)Ks^2 + KK_Ds + KK_P} \tag{2.26a}$$

2.

$$\frac{KK_P}{M_aM_ms^4 + K_DM_as^3 + (M_a + M_m)Ks^2 + KK_Ds + KK_P} \tag{2.26b}$$

3.

$$\frac{KK_P}{M_aM_ms^4 + ((M_a + M_m)K + K_PM_a)s^2 + KK_Ds + KK_P} \tag{2.26c}$$

4.

$$\frac{KK_P}{M_aM_ms^4 + K_DM_as^3 + ((M_a + M_m)K + K_PM_a)s^2 + KK_Ds + KK_P} \tag{2.26d}$$

Using Routh's algorithm [Franklin et al., 2002] makes it possible to analyse the stability conditions for the four controllers. Routh's algorithm says that a system is stable if and only if all the elements in the first column of the Routh array are positive, see Franklin et al. [2002] for definition of the Routh array. The Routh arrays for the four controllers show that

1. feedback from arm position and arm velocity is unstable independent of the parameters K_P and K_D.
2. feedback from arm position and motor velocity is stable if $K_D > 0$ and $0 < K_P < K$.
3. feedback from motor position and arm velocity is unstable independent of the parameters K_P and K_D.
4. feedback from motor position and motor velocity is stable for all $K_P, K_D > 0$.

It means that if sensors on the arm side of the gearbox are introduced, it is still necessary to have the motor velocity included in the feedback loop in order to get a stable system. If damping and friction are introduced, then the system can be stabilised without need of the motor velocity but the stability margin can be very low.

For a robot with more than one joint it is more complicated to prove stability. To do so, Lyapunov stability and LaSalle's theorem [Khalil, 2002] have to be used. It can be shown that a rigid joint manipulator affected by gravity is stabilised by a PD controller. However, in order to have the joint angles to converge to the desired joint angles a gravity compensating term must be included in the controller, see Chung et al. [2008] for details. Similar proofs for flexible joint manipulators exist in e.g. De Luca and Book [2008].

In Tomei [1991] it is shown that a flexible joint robot can be robustly stabilised by a decentralised PD controller with motor positions as measurement. Robustness with respect to model parameters is also discussed. Moreover, Rocco [1996] presents stability results, including explicit stability regions, for a rigid joint manipulator using PID controllers.

2.3.2 Feed-forward Control

Independent joint control is insufficient for trajectory tracking. When the robot should follow a desired trajectory, restrictions on the path are required not to excite the flexibilities and a model-based controller is necessary. Introducing a feed-forward controller that takes the dynamics of the robot into account makes it possible to follow the desired trajectory without exciting the flexibilities. The feed-forward controller, for a rigid joint manipulator, takes the form

$$\mathbf{u}_{ff} = \widehat{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \widehat{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \widehat{G}(\mathbf{q}_d), \tag{2.27}$$

where $\widehat{M}$, $\widehat{C}$, and $\widehat{G}$ are the models used in the control system. Note that the feed-forward controller requires that the reference signal $\mathbf{q}_d$ is twice differentiable with respect to time. If the model is exact and no disturbances are present, then the feed-forward controller achieves perfect trajectory tracking. Model errors and disturbances require that independent joint control also has to be present, giving the total control signal as

$$\mathbf{u} = \underbrace{\widehat{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \widehat{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \widehat{G}(\mathbf{q}_d)}_{\text{feed-forward}} + \underbrace{\mathbf{K}_P \cdot (\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_D \cdot (\dot{\mathbf{q}}_d - \dot{\mathbf{q}})}_{\text{feedback}}. \tag{2.28}$$

The feed-forward controller design is a more difficult problem for flexible joint models. It depends on the complexity of the model, if for example the extended flexible joint model is used, then a high-index differential algebraic equation (DAE) has to be solved [Moberg and Hanssen, 2009].

2.3.3 Feedback Linearisation

Feedback linearisation [Khalil, 2002], also known as exact linearisation in the control literature, is a control method similar to feed-forward. Instead of having the dynamical model in feed-forward it is used in a static feedback loop to cancel the non-linear terms. For a rigid joint manipulator, the feedback is given by

$$\mathbf{u} = \widehat{M}(\mathbf{q})\mathbf{v} + \widehat{C}(\mathbf{q}, \dot{\mathbf{q}}) + \widehat{G}(\mathbf{q}) \tag{2.29}$$

where $\mathbf{q}$ and $\dot{\mathbf{q}}$ are the measured joint positions and velocities, and $\mathbf{v}$ is a signal to chose. In case of no model mismatch, the feedback signal (2.29) gives the decoupled system $\ddot{\mathbf{q}} = \mathbf{v}$. The signal $\mathbf{v}$ can be obtained as the output from e.g. a PD controller, i.e., independent joint control, according to

$$\mathbf{v} = \ddot{\mathbf{q}}_d + \mathbf{K}_P \cdot (\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_D \cdot (\dot{\mathbf{q}}_d - \dot{\mathbf{q}}). \tag{2.30}$$

The drawback is that it is computational demanding to calculate the inverse dynamical model in the feedback loop as well as the need of the full state trajectory. Robustness properties for the controller are discussed in Bascetta and Rocco [2010].

For flexible joint models, the problem gets more involved, as was the case for the feed-forward controller. In Spong [1987] it is shown that a static feedback loop can linearise and decouple the model in (2.21) when the friction and damping terms are excluded. The complete flexible joint model, i.e., when the matrix $S(\mathbf{q}_a)$ is included in (2.21), with the friction and damping terms excluded is proved to be linearisable using a dynamic non-linear feedback law in De Luca and Lanari [1995]. So far the damping and friction terms have been neglected. In De Luca et al. [2005] it is shown that it is possible to achieve input-output linearisation with a static or dynamic non-linear feedback loop. Input-output linearisation does not eliminate all the non-linearities, instead the so called zero-dynamics remains. However, the flexible joint model gives stable zero-dynamics, hence input-output linearisation can be used for the model.

3

Estimation Theory

DIFFERENT TECHNIQUES for non-linear estimation are presented in this chapter. The estimation problem for the discrete time non-linear state space model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k; \boldsymbol{\theta}), \tag{3.1a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k, \mathbf{v}_k; \boldsymbol{\theta}), \tag{3.1b}$$

is to find the state vector $\mathbf{x}_k \in \mathbb{R}^{n_x}$ at time k and the unknown model parameters $\boldsymbol{\theta}$ given the measurements $\mathbf{y}_i \in \mathbb{R}^{n_y}$ for $i = 1, \ldots, l$. Here, l can be smaller, larger or equal to k depending on the method that is used. In this work the focus is on non-linear models with additive process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ given by

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k; \boldsymbol{\theta}) + g(\mathbf{x}_k; \boldsymbol{\theta})\mathbf{w}_k, \tag{3.2a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k; \boldsymbol{\theta}) + \mathbf{v}_k, \tag{3.2b}$$

where the *probability density functions* (PDFs) for the process noise $p_{\mathbf{w}}(\mathbf{w}; \boldsymbol{\theta})$, and measurement noise $p_{\mathbf{v}}(\mathbf{v}; \boldsymbol{\theta})$, are known except for some unknown parameters.

The estimation problem can be divided into the filtering problem where only previous measurements up to the present time are available, i.e., $l = k$, see Section 3.1 and the smoothing problem where both previous and future measurements are available, i.e., $l > k$, see Section 3.2. For the case with estimation of the unknown parameters $\boldsymbol{\theta}$, the expectation maximisation algorithm can be used as described in Section 3.3.

3.1 The Filtering Problem

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

$$\mathbf{y}_{1:k} = \{\mathbf{y}_1, \dots, \mathbf{y}_k\}, \tag{3.3}$$

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

$$\mathbf{x}_{k+1} \sim p(\mathbf{x}_{k+1}|\mathbf{x}_k), \tag{3.4a}$$

$$\mathbf{y}_k \sim p(\mathbf{y}_k|\mathbf{x}_k), \tag{3.4b}$$

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

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

and the Markov property for the state space model,

$$p(\mathbf{x}_n|\mathbf{x}_1, \dots \mathbf{x}_{n-1}) = p(\mathbf{x}_n|\mathbf{x}_{n-1}), \tag{3.6}$$

repeatedly, the optimal solution for the Bayesian inference [Jazwinski, 1970] can be obtained according to

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})}, \tag{3.7a}$$

$$p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k}) = \int_{\mathbb{R}^{n_x}} p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k})\,\mathrm{d}\mathbf{x}_k, \tag{3.7b}$$

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

$$p(\mathbf{y}_k|\mathbf{y}_{1:k-1}) = \int_{\mathbb{R}^{n_x}} p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})\,\mathrm{d}\mathbf{x}_k. \tag{3.7c}$$

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 non-linear and non-Gaussian systems, the posterior density cannot 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 and the particle filter. Another approximative solution not considered here is the *unscented Kalman filter* (UKF) [Julier et al., 1995].

3.1.1 The Extended Kalman Filter

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 non-linear system equations around the previous estimate. The approximation is acceptable if the non-linearity is almost linear or if the *signal to noise* *ratio* (SNR) is high. The Taylor expansion requires derivatives of the non-linear system equations, which can be obtained by symbolic or numeric differentiation.

The process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ are assumed to be Gaussian with zero means and covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively. The time update, $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$, and the measurement update, $\hat{\mathbf{x}}_{k|k}$ and $\mathbf{P}_{k|k}$, for the EKF with the non-linear 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 $\hat{\mathbf{x}}_{k|k}$, $\mathbf{P}_{k|k}$, $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ denotes estimates of the state vector $\mathbf{x}$ and covariance matrix $\mathbf{P}$ 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 [Gustafsson, 2010].

Algorithm 1 The Extended Kalman Filter (EKF)

Initialisation

$$\hat{\mathbf{x}}_{0|0} = \mathbf{x}_0 \tag{3.8a}$$

$$\mathbf{P}_{0|0} = \mathbf{P}_0 \tag{3.8b}$$

Time update

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}) \tag{3.9a}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^{\mathsf{T}} + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^{\mathsf{T}} \tag{3.9b}$$

$$\mathbf{F}_{k-1} = \left.\frac{\partial f\,(\mathbf{x}, \mathbf{u}_{k-1})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}} \tag{3.9c}$$

$$\mathbf{G}_{k-1} = g(\hat{\mathbf{x}}_{k-1|k-1}) \tag{3.9d}$$

Measurement update

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}} + \mathbf{R}_k\right)^{-1} \tag{3.10a}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1}, \mathbf{u}_k)\right) \tag{3.10b}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{P}_{k|k-1} \tag{3.10c}$$

$$\mathbf{H}_k = \left.\frac{\partial h(\mathbf{x}, \mathbf{u}_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}} \tag{3.10d}$$

3.1.2 The Particle Filter

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 $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ by a large set of N particles $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N}$, where each particle has a 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 particle filter has in practice been inapplicable due to degeneracy of the particles. The problem was solved in Gordon et al. [1993] by introducing a resampling 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 $p^{\text{prop}}(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \mathbf{y}_{k+1})$ 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., $p(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)})$, is a good choice [Gordon et al., 1993]. Using the conditional prior, the weight update can be written as $w_k^{(i)} = w_{k-1}^{(i)} p(\mathbf{y}_k|\mathbf{x}_k^{(i)})$. The optimal proposal should be to use the conditional density $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)$ [Doucet et al., 2000]. The problem is that it is difficult to sample from it and also to calculate the weights. In Paper A the optimal proposal density, approximated by an EKF [Doucet et al., 2000; Gustafsson, 2010], is used for experimental data. The approximated proposal density can be written as

$$p^{\text{prop}}(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k) \approx \mathcal{N}\left(\mathbf{x}_k; f(\mathbf{x}_{k-1}^{(i)}) + \mathbf{K}_k^{(i)}(\mathbf{y}_k - \hat{\mathbf{y}}_k^{(i)}), \left(\mathbf{H}_k^{(i),\mathsf{T}} \mathbf{R}_k^{\dagger} \mathbf{H}_k^{(i)} + \mathbf{Q}_{k-1}^{\dagger}\right)^{\dagger}\right),$$

where † denotes the pseudo-inverse, and where

$$\mathbf{K}_k^{(i)} = \mathbf{Q}_{k-1} \mathbf{H}_k^{(i),\mathsf{T}} \left(\mathbf{H}_k^{(i)} \mathbf{Q}_{k-1} \mathbf{H}_k^{(i),\mathsf{T}} + \mathbf{R}_k\right)^{-1}, \tag{3.11a}$$

$$\mathbf{H}_k^{(i)} = \left.\frac{\partial h(\mathbf{x}_k)}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k = f(\mathbf{x}_{k-1}^{(i)})}, \tag{3.11b}$$

$$\hat{\mathbf{y}}_k^{(i)} = h(f(\mathbf{x}_{k-1}^{(i)})). \tag{3.11c}$$

The matrices in (3.11) are evaluated for each particle. The approximated optimal proposal density gives a weight update according to

$$w_k^{(i)} = w_{k-1}^{(i)} p(\mathbf{y}_k|\mathbf{x}_{k-1}^{(i)}), \tag{3.12a}$$

$$p(\mathbf{y}_k|\mathbf{x}_{k-1}^{(i)}) = \mathcal{N}\left(\mathbf{y}_k; \hat{\mathbf{y}}_k^{(i)}, \mathbf{H}_k^{(i)} \mathbf{Q}_{k-1} \mathbf{H}_k^{(i),\mathsf{T}} + \mathbf{R}_k\right). \tag{3.12b}$$

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

$$\hat{\mathbf{x}}_{k|k} = \underset{\hat{\mathbf{x}}_k}{\arg\min}\ \mathrm{E}\left[(\hat{\mathbf{x}}_k - \mathbf{x}_k)^{\mathsf{T}} (\hat{\mathbf{x}}_k - \mathbf{x}_k) | \mathbf{y}_{1:k}\right], \tag{3.13}$$

which has the solution

$$\hat{\mathbf{x}}_{k|k} = \mathrm{E}\left[\mathbf{x}_k|\mathbf{y}_{1:k}\right] = \int_{\mathbb{R}^{n_x}} \mathbf{x}_k p\left(\mathbf{x}_k|\mathbf{y}_{1:k}\right) \mathrm{d}\mathbf{x}_k \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{x}_k^{(i)}. \tag{3.14}$$

Algorithm 2 The Particle Filter (PF)

```
1: Generate $N$ samples $\{\mathbf{x}_0^{(i)}\}_{i=1}^N$ from $p(\mathbf{x}_0)$.
2: Compute
       $$w_k^{(i)} = w_{k-1}^{(i)} \frac{p(\mathbf{y}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{p^{\text{prop}}(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)}$$
   and normalise, i.e., $\bar{w}_k^{(i)} = w_k^{(i)} / \sum_{j=1}^N w_k^{(j)},\ i = 1, \ldots, N$.
3: [Optional]. Generate a new set $\{\mathbf{x}_k^{(i_\star)}\}_{i=1}^N$ by resampling with replacement $N$
   times from $\{\mathbf{x}_k^{(i)}\}_{i=1}^N$, with probability $\bar{w}_k^{(i)} = \Pr\{\mathbf{x}_k^{(i_\star)} = \mathbf{x}_k^{(i)}\}$ and let $w_k^{(i)} = 1/N$,
   $i = 1, \ldots, N$.
4: Generate predictions from the proposal density
       $$\mathbf{x}_{k+1}^{(i)} \sim p^{\text{prop}}(\mathbf{x}_{k+1}|\mathbf{x}_k^{(i_\star)}, \mathbf{y}_{k+1}),\ i = 1, \ldots, N.$$
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 $p(\mathbf{x}_k|\mathbf{y}_{1:l})$, 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 non-linear and non-Gaussian. Different types of smoothing problems are possible, e.g. fixed-lag, fixed-point and fixed-interval smoothing [Gustafsson, 2010]. In this thesis, 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 off-line method that uses all available measurements $\mathbf{y}_{1:N}$. 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 $\boldsymbol{\theta}$ such that the measurements $\mathbf{y}_{1:N}$ become as likely as possible. In other words,

$$\hat{\boldsymbol{\theta}}^{\text{ML}} = \arg\max_{\boldsymbol{\theta} \in \Theta} p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}), \tag{3.17}$$

where $p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N})$ is the PDF of the observations, i.e., the likelihood, parametrised with the parameter $\boldsymbol{\theta}$. Usually, the logarithm of the likelihood PDF,

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}) = \log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}), \tag{3.18}$$

Algorithm 3 The Extended Kalman Smoother (EKS)

```
1: Run the EKF and store the time and measurement updates, $\hat{\mathbf{x}}_{k|k-1}$, $\hat{\mathbf{x}}_{k|k}$, $\mathbf{P}_{k|k-1}$and $\mathbf{P}_{k|k}$.
2: Initiate the backward time recursion,
       $\hat{\mathbf{x}}^s_{N|N} = \hat{\mathbf{x}}_{N|N},$   (3.15a)
       $\mathbf{P}^s_{N|N} = \mathbf{P}_{N|N}.$   (3.15b)
3: Apply the backward time recursion for $k = N-1, \dots, 1$,
       $\hat{\mathbf{x}}^s_{k|N} = \hat{\mathbf{x}}_{k|k} + \mathbf{P}_{k|k}\mathbf{F}_k^{\mathsf{T}}\mathbf{P}^{-1}_{k+1|k}\left(\hat{\mathbf{x}}^s_{k+1|N} - \hat{\mathbf{x}}_{k+1|k}\right),$   (3.16a)
       $\mathbf{P}^s_{k|N} = \mathbf{P}_{k|k} + \mathbf{P}_{k|k}\mathbf{F}_k^{\mathsf{T}}\mathbf{P}^{-1}_{k+1|k}\left(\mathbf{P}^s_{k+1|N} - \mathbf{P}_{k+1|k}\right)\mathbf{P}^{-1}_{k+1|k}\mathbf{F}_k\mathbf{P}_{k|k},$   (3.16b)
       $\mathbf{F}_k = \left.\frac{\partial f(\mathbf{x}, \mathbf{u}_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k}}.$   (3.16c)
```

is used and then the problem is to find the parameter $\boldsymbol{\theta}$ that maximises (3.18). These two problems are equivalent since the logarithm is a monotonic function. The solution can still be hard to find which has lead to the development of the *expectation maximisation* (EM) algorithm [Dempster et al., 1977]. The EM algorithm is an ML estimator for models with latent variables. Let all of the the latent variables be denoted by Z.

Take now the joint log-likelihood function

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, Z) = \log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, Z) \tag{3.19}$$

of the observed variables $\mathbf{y}_{1:N}$ and the latent variables Z. The latent variables are unknown, hence the joint log-likelihood function cannot be calculated. Instead, the expected value of (3.19) given $\mathbf{y}_{1:N}$ has to be derived. The expected value is given by

$$\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) = \mathrm{E}_{\boldsymbol{\theta}_l}\left[\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, Z)|\mathbf{y}_{1:N}\right], \tag{3.20}$$

where $\mathrm{E}_{\boldsymbol{\theta}_l}[\,\cdot\,|\,\cdot\,]$ is the conditional mean with respect to a PDF defined by the parameter $\boldsymbol{\theta}_l$, and $p_{\boldsymbol{\theta}}(\,\cdot\,)$ means that the PDF is parametrised by $\boldsymbol{\theta}$. The sought parameter $\boldsymbol{\theta}$ is now given iteratively by

$$\boldsymbol{\theta}_{l+1} = \underset{\boldsymbol{\theta}\in\Theta}{\arg\max}\ \Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l). \tag{3.21}$$

The connection to the likelihood estimate $\hat{\boldsymbol{\theta}}^{\mathrm{ML}}$ comes from the fact that [Dempster et al., 1977] any $\boldsymbol{\theta}$, such that

$$\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) > \Gamma(\boldsymbol{\theta}_l; \boldsymbol{\theta}_l), \tag{3.22}$$

implies that

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}) > L_{\boldsymbol{\theta}_l}(\mathbf{y}_{1:N}). \tag{3.23}$$

Hence, maximising $\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l)$ provides a sequence $\boldsymbol{\theta}_l$, $l = 1, 2, \dots$, which approximates $\hat{\boldsymbol{\theta}}^{\mathrm{ML}}$ better and better for every iteration. The EM algorithm is summarised in Algorithm 4 and it can be stopped when

$$\left|L_{\boldsymbol{\theta}_{l+1}}(\mathbf{y}_{1:N}) - L_{\boldsymbol{\theta}_l}(\mathbf{y}_{1:N})\right| \leq \epsilon_L, \tag{3.24}$$

or when

$$\|\boldsymbol{\theta}_{l+1} - \boldsymbol{\theta}_l\| \leq \epsilon_\theta. \tag{3.25}$$

Example 3.1 shows how the EM algorithm can be used for identification of parameters in state space models.

Algorithm 4 The Expectation Maximisation (EM) Algorithm

```
1: Select an initial value $\boldsymbol{\theta}_0$ and set $l = 0$.
2: Expectation Step (E-step): Calculate
       $\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) = \mathrm{E}_{\boldsymbol{\theta}_l}\left[\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, Z) | \mathbf{y}_{1:N}\right].$
3: Maximisation Step (M-step): Compute
       $\boldsymbol{\theta}_{l+1} = \arg\max_{\boldsymbol{\theta} \in \Theta} \Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l).$
4: If converged, stop. If not, set $l = l + 1$ and go to step 2.
```

Example 3.1: The EM Algorithm for System Identification

System identification is about to determine the unknown model parameters $\boldsymbol{\theta}$ given observations $\mathbf{y}_{1:N}$. Here the EM algorithm will be used, where all details can be found in Gibson and Ninness [2005]. Consider the discrete-time state space model

$$x_{k+1} = a x_k + w_k, \tag{3.26a}$$

$$y_k = c x_k + v_k, \tag{3.26b}$$

where the process noise $w_k \sim \mathcal{N}(0, 0.1)$, the measurement noise $w_k \sim \mathcal{N}(0, 0.1)$ and the unknown parameters are $\boldsymbol{\theta} = \begin{pmatrix} a & c \end{pmatrix}^{\mathsf{T}}$. Following Algorithm 4 the first is to choose the latent variables Z and then derive $p_{\boldsymbol{\theta}}(y_{1:N}, Z)$. The latent variables are simply chosen as $Z = x_{1:N+1}$, and Bayes' rule together with the Markov property of the state space model (3.26) give

$$p_{\boldsymbol{\theta}}(y_{1:N}, x_{1:N+1}) = p_{\boldsymbol{\theta}}(x_1) \prod_{k=1}^{N} p_{\boldsymbol{\theta}}(x_{k+1}, y_k | x_k), \tag{3.27}$$

where

$$p_{\boldsymbol{\theta}}(\xi_k | x_k) \sim \mathcal{N}(\boldsymbol{\theta} x_k, \boldsymbol{\Pi}), \quad \xi_k = \begin{pmatrix} x_{k+1} \\ y_k \end{pmatrix}, \quad \boldsymbol{\Pi} = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}. \tag{3.28}$$

Using the definition of the normal distribution gives

$$\log p_{\boldsymbol{\theta}}(y_{1:N}, x_{1:N+1}) \propto \operatorname{tr} \boldsymbol{\Pi}^{-1} \left(\sum_{k=1}^{N} \xi_k x_k \right) \boldsymbol{\theta}^{\mathsf{T}} - \frac{1}{2} \operatorname{tr} \boldsymbol{\Pi}^{-1} \boldsymbol{\theta} \left(\sum_{k=1}^{N} x_k^2 \right) \boldsymbol{\theta}^{\mathsf{T}}, \tag{3.29}$$

Table 3.1: Estimated model parameters $\hat{a}$ and $\hat{c}$ in Example 3.1 averaged over 1000 MC simulations for different number of data points.

N	50	100	500	1000	1500	5000	10000
$\hat{a}$	0.2190	0.2479	0.2923	0.2874	0.2954	0.3002	0.3004
$\hat{c}$	0.5608	0.5788	0.5963	0.5965	0.5990	0.5980	0.6002

where all terms independent of $\boldsymbol{\theta}$ are omitted and tr is the trace operator. To obtain $\Gamma(\boldsymbol{\theta};\boldsymbol{\theta}_l)$ the expected value of $\log p_{\boldsymbol{\theta}}(y_{1:N}, x_{1:N+1})$ is calculated giving

$$\Gamma(\boldsymbol{\theta};\boldsymbol{\theta}_l) \propto \operatorname{tr} \boldsymbol{\Pi}^{-1}\left(\sum_{k=1}^{N} \mathrm{E}_{\boldsymbol{\theta}_l}\left[\xi_k x_k | y_{1:N}\right]\right)\boldsymbol{\theta}^{\mathsf{T}} - \frac{1}{2}\operatorname{tr} \boldsymbol{\Pi}^{-1}\boldsymbol{\theta}\left(\sum_{k=1}^{N} \mathrm{E}_{\boldsymbol{\theta}_l}\left[x_k^2 | y_{1:N}\right]\right)\boldsymbol{\theta}^{\mathsf{T}}.$$

Here, the terms involving the expected values can be calculated using a Kalman smoother, see Gibson and Ninness [2005] for details. Next step is to maximise $\Gamma(\boldsymbol{\theta};\boldsymbol{\theta}_l)$. If

$$\sum_{k=1}^{N} \mathrm{E}_{\boldsymbol{\theta}_l}\left[x_k^2 | y_{1:N}\right] > 0, \tag{3.30}$$

then the Hessian of $\Gamma(\boldsymbol{\theta};\boldsymbol{\theta}_l)$ is negative definite and the maximising argument can be obtained by taking the derivative of $\Gamma(\boldsymbol{\theta};\boldsymbol{\theta}_l)$, with respect to $\boldsymbol{\theta}$, equal to zero. Derivative rules for the trace operator can be found in Lütkepohl [1996]. The maximising argument is finally given by

$$\boldsymbol{\theta}_{l+1} = \mathrm{E}_{\boldsymbol{\theta}_l}\left[\sum_{k=1}^{N} \xi_k x_k \middle| y_{1:N}\right] \Big/ \mathrm{E}_{\boldsymbol{\theta}_l}\left[\sum_{k=1}^{N} x_k^2 \middle| \mathbf{y}_{1:N}\right]. \tag{3.31}$$

Identification of a and c has been performed for a simulated system, with the true values $a^* = 0.3$ and $c^* = 0.6$, for different values of the number of samples N and $\boldsymbol{\theta}_0 = \begin{pmatrix} 0.25 & 0.5 \end{pmatrix}^{\mathsf{T}}$. Table 3.1 shows the estimates $\hat{a}$ and $\hat{c}$ averaged over 1000 MC simulations, where it can be seen that the estimates approach the true values when the number of samples increases. Figure 3.1 shows how the estimates converge for 10 MC simulations using $N = 5000$.

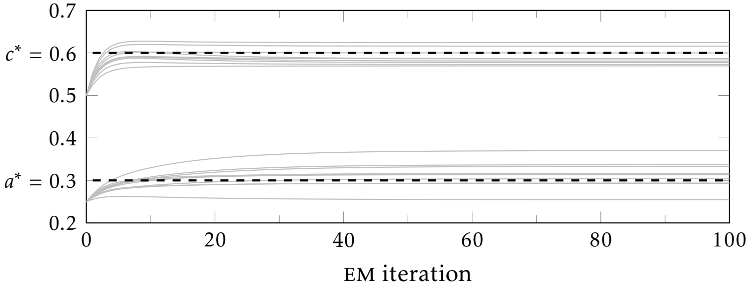


Figure 3.1: The estimate $\hat{\boldsymbol{\theta}}$ during 100 iterations for 10 MC simulations.

4 Control Theory

THREE DIFFERENT types of control methods, which are used in this thesis, will be introduced in this chapter. First, the general $\mathcal{H}_\infty$-control method and the loop shaping method, which are feedback controllers, are presented in Sections 4.1 and 4.2, respectively. Second, the idea of iterative learning control is explained in Section 4.3, with focus on the norm-optimal design method.

4.1 $\mathcal{H}_\infty$ Control

For design of $\mathcal{H}_\infty$ controllers the system

$$\begin{pmatrix} \mathbf{z} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{u} \end{pmatrix} = P(s) \begin{pmatrix} \mathbf{w} \\ \mathbf{u} \end{pmatrix} \tag{4.1}$$

is considered, where $\mathbf{w}$ are the exogenous input signals (disturbances and references), $\mathbf{u}$ is the control signal, $\mathbf{y}$ are the measurements and $\mathbf{z}$ are the exogenous output signals. Using a controller $\mathbf{u} = K(s)\mathbf{y}$, see Figure 4.1, the system from $\mathbf{w}$ to $\mathbf{z}$ can be written as

$$\mathbf{z} = \Big(P_{11}(s) + P_{12}(s)K(s)(\mathbf{I} - P_{22}(s)K(s))^{-1}P_{21}(s)\Big)\mathbf{w} = F_l(P,K)\mathbf{w}, \tag{4.2}$$

where $F_l(P,K)$ denotes the lower *linear fractional transformation* (LFT) [Skogestad and Postletwaite, 2005]. The $\mathcal{H}_\infty$ controller is the controller that minimises

$$\|F_l(P,K)\|_\infty = \max_\omega \bar{\sigma}\left(F_l(P,K)(i\omega)\right), \tag{4.3}$$

where $\bar{\sigma}(\,\cdot\,)$ denotes the maximal singular value. It is not always necessary and sometimes not even possible to find the optimal $\mathcal{H}_\infty$ controller. Instead, a subop-

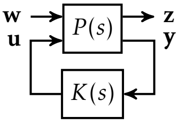


Figure 4.1: The system $P(s)$ in connection with the controller $K(s)$ symbolising the LFT *$F_l(P,K)$.*

timal controller is derived such that

$$\|F_l(P,K)\|_\infty < \gamma, \tag{4.4}$$

where γ can be reduced iteratively until a satisfactory controller is obtained. The aim is to get $\gamma \approx 1$, meaning that the disturbances are not magnified. A stabilising proper controller exists if a number of assumptions are fulfilled as discussed in Skogestad and Postletwaite [2005]. Furthermore, efficient iterative algorithms to find $K(s)$, such that (4.4) is fulfilled, exist [Skogestad and Postletwaite, 2005; Zhou et al., 1996]. Note that the resulting $\mathcal{H}_\infty$ controller has the same state dimension as P.

A common design method is to construct $P(s)$ by augmenting the original system $\mathbf{y} = G(s)\mathbf{u}$ with the weighting functions $W_\mathbf{u}(s)$, $W_S(s)$, and $W_T(s)$, see Figure 4.2, such that the system $F_l(P,K)$ can be written as

$$F_l(P,K) = \begin{pmatrix} W_\mathbf{u}(s)G_\mathbf{wu}(s) \\ -W_T(s)T(s) \\ W_S(s)S(s) \end{pmatrix}, \tag{4.5}$$

where $S(s) = (\mathbf{I} + G(s)K(s))^{-1}$ is the sensitivity function, $T(s) = \mathbf{I} - S(s)$ is the complementary sensitivity function, and $G_\mathbf{wu}(s) = -K(s)(\mathbf{I} + G(s)K(s))^{-1}$ is the transfer function from $\mathbf{w}$ to $\mathbf{u}$. Equations (4.4) and (4.5) give

$$\bar{\sigma}\left(W_\mathbf{u}(i\omega)G_\mathbf{wu}(i\omega)\right) < \gamma,\ \forall\omega, \tag{4.6a}$$

$$\bar{\sigma}\left(W_T(i\omega)T(i\omega)\right) < \gamma,\ \forall\omega, \tag{4.6b}$$

$$\bar{\sigma}\left(W_S(i\omega)S(i\omega)\right) < \gamma,\ \forall\omega. \tag{4.6c}$$

The transfers functions $G_\mathbf{wu}(s)$, $S(s)$, and $T(s)$ can now be shaped to satisfy the requirements by choosing the weighting functions $W_\mathbf{u}(s)$, $W_S(s)$, and $W_T(s)$. The aim is to get a value of γ close to 1, which in general is hard to achieve and it requires good insight in the design method as well as the system dynamics. Note that the design process is a trade-off between the requirements of $S(s)$ and $T(s)$ since $S(s) + T(s) = \mathbf{I}$. For example, both $S(s)$ and $T(s)$ cannot be small at the same frequency range. For more details about the design method, see e.g. Skogestad and Postletwaite [2005]; Zhou et al. [1996].

4.1.1 Mixed-$\mathcal{H}_\infty$ Control

The mixed-$\mathcal{H}_\infty$ controller design [Pipeleers and Swevers, 2013; Zavari et al., 2012] is a modification of the standard $\mathcal{H}_\infty$-design method. Instead of choosing the

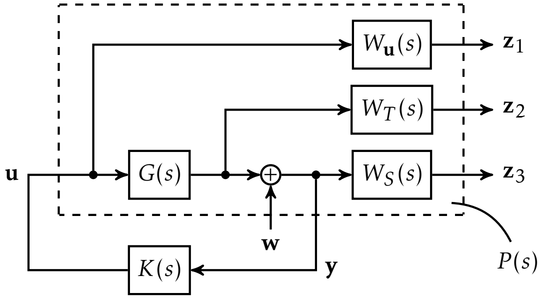


Figure 4.2: *The original system $G(s)$ augmented with the weighting functions $W_{\mathbf{u}}(s)$, $W_S(s)$, and $W_T(s)$, giving the system $P(s)$. Moreover, the system $P(s)$ is in connection with the controller $K(s)$.*

weighting functions in (4.5) such that the $\mathcal{H}_\infty$-norm of all weighted transfer functions satisfies (4.6), the modified method divides the problem into design constraints and design objectives. The controller can now be found as the solution to

$$\begin{aligned} &\underset{K(s)}{\text{minimise}} && \gamma && (4.7\text{a}) \\ &\text{subject to} && \|W_P(s)S(s)\|_\infty < \gamma && (4.7\text{b}) \\ & && \|M_S(s)S(s)\|_\infty < 1 && (4.7\text{c}) \\ & && \|W_{\mathbf{u}}(s)G_{\mathbf{wu}}(s)\|_\infty < 1 && (4.7\text{d}) \\ & && \|W_T(s)T(s)\|_\infty < 1 && (4.7\text{e}) \end{aligned}$$

where γ not necessarily has to be close to 1. Here, the weighting function $W_S(s)$ has been replaced with $M_S(s)$ and $W_P(s)$ to separate the design constraints and the design objectives. The method can be generalised to other control structures and in its general form formulated as a multi-objective optimisation problem. More details about the general form and how to solve the optimisation problem are presented in Pipeleers and Swevers [2013]; Zavari et al. [2012].

4.2 Loop Shaping

The loop shaping method was first presented in McFarlane and Glover [1992] and is based on robust stabilisation of a normalised coprime factorisation of the system as described in Glover and McFarlane [1989]. Robust stabilisation of a normalised coprime factorisation proceeds as follows. Let the system $G(s)$ be described by its left coprime factorisation $G(s) = M(s)^{-1}N(s)$, where $M(s)$ and $N(s)$ are stable transfer functions. The set of perturbed plants

$$G_p(s) = \left\{ (M(s) + \Delta_M(s))^{-1}(N(s) + \Delta_N(s)) : \left\| \begin{pmatrix} \Delta_N(s) & \Delta_M(s) \end{pmatrix} \right\|_\infty < \epsilon \right\}, \qquad (4.8)$$

where $\Delta_M(s)$, and $\Delta_N(s)$ are stable unknown transfer functions representing the uncertainties and ϵ is the stability margin, is robustly stabilised by the controller

$K(s)$ if and only if the nominal feedback system is stable and

$$\left\| \begin{pmatrix} K(s) \\ \mathbf{I} \end{pmatrix} (\mathbf{I} - G(s)K(s))^{-1} M(s)^{-1} \right\|_{\infty} \leq \frac{1}{\epsilon}. \tag{4.9}$$

Given the state space model

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \tag{4.10a}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t), \tag{4.10b}$$

then the maximal stability margin is given by

$$\frac{1}{\epsilon_{\max}} = \sqrt{1 + \rho(\mathbf{XY})} \triangleq \gamma_{\min}, \tag{4.11}$$

where $\rho(\,\cdot\,)$ is the spectral radius. For a strictly proper model, i.e., $\mathbf{D} = \mathbf{0}$, the two matrices $\mathbf{X}$ and $\mathbf{Y}$ are the unique and positive definite solutions to the algebraic Riccati equations

$$\mathbf{AY} + \mathbf{YA}^{\mathsf{T}} - \mathbf{YC}^{\mathsf{T}}\mathbf{CY} + \mathbf{BB}^{\mathsf{T}} = 0, \tag{4.12a}$$

$$\mathbf{XA} + \mathbf{A}^{\mathsf{T}}\mathbf{X} - \mathbf{XBB}^{\mathsf{T}}\mathbf{X} + \mathbf{C}^{\mathsf{T}}\mathbf{C} = 0. \tag{4.12b}$$

The two Riccati equations have a unique solution if the state space realisation in (4.10) is minimal. A non-proper model, i.e., $\mathbf{D} \neq \mathbf{0}$, also has an explicit solution but the corresponding Riccati equations are more extensive [Skogestad and Postletwaite, 2005].

Given $\gamma > \gamma_{\min}$, then a controller in closed form is given by

$$K(s) = \left(\begin{array}{c|c} \mathbf{A} - \mathbf{BB}^{\mathsf{T}}\mathbf{X} + \gamma^2\mathbf{L}^{-\mathsf{T}}\mathbf{YC}^{\mathsf{T}}\mathbf{C} & \gamma^2\mathbf{L}^{-\mathsf{T}}\mathbf{YC}^{\mathsf{T}} \\ \hline \mathbf{B}^{\mathsf{T}}\mathbf{X} & \mathbf{0} \end{array} \right), \tag{4.13a}$$

$$\mathbf{L} = (1 - \gamma^2)\mathbf{I} + \mathbf{XY}. \tag{4.13b}$$

From the definition of $\gamma_{\min}$ in (4.11) it holds that $\mathbf{L}$ becomes singular if $\gamma = \gamma_{\min}$, hence the controller $K(s)$ cannot be obtained. Implementing the solution in software is not difficult, however the MATLAB function `ncfsyn`, included in the Robust Control Toolbox , is a good alternative to use.

In practice a value $\gamma < 4$ is to aim for, otherwise the robustness properties become too poor. Having $\gamma < 4$ corresponds to $\epsilon > 0.25$ which means that a coprime uncertainty of at least 25 % is achieved. For a SISO system, the stability margin $\epsilon = 0.25$ corresponds to 4.4 dB gain margin and 29° phase margin.

For loop shaping [McFarlane and Glover, 1992], the system $G(s)$ is first pre- and post-multiplied with weighting functions $W_1(s)$ and $W_2(s)$, see Figure 4.3, such that the shaped system $G_s(s) = W_2(s)G(s)W_1(s)$ has desired properties. The desired properties are usually requirements on the cut-off frequency, the low frequency gain, and the phase margin. The controller $K_s(s)$ is then obtained using (4.13) applied on the system $G_s(s)$. The final controller $K(s)$ is given by

$$K(s) = W_1(s)K_s(s)W_2(s). \tag{4.14}$$

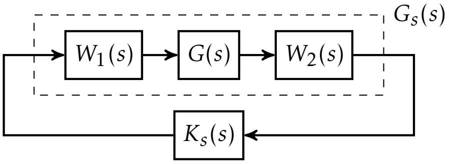


Figure 4.3: *Block diagram of the loop shaping procedure. The system $G(s)$ is pre- and post-multiplied with the weighting functions $W_1(s)$ and $W_2(s)$. The controller $K_s(s)$ is obtained from* (4.13) *using the shaped system $G_s(s)$.*

Note that the structure in Figure 4.3 for loop shaping can be rewritten as a standard $\mathcal{H}_\infty$ problem according to Figure 4.1, see Zhou et al. [1996] for details, and the methods in Section 4.1 can be used for controller synthesis.

Loop shaping is a simple to use method which does not require any problem dependent uncertainty model. Instead, the normalised coprime factorisation and the perturbed plant in (4.8) give a quite general uncertainty description. Compared to the $\mathcal{H}_\infty$ method described in Section 4.1, the solution does not require any γ iterations. The choice of the weighting functions $W_1(s)$ and $W_2(s)$ requires good understanding of the system to be controlled. The following working progress can be useful

1. Scale $G(s)$ in order to improve conditioning of the system. Also, reordering the inputs and outputs such that the system is as diagonal as possible is to prefer. How to do the reordering can be obtained using the *relative gain array* (RGA) [Skogestad and Postletwaite, 2005].

2. Choose $W_1(s)$ and $W_2(s)$. In Skogestad and Postletwaite [2005] a detailed description of how to chose the weighting functions is presented where $W_1(s)$ is composed as the product of several matrices which does different work, such as shape of the singular values, alignment of the singular values at the desired bandwidth, and control over the actuator usage.

3. Calculate $K_s(s)$ using (4.13). Modify the weighting functions $W_1(s)$ and $W_2(s)$ until $\gamma < 4$.

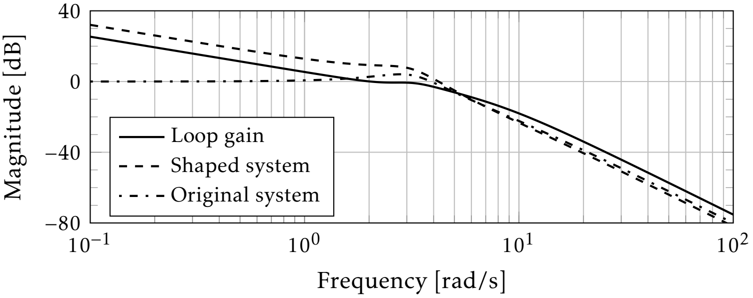
4. Calculate the controller $K(s)$ using (4.14), and test the performance. Modify the weighting functions until the desired performance is achieved.

More details about the robust stabilisation of the normalised coprime factorisation, as well as the loop shaping synthesis can be found in e.g. Skogestad and Postletwaite [2005]; Zhou et al. [1996].

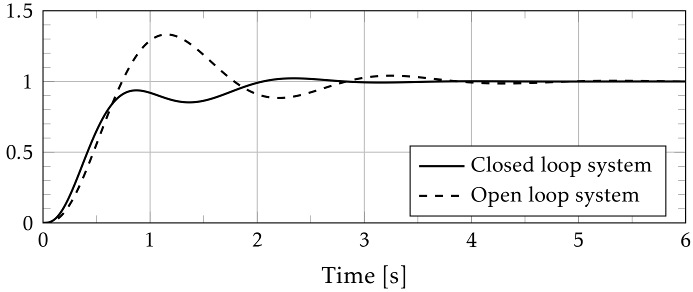
Example 4.1: Loop shaping for a SISO system

Let the nominal SISO system be given by

$$G(s) = \frac{100}{s^3 + 12s^2 + 30s + 100}. \tag{4.15}$$



***(a)** The Bode diagrams for the loop gain $K(s)G(s)$, the shaped system $G_s(s)$, and the original system $G(s)$.*



***(b)** Step response for the open loop system and the closed loop system.*

***Figure 4.4:** Results for Example 4.1.*

Design a controller using loop shaping that attenuates the oscillations without changing the cut-off frequency.

$G(s)$ is a SISO system, hence it is possible to let $W_2(s) = 1$ and only focus on $W_1(s)$. Integral action, i.e., a pole in the origin for $W_1(s)$, is added to get an improvement of the low frequency performance. Moreover, to improve the phase margin, a zero in $s = -5$ is added to achieve a slope of -1, instead of -2, at the cut-off frequency for the loop gain. The desired cut-off frequency is then achieved by multiplying with 0.8. The weighting functions are finally given as

$$W_1(s) = 0.8\frac{s+5}{s}, \quad W_2(s) = 1. \tag{4.16}$$

The resulting controller, using `ncfsyn`, gives $\gamma = 2.38$, hence a stability margin of 42 % is achieved. This corresponds to 7.8 dB gain margin and 50° phase margin. The Bode diagrams of the shaped system $G_s(s)$, the loop gain $G(s)K(s)$, and the original system $G(s)$ are shown in Figure 4.4a. Step responses of the open loop system and the closed loop system are presented in Figure 4.4b. It can be seen that the controller attenuates the oscillations and keeps the time constant of the open loop system.

4.3 Iterative Learning Control

Iterative learning control (ILC) is a way to improve the performance of systems that perform the same task repeatedly. An industrial manipulator performing arc welding or laser cutting is a good example of such a system. The performance of the system is improved by adding a correction signal, calculated off-line, to the system. At each iteration the correction signal is updated using the correction signal and the control error, both from previous iteration. The ILC concept has been used, to some extent, from the beginning of the 1970s. However, it was not until 1984, when the three articles Arimoto et al. [1984a], Casalino and Bartolini [1984] and Craig [1984] were published, independently from each other, that ILC became a widespread idea. The name iterative learning control originates from the article Arimoto et al. [1984b]. The following six assumptions are essential for the concept of ILC [Arimoto, 1990]

1. Every iteration consists of a fixed number of samples.
2. A desired output $\mathbf{r}(t)$ is given a priori.
3. The system is initialised at the beginning of each iteration with the same settings, i.e., $\mathbf{x}_k(0) = \mathbf{x}_0$ for all k.
4. The system dynamics are invariant in the iteration domain.
5. The controlled variable $\mathbf{z}_k(t)$ can be measured and used in the ILC algorithm.
6. The system dynamics are invertible, i.e., the desired output $\mathbf{r}(t)$ corresponds to a unique input $\mathbf{u}_d(t)$.

In practice, it is not possible to initialise the system at the beginning of each iteration with exact same settings. The system is, most probably, also affected with input disturbances, induced during the iterations, and measurement noise. Therefore, assumptions 3, 4, and 5 are relaxed and replaced with [Arimoto, 1998]

3'. The system is initialised at the beginning of each iteration in a neighbourhood of $\mathbf{x}_0$, i.e., $\|\mathbf{x}_k(0) - \mathbf{x}_0\| \leq \epsilon_1$ for all k.

4'. The norm of the input disturbances, induced during the iterations, is bounded.

5'. The controlled variable $\mathbf{z}_k(t)$ can be measured with bounded noise and used in the ILC algorithm.

Assumption 5 can be relaxed even more to the case where the controlled variable $\mathbf{z}_k(t)$ is not directly measured. Instead, the measurements $\mathbf{y}_k(t)$ are used in an estimation algorithm in order to estimate the controlled variable, which can be used in the ILC algorithm. The idea of estimation-based ILC has been investigated in Wallén et al. [2009], Wallén et al. [2011a], and Wallén et al. [2011b]. Paper G deals with estimation-based ILC in combination with norm-optimal ILC.

Over the years, several surveys of the field of ILC have been published, e.g. Moore [1998]; Ahn et al. [2007]; Bristow et al. [2006]. Moore [1998] covers the published literature up to 1998 and Ahn et al. [2007] continues from 1998 to 2004, with more than 500 references.

4.3.1 System Description

Before discussing different ILC algorithms, as well as convergence and stability properties, it is of interest to introduce a description of the system. The batch formulations of the system dynamics and the ILC algorithm are very useful for the ILC concept. Especially, the stability and convergence properties are simple to show using the batch formulation. Design of ILC algorithms, such as the norm-optimal ILC method, can also be performed using the batch formulation. Here, only the time domain description, including the batch formulation, is presented but frequency domain descriptions are sometimes used as well [Norrlöf and Gunnarsson, 2002].

A general system is shown in Figure 4.5 with reference $\mathbf{r}(t)$, ILC signal $\mathbf{u}_k(t)$, process disturbance $\mathbf{w}_k(t)$, and measurement disturbance $\mathbf{v}_k(t)$ as inputs. There are two types of output signals from the system, $\mathbf{z}_k(t)$ denotes the controlled variable and $\mathbf{y}_k(t)$ the measured variable. All signals are at ILC iteration k and time t. The system is not restricted to open loop systems. It is possible to design an ILC algorithm for systems with an existing feedback loop. For example, this is important for industrial manipulators, since it is not desirable to remove the feedback loops which are stabilising the manipulator and have good disturbance rejection. Instead, the ILC signal should only catch the small variations in the error to improve the performance. The system is commonly described in discrete time by

$$\mathbf{z}_k(t) = S_{\mathbf{r}}(q)\mathbf{r}(t) + S_{\mathbf{u}}(q)\mathbf{u}_k(t) + S_{\mathbf{w}}(q)\mathbf{w}_k(t), \tag{4.17a}$$

$$\mathbf{y}_k(t) = \mathbf{z}_k(t) + S_{\mathbf{v}}(q)\mathbf{v}_k(t), \tag{4.17b}$$

where $S_{\mathbf{r}}(q)$, $S_{\mathbf{u}}(q)$, $S_{\mathbf{v}}(q)$, and $S_{\mathbf{w}}(q)$ are discrete-time transfer functions relating the input signals to the output signals. The assumption here is that the controlled signal is measured with noise. An extension to this assumption would be to have dynamical couplings between $\mathbf{z}_k(t)$ and $\mathbf{y}_k(t)$, which is a more realistic model.

For the batch formulation, also known as *lifted system representation*, let

$$\bar{\mathbf{r}} = \begin{pmatrix} \mathbf{r}(0) \\ \vdots \\ \mathbf{r}(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_z}, \quad \bar{\mathbf{u}}_k = \begin{pmatrix} \mathbf{u}_k(0) \\ \vdots \\ \mathbf{u}_k(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_u}, \tag{4.18a}$$

$$\bar{\mathbf{v}}_k = \begin{pmatrix} \mathbf{v}_k(0) \\ \vdots \\ \mathbf{v}_k(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_v}, \quad \bar{\mathbf{w}}_k = \begin{pmatrix} \mathbf{w}_k(0) \\ \vdots \\ \mathbf{w}_k(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_w}, \tag{4.18b}$$

$$\bar{\mathbf{z}}_k = \begin{pmatrix} \mathbf{z}_k(0) \\ \vdots \\ \mathbf{z}_k(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_z}, \quad \bar{\mathbf{y}}_k = \begin{pmatrix} \mathbf{y}_k(0) \\ \vdots \\ \mathbf{y}_k(N-1) \end{pmatrix} \in \mathbb{R}^{Nn_y}, \tag{4.18c}$$

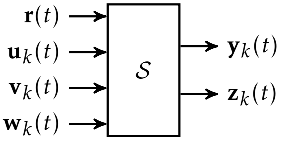


Figure 4.5: *System $\mathcal{S}$ with reference $\mathbf{r}(t)$, ILC input $\mathbf{u}_k(t)$, process disturbance $\mathbf{w}_k(t)$, measurement disturbance $\mathbf{v}_k(t)$, measured variable $\mathbf{y}_k(t)$ and controlled variable $\mathbf{z}_k(t)$ at ILC iteration k and time t.*

which are known as super-vectors. The system (4.17) can then be written as

$$\overline{\mathbf{z}}_k = \mathbf{S}_{\mathbf{r}}\overline{\mathbf{r}} + \mathbf{S}_{\mathbf{u}}\overline{\mathbf{u}}_k + \mathbf{S}_{\mathbf{w}}\overline{\mathbf{w}}_k, \tag{4.19a}$$

$$\overline{\mathbf{y}}_k = \overline{\mathbf{z}}_k + \mathbf{S}_{\mathbf{v}}\overline{\mathbf{v}}_k, \tag{4.19b}$$

where the matrices $\mathbf{S}_{\mathbf{r}}$, $\mathbf{S}_{\mathbf{u}}$, $\mathbf{S}_{\mathbf{v}}$, and $\mathbf{S}_{\mathbf{w}}$ are Toeplitz matrices formed from the impulse response coefficients for the corresponding transfer functions.

4.3.2 ILC Algorithms

A first order ILC algorithm can be formulated as

$$\mathbf{u}_{k+1}(t) = \mathcal{F}\left(\{\mathbf{u}_k(i)\}_{i=0}^{N-1}, \{\mathbf{e}_k(i)\}_{i=0}^{N-1}\right), \quad t = 0, \ldots, N-1, \tag{4.20}$$

where $\mathcal{F}(\,\cdot\,)$ is a function that can be either linear or non-linear, and $\mathbf{e}_k(t) = \mathbf{r}(t) - \mathbf{y}_k(t)$ is the tracking error, where $\mathbf{r}(t)$ is the reference and $\mathbf{y}_k(t)$ is the measured signal. The main objective of ILC is to determine the function $\mathcal{F}(\,\cdot\,)$ such that the tracking error converges to zero, i.e.,

$$\lim_{k\to\infty} \|\mathbf{e}_k(t)\|_2 = 0, \quad t = 0, \ldots, N-1, \tag{4.21}$$

in as few iterations as possible. Since information from previous iterations, at all time instances, is used, it is possible to use non-causal filtering. In practice, it is not possible to make the error vanish completely for some systems. In literature, the reasons are mostly said to be due to disturbances, and if a model based ILC algorithm has been used, also model errors affect the result. However, it can also be a fundamental problem with the ILC concept that the error cannot vanish completely. In Paper H the aspect of controllability, along the iterations, is considered and analysed. Convergence and stability of the ILC algorithm (4.20) are important properties to be able to guarantee that the sequence of signals $\mathbf{u}_k(t)$, $k = 0, 1, \ldots$ converges to a bounded correction signal, giving as low control error as possible. Convergence and stability properties for the update of the correction signal are presented in Section 4.3.3.

An update of the correction signal $\mathbf{u}_k(t)$ according to (4.20) is known as a first order ILC algorithm. Higher orders of ILC algorithms in the form

$$\mathbf{u}_{k+1}(t) = \mathcal{F}\left(\{\mathbf{u}_k(i)\}_{i=0}^{N-1}, \{\mathbf{u}_{k-1}(i)\}_{i=0}^{N-1}, \ldots, \{\mathbf{e}_k(i)\}_{i=0}^{N-1}, \{\mathbf{e}_{k-1}(i)\}_{i=0}^{N-1}, \ldots\right), \tag{4.22}$$

for $t = 0, \ldots, N-1$ can also be possible to use. See for example Bien and Huh

[1989]; Moore and YangQuan [2002]; Gunnarsson and Norrlöf [2006].

A widely used ILC algorithm, here shown for SISO systems, is

$$u_{k+1}(t) = \mathcal{Q}(q)(u_k(t) + \mathcal{L}(q)e_k(t)), \tag{4.23}$$

where q is the time-shift operator, t is time and k is the ILC iteration index. The filters $\mathcal{Q}(q)$ and $\mathcal{L}(q)$ are linear, possible non-causal, filters. Letting $\mathcal{Q}(q) = 1$ and taking $\mathcal{L}(q)$ as the inverse of the system, makes the error converge to zero in one iteration. Inverting a system can, if possible, be very complicated. Even for linear systems there exist difficulties. There are in principal three cases for linear systems where an inverse does not exist or it has undesirable properties. First, if the system has more poles than zeros, i.e., the system is strictly proper, then the inverse cannot be realised. Second, a non-minimum phase system results in an unstable inverse. Finally, systems with time delays are not possible to invert. Instead of inverting the system it is common to choose $\mathcal{L}(q) = \gamma q^{\delta}$, where $0 < \gamma < 1$ and δ a positive integer, are the design variables. The parameters γ and δ are chosen such that the stability criteria, see Section 4.3.3, is satisfied. Usually, δ is chosen as the time delay of the system. The filter $\mathcal{Q}(q)$ is introduced in order to restrict the high frequency influence from the error. However, including $\mathcal{Q}(q)$ makes the ILC algorithm converging slower and to a non zero error. It is usually enough to choose $\mathcal{Q}(q)$ as a simple low-pass filter of low order. The cut-off frequency is chosen such that the bandwidth of the ILC algorithm is large enough without letting through too much noise. In the design of the filters, there is a trade-off between convergence speed, error reduction and plant knowledge used in the design process.

Using the ILC algorithm (4.23) together with the system (4.17), where the disturbances are omitted, gives the ILC system

$$u_{k+1}(t) = \mathcal{Q}(q)\Big(1 - \mathcal{L}(q)S_u(q)\Big)u_k(t) + \mathcal{Q}(q)\mathcal{L}(q)\Big(1 - S_r(q)\Big)r(t). \tag{4.24}$$

The batch form of the ILC algorithm (4.23) is given by

$$\overline{\mathbf{u}}_{k+1} = \boldsymbol{\mathcal{Q}}(\overline{\mathbf{u}}_k + \boldsymbol{\mathcal{L}}\overline{\mathbf{e}}_k), \tag{4.25}$$

where

$$\overline{\mathbf{e}}_k = \overline{\mathbf{r}} - \overline{\mathbf{y}}_k = \begin{pmatrix} e_k(0) \\ \vdots \\ e_k(N-1) \end{pmatrix}. \tag{4.26}$$

The matrices $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{L}}$ in (4.25) can be formed from the impulse response coefficients of the filters in (4.23), or be the design variables directly. Together with the system (4.19), where the disturbances are omitted, gives the ILC system in batch form as

$$\overline{\mathbf{u}}_{k+1} = \boldsymbol{\mathcal{Q}}(\mathbf{I} - \boldsymbol{\mathcal{L}}\mathbf{S}_{\mathbf{u}})\overline{\mathbf{u}}_k + \boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{L}}(\mathbf{I} - \mathbf{S}_{\mathbf{r}})\overline{\mathbf{r}}. \tag{4.27}$$

Other choices of ILC algorithms, such as the PD- and PI-type, among others, can be found in e.g. Arimoto [1990]; Moore [1993]; Bien and Xu [1998]; Bristow et al.

[2006]. In this thesis, the norm-optimal ILC algorithm [Amann et al., 1996a,b; Gunnarsson and Norrlöf, 2001] is considered in Paper G. A short introduction to norm-optimal ILC is given below.

Norm-optimal ILC

In this section, the norm-optimal ILC algorithm [Amann et al., 1996a,b; Gunnarsson and Norrlöf, 2001] for the case $\mathbf{y}_k(t) = \mathbf{z}_k(t)$, i.e., with the noise term in (4.17) omitted, will be explained. Norm-optimal ILC using an estimate of the controlled variable is considered in Paper G. The method solves

$$\begin{aligned} &\underset{\mathbf{u}_{k+1}(\cdot)}{\text{minimise}} && \frac{1}{2}\sum_{t=0}^{N-1} \|\mathbf{e}_{k+1}(t)\|^2_{\mathbf{W_e}} + \|\mathbf{u}_{k+1}(t)\|^2_{\mathbf{W_u}} \\ &\text{subject to} && \frac{1}{2}\sum_{t=0}^{N-1} \|\mathbf{u}_{k+1}(t) - \mathbf{u}_k(t)\|^2 \leq \delta, \end{aligned} \tag{4.28}$$

where $\mathbf{e}_{k+1}(t) = \mathbf{r}(t) - \mathbf{z}_{k+1}(t)$ is the error, $\mathbf{W_e} \in \mathbb{S}^{n_z}_{++}$, and $\mathbf{W_u} \in \mathbb{S}^{n_u}_{++}$ are weight matrices for the error and the ILC control signal, respectively. Let the Lagrangian function [Nocedal and Wright, 2006] be defined by

$$\mathfrak{L}(\mathbf{u}_{k+1}(t), \lambda) \triangleq \frac{1}{2}\sum_{t=0}^{N-1}\Big[\|\mathbf{e}_{k+1}(t)\|^2_{\mathbf{W_e}} + \|\mathbf{u}_{k+1}(t)\|^2_{\mathbf{W_u}} + \lambda\,\|\mathbf{u}_{k+1}(t) - \mathbf{u}_k(t)\|^2\Big] - \lambda\delta,$$

where $\lambda \in \mathbb{R}_+$ is the Lagrange multiplier. The first-order sufficient conditions for optimality are given by the *Karush-Kuhn-Tucker* (KKT) conditions [Nocedal and Wright, 2006]. The solution can now be found using the KKT conditions for a fixed value of δ giving $\mathbf{u}^*_{k+1}(t)$ and λ^*. However, there is no predetermined value of δ, hence δ will be a tuning parameter. The solution procedure can be simplified if it is assumed that λ is a tuning parameter instead of δ. The solution becomes simpler to obtain since there is no need to calculate an optimal value for λ. The KKT conditions implies now that the constraint in (4.28) will always be satisfied with equality and where the value of δ depends indirectly of the value of λ. Moreover, the value of δ will decrease when number of iterations increase. Finally, the optimum is obtained where the gradient of the Lagrangian function with respect to $\mathbf{u}_{k+1}(t)$ equals zero. The second-order sufficient condition for optimality is that the Hessian of the Lagrangian function with respect to $\mathbf{u}_{k+1}(t)$ is greater than zero.

Using the super-vectors $\overline{\mathbf{u}}_k$, $\overline{\mathbf{z}}_k$, and $\overline{\mathbf{r}}$ from (4.18) gives

$$\mathfrak{L} = \frac{1}{2}\Big[\overline{\mathbf{e}}^{\mathsf{T}}_{k+1}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\overline{\mathbf{e}}_{k+1} + \overline{\mathbf{u}}^{\mathsf{T}}_{k+1}\boldsymbol{\mathcal{W}}_{\mathbf{u}}\overline{\mathbf{u}}_{k+1} + \lambda(\overline{\mathbf{u}}_{k+1} - \overline{\mathbf{u}}_k)^{\mathsf{T}}(\overline{\mathbf{u}}_{k+1} - \overline{\mathbf{u}}_k)\Big], \tag{4.29}$$

where $\overline{\mathbf{e}}_{k+1} = \overline{\mathbf{r}} - \overline{\mathbf{z}}_{k+1}$ and

$$\boldsymbol{\mathcal{W}}_{\mathbf{e}} = \mathbf{I}_N \otimes \mathbf{W_e} \in \mathbb{S}^{Nn_z}_{++}, \tag{4.30a}$$

$$\boldsymbol{\mathcal{W}}_{\mathbf{u}} = \mathbf{I}_N \otimes \mathbf{W_u} \in \mathbb{S}^{Nn_u}_{++}. \tag{4.30b}$$

Here, $\mathbf{I}_N$ is the $N \times N$ identity matrix and $\otimes$ denotes the Kronecker product.

In (4.29), the term including $\lambda\delta$ is omitted since it does not depend on $\bar{\mathbf{u}}_{k+1}$.

The objective function (4.29) using the batch model in (4.19), without the noise terms, and $\bar{\mathbf{e}}_{k+1} = \bar{\mathbf{r}} - \bar{\mathbf{z}}_{k+1}$ becomes

$$\mathfrak{L} = \frac{1}{2}\bar{\mathbf{u}}_{k+1}^{\mathsf{T}} \left(\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}} + \boldsymbol{\mathcal{W}}_{\mathbf{u}} + \lambda\mathbf{I}\right)\bar{\mathbf{u}}_{k+1} - \left(((\mathbf{I} - \mathbf{S}_{\mathbf{r}})\,\bar{\mathbf{r}})^{\mathsf{T}}\, \boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}} + \lambda\bar{\mathbf{u}}_{k}^{\mathsf{T}}\right)\bar{\mathbf{u}}_{k+1}, \quad (4.31)$$

where all terms independent of $\bar{\mathbf{u}}_{k+1}$ are omitted.

As mentioned before, the minimum is obtained when the gradient of $\mathfrak{L}$ equals zero. The second-order sufficient condition is fulfilled, since the Hessian matrix $\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}} + \boldsymbol{\mathcal{W}}_{\mathbf{u}} + \lambda\mathbf{I}$ is positive definite when $\boldsymbol{\mathcal{W}}_{\mathbf{e}} \in \mathbb{S}_{++}$, $\boldsymbol{\mathcal{W}}_{\mathbf{u}} \in \mathbb{S}_{++}$, and $\lambda \in \mathbb{R}_{+}$. By solving the gradient with respect to $\bar{\mathbf{u}}_{k+1}$ equal to zero, and using (4.19) together with $\bar{\mathbf{e}}_k = \bar{\mathbf{r}} - \bar{\mathbf{z}}_k$ to eliminate the terms involving $\bar{\mathbf{r}}$ and $\mathbf{x}_0$ gives

$$\bar{\mathbf{u}}_{k+1} = \boldsymbol{\mathcal{Q}}(\bar{\mathbf{u}}_k + \boldsymbol{\mathcal{L}}\bar{\mathbf{e}}_k), \quad (4.32a)$$

$$\boldsymbol{\mathcal{Q}} = (\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}} + \boldsymbol{\mathcal{W}}_{\mathbf{u}} + \lambda\mathbf{I})^{-1}(\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}}), \quad (4.32b)$$

$$\boldsymbol{\mathcal{L}} = (\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}})^{-1}\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\boldsymbol{\mathcal{W}}_{\mathbf{e}}, \quad (4.32c)$$

which is in the same form as (4.25). The matrices $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{L}}$ should not be confused with the filters $\mathcal{Q}(q)$ and $\mathcal{L}(q)$ in (4.23).

4.3.3 Convergence and Stability Properties

An important property for an ILC algorithm is that (4.20) is stable, i.e., $\mathbf{u}_k$, given by (4.24) or (4.27), should converge to a bounded signal. The convergence and stability properties are, for simplicity, analysed using the batch formulation in (4.27) of the ILC system. The ILC system is in fact a discrete-time linear system with the iterations index as the time. It is therefore possible to use standard stability results from linear system theory, see e.g. Rugh [1996]. Two main results about stability and convergence from Norrlöf and Gunnarsson [2002] are stated in Theorems 4.1 and 4.2. The reader is referred to Norrlöf and Gunnarsson [2002] for the details and more results, e.g. for the frequency domain.

Theorem 4.1 (Stability [Norrlöf and Gunnarsson, 2002, Corollary 3]). *The system* (4.27) *is stable if and only if*

$$\rho(\boldsymbol{\mathcal{Q}}(\mathbf{I} - \boldsymbol{\mathcal{L}}\mathbf{S}_{\mathbf{u}})) < 1. \quad (4.33)$$

Theorem 4.2 (Monotone convergence [Norrlöf and Gunnarsson, 2002, Theorem 9]). *If the system* (4.27) *satisfies*

$$\bar{\sigma}(\boldsymbol{\mathcal{Q}}(\mathbf{I} - \boldsymbol{\mathcal{L}}\mathbf{S}_{\mathbf{u}})) < 1, \quad (4.34)$$

then the system is stable and

$$\|\bar{\mathbf{u}}_\infty - \bar{\mathbf{u}}_k\|_2 \le \zeta^k \, \|\bar{\mathbf{u}}_\infty - \bar{\mathbf{u}}_0\|_2\,, \quad (4.35)$$

with $0 \le \zeta < 1$ *and*

$$\bar{\mathbf{u}}_\infty = (\mathbf{I} - \boldsymbol{\mathcal{Q}}(\mathbf{I} - \boldsymbol{\mathcal{L}}\mathbf{S}_{\mathbf{u}}))^{-1}\,\boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{L}}(\mathbf{I} - \mathbf{S}_{\mathbf{r}})\bar{\mathbf{r}}. \quad (4.36)$$

Nothing can unfortunately be said about monotonicity for the control error. However, it is still possible to calculate what the stationary error becomes as stated in Corollary 4.1.

Corollary 4.1. *The stationary control error* $\bar{\mathbf{e}}_\infty = \bar{\mathbf{r}} - \bar{\mathbf{y}}_\infty$ *is given by*

$$\bar{\mathbf{e}}_\infty = \left(\mathbf{I} - \mathbf{S_u}(\mathbf{I} - \mathcal{Q}(\mathbf{I} - \mathcal{L}\mathbf{S_u}))^{-1}\mathcal{Q}\mathcal{L}\right)(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}}. \tag{4.37}$$

Proof: The result follows from (4.36) and (4.19), without the noise terms. □

It should actually be the true systems $\mathbf{S}^0_\mathbf{u}(q)$ and $\mathbf{S}^0_\mathbf{r}(q)$ that are used in the ILC algorithm (4.24) and (4.27) instead of $\mathbf{S_u}(q)$ and $\mathbf{S_r}(q)$ when convergence and stability are investigated. However, the models must of course be used for obvious reasons.

Norm-optimal ILC

For norm-optimal ILC the ILC system (4.27) is stable independent of the system used. This special result is given in Theorem 4.3.

Theorem 4.3. ***(Stability and monotonic convergence for norm-optimal ILC):*** *The ILC system* (4.27) *is stable and monotonically convergent for all system descriptions in* (4.19) *using* $\mathcal{Q}$ *and* $\mathcal{L}$ *from* (4.32).

Proof: From Theorem 4.2 it holds that the iterative system (4.27) is stable and converges monotonically if $\bar{\sigma}\left(\mathcal{Q}(\mathbf{I} - \mathcal{L}\mathbf{S_u})\right) < 1$. Using $\mathcal{Q}$ and $\mathcal{L}$ from (4.32) gives

$$\bar{\sigma}\left(\mathcal{Q}(\mathbf{I} - \mathcal{L}\mathbf{S_u})\right) = \bar{\sigma}\left(\left(\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\mathbf{S_u} + \mathcal{W}_\mathbf{u} + \lambda\mathbf{I}\right)^{-1}\lambda\right) < 1 \tag{4.38}$$

independent of the system description $\mathbf{S_u}$ since $\mathcal{W}_\mathbf{e} \in \mathbb{S}_{++}$, $\mathcal{W}_\mathbf{u} \in \mathbb{S}_{++}$, and $\lambda \in \mathbb{R}_+$. □

From Theorem 4.2 and Corollary 4.1 it holds that the asymptotic control signal and error becomes

$$\bar{\mathbf{u}}_\infty = \left(\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\mathbf{S_u} + \mathcal{W}_\mathbf{u}\right)^{-1}\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}}, \tag{4.39}$$

$$\bar{\mathbf{e}}_\infty = \left(\mathbf{I} - \mathbf{S_u}\left(\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\mathbf{S_u} + \mathcal{W}_\mathbf{u}\right)^{-1}\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\right)(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}}. \tag{4.40}$$

If $\mathcal{W}_\mathbf{u} = 0$ and $\mathbf{S_u}$ invertible then

$$\bar{\mathbf{u}}_\infty = \left(\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\mathbf{S_u}\right)^{-1}\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}} = \mathbf{S_u^{-1}}(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}}, \tag{4.41}$$

$$\bar{\mathbf{e}}_\infty = \left(\mathbf{I} - \mathbf{S_u}\left(\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\mathbf{S_u}\right)^{-1}\mathbf{S_u^\mathsf{T}}\mathcal{W}_\mathbf{e}\right)(\mathbf{I} - \mathbf{S_r})\bar{\mathbf{r}} = 0. \tag{4.42}$$

It means that the norm-optimal ILC algorithm coincide with inversion of the system in (4.19). The assumption about $\mathbf{S_u}$ invertible is for most cases not applicable, e.g. the number of inputs and outputs to $\mathbf{S_u}$ must be equal. If $\mathbf{S_u}$ is not invertible but still $\mathcal{W}_\mathbf{u} = 0$, then the norm-optimal ILC algorithm gives a weighted least square solution of the inverse of the system with the weighting matrix $\mathcal{W}_\mathbf{e}$. However, in general $\mathcal{W}_\mathbf{u} \neq 0$ is used in order to handle model errors. Example 4.2 shows how the performance changes for different values of the weight matrices.

Example 4.2: Norm-optimal ILC

The performance of the norm-optimal ILC algorithm for different values of the tuning parameters is analysed on a flexible joint model. The model in continuous-time, using the state vector $\mathbf{x} = \begin{pmatrix} q_a & \dot{q}_a & q_m^a & \dot{q}_m^a \end{pmatrix}^{\mathsf{T}}$, is given by

$$\dot{\mathbf{x}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{M_a} & -\frac{d}{M_a} & \frac{k}{M_a} & \frac{d}{M_a} \\ 0 & 0 & 0 & 1 \\ \frac{k}{M_m} & \frac{d}{M_m} & -\frac{k}{M_m} & -\frac{f+d}{M_m} \end{pmatrix} \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{k_\tau}{M_m} \end{pmatrix} \tau, \tag{4.43}$$

with $k = 8$, $d = 0.0924$, $M_a = 0.0997$, $M_m = 0.0525$, $f = 1.7825$ $k_\tau = 0.61$. A discrete-time model is obtained using zero order hold sampling with a sample time $T_s = 0.01$ s. The model is stabilised with a P-controller with gain 1, and the output is chosen as q_a. No process noise and measurement noise are present. The reference signal is a step filtered four times through an FIR filter of length $n = 100$ with all coefficients equal to $1/n$. Five different configurations of the norm-optimal ILC algorithm, shown in Table 4.1, are used with $\mathbf{W_e} = 10^4$ for all five tests. The performance is evaluated using the relative reduction of the RMSE in percent with respect to the error when no ILC signal is applied, i.e.,

$$\rho_k = 100 \sqrt{\frac{1}{N} \sum_{t=1}^{N} e_k(t)^2} \Bigg/ \sqrt{\frac{1}{N} \sum_{t=1}^{N} e_0(t)^2}. \tag{4.44}$$

The relative reduction of the RMSE is shown in Figure 4.6. It can be seen that the convergence speed depends on λ and that the absolute error depends on $\mathbf{W_u}$.

Table 4.1: *Parameters for the norm-optimal ILC algorithm in Example 4.2.*

Test	1	2	3	4	5
$\mathbf{W_u}$	0.1	0.01	1	0.1	0.1
λ	1	1	1	0.1	10

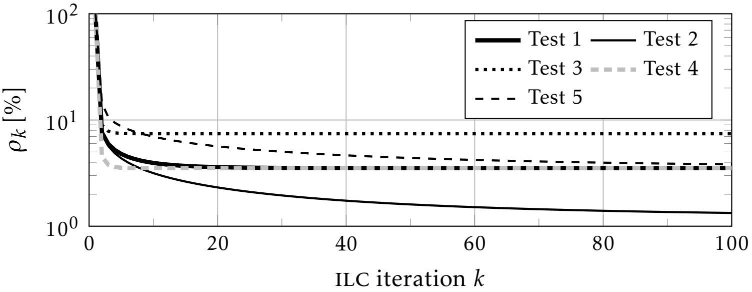


Figure 4.6: *Result for Example 4.2. Performance for norm-optimal ILC for different settings of the parameters* $\mathbf{W_u}$ *and* λ*.*

5 Concluding Remarks

THIS CHAPTER concludes the work in the thesis and gives a brief summary of the included publications in Part II. Possible directions for future work are also discussed.

5.1 Summary

New lightweight robot structures require more advanced controllers than before. Nowadays, the controllers are divided into a feedback controller and a feed-forward controller. The feedback controller usually consists of a single PID controller for each joint, and the feed-forward controller requires complex models of the robot structure. The more complex models, the more difficult will it be to derive and succeed in implementing the controller in an industrial environment, since a high-index DAE may have to be solved in real time. Instead, other control structures are needed. This thesis presents some ideas in this topic, based on sensor fusion methods for estimating the end-effector position. The thesis also presents more advanced control methods to be used in the feedback loop. The idea is not to remove the feed-forward controller completely since it is needed for trajectory tracking. Instead, the intention is to improve the feedback controller such that less complex models may be used in the feed-forward controller.

The estimation of the end-effector position is performed by combining a triaxial accelerometer at the end-effector and the motor angular positions. The estimation problem is formulated in a Bayesian setting, where the *extended Kalman filter* (EKF) and the *particle filter* (PF) have been used (Papers A and B). Experimental data for a two *degrees-of-freedom* (DOF) manipulator has been used to evaluate the estimation performance. Different types of estimators are used, where both the estimation model and the filter differ. The three observers with the best performance are

a) an EKF using a non-linear dynamic model,

b) a particle filter using a linear dynamic model,

c) an EKF with a non-linear model, where the acceleration of the end-effector is used as an input instead of a measurement.

The performance of these three observers is 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 on implementing the filter and get it to work, whereas most of the time for a) was spent on modelling the dynamics. The estimation methods in this thesis are general and can be extended to higher degrees of freedom robots and additional sensors, such as gyroscopes and camera systems. The main effect is more complex state space descriptions, a more problematic filter tuning, and longer computation time.

In order to have good performance it is essential to have good models and good knowledge of the measurement and process noise. The models are often given in continuous time and the filters operate in discrete time. The problem with discretisation of the continuous-time models has been investigated (Paper C). Moreover, a method to estimate the process noise covariance matrix has been proposed using the EM algorithm (Paper D). A great advantage with the EM method, compared to other methods that have been suggested in the literature, is that the true position of the end-effector is not needed.

Although most of the observers in this thesis, which have been implemented in MATLAB, are not running in real-time it is possible to use the estimates in off-line methods such as *iterative learning control* (ILC), system identification, and diagnosis. Clearly, the computation time can be decreased by optimising the MATLAB code or by using another programming language, e.g. C++. The estimation-based ILC framework, in particular for the norm-optimal ILC algorithm, has been considered in the thesis (Paper G). The algorithm has been extended to incorporate the full probability density function of the estimated control quantity. The estimation-based ILC has also made it possible to directly extend the norm-optimal ILC algorithm to non-linear systems using linearisation. Moreover, output controllability in the iteration domain, or target path controllability in the time domain, for systems that are controlled using ILC, are important to investigate (Paper H), to be able to draw conclusions about how the control error will converge.

The direct use of the accelerometer measurements in the feedback loop has also been considered with $\mathcal{H}_\infty$ methods (Paper E). It is shown that the performance can be significantly increased using $\mathcal{H}_\infty$ controllers without the extra accelerometer measurements. However, the manipulator is usually described by non-linear models, which makes it difficult to achieve a robust controller in the whole range of operation. A method to handle the non-linear flexibility is proposed, where robust stability is achieved for the whole range of operation, whereas robust performance is only ensured for a specific linearisation point (Paper F). Adding more sensors such as accelerometers at the end-effector increases the performance even more, though the tuning of the controllers becomes more difficult.

5.2 Future Work

A natural continuation is to extend the estimation problem to cover the complete six DOF robot. The sensor system could be extended with a gyroscope to get measurements of the rotation of the end-effector and not only the translation. It may not be possible to achieve good estimation performance if only one *inertial measurement unit* (IMU) is mounted at the end-effector. Instead, several IMUs should be mounted on well-chosen positions of the manipulator. Positioning of the IMUs is a complicated problem in itself. The ideal solution would be to formulate an optimisation problem to find the most informative positions of the IMUs to be used for state estimation. Another measurement to consider is the arm angular position, i.e., a measurement on the arm side of the gearbox.

A sensitivity analysis should also be considered to be able to find out how the observers behave. 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 could be to use the EM algorithm to estimate both the parameters and the states. The EM algorithm is in its general form an off-line method, however on-line solutions exist for special model structures.

It is also interesting to investigate the tuning of the noise covariance matrices in more details, for example, by having time varying matrices that increase when the path changes drastically. This can be done in several ways, e.g. find out when the path changes using the measured data, or using the programmed path. Another solution could be to use the *interacting multiple model* (IMM) filter.

The $\mathcal{H}_\infty$-control methods should of course also be extended to multi-DOF manipulators and experimental evaluations should be performed. Except for handling non-linearities it becomes even more difficult for the user to choose the weights compared to the single joint system considered in this thesis. To manage this increased complexity for the user it can be interesting to formulate an optimisation problem giving, in some sense, optimal weights that are used in the $\mathcal{H}_\infty$-synthesis method.

Bibliography

ABB Robotics. Product specification – ABB IRB4600. Document ID: 3HAC028284-001. Revision: N, 2013.

ABB Robotics. Company homepage. URL: `http://www.abb.com`, accessed February 2014.

Hyo-Sung Ahn, YangQuan Chen, and Kevin L. Moore. Iterative learning control: Brief survey and categorization. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 37(6):1099–1121, November 2007.

Notker Amann, David H. Owens, and Eric Rogers. Iterative learning control using optimal feedback and feedforward actions. *International Journal of Control*, 65(2):277–293, 1996a.

Notker Amann, David H. Owens, and Eric Rogers. Iterative learning control for discrete-time systems with exponential rate of convergence. *IEE Proceedings Control Theory and Applications*, 143(2):217–224, March 1996b.

Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Information and System Sciences Series. Prentice Hall Inc., Englewood Cliffs, NJ, USA, 1979.

Tohid Ardeshiri, 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/September 2011.

Suguru Arimoto. Learning control theory for robotic motion. *International Journal of Adaptive Control and Signal Processing*, 4(6):543–564, 1990.

Suguru Arimoto. *Iterative Learning Control: Analysis, Design, Integration and Application*, chapter A Brief History of Iterative Learning Control, pages 3–7. Kluwer Academic Publishers, Boston, MA, USA, 1998.

Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2):123–140, 1984a.

Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Iterative learning control for robot systems. In *Proceedings of the IEEE International Conference on Industrial Electronics, Control, and Instrumentation*, Tokyo, Japan, October 1984b.

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.

Karl-Johan Åström and Carlos Canudas de Wit. Revisiting the LuGre friction model. *IEEE Control Systems Magazine*, 28(6):101–114, December 2008.

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 Enginering, 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 Enginering, Linköping University, SE-581 83 Linköping, Sweden, June 2011a.

Patrik Axelsson. *On Sensor Fusion Applied to Industrial Manipulators*. Linköping Studies in Science and Technology. Licentiate Thesis No. 1511, Linköping University, SE-581 83 Linköping, Sweden, December 2011b.

Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 126–132, Dubrovnik, Croatia, September 2012.

Patrik Axelsson and Fredrik Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *Submitted to IEEE Transactions on Automatic Control*, 2012.

Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 283–288, Dubrovnik, Croatia, September 2012.

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

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/September 2011.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Tool position estimation of a flexible industrial robot using recursive Bayesian methods. In *Proceedings* *of the IEEE International Conference on Robotics and Automation*, pages 5234–5239, St. Paul, MN, USA, May 2012a.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012b.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based ILC using particle filter with application to industrial manipulators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1740–1745, Tokyo, Japan, November 2013.

Patrik Axelsson, Daniel Axehill, Torkel Glad, and Mikael Norrlöf. Controllability aspects for iterative learning control. *Submitted to International Journal of Control*, 2014a.

Patrik Axelsson, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-controller design methods applied to one joint of a flexible industrial manipulator. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014b.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based norm-optimal iterative learning control. *Submitted to Systems & Control Letters*, 2014c.

Patrik Axelsson, Goele Pipeleers, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-synthesis method for control of non-linear flexible joint models. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014d.

Luca Bascetta and Paolo Rocco. Revising the robust-control design for rigid robot manipulators. *IEEE Transactions on Robotics*, 26(1):180–187, February 2010.

Zeungnam Bien and Kyung M. Huh. Higher-order iterative learning control algorithm. *IEE Proceedings, Part D, Control Theory and Applications*, 136(3): 105–112, May 1989.

Zeungnam Bien and Jian-Xin Xu, editors. *Iterative Learning Control: Analysis, Design, Integration and Application*. Kluwer Academic Publishers, Boston, MA, USA, 1998.

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.

Douglas A. Bristow, Marina Tharayil, and Andrew G. Alleyne. A survey of iterative learning control — A learning-based method for high-performance tracking control. *IEEE Control Systems Magazine*, 26(3):96–114, June 2006.

Torgny Brogårdh. Present and future robot control development—an industrial perspective. *Annual Reviews in Control*, 31(1):69–79, 2007.

André Carvalho Bittencourt and Patrik Axelsson. Modeling and experiment design for identification of wear in a robot joint under load and temperature uncertainties based on friction data. *IEEE/ASME Transactions on Mechatronics*, 2013. DOI: 10.1109/TMECH.2013.2293001.

André Carvalho Bittencourt and Svante Gunnarsson. Static friction in a robot joint – modeling and identification of load and temperature effects. *Journal of Dynamic Systems, Measurement, and Control*, 134(5), September 2012.

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.

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/September 2011.

Giuseppe Casalino and Giorgio Bartolini. A learning procedure for the control of movements of robotic manipulators. In *Proceedings of the IASTED Symposium on Robotics and Automation*, pages 108–111, New Orleans, LA, USA, November 1984.

Wenjie Chen and Masayoshi Tomizuka. Direct joint space state estimation in robots with multiple elastic joints. *IEEE/ASME Transactions on Mechatronics*, 19(2):697–706, April 2014. DOI: 10.1109/TMECH.2013.2255308.

Wankyun Chung, Li-Chen Fu, and Su-Hau Hsu. *Springer Handbook of Robotics*, chapter Motion Control, pages 133–159. Springer-Verlag, Berlin, Heidelberg, Germany, 2008.

John J. Craig. Adaptive control of manipulators through repeated trials. In *Proceedings of the American Control Conference*, pages 1566–1572, San Diego, CA, USA, June 1984.

John J. Craig. *Introduction to Robotics Mechanics and Control*. Addison Wesley, Menlo Park, CA, USA, second edition, 1989.

Crossbow Technology. Accelerometers, High Sensitivity, LF Series, CXL02LF3, January 2004. Available at `http://www.xbow.com`.

Alessandro De Luca and Wayne Book. *Springer Handbook of Robotics*, chapter Robots with Flexible Elements, pages 287–319. Springer-Verlag, Berlin, Heidelberg, Germany, 2008.

Alessandro De Luca and Leonardo Lanari. Robots with elastic joints are linearizable via dynamic feedback. In *Proceedings of the 34th IEEE Conference on Decision and Control*, pages 3895–3897, New Orleans, LA, USA, December 1995.

Alessandro De Luca, Riccardo Farina, and Pasquale Lucibello. On the control of robots with visco-elastic joints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4297–4302, Barcelona, Spain, April 2005.

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.

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.

Jaques Denavit and Richard S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Journal of Applied Mechanics*, 22:215–221, 1955.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10 (3):197–208, 2000.

Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, NY, USA, 2001.

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.

FANUC Robotics. Company homepage. URL: `http://www.fanucrobotics.com`, accessed February 2014.

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.

Ronald A. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Mathematics*, 41:155–160, 1912.

Ronald A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society, Series A*, 222:309–368, 1922.

Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall Inc., Upper Saddle River, NJ, USA, fourth edition, 2002.

Stuart Gibson and Brett Ninness. Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682, October 2005.

Keith Glover and Duncan McFarlane. Robust stabilization of normalized coprime factor plant descriptions with $\mathcal{H}_\infty$-bounded uncertainty. *IEEE Transactions on Automatic Control*, 34(8):821–830, August 1989.

Herbert Goldstein, Charles Poole, and John Safko. *Classical Mechanics*. Addison Wesley, San Francisco, CA, USA, third edition, 2002.

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.

Svante Gunnarsson and Mikael Norrlöf. On the design of ILC algorithms using optimization. *Automatica*, 37(12):2011–2016, December 2001.

Svante Gunnarsson and Mikael Norrlöf. On the disturbance properties of high order iterative learning control algorithms. *Automatica*, 42(11):2031–2034, November 2006.

Fredrik Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur, Lund, Sweden, first edition, 2010.

Robert Henriksson, Mikael Norrlöf, 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.

Mrdjan Jankovic. Observer based control for elastic joint robots. *IEEE Transactions on Robotics and Automation*, 11(4):618–623, August 1995.

Rahim Jassemi-Zargani and Dan Necsulescu. 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.

Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64. Academic Press, New York, NY, USA, 1970.

Soo Jeon, Masayoshi Tomizuka, and Tetsuaki Katou. Kinematic Kalman filter (KKF) for robot end-effector sensing. *Journal of Dynamic Systems, Measurement, and Control*, 131(2), March 2009.

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, WA, USA, June 1995.

Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, 2000.

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.

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.

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.

Hassan K. Khalil. *Nonlinear Systems*. Prentice Hall Inc., Upper Saddle River, NJ, USA, third edition, 2002.

K. Kosuge, M. Umetsu, and K. Furuta. Robust linearization and control of robot arm using acceleration feedback. In *Proceedings of the IEEE International Conference on Control and Applications*, pages 161–165, Jerusalem, Israel, April 1989.

Krzysztof Kozłowski. *Modelling and Identification in Robotics*. Advances in Industrial Control. Springer, London, UK, 1998.

KUKA. Company homepage. URL: `http://www.kuka-ag.de/en/`, accessed February 2014.

Jae Young Lee, Je Sung Yeon, and Jong Hyeon Park. Robust nonlinear control for flexible joint robot manipulators. In *Proceedings of the SICE Annual Conference*, pages 440–445, Takamatsu, Japan, September 2007.

Leica Geosystems. Company homepage. URL: `http://metrology.leica-geosystems.com/en/index.htm`, accessed February 2014.

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.

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.

Helmut Lütkepohl. *Handbook of Matrices*. John Wiley & Sons, Chichester, West Sussex, England, 1996.

Duncan McFarlane and Keith Glover. A loop shaping design procedure using $\mathcal{H}_\infty$ synthesis. *IEEE Transactions on Automatic Control*, 37(6):759–769, June 1992.

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.

Yoshihiko Miyasato. Nonlinear adaptive $\mathcal{H}_\infty$ control of robotic manipulators under constraint. In *Proceedings of the 17th IFAC World Congress*, pages 4090–4095, Seoul, Korea, July 2008.

Yoshihiko Miyasato. Nonlinear adaptive $\mathcal{H}_\infty$ control of constrained robotic manipulators with input nonlinearity. In *Proceedings of the American Control Conference*, pages 2000–2005, St. Louis, MO, USA, July 2009.

Stig Moberg. *Modeling and Control of Flexible Manipulators*. Linköping Studies in Science and Technology. Dissertations No. 1349, Linköping University, SE-581 83 Linköping, Sweden, December 2010.

Stig Moberg and Sven Hanssen. Inverse dynamics of flexible manipulators. In *Proceedings of the Conference on Multibody Dynamics*, Warsaw, Poland, June 2009.

Stig Moberg, Jonas Öhr, and Svante Gunnarsson. A benchmark problem for robust feedback control of a flexible manipulator. *IEEE Transactions on Control Systems Technology*, 17(6):1398–1405, November 2009.

Stig Moberg, Erik Wernholt, Sven Hanssen, and Torgny Brogårdh. Modeling and parameter estimation of robot manipulators using extended flexible joint models. *Journal of Dynamic Systems, Measurement, and Control*, 136(3), May 2014. DOI: doi:10.1115/1.4026300.

Kevin L. Moore. *Iterative Learning Control for Deterministic Systems*. Advances in Industrial Control. Springer-Verlag, London, UK, 1993.

Kevin L. Moore. Iterative learning control: An expository overview. *Applied and Computational Control, Signals, and Circuits*, 1(1):151–214, 1998.

Kevin L. Moore and Chen YangQuan. On monotonic convergence of high order iterative learning update laws. In *Proceedings of the 15th IFAC World Congress*, pages 852–857, Barcelona, Spain, July 2002.

Motoman. Company homepage. URL: `http://www.motoman.eu/`, accessed February 2014.

Salvatore Nicosia and Patrizio Tomei. State observers for rigid and elastic joint robots. *Robotics and Computer-Integrated Manufacturing*, 9(2):113–120, 1992.

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.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, NY, USA, second edition, 2006.

Shimon Y. Nof, editor. *Handbook of Industrial Robotics*. John Wiley & Sons, Hoboken, NJ, USA, second edition, 1999.

Mikael Norrlöf and Svante Gunnarsson. Time and frequency domain convergence properties in iterative learning control. *International Journal of Control*, 75(14):1114–1126, 2002.

Goele Pipeleers and Jan Swevers. MATLAB-software `mixedHinfsyn`, 2013. Available at `http://set.kuleuven.be/optec/Software/mixedhinfsyn`.

Gerasimos G. Rigatos. Particle filtering for state estimation in nonlinear industrial systems. *IEEE Transactions on Instrumentation and Measurement*, 58(11): 3885–3900, November 2009.

Paolo Rocco. Stability of PID control for industrial robot arms. *IEEE Transactions on Robotics and Automation*, 12(4):606–614, August 1996.

Wilson J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1996.

Hansjörg G. Sage, Michel F. de Mathelin, Gabriel Abba, Jacques A. Gangloff, and Eric Ostertag. Nonlinear optimization of robust $\mathcal{H}_\infty$ controllers for industrial robot manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2352–2357, Albuquerque, NM, USA, April 1997.

Hansjörg G. Sage, Michel F. de Mathelin, and Eric Ostertag. Robust control of robot manipulators: A survey. *International Journal of Control*, 72(16):1498–1522, 1999.

Lorenzo Sciavicco and Bruno Siciliano. *Modelling and Control of Robot Manipulators*. Springer, London, UK, second edition, 2000.

Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer-Verlag, Berlin, Heidelberg, Germany, 2008.

Sigurd Skogestad and Ian Postletwaite. *Multivariable Feedback Control, Analysis and Design*. John Wiley & Sons, Chichester, West Sussex, England, second edition, 2005.

Y. D. Song, A. T. Alouani, and J. N. Anderson. Robust path tracking control of industrial robots: An $\mathcal{H}_\infty$ approach. In *Proceedings of the IEEE Conference on Control Applications*, pages 25–30, Dayton, OH, USA, September 1992.

Mark W. Spong. Modeling and control of elastic joint robots. *Journal of Dynamic Systems, Measurement, and Control*, 109:310–319, December 1987.

Mark W. Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons, Hoboken, NJ, USA, 2006.

Wayne L. Stout and M. Edwin Sawan. Application of H-infinity theory to robot manipulator control. In *Proceedings of the IEEE Conference on Control Applications*, pages 148–153, Dayton, OH, USA, September 1992.

Tatiana Taveira, Adriano Siqueira, and Marco Terra. Adaptive nonlinear $\mathcal{H}_\infty$ controllers applied to a free-floating space manipulator. In *Proceedings of the IEEE Conference on Control Applications*, pages 1476–1481, Munich, Germany, October 2006.

Patrizio Tomei. An observer for flexible joint robots. *IEEE Transactions on Automatic Control*, 35(6):739–743, June 1990.

Patrizio Tomei. A simple PD controller for robots with elastic joints. *IEEE Transactions on Automatic Control*, 36(10):1208–1213, October 1991.

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.

Niklas Wahlström, Patrik Axelsson, and Fredrik Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014.

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.

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.

Johanna Wallén, Isolde Dressler, Anders Robertsson, Mikael Norrlöf, and Svante Gunnarsson. Observer-based ILC applied to the Gantry-Tau parallel kinematic robot. In *Proceedings of the 18th IFAC World Congress*, pages 992–998, Milano, Italy, August/September 2011a.

Johanna Wallén, Mikael Norrlöf, and Svante Gunnarsson. A framework for analysis of observer-based ILC. *Asian Journal of Control*, 13(1):3–14, January 2011b.

Lars Westerlund. *The Extended Arm of Man. A History of the Industrial Robot.* Informationsförlaget, Stockholm, Sweden, 2000.

W. L. Xu and J. D. Han. Joint acceleration feedback control for robots: Analysis, sensing and experiments. *Robotics and Computer-Integrated Manufacturing*, 16(5):307–320, October 2000.

Je Sung Yeon and Jong Hyeon Park. Practical robust control for flexible joint robot manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3377–3382, Pasadena, CA, USA, May 2008.

Jongguk Yim and Jong Hyeon Park. Nonlinear $\mathcal{H}_\infty$ control of robotic manipulator. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 866–871, Tokyo, Japan, October 1999.

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`, 19 October 2004.

Keivan Zavari, Goele Pipeleers, and Jan Swevers. Multi-$\mathcal{H}_\infty$ controller design and illustration on an overhead crane. In *Proceedings of the IEEE Conference on Control Applications. Part of the IEEE Multi-Conference on Systems and Control*, pages 670–674, Dubrovnik, Croatia, October 2012.

Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall Inc., Upper Saddle River, NJ, USA, 1996.

Part II

Publications

Paper A

Bayesian State Estimation of a Flexible Industrial Robot

Authors: Patrik Axelsson, Rickard Karlsson and Mikael Norrlöf

Edited version of the paper:

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012b.

Bayesian State Estimation of a Flexible Industrial Robot

Patrik Axelsson[*], Rickard Karlsson[**] and Mikael Norrlöf[*]

[*]Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
mikael.norrlof@liu.se

[**]Nira Dynamics
Teknikringen 6
SE-583 30 Linköping, Sweden
rickard.karlsson
@niradynamics.se

Abstract

A sensor fusion method for state estimation of a flexible industrial robot is developed. By measuring the acceleration at the end-effector, the accuracy of the arm angular position, as well as the estimated position of the end-effector are improved. The problem is formulated in a Bayesian estimation framework and two solutions are proposed; the extended Kalman filter and the particle filter. In a simulation study on a realistic flexible industrial robot, the angular position performance is shown to be close to the fundamental Cramér-Rao lower bound. The technique is also verified in experiments on an ABB robot, where the dynamic performance of the position for the end-effector is significantly improved.

1 Introduction

Modern industrial robot control is usually based only on measurements from the motor angles of the manipulator. However, the ultimate goal is to move the tool according to a predefined path. In Gunnarsson et al. [2001] a method for improving the absolute accuracy of a standard industrial manipulator is described, where improved accuracy is achieved through identification of unknown or uncertain parameters in the robot system, and applying the *iterative learning control* (ILC) method [Arimoto et al., 1984; Moore, 1993], using additional sensors to measure the actual tool position. The aim of this paper is to evaluate Bayesian estimation techniques for sensor fusion and to improve the estimate of the tool position from measurements of the acceleration at the end-effector. The improved accuracy at the end-effector is needed in demanding applications such as laser cutting, where low cost sensors such as accelerometers are a feasible choice.

Two Bayesian state estimation techniques, the *extended Kalman filter* (EKF) and the *particle filter* (PF), are applied to a standard industrial manipulator and the result is evaluated with respect to the tracking performance in terms of position

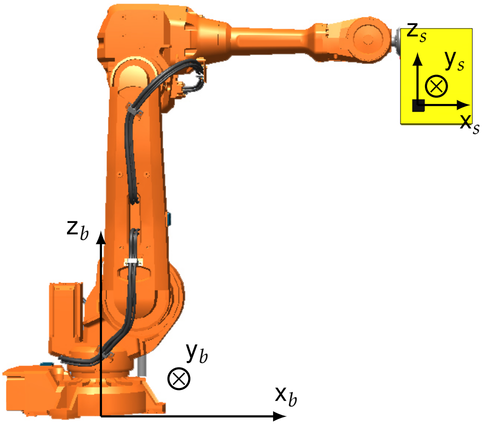


Figure 1: *The* ABB IRB4600 *robot with the accelerometer. The base coordinate system,* (x_b, y_b, z_b)*, and the coordinate system for the sensor (accelerometer),* (x_s, y_s, z_s)*, are also shown.*

accuracy of the tool. The main contribution in this paper compared to previous papers in the field is the combination of: i) the evaluation of estimation results in relation to the *Cramér-Rao lower bound* (CRLB); ii) the utilisation of motor angle measurement and accelerometer measurement in the filters; iii) the experimental evaluation on a commercial industrial robot, see Figure 1; iv) the extensive comparison of EKF and PF, and finally; v) the use of a manipulator model including a complete model of the manipulator's flexible modes. In addition, the utilisation of the calibration of the accelerometer sensor from Axelsson and Norrlöf [2012] and the proposal density for the PF using an EKF [Doucet et al., 2000; Gustafsson, 2010] are non standard.

Traditionally, many non-linear Bayesian estimation problems are solved using the EKF [Anderson and Moore, 1979; Kailath et al., 2000]. When the dynamic models and measurements are highly non-linear and the measurement noise is not Gaussian, linearised methods may not always be a good approach. The PF [Gordon et al., 1993; Doucet et al., 2001] provides a general solution to many problems where linearisation and Gaussian approximations are intractable or would yield too low performance.

Bayesian techniques have traditionally been applied in mobile robot applications, see e.g. Kwok et al. [2004]; Jensfelt [2001], and Doucet et al. [2001, Ch. 19]. In the industrial robotics research area one example is Jassemi-Zargani and Necsulescu [2002] where an EKF is used to improve the trajectory tracking for a rigid 2-*degrees-of-freedom* (DOF) robot using arm angle measurements and tool acceleration measurements. The extension to several DOF is presented in Quigley et al. [2010], where the EKF is used on a robot manipulator with seven DOF and three accelerometers. A method for accelerometer calibration with respect to orientation is also presented. The idea of combining a vision sensor, accelerometers, and gyros when estimating the tool position is explored in Jeon et al. [2009] for a 2-DOF manipulator, using a kinematic Kalman filter. Another way is to use the acceleration of the tool as an input instead of a measurement as described in De Luca et al. [2007], where it is assumed that the friction is neglected, the damping and spring are assumed linear. As a result, the estimation can be done using a linear time invariant observer with dynamics based upon pole placement. For flexible link robots the Kalman filter has been investigated in Li and Chen [2001] for a single link, where the joint angle and the acceleration of the tool are used as measurements. Moreover, in Lertpiriyasuwat et al. [2000] the extended Kalman filter has been used for a two link manipulator using the joint angles and the tool position as measurements. In both cases, the manipulator is operating in a plane perpendicular to the gravity field. Sensor fusion techniques using particle filters have so far been applied to very few industrial robotic applications [Rigatos, 2009; Karlsson and Norrlöf, 2004, 2005], and only using simulated data. The PF method is also motivated since it provides the possibility to design control laws and perform diagnosis in a much more advanced way, making use of the full posterior *probability density function* (PDF). The PF also enables incorporation of hard constraints on the system parameters, and it provides a benchmark for simpler solutions, such as given by the EKF.

This paper extends the simulation studies introduced in Karlsson and Norrlöf [2004, 2005] with experimental results. A performance evaluation in a realistic simulation environment for both the EKF and the PF is presented and it is analysed using the *Cramér-Rao lower bound* (CRLB) [Bergman, 1999; Kay, 1993]. In addition to Karlsson and Norrlöf [2004, 2005], experimental data, from a state of the art industrial robot, is used for evaluation of the proposed methods. A detailed description of the experimental setup is given and also modifications of the PF for experimental data are presented.

The paper is organised as follows. In Section 2, the Bayesian theory estimation is introduced and the concept of the CRLB is presented. The robot, estimation, and sensor models, are presented in Section 3. The performance of the EKF and of the PF are compared to the Cramér-Rao lower bound limit for simulated data in Section 4. In Section 5 the experimental setup and performance are presented. Finally, Section 6 contains conclusive remarks and future work.

2 Bayesian Estimation

Consider the discrete state-space model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \tag{1a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k, \tag{1b}$$

with state variables $\mathbf{x}_k \in \mathbb{R}^n$, input signal $\mathbf{u}_k$ and measurements $\mathbf{y}_{1:k} = \{\mathbf{y}_i\}_{i=1}^k$, with known *probability density functions* (PDFs) for the process noise, $p_{\mathbf{w}}(\mathbf{w})$, and measurement noise $p_{\mathbf{v}}(\mathbf{v})$. The non-linear posterior prediction density $p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k})$ and filtering density $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ for the Bayesian inference [Jazwinski, 1970] are given by

$$p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k}) = \int\limits_{\mathbb{R}^{n_x}} p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k})\,\mathrm{d}\mathbf{x}_k, \tag{2a}$$

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})}. \tag{2b}$$

For the important special case of linear-Gaussian dynamics and linear-Gaussian observations, the Kalman filter [Kalman, 1960] provides the solution. For non-linear and non-Gaussian systems, the PDF cannot, in general, be expressed with a finite number of parameters. Instead approximative methods are used. This is usually done in two ways; either by approximating the system or by approximating the posterior PDF, see for instance, Sorenson [1988]; Arulampalam et al. [2002]. Here, two different approaches of solving the Bayesian equations are considered; *extended Kalman filter* (EKF) , and *particle filter* (PF). The EKF will solve the problem using a linearisation of the system and assuming Gaussian noise. The PF on the other hand will approximately solve the Bayesian equations by stochastic integration. Hence, no linearisation errors occur. The PF can also handle non-Gaussian noise models where the PDFs are known only up to a normalisation constant. Also, hard constraints on the state variables can easily be incorporated in the estimation problem.

2.1 The Extended Kalman Filter (EKF)

For the special case of linear dynamics, linear measurements and additive Gaussian noise, the Bayesian recursions in (2) have an analytical solution given by the Kalman filter. For many non-linear problems, the noise assumptions and the non-linearity are such that a linearised solution will be a good approximation. This is the idea behind the EKF [Anderson and Moore, 1979; Kailath et al., 2000] where the model is linearised around the previous estimate. The time update and measurement update for the EKF are

$$\text{TU:} \quad \begin{cases} \hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{0}), \\ \mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^{\mathsf{T}} + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^{\mathsf{T}}, \end{cases} \tag{3a}$$

$$\text{MU:} \quad \begin{cases} \mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}} + \mathbf{R}_k\right)^{-1}, \\ \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1})\right), \\ \mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{P}_{k|k-1}, \end{cases} \tag{3b}$$

where the linearised matrices are given as

$$\mathbf{F}_{k-1} = \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{0})|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}, \tag{4a}$$

$$\mathbf{G}_{k-1} = \nabla_{\mathbf{w}} f(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w})|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}, \tag{4b}$$

$$\mathbf{H}_k = \nabla_{\mathbf{x}} h(\mathbf{x})|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}. \tag{4c}$$

In (3), $\mathbf{P}_{k|k}$ and $\mathbf{P}_{k|k-1}$ denote the covariance matrices for the estimation errors at time k given measurements up to time k and $k-1$, and the noise covariances are given as

$$\mathbf{Q}_k = \mathrm{Cov}(\mathbf{w}_k), \ \mathbf{R}_k = \mathrm{Cov}(\mathbf{v}_k), \tag{5}$$

where the process noise and measurement noise are assumed zero mean processes.

2.2 The Particle Filter (PF)

The PF from Doucet et al. [2001]; Gordon et al. [1993]; Ristic et al. [2004] provides an approximate solution to the discrete time Bayesian estimation problem formulated in (2), by updating an approximate description of the posterior filtering density. Let $\mathbf{x}_k$ denote the state of the observed system and $\mathbf{y}_{1:k} = \{\mathbf{y}_i\}_{i=1}^{k}$ be the set of observed measurements until present time. The PF approximates the density $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ by a large set of N samples (particles), $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N}$, where each particle has an assigned relative weight, $w_k^{(i)}$, chosen such that all weights sum to unity. The location and weight of each particle reflect the value of the density in the region of the state space. The PF updates the particle location in the state space and the corresponding weights recursively with each new observed measurement. Using the samples (particles) and the corresponding weights, the Bayesian equations can be approximately solved. To avoid divergence, a resampling step is introduced [Gordon et al., 1993]. The PF is summarised in Algorithm 1, where the proposal distribution $p^{\mathrm{prop}}(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \mathbf{y}_{k+1})$ can be chosen arbitrary as long as it is possible to draw samples from it.

The estimate for each time, k, is often chosen as the minimum mean square estimate, i.e.,

$$\hat{\mathbf{x}}_{k|k} = \mathrm{E}[\mathbf{x}_k] = \int_{\mathbb{R}^{n_x}} \mathbf{x}_k p(\mathbf{x}_k|\mathbf{y}_{1:k})\,\mathrm{d}\mathbf{x}_k \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{x}_k^{(i)}, \tag{6}$$

but other choices, such as the ML-estimate, might be of interest. There exist theoretical limits [Doucet et al., 2001] that the approximated PDF converges to the true as the number of particles tends to infinity.

2.3 Cramér-Rao Lower Bound

When different estimators are used, it is fundamental to know the best possible achievable performance. As mentioned previously, the PF will approach the true PDF asymptotically. In practice only approximations are possible since the number of particles are finite. For other estimators, such as the EKF, it is important to know how much the linearisation or model structure used, will affect the performance. The *Cramér-Rao lower bound* (CRLB) is such a characteristic for the second order moment [Kay, 1993; Cramér, 1946]. Here, only state-space models with additive Gaussian noise are considered. The theoretical posterior CRLB for a general dynamic system was derived in Van Trees [1968]; Tichavský et al. [1998]; Bergman [1999]; Doucet et al. [2001]. Here a continuous-time system is consid-

Algorithm 1 The Particle Filter (PF)

```
1: Generate $N$ samples $\{\mathbf{x}_0^{(i)}\}_{i=1}^N$ from $p(\mathbf{x}_0)$.
2: Compute
       $w_k^{(i)} = w_{k-1}^{(i)} \frac{p(\mathbf{y}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{p^{\text{prop}}(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)}$
   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 $\{\mathbf{x}_k^{(i_\star)}\}_{i=1}^N$ by resampling with replacement $N$
   times from $\{\mathbf{x}_k^{(i)}\}_{i=1}^N$, with probability $\bar{w}_k^{(i)} = \Pr\{\mathbf{x}_k^{(i_\star)} = \mathbf{x}_k^{(i)}\}$ and let $w_k^{(i)} = 1/N$,
   $i = 1, \dots, N$.
4: Generate predictions from the proposal density
       $\mathbf{x}_{k+1}^{(i)} \sim p^{\text{prop}}(\mathbf{x}_{k+1}|\mathbf{x}_k^{(i_\star)}, \mathbf{y}_{k+1}),\ i = 1, \dots, N.$
5: Increase $k$ and continue to step 2.
```

ered. By first linearising and then discretising the system, the fundamental limit can in practice be calculated as the stationary solution for every k, $\bar{\mathbf{P}} = \bar{\mathbf{P}}(\mathbf{x}_k^{\text{TRUE}})$, of the Riccati recursions in the EKF, where the linearisations are around the true state trajectory, $\mathbf{x}_k^{\text{TRUE}}$. Note that the approximation is fairly accurate for the industrial robot application due to a high sample rate and a small process noise. The predicted value of the stationary covariance for each time t, i.e., for each point in the state-space, $\mathbf{x}_k^{\text{TRUE}}$, is denoted $\bar{\mathbf{P}}_p$ and given as the solution to

$$\bar{\mathbf{P}}_p = \bar{\mathbf{F}}(\bar{\mathbf{P}}_p - \bar{\mathbf{K}}\bar{\mathbf{H}}\bar{\mathbf{P}}_p)\bar{\mathbf{F}}^{\mathsf{T}} + \bar{\mathbf{G}}\mathbf{Q}\bar{\mathbf{G}}^{\mathsf{T}}. \tag{7}$$

where the linearised matrices $\bar{\mathbf{F}}$, $\bar{\mathbf{G}}$ and $\bar{\mathbf{H}}$ are evaluated around the true trajectory, $\mathbf{x}_k^{\text{TRUE}}$, and

$$\bar{\mathbf{K}} = \bar{\mathbf{P}}_p\bar{\mathbf{H}}^{\mathsf{T}}(\bar{\mathbf{H}}\bar{\mathbf{P}}_p\bar{\mathbf{H}}^{\mathsf{T}} + \mathbf{R})^{-1}. \tag{8}$$

The CRLB limit can now be calculated as

$$\bar{\mathbf{P}} = \bar{\mathbf{P}}_p - \bar{\mathbf{K}}\bar{\mathbf{H}}\bar{\mathbf{P}}_p, \tag{9}$$

for each point along the true state-trajectory.

3 Dynamic Models

In this section a continuous-time 2-DOF robot model is discussed. The model is simplified and transformed into discrete time, to be used by the EKF and PF. The measurements are in both cases angle measurements from the motors, with or without acceleration information from the end-effector.

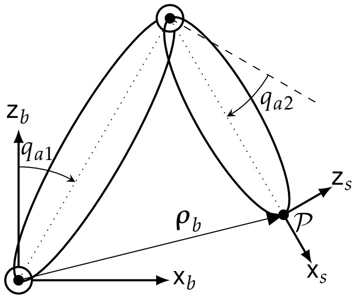


Figure 2: *A 2-DOF robot model. The links are assumed to be rigid and the joints are described by a two mass system connected by a spring damping pair.*

3.1 Robot Model

The robot model used in this work is a joint flexible two-axes model, see Figure 2. The model corresponds to axes two and three of a serial 6-DOF industrial robot like the one in Figure 1. A common assumption of the dynamics of the robot is that the transmission can be approximated by two or three masses connected by springs and dampers. The coefficients in the resulting model can be estimated from an identification experiment, see for instance Kozłowski [1998]. Here, it will be assumed that the transmission can be modelled as a two mass system and that the links are rigid.

The dynamic model can be described by a torque balance for the motors and the arms. A common way to obtain the dynamic model in industrial robotics is to use Lagrange's equation as described in Sciavicco and Siciliano [2000]. The equation describing the torque balance for the motor becomes

$$\mathbf{M}_m\ddot{\mathbf{q}}_m^a = -\mathbf{F}_m\dot{\mathbf{q}}_m^a - \mathbf{K}\cdot(\mathbf{q}_m^a - \mathbf{q}_a) - \mathbf{D}\cdot(\dot{\mathbf{q}}_m^a - \dot{\mathbf{q}}_a) + \boldsymbol{\tau}_m^a, \tag{10}$$

where $\mathbf{M}_m$ is the motor inertia matrix, $\mathbf{q}_m^a = \begin{pmatrix} q_m^1/\eta_1 & q_m^2/\eta_2 \end{pmatrix}^{\mathsf{T}}$ the motor angles on the arm side of the gear box, $\mathbf{q}_a = \begin{pmatrix} q_a^1 & q_a^2 \end{pmatrix}^{\mathsf{T}}$ the arm angles, η_i the gear ratio, $\mathbf{F}_m$ the viscous friction at the motor, $\mathbf{K}$ the spring constant and $\mathbf{D}$ the damping coefficient. No couplings between motor one and two implies that $\mathbf{M}_m$ is a diagonal matrix. The parameters $\mathbf{F}_m$, $\mathbf{K}$, and $\mathbf{D}$ are two by two diagonal matrices, where the diagonal element ii corresponds to joint i. The inputs to the system are the motor torques, $\boldsymbol{\tau}_m^a = \begin{pmatrix} \tau_m^1\eta_1 & \tau_m^2\eta_1 \end{pmatrix}^{\mathsf{T}}$. The corresponding relation for the arm becomes a non-linear equation

$$M_a(\mathbf{q}_a)\ddot{\mathbf{q}}_a + C(\mathbf{q}_a, \dot{\mathbf{q}}_a)\dot{\mathbf{q}}_a + G(\mathbf{q}_a) = \mathbf{K}\cdot(\mathbf{q}_m^a - \mathbf{q}_a) + \mathbf{D}\cdot(\dot{\mathbf{q}}_m^a - \dot{\mathbf{q}}_a), \tag{11}$$

where $M_a(\cdot)$ is the arm inertia matrix, $C(\cdot)$ the Coriolis- and centrifugal terms and $G(\cdot)$ the gravitational torque. Here, it is assumed that there are no couplings between the arms and motors, which is valid if the gear ratio is high [Spong, 1987]. A more detailed model of the robot should include non-linear friction such as Coulomb friction. An important extension would also be to model the non-linear spring characteristics in the gearboxes. In general, the gearbox is less stiff for torques close to zero and more stiff when high torques are applied. The extended flexible joint model proposed in Moberg [2010, Paper A], which improves the control accuracy, can also be used.

3.2 Estimation Model

The estimation model has to reflect the dynamics in the true system. A straight forward choice of estimation model is the state space equivalent of (10) and (11), which gives a non-linear dynamic model with eight states (motor and arm angular positions and velocities). This gives both a non-linear state space model and a non-linear measurement model. Instead, a linear state space model is suggested with arm angles, velocities and accelerations as state variables, in order to simplify the time update for the PF. Note that the measurement model is still non-linear in this case. Bias states compensating for measurement and model errors have shown to improve the accuracy and are therefore also included. The state vector is now given as

$$\mathbf{x}_k = \begin{pmatrix} \mathbf{q}_{a,k}^{\mathsf{T}} & \dot{\mathbf{q}}_{a,k}^{\mathsf{T}} & \ddot{\mathbf{q}}_{a,k}^{\mathsf{T}} & \mathbf{b}_{m,k}^{\mathsf{T}} & \mathbf{b}_{\ddot{\rho},k}^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}}, \tag{12}$$

where $\mathbf{q}_{a,k} = \begin{pmatrix} q_{a,k}^1 & q_{a,k}^2 \end{pmatrix}^{\mathsf{T}}$ contains the arm angles from joint two and three in Figure 1, $\dot{\mathbf{q}}_{a,k}$ is the angular velocity, $\ddot{\mathbf{q}}_{a,k}$ is the angular acceleration, $\mathbf{b}_{m,k} = \begin{pmatrix} b_{m,k}^1 & b_{m,k}^2 \end{pmatrix}^{\mathsf{T}}$ contains the bias terms for the motor angles, and $\mathbf{b}_{\ddot{\rho},k} = \begin{pmatrix} b_{\ddot{\rho},k}^1 & b_{\ddot{\rho},k}^2 \end{pmatrix}^{\mathsf{T}}$ contains the bias terms for the acceleration at time k. The bias states are used to handle model errors in the measurement equation but also to handle drifts in the measured signals, especially in the acceleration signals. The first three states are given by a constant acceleration model discretised with zero order hold, and the bias states are modelled as random walk. This yields the following state space model in discrete time

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{G}_{\mathbf{u}}\mathbf{u}_k + \mathbf{G}_{\mathbf{w}}\mathbf{w}_k, \tag{13a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k, \tag{13b}$$

where

$$\mathbf{F} = \left(\begin{array}{ccc|cc} \mathbf{I} & T_s\mathbf{I} & T_s^2/2\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & T_s\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right), \quad \mathbf{G}_{\mathbf{w}} = \left(\begin{array}{c|cc} \frac{T_s^3}{6}\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \frac{T_s^2}{2}\mathbf{I} & \mathbf{0} & \mathbf{0} \\ T_s\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right), \quad \mathbf{G}_{\mathbf{u}} = \left(\begin{array}{c} \frac{T_s^3}{6}\mathbf{I} \\ \frac{T_s^2}{2}\mathbf{I} \\ T_s\mathbf{I} \\ \hline \mathbf{0} \\ \mathbf{0} \end{array}\right). \tag{14}$$

The input, $\mathbf{u}_k$, is the arm jerk reference, i.e., the differentiated arm angular acceleration reference. The process noise, $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ are considered Gaussian with zero mean and covariances, $\mathbf{Q}_k$ and $\mathbf{R}_k$ respectively. The sample time is denoted T_s and $\mathbf{I}$ and $\mathbf{0}$ are two by two identity and null matrices. The sensor model (13b) is described in full detail in the next section.

3.3 Sensor Model

The available measurements are motor angular positions from resolvers and the acceleration of the end-effector from the accelerometer. The sensor model is thus given by

$$h(\mathbf{x}_k) = \begin{pmatrix} \mathbf{q}^a_{m,k} + \mathbf{b}_{m,k} \\ \ddot{\boldsymbol{\rho}}_k + \mathbf{b}_{\ddot{\rho},k} \end{pmatrix}, \tag{15}$$

where $\mathbf{q}^a_{m,k} = \begin{pmatrix} q^1_{m,k}/\eta_1 & q^2_{m,k}/\eta_2 \end{pmatrix}^{\mathsf{T}}$ are the motor angles and $\ddot{\boldsymbol{\rho}}_k = \begin{pmatrix} \rho^{\mathsf{x}}_k & \rho^{\mathsf{z}}_k \end{pmatrix}^{\mathsf{T}}$ is the Cartesian acceleration vector in the accelerometer frame $O\mathsf{x}_s\mathsf{z}_s$, see Figure 2. With the simplified model described in Section 3.1, the motor angles $\mathbf{q}_{m,k}$ are computed from (11) according to

$$\mathbf{q}^a_{m,k} = \mathbf{q}_{a,k} + \mathbf{K}^{-1} \cdot \left(M_a(\mathbf{q}_{a,k})\ddot{\mathbf{q}}_{a,k} + G(\mathbf{q}_{a,k}) + C(\mathbf{q}_{a,k}, \dot{\mathbf{q}}_{a,k})\dot{\mathbf{q}}_{a,k} - \mathbf{D} \cdot (\dot{\mathbf{q}}^a_{m,k} - \dot{\mathbf{q}}_{a,k}) \right). \tag{16}$$

Here, the motor angular velocity $\dot{\mathbf{q}}^a_m$ can be seen as an input signal to the sensor model. The damping term $\mathbf{D} \cdot (\dot{\mathbf{q}}^a_m - \dot{\mathbf{q}}_a)$ is small compared to the other terms and is therefore neglected.

The approach is similar to the one suggested in Gunnarsson and Norrlöf [2004], which uses the relation given by (11) in the case when the system is scalar and linear. The results presented here are more general, since a multi-variable non-linear system is considered.

The acceleration in frame $O\mathsf{x}_s\mathsf{z}_s$, in Figure 2, measured by the accelerometer, can be expressed as

$$\ddot{\rho}_{s,k} = \mathcal{R}_{s/b}(\mathbf{q}_{a,k}) \left(\ddot{\rho}_{b,k}(\mathbf{q}_{a,k}) + \mathbf{g}_b \right), \tag{17}$$

where $\mathcal{R}_{s/b}(\mathbf{q}_{a,k})$ is the rotation matrix from $O\mathsf{x}_b\mathsf{z}_b$ to $O\mathsf{x}_s\mathsf{z}_s$, $\mathbf{g}_b = \begin{pmatrix} 0 & g \end{pmatrix}^{\mathsf{T}}$ is the gravity vector and $\ddot{\rho}_{b,k}(\mathbf{q}_{a,k})$ is the second time derivative of the vector $\boldsymbol{\rho}_{b,k}(\mathbf{q}_{a,k})$, see Figure 2. The vector $\boldsymbol{\rho}_{b,k}(\mathbf{q}_{a,k})$ is described by the forward kinematics [Sciavicco and Siciliano, 2000] which is a non-linear mapping from joint angles to Cartesian coordinates, i.e.,

$$\boldsymbol{\rho}_{b,k}(\mathbf{q}_{a,k}) = \begin{pmatrix} \mathsf{x}^{\mathrm{ACC}} \\ \mathsf{z}^{\mathrm{ACC}} \end{pmatrix} = \Upsilon_{\mathrm{ACC}}(\mathbf{q}_{a,k}), \tag{18}$$

where $\mathsf{x}^{\mathrm{ACC}}$ and $\mathsf{z}^{\mathrm{ACC}}$ are the position of the accelerometer expressed in frame $O\mathsf{x}_b\mathsf{z}_b$. Differentiation of $\boldsymbol{\rho}_{b,k}$ twice, with respect to time, gives

$$\ddot{\rho}_{b,k}(\mathbf{q}_{a,k}) = \mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_{a,k})\ddot{\mathbf{q}}_{a,k} + \left(\sum_{i=1}^{2} \frac{\partial \mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_{a,k})}{\partial q^{(i)}_{a,k}} \dot{q}^{(i)}_{a,k} \right) \dot{\mathbf{q}}_{a,k}, \tag{19}$$

where $q^{(i)}_{a,k}$ is the ith element of $\mathbf{q}_{a,k}$ and $\mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_{a,k})$ is the Jacobian of $\Upsilon_{\mathrm{ACC}}(\mathbf{q}_{a,k})$, i.e.,

$$\mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_a) = \nabla_{\mathbf{q}_a} \Upsilon_{\mathrm{ACC}}(\mathbf{q}_a). \tag{20}$$

Both the position model (16) for the motors and the acceleration model (19) are now a function of the state variables $\mathbf{q}_{a,k}$, $\dot{\mathbf{q}}_{a,k}$, and $\ddot{\mathbf{q}}_{a,k}$.

Remark 1. If the non-linear dynamics (10) and (11), are used, see Section 3.1, the relation in (16) becomes linear since $\mathbf{q}_{m,k}^{a}$ is a state variable. However, the relation in (19) becomes more complex since $\ddot{\mathbf{q}}_{a,k}$ is no longer a state, but has to be computed using (11).

4 Analysis

4.1 Simulation Model

In order to perform *Cramér-Rao lower bound* (CRLB) analysis, the true robot trajectory must be known. Hence, in practice this must be conducted in a simulation environment since not all state variables are available as measurements. In the sequel, the simulation model described in Karlsson and Norrlöf [2005] is used, where the CRLB analysis is compared to Monte Carlo simulations of the EKF and PF.

4.2 Cramér-Rao Lower Bound Analysis of the Robot

In Section 2.3, the posterior *Cramér-Rao lower bound* (CRLB) was defined for a general non-linear system with additive Gaussian noise. In this section the focus is on the CRLB expression for the industrial robot presented in Section 3.1. Solving for the acceleration in (11) yields

$$\kappa(\mathbf{q}_a, \dot{\mathbf{q}}_a) \triangleq \ddot{\mathbf{q}}_a = M_a(\mathbf{q}_a)^{-1}\Big(\mathbf{K}\cdot(\mathbf{q}_m^a - \mathbf{q}_a) + \mathbf{D}\cdot(\dot{\mathbf{q}}_m^a - \dot{\mathbf{q}}_a) - G(\mathbf{q}_a) - C(\mathbf{q}_a, \dot{\mathbf{q}}_a)\dot{\mathbf{q}}_a\Big).$$

Here, the motor angular velocity, $\dot{\mathbf{q}}_m$, is considered as an input signal, hence not part of the state-vector, $\mathbf{x}(t) = \begin{pmatrix}\mathbf{q}_a^\mathsf{T} & \dot{\mathbf{q}}_a^\mathsf{T} & \ddot{\mathbf{q}}_a^\mathsf{T}\end{pmatrix}^\mathsf{T}$. The system can be written in state space form as

$$\dot{\mathbf{x}}(t) = \frac{d}{dt}\begin{pmatrix}\mathbf{q}_a \\ \dot{\mathbf{q}}_a \\ \ddot{\mathbf{q}}_a\end{pmatrix} = f^c(\mathbf{x}(t)) = \begin{pmatrix}\dot{\mathbf{q}}_a \\ \ddot{\mathbf{q}}_a \\ \Lambda(\mathbf{q}_a, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a)\end{pmatrix}, \tag{21a}$$

$$\Lambda(\mathbf{q}_a, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a) = \frac{d}{dt}\kappa(\mathbf{q}_a, \dot{\mathbf{q}}_q). \tag{21b}$$

The differentiation of κ is performed symbolically, using the MATLAB symbolic toolbox. According to Section 2.3 the CRLB is defined as the stationary Riccati solution of the EKF around the true trajectory, $\mathbf{x}_k^{\text{TRUE}}$. This is formulated for a discrete-time system. Hence, the continuous-time robot model from (21) must be discretised. This can be done by first linearising the system and then discretising it [Gustafsson, 2010]. The differentiation is done numerically around the true trajectory, to avoid the very complex symbolic gradient, and the result becomes,

$$\mathbf{A}^c = \nabla_{\mathbf{x}} f^c(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_k^{\text{TRUE}}} = \begin{pmatrix}\mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \frac{\partial\Lambda(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})}{\partial\mathbf{q}} & \frac{\partial\Lambda(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})}{\partial\dot{\mathbf{q}}} & \frac{\partial\Lambda(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})}{\partial\ddot{\mathbf{q}}}\end{pmatrix}. \tag{22}$$

The desired discrete-time system matrix is now given as

$$\bar{\mathbf{F}} = e^{\mathbf{A}^c \cdot T_s}, \tag{23}$$

where T_s is the sample time. The CRLB is presented in Figure 3.

4.3 Estimation Performance

The performance of the EKF and of the PF are compared against the *Cramér-Rao lower bound* (CRLB) calculated in Section 4.2, using simulated data. The model is implemented and simulated using the Robotics Toolbox [Corke, 1996] in MATLAB Simulink. The robot is stabilised using a single PID-controller. The estimation model and sensor model will not use the bias states described in Section 3.2 because no model errors or drift are included in the simulation. This means that only the upper left corner of the matrices in (14) are used.

The simulation study is based mainly around the EKF approach, since it is a fast method well suited for large Monte Carlo simulations. The PF is much slower, therefore, a smaller Monte Carlo study is performed. The Monte Carlo simulations use the following covariance matrices for the process and measurement noise

$$\mathbf{Q} = 4 \cdot 10^{-6} \cdot \mathbf{I}, \quad \mathbf{R} = \begin{pmatrix} 10^{-6} \cdot \mathbf{I} & \mathbf{0} \\ \mathbf{0} & 10^{-4} \cdot \mathbf{I} \end{pmatrix}. \tag{24}$$

The measurement covariance is basically given by the motor angle and accelerometer uncertainty, and the process noise covariance is considered as a filter design parameter. The system is simulated around the nominal trajectory and produces different independent noise realisations for the measurement noise in each simulation. The continuous-time Simulink model of the robot is sampled in 1 kHz. The data is then decimated to 100 Hz before any estimation method is applied.

The estimation performance is evaluated using the *root mean square error* (RMSE) which is defined as

$$\text{RMSE}(k) = \left(\frac{1}{N_{\text{MC}}} \sum_{j=1}^{N_{\text{MC}}} \| \mathbf{x}_k^{\text{TRUE}} - \hat{\mathbf{x}}_k^{(j)} \|_2^2 \right)^{1/2}, \tag{25}$$

where N_{MC} is the number of Monte Carlo simulations, $\mathbf{x}_k^{\text{TRUE}}$ is the true state vector and $\hat{\mathbf{x}}_k^{(j)}$ is the estimated state vector in Monte Carlo simulation j. Here, the state vector is divided up into states corresponding to angular position, angular velocity, and angular acceleration, before (25) is used.

EKF In Figure 3 the *root mean square error* (RMSE) from 500 Monte Carlo simulations is compared to the CRLB limit, both with and without acceleration measurements. The CRLB is computed as the square root of the trace for the covariance matrix part corresponding to the angular states. As seen, the RMSE is close the fundamental limit. The discrepancy is due to model errors, i.e., neglected damping term and the fact that the estimator uses a simplified system matrix consisting of integrators only. Also note that the accelerometer measurements reduce

Table 1: *The* RMSE *for arm-side angular position ($\mathbf{q}_a$), angular velocity ($\dot{\mathbf{q}}_a$) and angular acceleration ($\ddot{\mathbf{q}}_a$), with and without the accelerometer, using* 500 *Monte Carlo simulations.*

	Accelerometer	**No accelerometer**
RMSE q_a	$1.25 \cdot 10^{-5}$	$2.18 \cdot 10^{-5}$
RMSE $\dot{q}_a$	$7.57 \cdot 10^{-5}$	$4.08 \cdot 10^{-4}$
RMSE $\ddot{q}_a$	$1.23 \cdot 10^{-3}$	$3.91 \cdot 10^{-3}$

the estimation uncertainty. The results in Figure 3 are of course for the chosen trajectory, but the acceleration values are not that large, so greater differences will occur for larger accelerations. The RMSE, ignoring the initial transient is given in Table 1 for both angular position, velocity and acceleration. The improvements are substantial in angular position, but for control, the improvements in angular velocity and acceleration are important.

PF The proposal density $p^{\text{prop}}(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \mathbf{y}_{k+1})$ in Algorithm 1 is chosen as the conditional prior of the state vector, i.e., $p(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)})$, and resampling is selected every time, which gives

$$w_k^{(i)} = p(\mathbf{y}_k|\mathbf{x}_k^{(i)}), \quad i = 1, \ldots, N. \tag{26}$$

The particle filter is rather slow compared to the EKF for this model structure. Hence, the given MATLAB implementation of the system is not well suited for large Monte Carlo simulations. Instead, a small Monte Carlo study over a short part of the trajectory used for the EKF case is considered. The PF and the EKF are compared, and a small improvement in performance is noted. The result is given in Figure 4. One explanation for the similar results between the EKF and PF is that the non-linearities may not give a multi modal distribution, hence the point estimates are quite similar. The advantage with the PF is that it can utilise hard constraints on the state variables and it can also be used for control and diagnosis where the full posterior PDF is available. Even though the PF is slow, it gives more insight in the selection of simulation parameters than the EKF, where the filter performance is more dependent on the ratio between the process and measurement noise.

5 Experiments on an ABB IRB4600 Robot

The experiments were performed on an ABB IRB4600 industrial robot, like the one seen in Figure 1. To illuminate the tracking capacity of the filters, the servo tuning of the robot was not optimal, which introduces more oscillations in the path. The accelerometer used in the experiments is the triaxial accelerometer CXL02LF3 from Crossbow Technology, which has a range of ±2 g, and a sensitivity of 1 V/g [Crossbow Technology, 2004]. In the next sections the experimental setup and results are given.

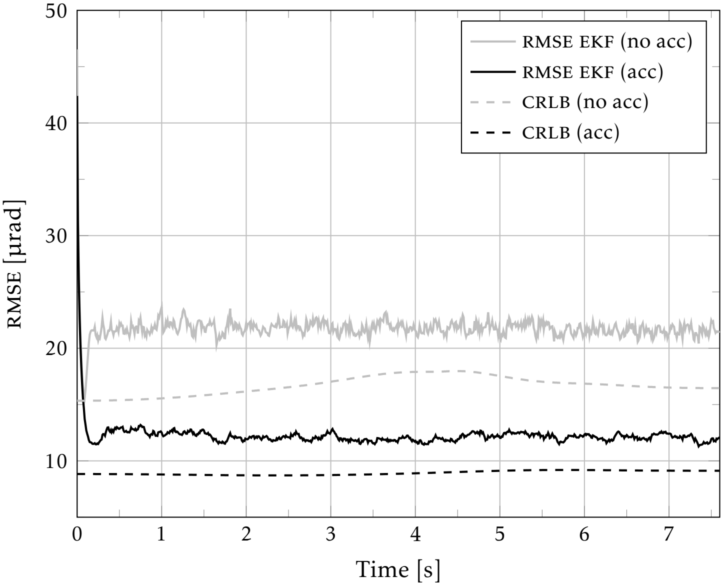


Figure 3: *Angular position* RMSE *from* 500 *Monte Carlo simulations using the* EKF *with and without accelerometer sensor are compared to the* CRLB *limit for every time, i.e., the square root of the trace of the angular position from the time-varying* CRLB *covariance.*

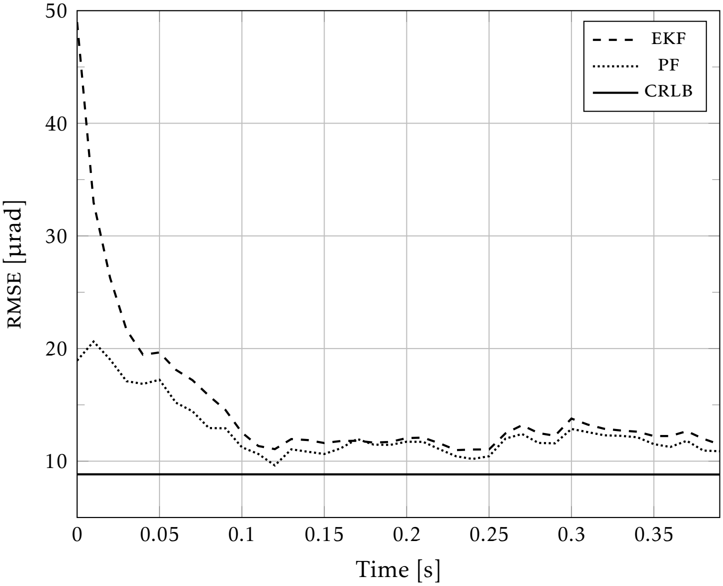


Figure 4: EKF *and* PF *angular position* RMSE *with external accelerometer signal from* 20 *Monte Carlo simulations.*

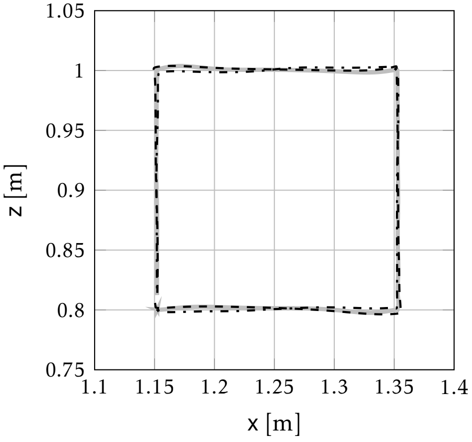


Figure 5: *The path start at the lower left corner and is counter-clockwise. A laser tracking system from Leica Geosystems has been used to measure the true tool position (solid). The estimated tool position using the* EKF *(dashed) and $\Upsilon_{TCP}(\mathbf{q}^a_{m,k})$ (dash-dot) are also shown.*

5.1 Experimental Setup

The orientation and position of the accelerometer were estimated using the method described in Axelsson and Norrlöf [2012]. All measured signals, i.e., acceleration, motor angles and arm angular acceleration reference, are synchronous and sampled with a rate of 2 kHz. The accelerometer measurements are filtered with a low pass filter before any estimation method is applied to better reflect the tool movement. The path used in the evaluation is illustrated in Figure 5, and it is programmed such that only joints two and three are moved. Moreover, the wrist is configured such that the couplings to joint two and three are minimised. It is not possible to get measurements of the true state variables $\mathbf{x}_k^{\text{TRUE}}$, as is the case for the simulation, instead, the true trajectory of the end-effector, more precisely the *tool centre point* (TCP), $\mathrm{x}_k^{\text{TCP}}$ and $\mathrm{z}_k^{\text{TCP}}$, 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/m and a sample rate of 1 kHz [Leica Geosystems, 2008]. The measured tool position is however not synchronised with the other measured signals, i.e., 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. The estimated data is 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.

5.2 Experimental Results

The only measured quantity to compare the estimates with is the measured tool position, as was mentioned in Section 5.1. Therefore, the estimated arm angles are used to compute an estimate of the TCP using the kinematic relation, i.e.,

$$\begin{pmatrix} \hat{x}^{\mathrm{TCP}} \\ \hat{z}^{\mathrm{TCP}} \end{pmatrix} = \Upsilon_{\mathrm{TCP}}(\hat{\mathbf{q}}_{a,k}), \tag{27}$$

where $\hat{\mathbf{q}}_{a,k}$ is the result from the EKF or the PF. Another simple way to estimate the tool position is to use the forward kinematic applied to the motor angles [1], i.e., $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$. In the evaluation study the estimates from the EKF, PF, and $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$ are compared to measurements from the Leica system. When computing the 2-norm of the RMSE the first 0.125 seconds are disregarded in order to evaluate the tracking performance only, and not include filter transients.

In the evaluation of the experiment, the focus is on position error only since the Leica laser reference system measures position only. However, the estimation technique presented is general, so the velocity estimates will be improved as well, which is important for many control applications. In simulations this has been verified, see Section 4.3 and Table 1. Since the position is based on integrating the velocity model, this will in general be true when applied to experimental data as well. However, the current measurement system cannot be used to verify this.

EKF Figure 5 shows that the estimated paths follow the true path. The performance of the estimates is better shown in Figures 6 and 7, where the four sides are magnified. At first, it can be noticed that $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$ cannot estimate the oscillations of the true path. This is not a surprise since the oscillations originates from the flexibilities in the gear boxes which are not taken care of in this straightforward way to estimate the TCP. However, as seen the accelerometer based sensor fusion method performs very well. It can also be noticed that the EKF estimate goes somewhat past the corners before it changes direction. An explanation to this phenomena can be that the jerk reference is used as an input to the estimation model. The jerk reference does not coincide with the actual jerk as a result of model errors and control performance. The initial transient for the EKF, due to incorrect initialisation of the filter, rapidly approaches the true path. In this case $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$ starts near the true path, but $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$ can start further away for another path. The position RMSE is presented in Figure 8, where the EKF with acceleration measurements shows a significantly improve in the performance. The 2-norm of the RMSE[2] for the EKF is reduced by 25 % compared to $\Upsilon_{\mathrm{TCP}}(\mathbf{q}^a_{m,k})$. This is based on the single experimental trajectory, but the result is in accordance with the simulation result and the theoretical calculations. Figure 8 also shows that the EKF converges fast. The MATLAB implementation of the EKF is almost real-time, and without losing performance the measurements can be slightly decimated (to approximately 200 Hz), yielding faster than real-time calculations.

[1] The motor angles are first transformed to the arm side of the gear box via the gear ratio.

[2] The RMSE is computed without considering the first 0.125 seconds where the EKF has a transient behavior.

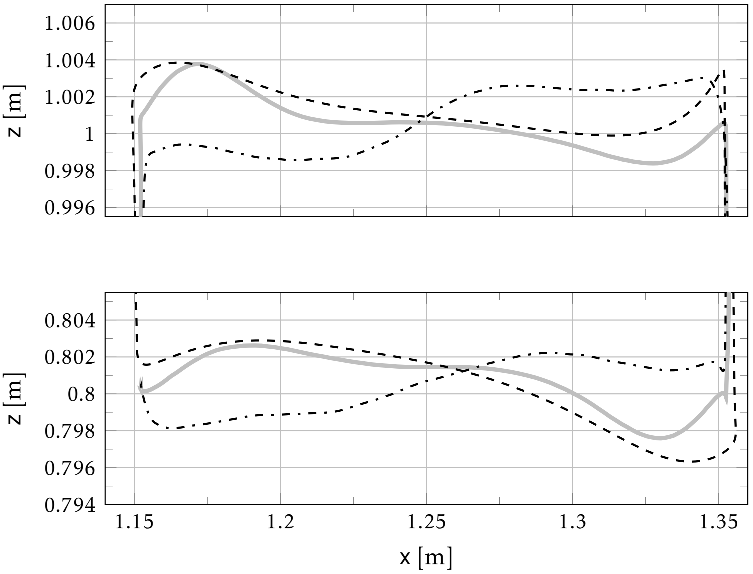


Figure 6: *The top side (upper diagram) and bottom side (lower diagram) of the square path in Figure 5 for the true tool position (solid) and tool position estimates using the* EKF *(dashed) and* $\Upsilon_{TCP}(\mathbf{q}^{a}_{m,k})$ *(dash-dot).*

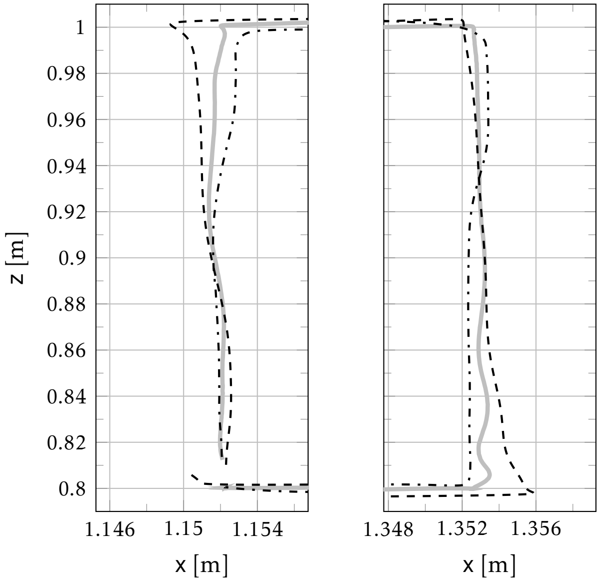


Figure 7: *The left side (left diagram) and right side (right diagram) of the square path in Figure 5 for the true tool position (solid) and tool position estimates using the* EKF *(dashed) and* $\Upsilon_{TCP}(\mathbf{q}^{a}_{m,k})$ *(dash-dot).*

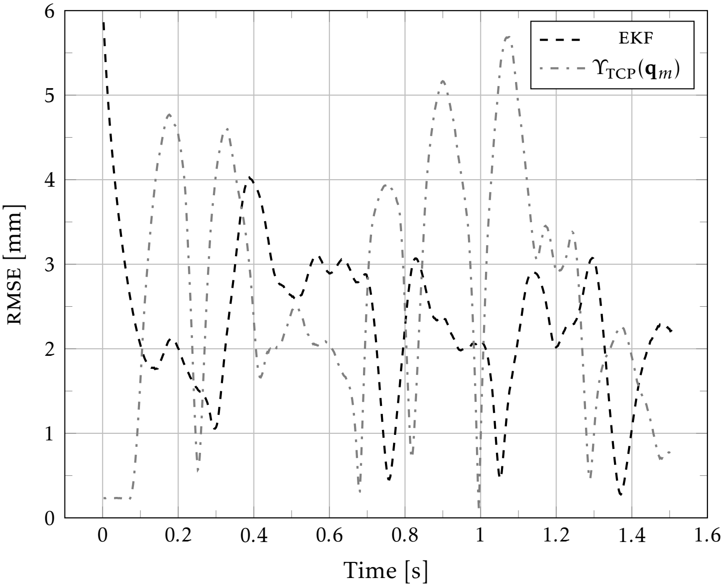


Figure 8: *Tool position RMSE for the EKF (dashed) and $\Upsilon_{TCP}(\mathbf{q}^a_{m,k})$ (dash-dot). The 2-norm of the RMSE-signals, without the first 0.125 seconds, are 0.1246 and 0.1655 for the EKF and $\Upsilon_{TCP}(\mathbf{q}^a_{m,k})$, respectively.*

PF The proposal density used during the simulation did not work properly for the experimental data due to a high *signal to noise ratio* (SNR) and also model errors. One could use an optimal proposal density [Doucet et al., 2000; Gustafsson, 2010] but the problem is that it is difficult to sample from that. Instead, the proposal density is approximated using an EKF, [Doucet et al., 2000; Gustafsson, 2010]

$$p^{\text{prop}}(\mathbf{x}_k|\mathbf{x}^{(i)}_{k-1}, \mathbf{y}_k) = \mathcal{N}(f(\mathbf{x}^{(i)}_{k-1}) + \mathbf{K}^{(i)}_k(\mathbf{y}_k - \hat{\mathbf{y}}^{(i)}_k), (\mathbf{H}^{(i)}_k \mathbf{R}^{\dagger}_k \mathbf{H}^{(i)}_k + \mathbf{Q}^{\dagger}_{k-1})^{\dagger}), \tag{28}$$

where † denotes the pseudo-inverse, and where the matrices are assumed to be evaluated for each particle state.

The result of the PF compared to the EKF can be found in Figure 9 and Figure 10. The PF performs better in the corners, i.e., the estimated path does not go past the corners before it changes. The motive that the PF can handle the problem with the jerk input better than the EKF can be that the particle cloud covers a larger area of the state space. The PF estimate is also closer to the true path, at least at the vertical sides. Figure 11 shows the RMSE for the PF which is below the RMSE for the EKF most of the time. The resulting 2-norm of the RMSE for the PF is 0.0818, which is approximately 66 % of the EKF and 49 % of $\Upsilon_{\text{TCP}}(\mathbf{q}^a_{m,k})$. Note that the transients are not included, i.e., the first 0.125 seconds are removed. The PF converges much faster than the EKF as can be seen clearly in Figure 11. The PF in the proposed implementation is far from real-time and the bias states are needed to control the model errors.

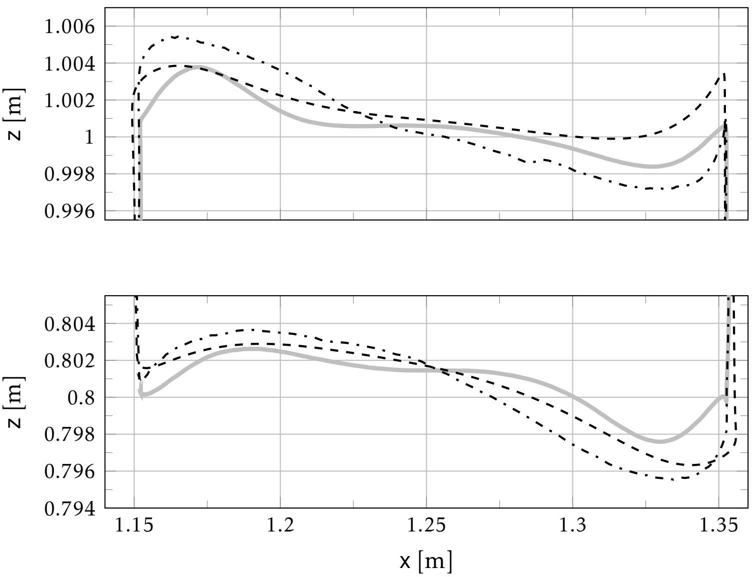


Figure 9: *The top side (upper diagram) and bottom side (lower diagram) of the square path in Figure 5 for the true tool position (solid) and tool position estimates using the* EKF *(dashed) and the* PF *(dash-dot).*

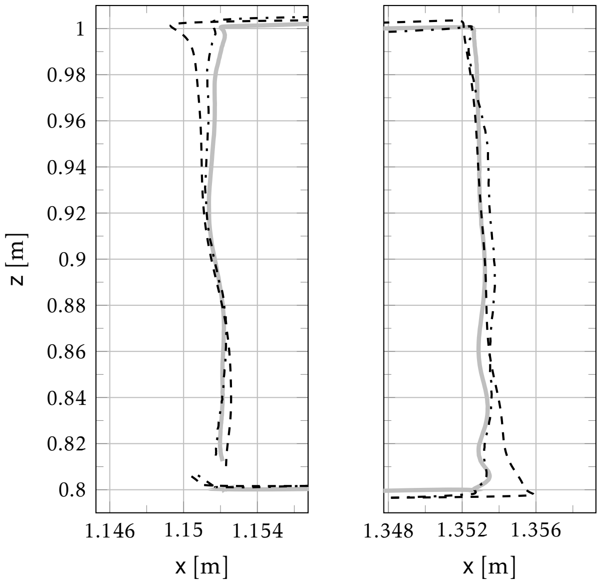


Figure 10: *The left side (left diagram) and right side (right diagram) of the square path in Figure 5 for the true tool position (solid) and tool position estimates using the* EKF *(dashed) and the* PF *(dash-dot).*

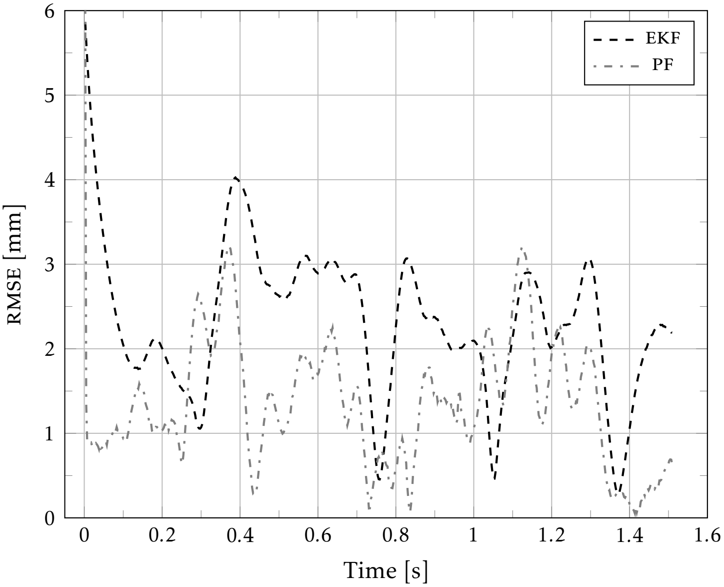


Figure 11: *Tool position RMSE for the EKF (dashed) and the PF (dash-dot). The 2-norm of the RMSE-signals, without the first 0.125 seconds, are 0.1246 and 0.0818 for the EKF and the PF, respectively.*

6 Conclusions and Future Work

A sensor fusion approach to find estimates of the tool position, velocity, and acceleration by combining a triaxial accelerometer at the end-effector and the measurements from the motor angles of an industrial robot is presented. The estimation is formulated as a Bayesian problem and two solutions are proposed; the extended Kalman filter and the particle filter. The algorithms were tested on simulated data from a realistic robot model as well as on experimental data.

Sufficiently accurate estimates are produced for simulated data, where the performance both with and without accelerometer measurements are close to the fundamental Cramér-Rao lower bound limit in Monte Carlo simulations. The dynamic performance for experimental data is also significantly better using the accelerometer method. The velocity estimates are also proven to be much more accurate when the filter uses information from the accelerometer. This is important for control design in order to give a well damped response at the robot arm.

Since the intended use of the estimates is to improve position control using an off-line method, like iterative learning control, there are no real-time issues using the computational demanding particle filter algorithm, however the extended Kalman filter runs in real-time in MATLAB. The estimation methods presented in this paper 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 a larger state space model giving more time-consuming calculations and also a more complex measurement equation. The most time-consuming step in the EKF is the matrix multiplications $\mathbf{F}_k \mathbf{P}_{k|k} \mathbf{F}_k^\mathsf{T}$. The two matrix multiplications require in total $4n^3$ flops[3]. For example, going from two to six DOF increases the computational cost with a factor of 27. For the PF it is not as easy to give a description of the increased computational complexity.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC and the SSF project Collaborative Localization.

[3] A flop is one of the elementary scalar operations $+, -, *, /$.

Bibliography

Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Information and System Sciences Series. Prentice Hall Inc., Englewood Cliffs, NJ, USA, 1979.

Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2):123–140, 1984.

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.

Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 283–288, Dubrovnik, Croatia, September 2012.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012.

Niclas Bergman. *Recursive Bayesian Estimation: Navigation and Tracking Applications*. Linköping Studies in Science and Technology. Dissertations No. 579, Linköping University, SE-581 83 Linköping, Sweden, May 1999. http://www.control.isy.liu.se/publications/.

Peter I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, March 1996.

Harald Cramér. *Mathematical Methods of Statistics*. Princeton University Press, 1946.

Crossbow Technology. Accelerometers, High Sensitivity, LF Series, CXL02LF3, January 2004. Available at `http://www.xbow.com`.

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.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10 (3):197–208, 2000.

Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, NY, USA, 2001.

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.

Svante Gunnarsson and Mikael Norrlöf. Iterative learning control of a flexible robot arm using accelerometers. In *Proceedings of the IEEE Conference on Control Applications*, pages 1012–1016, Taipei, Taiwan, September 2004.

Svante Gunnarsson, Mikael Norrlöf, Geir Hovland, Ulf Carlsson, Torgny Brogårdh, Tommy Svensson, and Stig Moberg. Pathcorrection for an industrial robot. European Patent Application No. EP1274546, April 2001.

Fredrik Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur, Lund, Sweden, first edition, 2010.

Rahim Jassemi-Zargani and Dan Necsulescu. 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.

Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64. Academic Press, New York, NY, USA, 1970.

Patric Jensfelt. *Approaches to Mobile Robot Localization in Indoor Environments*. PhD thesis, Royal Institute of Technology, Sweden, 2001. ISBN 91-7283-135-9.

Soo Jeon, Masayoshi Tomizuka, and Tetsuaki Katou. Kinematic Kalman filter (KKF) for robot end-effector sensing. *Journal of Dynamic Systems, Measurement, and Control*, 131(2), March 2009.

Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, 2000.

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.

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.

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.

Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Signal Processing Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, 1993.

Krzysztof Kozłowski. *Modelling and Identification in Robotics*. Advances in Industrial Control. Springer, London, UK, 1998.

Cody Kwok, Dieter Fox, and Marina Meilă. Real-time particle filters. *Proceedings of the IEEE*, 92(3):469–484, March 2004.

Leica Geosystems. Case study ABB robotics - Västerås, 2008. Available at `http://metrology.leica-geosystems.com/en/index.htm`.

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.

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.

Stig Moberg. *Modeling and Control of Flexible Manipulators*. Linköping Studies in Science and Technology. Dissertations No. 1349, Linköping University, SE-581 83 Linköping, Sweden, December 2010.

Kevin L. Moore. *Iterative Learning Control for Deterministic Systems*. Advances in Industrial Control. Springer-Verlag, London, UK, 1993.

Morgan Quigley, Reuben Brewer, Sai P. Soundararaj, Vijay Pradeep, Quoc Le, and Andrew Y. Ng. Low-cost accelerometers for robotic manipulator perception. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6168–6174, Taipei, Taiwan, October 2010.

Gerasimos G. Rigatos. Particle filtering for state estimation in nonlinear industrial systems. *IEEE Transactions on Instrumentation and Measurement*, 58(11): 3885–3900, November 2009.

Branko Ristic, Sanjeev Arulampalam, and Neil Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, Norwood, MA, USA, 2004.

Lorenzo Sciavicco and Bruno Siciliano. *Modelling and Control of Robot Manipulators*. Springer, London, UK, second edition, 2000.

Harold W. Sorenson. *Bayesian Analysis of Time Series and Dynamic Models*, chapter Recursive Estimation for Nonlinear Dynamic Systems, pages 126–165. Marcel Dekker Inc., 1988.

Mark W. Spong. Modeling and control of elastic joint robots. *Journal of Dynamic Systems, Measurement, and Control*, 109:310–319, December 1987.

Petr Tichavský, Carlos H. Muravchik, and Arye Nehorai. Posterior Cramér-Rao bounds for discrete-time nonlinear filtering. *IEEE Transactions on Signal Processing*, 46(5):1386–1396, May 1998.

Harry L. Van Trees. *Detection, Estimation and Modulation Theory, Part 1*. John Wiley & Sons, Hoboken, NJ, USA, 1968.

Paper B

Evaluation of Six Different Sensor Fusion Methods for an Industrial Robot using Experimental Data

Authors: Patrik Axelsson

Edited version of the paper:

Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 126–132, Dubrovnik, Croatia, September 2012.

Evaluation of Six Different Sensor Fusion Methods for an Industrial Robot using Experimental Data

Patrik Axelsson

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se

Abstract

Experimental evaluations for path estimation are performed on an ABB IRB4600 robot. Different observers using Bayesian techniques 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.

1 Introduction

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 non-linearities. The most crucial component, when it comes to flexibilities, is the gearbox. The gearbox has changed more and more to a flexible component, where the flexibilities have to be described by non-linear relations in the models in order to have good control performance. 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. 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. The performance is insufficient and the reason 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. 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 the gearbox as well as more measurements than only the motor angular positions. 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.

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 Necsulescu [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.

2 State Estimation

The estimation problem for the discrete time non-linear state space model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) + g(\mathbf{x}_k)\mathbf{w}_k, \tag{1a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k, \tag{1b}$$

is to find the state vector $\mathbf{x}_k \in \mathbb{R}^{n_x}$ at time k given the measurements $\mathbf{y}_k \in \mathbb{R}^{n_y}$ $k = 1, \ldots, N$. The estimation problem can be seen as calculation/approximation of the posterior density $p(\mathbf{x}_k | \mathbf{y}_{1:l})$ using all measurements up to time l, where $\mathbf{y}_{1:l} = \{\mathbf{y}_1, \ldots, \mathbf{y}_l\}$. There are two types of problems, filtering and smoothing. Filtering uses only measurements up to present time and smoothing uses future measurements also, i.e., $l = k$ for filtering and $l > k$ for smoothing. Using Bayes' law, and the Markov property for the state space model, repeatedly, the optimal solution for the Bayesian inference can be obtained. See Jazwinski [1970] for details. The solution to the Bayesian inference 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 have an analytical solution, which is known as the *Kalman filter* (KF). Approximative methods must be used for non-linear and non-Gaussian systems. Here three approximative solutions are considered; the *extended Kalman filter* (EKF), the *extended Kalman smoother* (EKS) and the *particle filter* (PF).

EKF The EKF [Anderson and Moore, 1979] solves the Bayesian recursions using a first order Taylor expansion of the non-linear system equations around the previous estimate. The process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ are assumed to be Gaussian with zero mean and covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively. The time and measurement updates are

$$\text{TU:} \quad \begin{cases} \hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}), \\ \mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^{\mathsf{T}} + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^{\mathsf{T}}, \end{cases} \tag{2a}$$

$$\text{MU:} \quad \begin{cases} \mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathsf{T}} + \mathbf{R}_k\right)^{-1}, \\ \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1}, \mathbf{u}_k)\right), \\ \mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{P}_{k|k-1}, \end{cases} \tag{2b}$$

where

$$\mathbf{F}_{k-1} = \left.\frac{\partial f\,(\mathbf{x}, \mathbf{u}_{k-1})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}, \quad \mathbf{G}_{k-1} = g(\hat{\mathbf{x}}_{k-1|k-1}), \quad \mathbf{H}_k = \left.\frac{\partial h(\mathbf{x}, \mathbf{u}_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}. \tag{3}$$

The notation $\hat{\mathbf{x}}_{k|k}$, $\mathbf{P}_{k|k}$, $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ means estimates of the state vector $\mathbf{x}$ and covariance matrix $\mathbf{P}$ at time k using measurements up to time k and $k-1$, respectively.

EKS The EKS [Yu et al., 2004] solves the Bayesian recursions in the same way as the EKF. The difference is that future measurements are available. First, the EKF equations are used forward in time, then the backward time recursion

$$\hat{\mathbf{x}}_{k|N}^{s} = \hat{\mathbf{x}}_{k|k} + \mathbf{P}_{k|k}\mathbf{F}_k^{\mathsf{T}}\mathbf{P}_{k+1|k}^{-1}\left(\hat{\mathbf{x}}_{k+1|N}^{s} - \hat{\mathbf{x}}_{k+1|k}\right), \tag{4a}$$

$$\mathbf{P}_{k|N}^{s} = \mathbf{P}_{k|k} + \mathbf{P}_{k|k}\mathbf{F}_k^{\mathsf{T}}\mathbf{P}_{k+1|k}^{-1}\left(\mathbf{P}_{k+1|N}^{s} - \mathbf{P}_{k+1|k}\right)\mathbf{P}_{k+1|k}^{-1}\mathbf{F}_k\mathbf{P}_{k|k}, \tag{4b}$$

is used, where $\mathbf{F}_k$ is given above.

PF The PF [Doucet et al., 2001; Gordon et al., 1993] solves the Bayesian recursions using stochastic integration. The PF approximates the posterior density $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ by a large set of N particles $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N}$, where each particle has an assigned relative weight $w_k^{(i)}$, chosen such that $\sum_{i=1}^{N} w_k^{(i)} = 1$. The PF updates the particle location and the corresponding weights recursively with each new observed measurement. 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 1, where the proposal density $p^{\text{prop}}(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \mathbf{y}_{k+1})$ can be chosen arbitrary as long as it is possible to draw samples from it. In this work the optimal proposal density, approximated by an EKF, is used. See Doucet et al. [2000], and Gustafsson [2010] for details. The state estimate for each sample k is often chosen as the minimum mean square estimate

$$\hat{\mathbf{x}}_{k|k} = \mathrm{E}\left[\mathbf{x}_k\right] = \int_{\mathbb{R}^{n_x}} \mathbf{x}_k p(\mathbf{x}_k|\mathbf{y}_{1:k})\,\mathrm{d}\mathbf{x}_k \approx \sum_{i=1}^{N} w_k^{(i)}\mathbf{x}_k^{(i)}. \tag{5}$$

Algorithm 1 The Particle Filter (PF)

```
1: Generate $N$ samples $\{\mathbf{x}_0^{(i)}\}_{i=1}^N$ from $p(\mathbf{x}_0)$.
2: Compute
       $w_k^{(i)} = w_{k-1}^{(i)} \dfrac{p(\mathbf{y}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{p^{\text{prop}}(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)}$
   and normalise, i.e., $\bar{w}_k^{(i)} = w_k^{(i)} / \sum_{j=1}^N w_k^{(j)}, \; i = 1, \ldots, N$.
3: [Optional]. Generate a new set $\{\mathbf{x}_k^{(i_\star)}\}_{i=1}^N$ by resampling with replacement $N$
   times from $\{\mathbf{x}_k^{(i)}\}_{i=1}^N$, with probability $\bar{w}_k^{(i)} = \Pr\{\mathbf{x}_k^{(i_\star)} = \mathbf{x}_k^{(i)}\}$ and let $w_k^{(i)} = 1/N$,
   $i = 1, \ldots, N$.
4: Generate predictions from the proposal density
       $\mathbf{x}_{k+1}^{(i)} \sim p^{\text{prop}}(\mathbf{x}_{k+1}|\mathbf{x}_k^{(i_\star)}, \mathbf{y}_{k+1}), \; i = 1, \ldots, N.$
5: Increase $k$ and continue to step 2.
```

3 Dynamic Models

3.1 Robot Model

This section describes a two-link non-linear flexible robot model, corresponding to joint two and three for a serial six DOF industrial robot. The dynamic robot model is a joint flexible two-axes model from Moberg et al. [2008], see Figure 1. Each link is modelled as a rigid-body and the joints are modelled as a spring damping pair with non-linear spring torque and linear damping. The deflection in each joint is given by the arm angle q_{ai} and the motor angle q_{mi}. Let

$$\mathbf{q}_a = \begin{pmatrix} q_{a1} & q_{a2} \end{pmatrix}^{\mathsf{T}}, \quad \mathbf{q}_m^a = \begin{pmatrix} q_{m1}/\eta_1 & q_{m2}/\eta_2 \end{pmatrix}^{\mathsf{T}}, \quad \boldsymbol{\tau}_m^a = \begin{pmatrix} \tau_{m1}\eta_1 & \tau_{m2}\eta_2 \end{pmatrix}^{\mathsf{T}},$$

where τ_{mi} is the motor torque and $\eta_i = q_{mi}/q_{ai} > 1$ is the gear ratio. A dynamic model can be derived as

$$M_a(\mathbf{q}_a)\ddot{\mathbf{q}}_a + C(\mathbf{q}_a, \dot{\mathbf{q}}_a) + G(\mathbf{q}_a) + N(\mathbf{q}) = 0, \tag{6a}$$

$$\mathbf{M}_m\ddot{\mathbf{q}}_m^a + F(\dot{\mathbf{q}}_m^a) - N(\mathbf{q}) = \tau_m^a, \tag{6b}$$

using Lagrange's equation, where $N(\mathbf{q}) = T(\mathbf{q}_a - \mathbf{q}_m^a) + D(\dot{\mathbf{q}}_a - \dot{\mathbf{q}}_m^a)$, $M_a(\,\cdot\,)$ and $\mathbf{M}_m$ are the inertia matrices for the arms and motors, respectively, $C(\,\cdot\,)$ is the Coriolis- and centrifugal terms, $G(\,\cdot\,)$ is the gravity torque, $F(\,\cdot\,)$ is the friction torque, $T(\,\cdot\,)$ is the spring stiffness torque and $D(\,\cdot\,)$ is the damping torque. See Moberg et al. [2008] for a full description of the model.

3.2 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

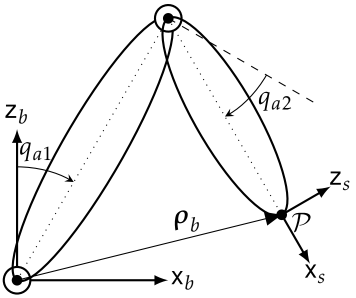


Figure 1: *A two degrees-of-freedom robot model. The links are assumed to be rigid and the joints are described by a two mass system connected by a spring damper pair. The accelerometer is attached in the point $\mathcal{P}$.*

to include a bias term. The acceleration is measured in a frame $O\mathrm{x}_s\mathrm{z}_s$ fixed to the accelerometer relative an inertial frame, which is chosen to be the world fixed base frame $O\mathrm{x}_b\mathrm{z}_b$, see Figure 1. The acceleration in $O\mathrm{x}_s\mathrm{z}_s$ can thus be expressed as

$$\ddot{\boldsymbol{\rho}}_s(\mathbf{q}_a) = \boldsymbol{\mathcal{R}}_{b/s}(\mathbf{q}_a)\left(\ddot{\boldsymbol{\rho}}_b(\mathbf{q}_a) + \mathbf{g}_b\right) + \mathbf{b}^{\mathrm{ACC}}, \tag{7}$$

where $\ddot{\boldsymbol{\rho}}_b(\mathbf{q}_a)$ is the acceleration due to the motion and $\mathbf{g}_b = \begin{pmatrix}0 & g\end{pmatrix}^{\mathsf{T}}$ models the gravitation, both expressed in the base frame $O\mathrm{x}_b\mathrm{z}_b$. The bias term is denoted by $\mathbf{b}^{\mathrm{ACC}}$ and is expressed in $O\mathrm{x}_s\mathrm{z}_s$. $\boldsymbol{\mathcal{R}}_{b/s}(\mathbf{q}_a)$ is a rotation matrix that represents the rotation from frame $O\mathrm{x}_b\mathrm{z}_b$ to frame $O\mathrm{x}_s\mathrm{z}_s$.

The vector $\ddot{\boldsymbol{\rho}}_b(\mathbf{q}_a)$ can be calculated as the second derivative of $\boldsymbol{\rho}_b(\mathbf{q}_a)$ which is shown in Figure 1. Using the forward kinematic relation, the vector $\boldsymbol{\rho}_b$ can be written as

$$\boldsymbol{\rho}_b(\mathbf{q}_a) = \Upsilon_{\mathrm{ACC}}(\mathbf{q}_a), \tag{8}$$

where Υ_{ACC} is a non-linear function. Taking the derivative of (8) with respect to time twice gives

$$\ddot{\boldsymbol{\rho}}_b(\mathbf{q}_a) = \frac{d^2}{dt^2}\Upsilon_{\mathrm{ACC}}(\mathbf{q}_a) = \mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_a)\ddot{\mathbf{q}}_a + \left(\frac{d}{dt}\mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_a)\right)\dot{\mathbf{q}}_a, \tag{9}$$

where $\mathbf{J}_{\mathrm{ACC}}(\mathbf{q}_a) \triangleq \frac{\partial \Upsilon_{\mathrm{ACC}}}{\partial \mathbf{q}_a}$ is the Jacobian matrix. The final expression for the acceleration measured by the accelerometer is given by (7) and (9).

3.3 Modelling of Bias

In (7) a bias component is included in the accelerometer model, which is unknown and may vary over time. The bias component $\mathbf{b}_k = \begin{pmatrix}b_k^1 & \dots & b_k^{n_b}\end{pmatrix}^{\mathsf{T}}$ can be modelled as a random walk, i.e.,

$$\mathbf{b}_{k+1} = \mathbf{b}_k + \mathbf{w}_k^{\mathbf{b}}, \tag{10}$$

where $\mathbf{w}_k^{\mathbf{b}} = \begin{pmatrix} w_k^{\mathbf{b},1} & \dots & w_k^{\mathbf{b},n_b} \end{pmatrix}^{\mathsf{T}}$ 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 by (1). The augmented model with the bias states can then be written as

$$\begin{pmatrix} \mathbf{x}_{k+1} \\ \mathbf{b}_{k+1} \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{b}_k \end{pmatrix} + \begin{pmatrix} g(\mathbf{x}_k) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w}_k \\ \mathbf{w}_k^{\mathbf{b}} \end{pmatrix}, \tag{11}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{C}\mathbf{b}_k + \mathbf{v}_k, \tag{12}$$

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

4 Estimation Models

Four different estimation models are presented using the robot and acceleration models described in Section 3. The process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$ are in all four estimation models assumed to be Gaussian with zero mean and covariance matrices $\mathbf{Q}$ and $\mathbf{R}$, respectively.

4.1 Non-linear Estimation Model

Let the state vector be

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1^{\mathsf{T}} & \mathbf{x}_2^{\mathsf{T}} & \mathbf{x}_3^{\mathsf{T}} & \mathbf{x}_4^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \mathbf{q}_a^{\mathsf{T}} & \mathbf{q}_m^{a,\mathsf{T}} & \dot{\mathbf{q}}_a^{\mathsf{T}} & \dot{\mathbf{q}}_m^{a,\mathsf{T}} \end{pmatrix}^{\mathsf{T}}, \tag{13}$$

where $\mathbf{q}_a = \begin{pmatrix} q_{a1} & q_{a2} \end{pmatrix}^{\mathsf{T}}$ are the arm positions and $\mathbf{q}_m^a = \begin{pmatrix} q_{m1}^a & q_{m2}^a \end{pmatrix}^{\mathsf{T}}$ are the motor positions on the arm side of the gearbox. Let also the input vector $\mathbf{u} = \boldsymbol{\tau}_m^a$. Taking the derivative of $\mathbf{x}$ with respect to time and using (6) give

$$\dot{\mathbf{x}} = \begin{pmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \\ M_a^{-1}(\mathbf{x}_1)\left(-C(\mathbf{x}_1, \mathbf{x}_3) - G(\mathbf{x}_1) - N(\mathbf{x})\right) \\ \mathbf{M}_m^{-1}\left(\mathbf{u} - F(\mathbf{x}_4) + N(\mathbf{x})\right) \end{pmatrix}. \tag{14}$$

In order to use the estimation methods described in Section 2 the continuous state space model (14) has to be discretised. The time derivative of the state vector can be approximated using Euler forward according to

$$\dot{\mathbf{x}} = \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{T_s}, \tag{15}$$

where T_s is the sample time. Taking the right hand side in (14) and (15) equal to each other give the non-linear discrete state space model

$$\mathbf{x}_{k+1} = \begin{pmatrix} \mathbf{x}_{1,k} + T_s\mathbf{x}_{3,k} \\ \mathbf{x}_{2,k} + T_s\mathbf{x}_{4,k} \\ \mathbf{x}_{3,k} + T_s M_a^{-1}(\mathbf{x}_{1,k})\left(-C(\mathbf{x}_{1,k}, \mathbf{x}_{3,k}) - G(\mathbf{x}_{1,k}) - N(\mathbf{x}_k)\right) \\ \mathbf{x}_{4,k} + T_s\mathbf{M}_m^{-1}\left(\mathbf{u}_k - F(\mathbf{x}_{4,k}) + N(\mathbf{x}_k)\right) \end{pmatrix}. \tag{16}$$

The noise is modelled as a torque disturbance on the arms and motors, giving a model according to (1a) where $f(\mathbf{x}_k, \mathbf{u}_k)$ is given by the right hand side in (16) and

$$g(\mathbf{x}_k) = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ T_s M_a^{-1}(\mathbf{x}_{1,k}) & \mathbf{0} \\ \mathbf{0} & T_s \mathbf{M}_m^{-1} \end{pmatrix}, \tag{17}$$

where $\mathbf{0}$ is a two by two null matrix and the noise $\mathbf{w}_k \in \mathbb{R}^4$.

The measurements are the motor positions $\mathbf{q}_m^a$ and the end-effector acceleration $\ddot{\boldsymbol{\rho}}_s^M(\mathbf{q}_a)$. The measurement model (1b) can therefore be written as

$$\mathbf{y}_k = \begin{pmatrix} \mathbf{x}_{2,k} \\ \mathcal{R}_{b/s}(\mathbf{x}_{1,k}) \left(\ddot{\boldsymbol{\rho}}_b(\mathbf{x}_k) + \mathbf{g}_b \right) \end{pmatrix} + \mathbf{v}_k, \tag{18}$$

where $\ddot{\boldsymbol{\rho}}_b(\mathbf{x}_k)$ is given by (9) and $\mathbf{v}_k \in \mathbb{R}^4$. In (9) are $\mathbf{q}_a$ and $\dot{\mathbf{q}}_a$ given as states, whereas $\ddot{\mathbf{q}}_a$ is given by the third row in (14). The accelerometer bias $\mathbf{b}_k^{\text{ACC}} = \begin{pmatrix} b_k^{\text{ACC,x}} & b_k^{\text{ACC,z}} \end{pmatrix}^{\mathsf{T}}$ is modelled as it is described in Section 3.3 with

$$\mathcal{C} = \begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix}, \tag{19}$$

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

4.2 Estimation Model with Linear Dynamic

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

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1^{\mathsf{T}} & \mathbf{x}_2^{\mathsf{T}} & \mathbf{x}_3^{\mathsf{T}} \end{pmatrix} = \begin{pmatrix} \mathbf{q}_a^{\mathsf{T}} & \dot{\mathbf{q}}_a^{\mathsf{T}} & \ddot{\mathbf{q}}_a^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}}, \tag{20}$$

where $\mathbf{q}_a = \begin{pmatrix} q_{a1} & q_{a2} \end{pmatrix}^{\mathsf{T}}$ are the arm positions. This yields the following state space model in discrete time

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{G}_{\mathbf{u}}\mathbf{u}_k + \mathbf{G}_{\mathbf{w}}\mathbf{w}_k, \tag{21}$$

where $\mathbf{u}_k$ is the input vector and the process noise $\mathbf{w}_k \in \mathbb{R}^2$. The constant matrices are given by

$$\mathbf{F} = \begin{pmatrix} \mathbf{I} & T_s\mathbf{I} & \frac{T_s^2}{2}\mathbf{I} \\ \mathbf{0} & \mathbf{I} & T_s\mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad \mathbf{G}_{\mathbf{u}} = \mathbf{G}_{\mathbf{w}} = \begin{pmatrix} \frac{T_s^3}{6}\mathbf{I} \\ \frac{T_s^2}{2}\mathbf{I} \\ T_s\mathbf{I} \end{pmatrix} \tag{22}$$

where $\mathbf{I}$ and $\mathbf{0}$ are two by two identity and null matrices, respectively. The input, $\mathbf{u}_k$, is the arm jerk reference, i.e., the differentiated arm angular acceleration reference.

The motor positions are calculated from (6a) where the spring is linear, i.e.,

$$T(\mathbf{q}_m^a - \mathbf{q}_a) = \begin{pmatrix} k_1 & 0 \\ 0 & k_2 \end{pmatrix} (\mathbf{q}_m^a - \mathbf{q}_a) = \mathbf{K} \cdot (\mathbf{q}_m^a - \mathbf{q}_a). \tag{23}$$

The damping term is small compared to the other terms [Karlsson and Norrlöf, 2005] and is therefore neglected for simplicity. The measurement model for the accelerometer is the same as in (18) where $\ddot{\mathbf{q}}_{a,k}$ is a state in this case. The measurement model can now be written as

$$\mathbf{y}_k = \begin{pmatrix} \mathbf{q}_{m,k}^a \\ \mathcal{R}_{b/s}(\mathbf{x}_{1,k}) \left(\ddot{\rho}_b(\mathbf{x}_k) + \mathbf{g}_b \right) \end{pmatrix} + \mathbf{v}_k. \tag{24}$$

where $\mathbf{v}_k \in \mathbb{R}^4$ and

$$\mathbf{q}_{m,k}^a = \mathbf{x}_{1,k} + \mathbf{K}^{-1} \left(M_a(\mathbf{x}_{1,k}) \mathbf{x}_{3,k} + C(\mathbf{x}_{1,k}, \mathbf{x}_{2,k}) + G(\mathbf{x}_{1,k}) \right), \tag{25a}$$

$$\ddot{\rho}_b(\mathbf{x}_k) = \mathbf{J}_{\text{ACC}}(\mathbf{x}_{1,k}) \mathbf{x}_{3,k} + \left(\frac{d}{dt} \mathbf{J}_{\text{ACC}}(\mathbf{x}_{1,k}) \right) \mathbf{x}_{2,k}. \tag{25b}$$

Once again, the accelerometer bias $\mathbf{b}_k^{\text{ACC}}$ is modelled according to Section 3.3. However, the estimation result is improved if bias components for the motor measurements also are included. The explanation is model errors in the measurement equation. The total bias component is $\mathbf{b}_k = \begin{pmatrix} \mathbf{b}_k^{q_m,\text{T}} & \mathbf{b}_k^{\text{ACC,T}} \end{pmatrix}^{\text{T}}$, where $\mathbf{b}_k^{q_m} = \begin{pmatrix} b_k^{q_{m1}} & b_k^{q_{m2}} \end{pmatrix}^{\text{T}}$ and $\mathbf{b}_k^{\text{ACC}} = \begin{pmatrix} b_k^{\text{ACC,x}} & b_k^{\text{ACC,z}} \end{pmatrix}^{\text{T}}$. The matrix $\mathcal{C}$ in (12) is for this model a four by four identity matrix.

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 (14) as the arm angular acceleration and use that as an input signal with (13) as state vector give the following model,

$$\dot{\mathbf{x}} = \begin{pmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \\ \ddot{\mathbf{q}}_a^{\text{IN}} \\ \mathbf{M}_m^{-1} \left(\mathbf{u} - F(\mathbf{x}_4) + N(\mathbf{x}) \right) \end{pmatrix}, \tag{26}$$

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

$$\dot{\mathbf{x}} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_m^{-1}\mathbf{K} & -\mathbf{M}_m^{-1}\mathbf{K} & \mathbf{M}_m^{-1}\mathbf{D} & -\mathbf{M}_m^{-1}(\mathbf{D} + \mathbf{F}_d) \end{pmatrix} \mathbf{x} + \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_m^{-1} \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{q}}_a^{\text{IN}} \\ \mathbf{u} \end{pmatrix}, \tag{27}$$

where

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

The linear state space model is discretised using *zero order hold* (ZOH). ZOH is used instead of Euler forward since it gives a better result and an explicit solution exist when the model is linear. The only remaining measurements are the motor positions, which give a linear measurement model according to

$$\mathbf{y}_k = \begin{pmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \end{pmatrix} \mathbf{x}_k + \mathbf{v}_k, \tag{28}$$

where $\mathbf{v}_k \in \mathbb{R}^2$. Note that the arm angular acceleration $\ddot{\mathbf{q}}_a^{\mathrm{IN}}$ is not measured direct, instead it has to be calculated from the accelerometer signal using (7) and (9), which is possible as long as the Jacobian $\mathbf{J}_{\mathrm{ACC}}(\mathbf{x}_{1,k})$ has full rank.

4.4 Non-linear Estimation Model with Acceleration as Input

The linear model presented in Section 4.3 is here reformulated as a non-linear model. Given the model in (26) and using Euler forward for discretisation give

$$\mathbf{x}_{k+1} = \begin{pmatrix} \mathbf{x}_{1,k} + T_s \mathbf{x}_{3,k} \\ \mathbf{x}_{2,k} + T_s \mathbf{x}_{4,k} \\ \mathbf{x}_{3,k} + T_s \ddot{\mathbf{q}}_{a,k}^{\mathrm{IN}} \\ \mathbf{x}_{4,k} + T_s \mathbf{M}_m^{-1} \left(\mathbf{u}_k - F(\mathbf{x}_{4,k}) + N(\mathbf{x}_k) \right) \end{pmatrix}. \tag{29}$$

The noise model is assumed to be the same as in Section 4.1, and the measurement is the same as in (28).

5 Experiments on an ABB IRB4600 Robot

5.1 Experimental Setup

The accelerometer used in the experiments is a triaxial accelerometer from Crossbow Technology, with a range of ±2 g, and a sensitivity of 1 V/g [Crossbow Technology, 2004]. The orientation and position of the accelerometer are estimated using the method described in Axelsson and Norrlöf [2012]. All measured signals 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 2 and it is programmed such that only joint two and three 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].

It is not possible to get measurements of the true state variables, as is the case for simulation, instead, only the true trajectory of the tool, more precise the TCP, x and z coordinates, 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 are therefore needed, which can introduce small errors. Another source of

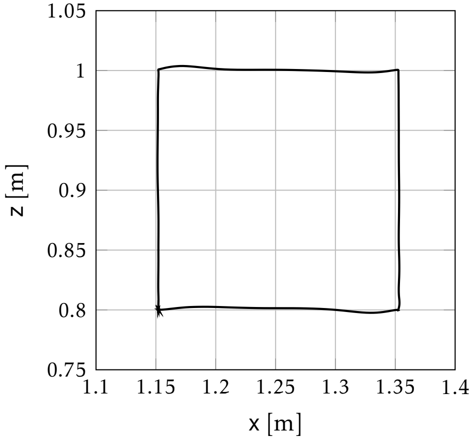


Figure 2: *Measured path for the end-effector used for experimental evaluations.*

error is the accuracy of the programmed TCP in the control system of the robot. The estimated data is 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.

5.2 Experimental Results

Six observers using the four different estimation models described in Section 4 are evaluated. The observers are based on the EKF, EKS, PF or a linear dynamic observer using pole placement [Franklin et al., 2002].

OBS1: EKF with the non-linear model in Section 4.1.

OBS2: EKS with the non-linear model in Section 4.1.

OBS3: EKF with the linear state model and non-linear measurement model in Section 4.2.

OBS4: PF with the linear state model and non-linear measurement model in Section 4.2.

OBS5: EKF with the non-linear model where the acceleration of the end-effector is input, see Section 4.4 .

OBS6: Linear dynamic observer using pole placement with the linear model where the acceleration of the end-effector is input, see Section 4.3. [De Luca et al., 2007]

The only measured quantity, to compare the estimates with, is the measured tool position, as was mentioned in Section 5.1. Therefore, the estimated arm angles are used to compute an estimate of the TCP using the kinematic relation, i.e.,

$$\begin{pmatrix} \hat{\mathsf{x}}_k \\ \hat{\mathsf{z}}_k \end{pmatrix} = \Upsilon_{\text{TCP}}(\hat{\mathbf{q}}_{a,k}), \tag{30}$$

where $\hat{\mathbf{q}}_{a,k}$ is the result from one of the six observers at time k. The result is presented with diagrams of the true and estimated paths for the horizontal parts of the path in Figure 2. The path error

$$\mathsf{e}_k = \sqrt{(\mathsf{x}_k - \hat{\mathsf{x}}_k)^2 + (\mathsf{z}_k - \hat{\mathsf{z}}_k)^2}, \tag{31}$$

where x_k, $\hat{\mathsf{x}}_k$, z_k and $\hat{\mathsf{z}}_k$ are the true and estimated position for the tool in the x- and z-direction at time k, as well as the *root mean square error* (RMSE)

$$\epsilon = \sqrt{\frac{1}{N}\sum_{k=1}^{N} \mathsf{e}_k^2}, \tag{32}$$

where N is the number of samples, are also used for 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 non-linear model) will give a better result than OBS1 (EKF with non-linear model) since the EKS uses both previous and future measurements. This is not the case as can be seen in Figure 3. The reason for this can be model errors and in particular the non-linearities in the joint stiffness.

One interesting observation is the higher orders of oscillations in the estimated paths. The oscillations can be reduced if the covariance matrix $\mathbf{Q}$ for the process noise is decreased. However, this leads to a larger path error. The RMSE values can be found in Table 1. The table shows that OBS2 is slightly better than OBS1.

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 $\mathbf{H}_k$ in the EKF 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. OBS2 is slower since an EKF is used first and then the backward time recursion in (4). 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 off-line.

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 mea-

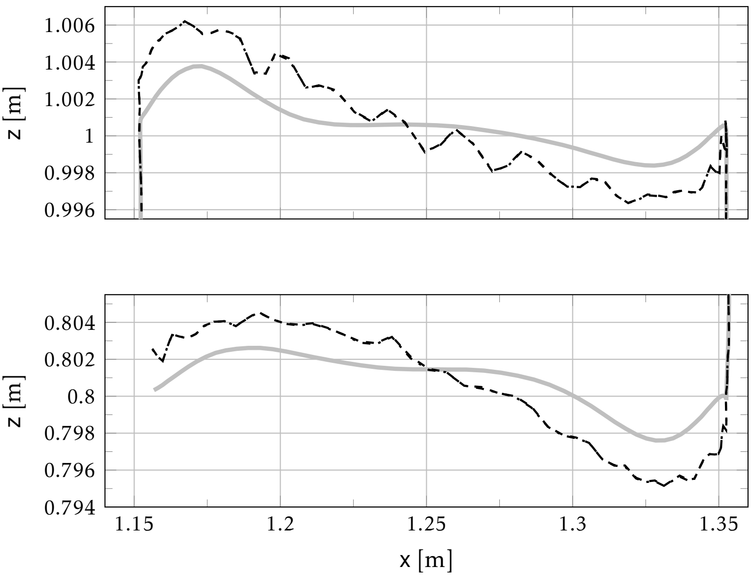


Figure 3: The two horizontal sides of the true path (solid), and the estimated path using OBS1 (dashed), and OBS2 (dash-dot).

Table 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
ϵ	1.5704	1.5664	2.3752	1.5606	1.6973	1.7624

surement can be used or not. OBS1 is chosen as the one that will be compared with the other observers.

OBS3 and OBS4. Figure 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 RMSE values of the path error for OBS3 and OBS4 are presented in Table 1. The RMSE for OBS4 is approximately two-thirds of the RMSE for OBS3.

The MATLAB implementation of OBS3 is almost real-time, just above four seconds, and the execution time for OBS4 is about ten hours. The execution time for OBS3 can be reduced to real-time without losing performance if the measurements are decimated to approximately 200 Hz.

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 compared with the other observers since the path error is of more interest in this paper.

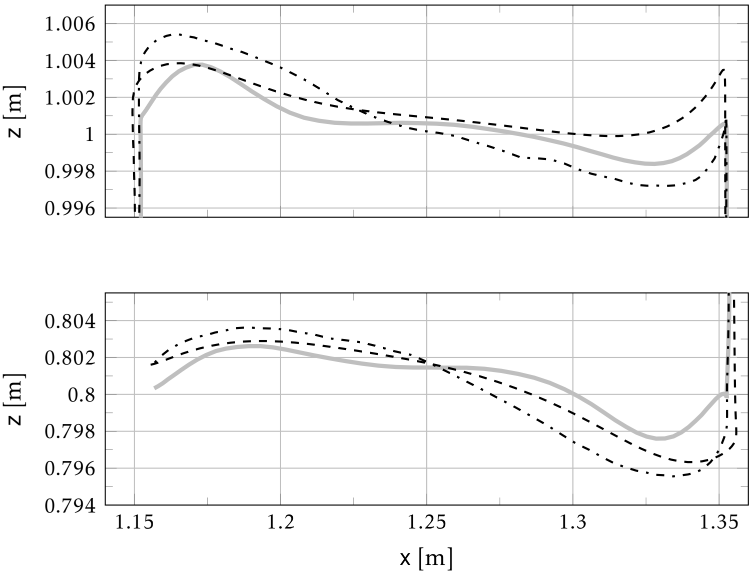


Figure 4: *The two horizontal sides of the true path (solid), and the estimated path using OBS3 (dashed), and OBS4 (dash-dot).*

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 5. It can also be seen that OBS5 (linear model with acceleration as input using EKF) performs a bit better. 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. The RMSE values of the path error are showed in Table 1.

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.

Summary

The three observers OBS1, OBS4, and OBS5 are the best ones from each pair. From Table 1 it can be seen that OBS1 and OBS4 have the same performance and that OBS5 is a bit worse, see also the path errors in Figure 6. 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

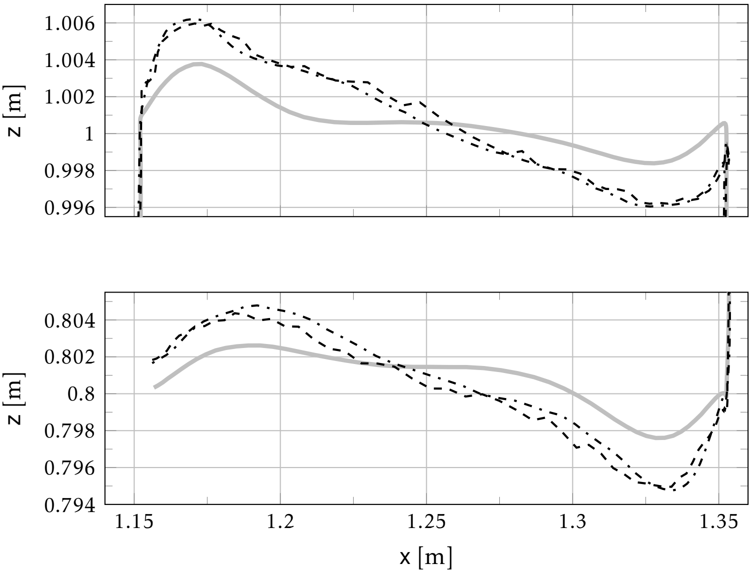


***Figure 5:** The two horizontal sides of the true path (solid), and the estimated path using OBS5 (dashed), and OBS6 (dash-dot).*

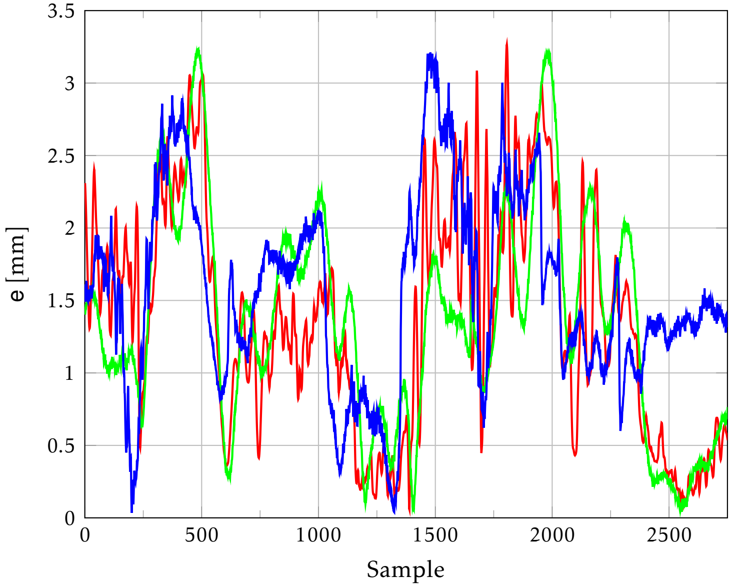


***Figure 6:** The path error for the estimated path using OBS1 (red), OBS4 (green), and OBS5 (blue).*

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 rigid body motion on the arm side is needed which is a difference from the other two. However, a non-linear 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. 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 remains.

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.

6 Conclusions

A sensor fusion approach to estimate the end-effector position 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 estimation problem and has been evaluated on experimental data from a state of the art industrial robot.

Different types of observers where both the estimation model and the filter were changed, have been used. The three observers with the best performance were

a) an EKF using a non-linear dynamic model,

b) a particle filter using a linear dynamic model, and

c) an EKF with a non-linear model, where the acceleration of the end-effector is used as an input instead of a measurement.

The performances of these three observers were 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.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC.

Bibliography

Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Information and System Sciences Series. Prentice Hall Inc., Englewood Cliffs, NJ, USA, 1979.

Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 126–132, Dubrovnik, Croatia, September 2012.

Patrik Axelsson and Mikael Norrlöf. Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 283–288, Dubrovnik, Croatia, September 2012.

Torgny Brogårdh. Present and future robot control development—an industrial perspective. *Annual Reviews in Control*, 31(1):69–79, 2007.

Crossbow Technology. Accelerometers, High Sensitivity, LF Series, CXL02LF3, January 2004. Available at `http://www.xbow.com`.

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.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10 (3):197–208, 2000.

Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, NY, USA, 2001.

Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall Inc., Upper Saddle River, NJ, USA, fourth edition, 2002.

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.

Fredrik Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur, Lund, Sweden, first edition, 2010.

Robert Henriksson, Mikael Norrlöf, 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.

Rahim Jassemi-Zargani and Dan Necsulescu. 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.

Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64. Academic Press, New York, NY, USA, 1970.

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.

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.

Leica Geosystems. Case study ABB robotics - Västerås, 2008. Available at `http://metrology.leica-geosystems.com/en/index.htm`.

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.

Gerasimos G. Rigatos. Particle filtering for state estimation in nonlinear industrial systems. *IEEE Transactions on Instrumentation and Measurement*, 58(11): 3885–3900, November 2009.

Erik Wernholt and Stig Moberg. Nonlinear gray-box identification using local models applied to industrial robots. *Automatica*, 47(4):650–660, April 2011.

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`, 19 October 2004.

Paper C

Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation With Applications to Kalman Filtering

Authors: Patrik Axelsson and Fredrik Gustafsson

Edited version of the paper:

Patrik Axelsson and Fredrik Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *Submitted to IEEE Transactions on Automatic Control,* 2012.

Discrete-time Solutions to the Continuous-time Differential Lyapunov Equation With Applications to Kalman Filtering

Patrik Axelsson and Fredrik Gustafsson

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
fredrik.gustafsson@liu.se

Abstract

Prediction and filtering of continuous-time stochastic processes often require a solver of a *continuous-time differential Lyapunov equation* (CDLE), for example the time update in the Kalman filter. Even though this can be recast into an *ordinary differential equation* (ODE), where standard solvers can be applied, the dominating approach in Kalman filter applications is to discretise the system and then apply the *discrete-time difference Lyapunov equation* (DDLE). To avoid problems with stability and poor accuracy, oversampling is often used. This contribution analyses over-sampling strategies, and proposes a novel low-complexity analytical solution that does not involve over-sampling. The results are illustrated on Kalman filtering problems in both linear and non-linear systems.

1 Introduction

Numerical solvers for *ordinary differential equations* (ODE) is a well studied area [Hairer et al., 1987]. The related area of Kalman filtering (state prediction and state estimation) in continuous-time models was also well studied during the first two decades of the Kalman filter, see for instance Jazwinski [1970], while the more recent literature such as the standard reference Kailath et al. [2000] focuses on discrete time filtering only. A specific example, with many applications in practice, is Kalman filtering based on a continuous-time state space model with discrete-time measurements, known as continuous-discrete filtering. The *Kalman filter* (KF) here involves a time update that integrates the first and second order moments from one sample time to the next one. The second order moment is a covariance matrix, and it governs a *continuous-time differential Lyapunov* *equation* (CDLE). The problem can easily be recast into a vectorised ODE problem and standard solvers can be applied. For linear ODE's, the time update of the linear KF can thus be solved analytically, and for non-linear ODE's, the time update of the extended KF has a natural approximation in continuous-time. One problem is the large dimension of the resulting ODE. Another possible explanation why the continuous-time update is not used is the common use of discrete-time models in Kalman filter applications, so practitioners often tend to discretise the state space model first to fit the discrete-time Kalman filter time update. Despite a closed form solution exists, this involves approximations that lead to well known problems with accuracy and stability. The *ad-hoc* remedy is to oversample the system, so a large number of small time updates are taken between the sampling times of the observations.

In literature, different methods are proposed to solve the continuous-discrete non-linear filtering problem using *extended Kalman filters* (EKF). A common way is to use a first or second order Taylor approximation as well as a Runge-Kutta method in order to integrate the first order moments, see e.g. LaViola [2003]; Rao et al. [2011]; Mallick et al. [2012]. They all have in common that the CDLE is replaced by the *discrete-time difference Lyapunov equation* (DDLE), used in discrete-time Kalman filters. A more realistic way is to solve the CDLE as is presented in Bagterp Jörgensen et al. [2007]; Mazzoni [2008], where the first and second order moments are integrated numerically. A comparison between different solutions is presented in Frogerais et al. [2012], where the method proposed by the authors discretises the stochastic differential equation (SDE) using a Runge-Kutta solver. The other methods in Frogerais et al. [2012] have been proposed in the literature before, e.g. LaViola [2003]; Mazzoni [2008]. Related work using different approximations to continuous integration problems in non-linear filtering also appears in Särkkä [2007]; Zhang et al. [2005] for unscented Kalman filters and Arasaratnam and Haykin [2009] for cubature Kalman filters.

This contribution takes a new look at this fundamental problem. First, we review different approaches for solving the CDLE in a coherent mathematical framework. Second, we analyse in detail the stability conditions for oversampling, and based on this we can explain why even simple linear models need a large rate of oversampling. Third, we make a new straightforward derivation of a low-complexity algorithm to compute the solution with arbitrary accuracy. Numerical stability and computational complexity are analysed for the different approaches. It turns out that the low-complexity algorithm has better numerical properties compared to the other methods, and at the same time a computational complexity in the same order. Fourth, the methods are extended to non-linear system where the *extended Kalman filter* (EKF) is used. We illustrate the results on both a simple second order linear spring-damper system, and a non-linear spring-damper system relevant for mechanical systems, in particular robotics.

2 Mathematical Framework and Background

2.1 Linear Stochastic Differential Equations

Consider the linear *stochastic differential equation* (SDE)

$$d\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)dt + \mathbf{G}d\boldsymbol{\beta}(t), \tag{1}$$

for $t \geq 0$, where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the state vector and $\boldsymbol{\beta}(t) \in \mathbb{R}^{n_\beta}$ is a vector of Wiener processes with $\mathrm{E}\left[d\boldsymbol{\beta}(t)d\boldsymbol{\beta}(t)^\mathsf{T}\right] = \mathbf{Q}dt$. The matrices $\mathbf{A} \in \mathbb{R}^{n_x \times n_x}$ and $\mathbf{G} \in \mathbb{R}^{n_x \times n_\beta}$ are here assumed to be constants, but they can also be time varying. It is also possible to include a control signal $\mathbf{u}(t)$ in (1) but that is omitted here for brevity.

Given an initial state $\hat{\mathbf{x}}(0)$ with covariance $\mathbf{P}(0)$, we want to solve the SDE to get $\hat{\mathbf{x}}(t)$ and $\mathbf{P}(t)$ at an arbitrary time instance. By multiplying both sides with the integrating factor $e^{-\mathbf{A}s}$ and integrating over the time interval gives

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0) + \underbrace{\int_0^t e^{\mathbf{A}(t-s)}\mathbf{G}d\boldsymbol{\beta}(s)\,\mathrm{d}s}_{\triangleq \mathbf{v}_d(t)}. \tag{2}$$

The goal is to get a discrete-time update of the mean and covariance, from $\hat{\mathbf{x}}(kh)$ and $\mathbf{P}(kh)$ to $\hat{\mathbf{x}}((k+1)h)$ and $\mathbf{P}((k+1)h)$, respectively. The time interval h may correspond to one sample interval, or be a fraction of it in the case of oversampling. The latter case will be discussed in detail later on. For simplicity, the time interval $[kh, (k+1)h]$ will be denoted as $[0, t]$ below.

The discrete-time equivalent noise $\mathbf{v}_d(t)$ has covariance given by

$$\mathbf{Q}_d(t) = \int_0^t e^{\mathbf{A}(t-s)}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}e^{\mathbf{A}^\mathsf{T}(t-s)}\,\mathrm{d}s = \int_0^t e^{\mathbf{A}\tau}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}e^{\mathbf{A}^\mathsf{T}\tau}\,\mathrm{d}\tau, \tag{3}$$

We immediately get an expression for the first and second order moments of the SDE solution over one time interval as

$$\hat{\mathbf{x}}(t) = e^{\mathbf{A}t}\hat{\mathbf{x}}(0), \tag{4a}$$

$$\mathbf{P}(t) = e^{\mathbf{A}t}\mathbf{P}(0)e^{\mathbf{A}^\mathsf{T}t} + \mathbf{Q}_d(t). \tag{4b}$$

From (2) and (3), we can also recover the continuous-time update formulas

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{A}\hat{\mathbf{x}}(t), \tag{5a}$$

$$\dot{\mathbf{P}}(t) = \mathbf{A}\mathbf{P}(t) + \mathbf{P}(t)\mathbf{A}^\mathsf{T} + \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}, \tag{5b}$$

by, a bit informally, taking the expectation of (2) and then dividing both sides with t and letting $t \to 0$. Here, (5a) is an ordinary ODE and (5b) is the *continuous-time differential Lyapunov equation* (CDLE).

Thus, there are two conceptually different alternatives. Either, solve the integral (3) defining $\mathbf{Q}_d(t)$ and use (4), or solve the ODE and CDLE in (5). These two well-known alternatives are outlined below.

2.2 Matrix Fraction Decomposition

There are two ways to compute the integral (3) described in literature. Both are based on computing the matrix exponential of the matrix

$$\mathbf{H} = \begin{pmatrix} \mathbf{A} & \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} \\ \mathbf{0} & -\mathbf{A}^\mathsf{T} \end{pmatrix}. \tag{6}$$

The result is a block matrix in the form

$$e^{\mathbf{H}t} = \begin{pmatrix} \mathbf{M}_1(t) & \mathbf{M}_2(t) \\ \mathbf{0} & \mathbf{M}_3(t) \end{pmatrix}, \tag{7}$$

where the structure implies that $\mathbf{M}_1(t) = e^{\mathbf{A}t}$ and $\mathbf{M}_3(t) = e^{-\mathbf{A}^\mathsf{T}t}$. As shown in Van Loan [1978], the solution to (3) can be computed as

$$\mathbf{Q}_d(t) = \mathbf{M}_2(t)\mathbf{M}_1(t)^\mathsf{T} \tag{8}$$

This is immediately verified by taking the time derivative of the definition (3) and the matrix exponential (7), and verifying that the involved Taylor expansions are equal.

Another alternative known as matrix fraction decomposition, which solves a matrix valued ODE, given in Grewal and Andrews [2008]; Särkkä [2006], is to compute $\mathbf{P}(t)$ directly. Using the initial conditions $\begin{pmatrix}\mathbf{P}(0) & \mathbf{I}\end{pmatrix}^\mathsf{T}$ for the ODE gives

$$\mathbf{P}(t) = \underbrace{\left(\mathbf{M}_1(t)\mathbf{P}(0) + \mathbf{M}_2(t)\right)}_{\triangleq \mathbf{C}(t)} \mathbf{M}_3(t)^{-1}. \tag{9}$$

The two alternatives in (8) and (9) are apparently algebraically the same.

There are also other alternatives described in literature. First, the integral in (3) can of course be solved with numerical methods such as the trapezoidal method or the rectangle method. In Rome [1969] the integral is solved analytically in the case that $\mathbf{A}$ is diagonalisable. However, not all matrices are diagonalisable, and even in such cases, this method is not numerically stable [Higham, 2008].

2.3 Vectorisation Method

The ODEs for the first and second order moments in (5) can be solved using a method based on vectorisation. The vectorisation approach for matrix equations is well known and especially for the CDLE, see e.g. Davison [1975]; Gajić and Qureshi [1995]. The method uses the fact that (5) can be converted to one single

ODE by introducing an extended state vector

$$\mathbf{z}(t) = \begin{pmatrix} \mathbf{z}_{\mathbf{x}}(t) \\ \mathbf{z}_{\mathbf{P}}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{x}(t) \\ \operatorname{vech} \mathbf{P}(t) \end{pmatrix}, \tag{10}$$

$$\dot{\mathbf{z}}(t) = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathbf{P}} \end{pmatrix} \mathbf{z}(t) + \begin{pmatrix} \mathbf{0} \\ \operatorname{vech} \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}} \end{pmatrix} = \mathbf{A}_{\mathbf{z}}\mathbf{z}(t) + \mathbf{B}_{\mathbf{z}}, \tag{11}$$

where $\mathbf{A}_{\mathbf{P}} = \mathbf{D}^{\dagger}\left(\mathbf{I} \otimes \mathbf{A} + \mathbf{A} \otimes \mathbf{I}\right)\mathbf{D}$. Here, vech denotes the half-vectorisation operator, $\otimes$ is the Kronecker product and $\mathbf{D}$ is a duplication matrix, see Appendix A for details.

The solution of the ODE (11) is given by [Rugh, 1996]

$$\mathbf{z}(t) = e^{\mathbf{A}_{\mathbf{z}}t}\mathbf{z}(0) + \int_0^t e^{\mathbf{A}_{\mathbf{z}}(t-s)}\, \mathrm{d}s \mathbf{B}_{\mathbf{z}}. \tag{12}$$

One potentially prohibitive drawback with the solution in (12) is its computational complexity, in particular for the matrix exponential. The dimension of the extended state $\mathbf{z}$ is $n_z = n_x + n_x(n_x + 1)/2$, giving a computational complexity of $\mathcal{O}(n_x^6)$. Section 4 presents a way to rewrite (12) to give a complexity of $\mathcal{O}(n_x^3)$ instead of $\mathcal{O}(n_x^6)$.

2.4 Discrete-time Recursion of the CDLE

The solution in (4b) using (8), the matrix fraction decomposition in (9), and the ODE (12) evaluated at the discrete-time instances $t = kh$ and $t = (k+1)h$ give the following recursive update formulas

$$\mathbf{P}((k+1)h) = e^{\mathbf{A}h}\mathbf{P}(kh)e^{\mathbf{A}^{\mathsf{T}}h} + \mathbf{M}_2(h)\mathbf{M}_1(h)^{\mathsf{T}} \tag{13}$$

$$\mathbf{P}((k+1)h) = \big(\mathbf{M}_1(h)\mathbf{P}(kh) + \mathbf{M}_2(h)\big)\mathbf{M}_3(h)^{-1} \tag{14}$$

$$\mathbf{z}((k+1)h) = e^{\mathbf{A}_{\mathbf{z}}h}\mathbf{z}(kh) + \int_0^h e^{\mathbf{A}_{\mathbf{z}}s}\, \mathrm{d}s \mathbf{B}_{\mathbf{z}}, \tag{15}$$

which can be used in the Kalman filter time update.

2.5 The Matrix Exponential

Sections 2.1 to 2.3 shows that the matrix exponential function is a working horse to solve the linear SDE. At this stage, numerical routines for the matrix exponential are important to understand. One key approach is based on the following identity and Taylor expansion [Moler and Van Loan, 2003]

$$e^{\mathbf{A}h} = \left(e^{\mathbf{A}h/m}\right)^m \approx \left(\mathbf{I} + \left(\frac{\mathbf{A}h}{m}\right) + \cdots + \frac{1}{p!}\left(\frac{\mathbf{A}h}{m}\right)^p\right)^m \triangleq e_{p,m}(\mathbf{A}h). \tag{16}$$

In fact, the Taylor expansion is a special case of a more general Padé approximation of $e^{\mathbf{A}h/m}$ [Moler and Van Loan, 2003], but this does not affect the discussion here.

The eigenvalues of $\mathbf{A}h/m$ are the eigenvalues of $\mathbf{A}$ scaled with h/m, and thus they can be arbitrarily small if m is chosen large enough for any given h. Further, the p'th order Taylor expansion converges faster for smaller eigenvalues of $\mathbf{A}h/m$. Finally, the power function $\mathbf{M}^m$ is efficiently implemented by squaring the matrix $\mathbf{M}$ in total $\log_2(m)$ times, assuming that m is chosen to be a power of 2. We will denote this approximation with $e_{p,m}(\mathbf{A}h)$.

A good approximation $e_{p,m}(\mathbf{A}h)$ is characterised by the following properties:

- Stability. If $\mathbf{A}$ has all its eigenvalues in the left half plane, then $e_{p,m}(\mathbf{A}h)$ should have all its eigenvalues inside the unit circle.
- Accuracy. If p and m are chosen large enough, the error $\left\| e^{\mathbf{A}h} - e_{p,m}(\mathbf{A}h) \right\|$ should be small.

Since the Taylor expansion converges, we have trivially that

$$\lim_{p\to\infty} e_{p,m}(\mathbf{A}h) = \left(e^{\mathbf{A}h/m}\right)^m = e^{\mathbf{A}h}. \tag{17a}$$

From the property $\lim_{x\to\infty}(1 + a/x)^x = e^a$, we also have

$$\lim_{m\to\infty} e_{p,m}(\mathbf{A}h) = e^{\mathbf{A}h}. \tag{17b}$$

Finally, from Higham [2008] we have that

$$\left\| e^{\mathbf{A}h} - e_{p,m}(\mathbf{A}h) \right\| \leq \frac{\|\mathbf{A}\|^{p+1}\, h^{p+1}}{m^p(p+1)!} e^{\|\mathbf{A}\|h}. \tag{17c}$$

However, for any finite p and $m > 1$, then all terms in the binomial expansion of $e_{p,m}(\mathbf{A}h)$ are different from the Taylor expansion of $e^{\mathbf{A}h}$, except for the first two terms which are always $\mathbf{I} + \mathbf{A}h$.

The complexity of the approximation $e_{p,m}(\mathbf{A}h)$, where $\mathbf{A} \in \mathbb{R}^{n_x \times n_x}$, is in the order of $(\log_2(m) + p)\, n_x^3$, where pn_x^3 multiplications are required to compute $\mathbf{A}^p$ and $\log_2(m) n_x^3$ multiplications are needed for squaring the Taylor expansion $\log_2(m)$ times.

Standard numerical integration routines can be recast into this framework as well. For instance, a standard tuning of the fourth order Runge-Kutta method for a linear ODE results in $e_{4,1}(\mathbf{A}h)$.

2.6 Solution using Approximate Discretisation

We have now outlined three methods to compute the exact time update in the discrete-time Kalman filter. These should be equivalent up to numerical issues, and will be treated as one approach in the sequel.

Another common approach in practice, in particular in Kalman filter applications, is to assume that the noise is piece-wise constant giving the discrete-time system

$$\mathbf{x}(k+1) = \mathbf{F}_h \mathbf{x}(k) + \mathbf{G}_h \mathbf{v}_h(k), \tag{18a}$$

$$\operatorname{Cov}(\mathbf{v}_h(k)) = \mathbf{Q}_h, \tag{18b}$$

where $\mathbf{F}_h = e_{p,m}(\mathbf{A}h)$, $\mathbf{G}_h = \int_0^h e^{\mathbf{A}t}\,\mathrm{d}t\mathbf{G}$, and $\mathbf{Q}_h = h\mathbf{Q}$. The discrete-time Kalman filter update equations

$$\hat{\mathbf{x}}(k+1) = \mathbf{F}_h \hat{\mathbf{x}}(k), \tag{19a}$$

$$\mathbf{P}(k+1) = \mathbf{F}_h \mathbf{P}(k) \mathbf{F}_h^{\mathsf{T}} + \mathbf{G}_h \mathbf{Q}_h \mathbf{G}_h^{\mathsf{T}}, \tag{19b}$$

are then used, where (19a) is a difference equation and (19b) is the *discrete-time difference Lyapunov equation* (DDLE). The update equations (19) are exact for the discrete-time model (18). However, there are several approximations involved in the discretisation step:

- First, $\mathbf{F}_h = e_{p,m}(\mathbf{A}h)$ is an approximation of the exact solution given by $\mathbf{F}_h = e^{\mathbf{A}h}$. It is quite common in practice to use Euler sampling defined by $\mathbf{F}_h = \mathbf{I} + \mathbf{A}h = e_{1,1}(\mathbf{A}h)$.
- Even without process noise, the update formula for $\mathbf{P}$ in (19b) is not equivalent to (5b).
- The discrete-time noise $\mathbf{v}_h(t)$ is an aggregation of the total effect of the Wiener process $d\beta(t)$ during the interval $[t, t+h]$, as given in (3). The conceptual drawback is that the Wiener process $d\beta(t)$ is not aware of the sampling time chosen by the user.

One common remedy is to introduce oversampling. This means that (19) is iterated m times using the sampling time h/m. When oversampling is used, the covariance matrix for the discrete-time noise $\mathbf{v}_h(k)$ should be scaled as $\mathbf{Q}_h = h\mathbf{Q}/m$. In this way, the problems listed above will asymptotically vanish as m increases. However, as we will demonstrate, quite large an m can be needed even for some quite simple systems.

2.7 Summary of Contributions

- Section 3 gives explicit conditions for an upper bound of the sample time h such that a stable continuous-time model remains stable after discretisation. The analysis treats stability of both $\mathbf{x}$ and $\mathbf{P}$, for the case of Euler sampling $e_{1,m}(\mathbf{A})$, for the solution of the SDE given by the ODE (11). Results for $p > 1$ are also briefly discussed. See Table 1 for a summary when the vectorised solution is used.
- Section 4 presents a reformulation of the solution to the ODE (11), where the computational complexity has been decreased from $(\log_2(m) + p)\left(n_x^2/2\right)^3$ to $(\log_2(m) + p + 43)\, n_x^3$.
- Section 5 shows how the computational complexity and the numerical properties differs between the different methods.

Table 1: *Summary of approximations $e_{p,m}(\mathbf{A}h)$ of $e^{\mathbf{A}h}$. The stability region ($h < h_{\max}$) is parametrised in λ_i which are the eigenvalues to $\mathbf{A}$. In the case of Runge-Kutta, only real eigenvalues are considered.*

Approach	p	m	Stability region ($h_{\max}$)
Euler sampling	1	1	$-\frac{2\mathfrak{Re}\{\lambda_i\}}{\|\lambda_i\|^2}$
Oversampled Euler	1	$m > 1$	$-\frac{2m\mathfrak{Re}\{\lambda_i\}}{\|\lambda_i\|^2}$
Runge-Kutta	4	1	$-\frac{2.7852}{\lambda_i}$, $\lambda_i \in \mathbb{R}$
Oversampled Runge-Kutta	4	$m > 1$	$-\frac{2.7852m}{\lambda_i}$, $\lambda_i \in \mathbb{R}$

- Section 6 presents a second order spring-damper example to demonstrate the advantages using a continuous-time update.

- Section 7 discusses implications for non-linear systems, and investigates a non-linear system inspired by applications in robotics.

3 Stability Analysis

It is known that the CDLE in (5b) has a unique positive solution $\mathbf{P}(t)$ if $\mathbf{A}$ is Hurwitz[1], $\mathbf{GQG}^\mathsf{T} \geq 0$, the pair $(\mathbf{A}, \sqrt{\mathbf{GQG}^\mathsf{T}})$ is controllable, and $\mathbf{P}(0) > 0$ [Gajić and Qureshi, 1995]. We want to show that a stable continuous-time system results in a stable discrete-time recursion. We therefore assume that the continuous-time ODE describing the state vector $\mathbf{x}(t)$ is stable, hence the eigenvalues λ_i, $i = 1, \ldots, n_x$ to $\mathbf{A}$ are assumed to be in the left half plane, i.e., $\mathfrak{Re}\{\lambda_i\} < 0$, $i = 1, \ldots, n_x$. It will also be assumed that the remaining requirements are fulfilled.

For the methods described in Section 2.2 we have that $\mathbf{H}$ has the eigenvalues $\pm\lambda_i$, $i = 1, \ldots, n_x$, where λ_i are the eigenvalues of $\mathbf{A}$. This follows from the structure of $\mathbf{H}$. Hence, the matrix exponential $e^{\mathbf{H}t}$ will have terms that tend to infinity and zero with the same exponential rate when t increases. However, the case $t = h$ is of most interest, where h is finite. Note that a small/large sample time depends strongly on the system dynamics. Even if the matrix $e^{\mathbf{H}t}$ is ill-conditioned, the product (8) and the ratio (9) can be limited under the assumptions above, for not too large values of t. Note that the solution in (9) is, as a matter of fact, based on the solution of an unstable ODE, see Grewal and Andrews [2008]; Särkkä [2006], but the ratio $\mathbf{C}(t)\mathbf{M}_3(t)^{-1}$ can still be bounded. Both of these methods can have numerical problems which will be discussed in Section 5.2.

[1] All eigenvalues are in the left half plane.

3.1 Stability for the Vectorisation Method using Euler Sampling

The stability analysis in this section is standard and a similar analysis has been performed in Hinrichsen and Pritchard [2005]. The difference is that the analysis in Hinrichsen and Pritchard [2005] investigates which discretisation methods that are stable for sufficiently small sample times. The analysis here is about to find an upper bound of the sample time such that a stable continuous-time model remains stable after discretisation.

The recursive solution (15) is stable for all h according to Lemma 3 in Appendix B, if the matrix exponential can be calculated exactly. Stability issues arise when $e^{\mathbf{A_z}h}$ has to be approximated by $e_{p,m}(\mathbf{A_z}h)$. In this section we derive an upper bound on h that gives a stable solution for $e_{1,m}(\mathbf{A_z}h)$, i.e., Euler sampling. The Taylor expansion and in particular Euler sampling is chosen due to its simplicity, the same approach is applicable to the Padé approximation as well. Higher orders of approximations using the Taylor expansion will be treated briefly at the end of this section.

From Section 2.3 we have that the matrix $\mathbf{A_z}$ is diagonal, which means that calculation of the matrix exponential $e^{\mathbf{A_z}h}$ can be separated into $e^{\mathbf{A}h}$ and $e^{\mathbf{A_P}h}$. From Lütkepohl [1996] it is known that the eigenvalues of $\mathbf{A_P}$ are given by $\lambda_i + \lambda_j$, $1 \leq i \leq j \leq n_x$, hence the ODE describing the CDLE is stable if $\mathbf{A}$ is Hurwitz. In order to keep the discrete-time system stable, the eigenvalues of both $e_{1,m}(\mathbf{A}h)$ and $e_{1,m}(\mathbf{A_P}h)$ need to be inside the unit circle. In Theorem 1 an explicit upper bound on the sample time h is given that makes the recursive solution to the continuous-time SDE stable.

Theorem 1. *The recursive solution to the* SDE (1)*, in the form of* (15)*, where the matrix exponential $e^{\mathbf{A_z}h}$ is approximated by $e_{1,m}(\mathbf{A_z}h)$, is stable if*

$$h < \min\left\{-\frac{2m\Re\mathfrak{e}\left\{\lambda_i+\lambda_j\right\}}{\left|\lambda_i+\lambda_j\right|^2},\ 1 \leq i \leq j \leq n_x\right\}, \tag{20}$$

where λ_i, $i = 1, \ldots, n_x$, are the eigenvalues to $\mathbf{A}$.

Corollary 1. *The bound in Theorem 1 becomes*

$$h < -\frac{2m}{\lambda_i}, \quad \lambda_i \in \mathbb{R}, \tag{21}$$

for real eigenvalues.

Proof: Start with the ODE describing the state vector $\mathbf{x}(t)$. The eigenvalues to $e_{1,m}(\mathbf{A}h) = (\mathbf{I} + \mathbf{A}h/m)^m$ are, according to Lemma 2 in Appendix B, given by $(1 + \lambda_i h/m)^m$. The eigenvalues are inside the unit circle if $|(1 + \lambda_i h/m)^m| < 1$, where

$$\left|\left(1+\frac{\lambda_i h}{m}\right)^m\right| = \left(\frac{1}{m}\sqrt{m^2 + 2a_i hm + (a_i^2 + b_i^2)h^2}\right)^m. \tag{22}$$

In (22), the parametrisation $\lambda_i = a_i + ib_i$ has been used. Solving $|(1 + \lambda_i h/m)^m| < 1$

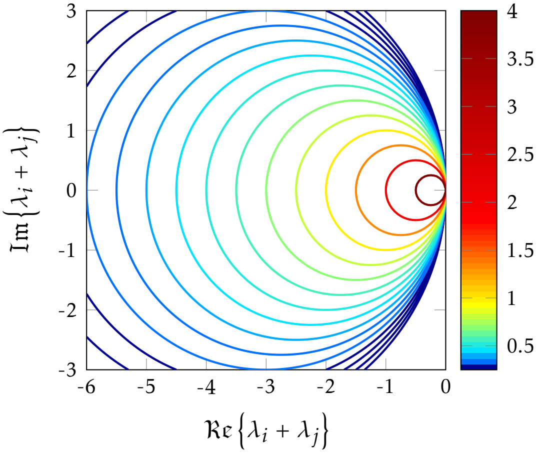


Figure 1: *Level curves of* (20)*, where the colours indicate the values on h.*

for h and using the fact $|\lambda_i|^2 = a_i^2 + b_i^2$ give

$$h < -\frac{2ma_i}{|\lambda_i|^2}. \tag{23}$$

Similar calculations for the ODE describing vech $\mathbf{P}(t)$ give

$$h < -\frac{2m(a_i + a_j)}{\left|\lambda_i + \lambda_j\right|^2}, \quad 1 \le i \le j \le n_x. \tag{24}$$

Using $\lambda_i = \lambda_j$ in (24) gives

$$-\frac{2m(a_i + a_j)}{\left|\lambda_i + \lambda_j\right|^2} = -\frac{4ma_i}{|2\lambda_i|^2} = -\frac{ma_i}{|\lambda_i|^2}, \tag{25}$$

which is half as much as the bound in (23), hence the upper bound for h is given by (24). □

Theorem 1 shows that the sample time can be decreased if the absolute value of the eigenvalues are increased, but also if the real part approaches zero. The level curves of (20) for $h = c =$ constant in the complex plane are given by

$$-\frac{2a_{ij}m}{a_{ij}^2 + b_{ij}^2} = c \Leftrightarrow \left(a_{ij} + \frac{m}{c}\right)^2 + b_{ij}^2 = \frac{m^2}{c^2} \tag{26}$$

where $a_{ij} = \mathfrak{Re}\left\{\lambda_i + \lambda_j\right\}$ and $b_{ij} = \mathfrak{Im}\left\{\lambda_i + \lambda_j\right\}$. Equation (26) is the description of a circle with radius m/c centred in the point $(-m/c, 0)$. The level curves are shown in Figure 1, where it can be seen how the maximal sample time depends on the magnitude and direction of the eigenvalues.

Stability conditions of $e_{p,m}(\mathbf{A_z}h)$ for $p > 1$ can be carried out in the same way as for $p = 1$. For $p = 2$, the calculations can be done analytically and this results again in (20),

$$h < \min\left\{-\frac{2m\Re\mathfrak{e}\left\{\lambda_i + \lambda_j\right\}}{\left|\lambda_i + \lambda_j\right|^2},\ 1 \leq i \leq j \leq n_x\right\}. \tag{27}$$

It means that though the accuracy has increased, recall (17c), the stability condition remains the same.

Increasing the order of approximation even more, results in a higher order polynomial inequality that has to be solved. A numerical solution is therefore preferred. The stability bound for h/m will actually increase when $p > 2$ increases. For example, $e_{4,m}(\mathbf{A}h)$, which corresponds to the Runge-Kutta solution for a linear ODE gives, for real eigenvalues,

$$h < -\frac{2.7852m}{\lambda_i}, \quad \lambda_i \in \mathbb{R}. \tag{28}$$

This is less conservative than the corresponding bound in (21).

4 Reformulation of the Vectorised Solution for the CDLE

The solution to the ODE describing the second order moment given by (12) can be computed efficiently using the following lemma.

Lemma 1. *The solution to the vectorised CDLE*

$$\mathit{vech}\,\dot{\mathbf{P}}(t) = \mathbf{A_P}\mathit{vech}\,\mathbf{P}(t) + \mathit{vech}\,\mathbf{GQG}^\mathsf{T}, \tag{29}$$

is given by

$$\mathit{vech}\,\mathbf{P}(t) = \mathbf{F_P}(t)\mathit{vech}\,\mathbf{P}(0) + \mathbf{G_P}(t)\mathit{vech}\,\mathbf{GQG}^\mathsf{T}. \tag{30}$$

where $\mathbf{F}_P(t)$ and $\mathbf{G}_P(t)$ are given by

$$\exp\left[\begin{pmatrix}\mathbf{A_P} & \mathbf{I}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}t\right] = \begin{pmatrix}\mathbf{F_P}(t) & \mathbf{G_P}(t)\\ \mathbf{0} & \mathbf{0}\end{pmatrix} \tag{31}$$

Proof: The derivation in Appendix A gives that

$$\mathbf{F_P}(t) = e^{\mathbf{A_P}t}, \tag{32}$$

$$\mathbf{G_P}(t) = \mathbf{A_P^{-1}}(e^{\mathbf{A_P}t} - \mathbf{I}). \tag{33}$$

It is now easily verified that the Taylor expansion of (31) and (32) as

$$\mathbf{A_P^{-1}}(e^{\mathbf{A_P}t} - \mathbf{I}) = \mathbf{I}t + \mathbf{A_P}\frac{t^2}{2!} + \mathbf{A_P^2}\frac{t^3}{3!} + \mathbf{A_P^3}\frac{t^4}{4!} + \ldots, \tag{34}$$

are the same. □

The last result is important of two reasons. First, we avoid inversion of the large matrix $\mathbf{A_P}$ by solving a matrix exponential instead. Second, the condition that $\mathbf{A_P}$ has to be non-singular is relaxed.

The new solution based on Lemma 1 is presented in Theorem 2. The solution requires the matrix exponential and the solution of an algebraic Lyapunov equation for which efficient numerical solvers exist.

Remark 1. In contrast to Lemma 1, the solution to the CDLE in (5b) presented in Theorem 2 actually requires $\mathbf{A_P}$ to be non-singular. The eigenvalues to $\mathbf{A_P}$ are given by $\lambda_i + \lambda_j$, $1 \le i \le j \le n_x$ [Lütkepohl, 1996], so we have that $\mathbf{A_P}$ is non-singular when the eigenvalues of $\mathbf{A}$ are not mirrored in the imaginary axis. Eigenvalues in the origin is a special case of this.

Theorem 2. *Let* $\mathbf{Q}$ *be positive definite and assume that the eigenvalues of* $\mathbf{A}$ *are not mirrored in the imaginary axis. Then the solution of the* CDLE (5b) *is given by*

$$\mathbf{P}(t) = e^{\mathbf{A}t}\mathbf{P}(0)e^{\mathbf{A}^\mathsf{T}t} + \mathbf{Q}_d(t), \tag{35a}$$

$$\mathbf{A}\mathbf{Q}_d(t) + \mathbf{Q}_d(t)\mathbf{A}^\mathsf{T} + \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} - e^{\mathbf{A}t}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}e^{\mathbf{A}^\mathsf{T}t} = \mathbf{0}, \tag{35b}$$

where $\mathbf{Q}_d(t)$ *is a unique and positive definite solution to* (3).

Proof: Taylor expansion of the matrix exponential gives

$$e^{\mathbf{A_P}t} = \mathbf{I} + \mathbf{A_P}t + \frac{\mathbf{A}_\mathbf{P}^2t^2}{2!} + \frac{\mathbf{A}_\mathbf{P}^3t^3}{3!} + \dots \tag{36}$$

Using (73) in Appendix C, each term in the Taylor expansion can be rewritten according to

$$\mathbf{A}_\mathbf{P}^k t^k = \mathbf{D}^\dagger(\mathbf{I}\otimes\mathbf{A} + \mathbf{A}\otimes\mathbf{I})^k t^k\mathbf{D}, \tag{37}$$

hence

$$e^{\mathbf{A_P}t} = \mathbf{D}^\dagger e^{(\mathbf{I}\otimes\mathbf{A}+\mathbf{A}\otimes\mathbf{I})t}\mathbf{D} \overset{(71),(72)}{=} \mathbf{D}^\dagger(e^{\mathbf{A}t}\otimes e^{\mathbf{A}t})\mathbf{D}. \tag{38}$$

The first term in (30) can now be written as

$$\begin{aligned} e^{\mathbf{A_P}t}\operatorname{vech}\mathbf{P}(0) &= \mathbf{D}^\dagger(e^{\mathbf{A}t}\otimes e^{\mathbf{A}t})\mathbf{D}\operatorname{vech}\mathbf{P}(0) = \mathbf{D}^\dagger(e^{\mathbf{A}t}\otimes e^{\mathbf{A}t})\operatorname{vec}\mathbf{P}(0) \\ &\overset{(70)}{=} \mathbf{D}^\dagger\operatorname{vec} e^{\mathbf{A}t}\mathbf{P}(0)e^{\mathbf{A}^\mathsf{T}t} = \operatorname{vech} e^{\mathbf{A}t}\mathbf{P}(0)e^{\mathbf{A}^\mathsf{T}t}. \end{aligned} \tag{39}$$

Similar calculations give

$$e^{\mathbf{A_P}t}\operatorname{vech}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} = \mathbf{D}^\dagger\operatorname{vec} e^{\mathbf{A}t}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}e^{\mathbf{A}^\mathsf{T}t} \triangleq \operatorname{vech}\widehat{\mathbf{Q}}(t). \tag{40}$$

The last term in (30) can then be rewritten according to

$$\mathbf{A}_\mathbf{P}^{-1}(e^{\mathbf{A_P}t} - \mathbf{I})\operatorname{vech}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} = \mathbf{A}_\mathbf{P}^{-1}\operatorname{vech}(\widehat{\mathbf{Q}}(t) - \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}) \triangleq \operatorname{vech}\mathbf{Q}_d(t), \tag{41}$$

where it is assumed that $\mathbf{A_P}$ is invertible. Equation (41) can be seen as the solution of the linear system of equations $\mathbf{A_P}\operatorname{vech}\mathbf{Q}_d(t) = \operatorname{vech}(\widehat{\mathbf{Q}}(t) - \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T})$. Using the derivation in (65) in Appendix A backwards gives that $\mathbf{Q}_d(t)$ is the solution to the algebraic Lyapunov equation

$$\mathbf{A}\mathbf{Q}_d(t) + \mathbf{Q}_d(t)\mathbf{A}^\mathsf{T} + \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} - \widehat{\mathbf{Q}}(t) = \mathbf{0}. \tag{42}$$

Combining (39) and (41) gives that (30) can be written as

$$\mathbf{P}(t) = e^{\mathbf{A}t}\mathbf{P}(0)e^{\mathbf{A}^\mathsf{T}t} + \mathbf{Q}_d(t), \tag{43}$$

where $\mathbf{Q}_d(t)$ is the solution to (42).

It is easily verified that $\mathbf{Q}_d(t)$ in (3) satisfies (42) and it is well known that the Lyapunov equation has a unique solution iff the eigenvalues of $\mathbf{A}$ are not mirrored in the imaginary axis [Gajić and Qureshi, 1995]. Moreover, the assumption that $\mathbf{Q}$ is positive definite gives from (3) that $\mathbf{Q}_d(t)$ is positive definite, hence the solution to (42) is unique and guaranteed to be positive definite under the assumptions on $\mathbf{A}$. □

If Lemma 1 is used directly, a matrix exponential of a matrix of dimension $n_x(n_x + 1) \times n_x(n_x + 1)$ is required. Now, only the Lyapunov equation (35b) has to be solved, where the dimensions of the matrices are $n_x \times n_x$. The computational complexity for solving the Lyapunov equation is $35n_x^3$ [Higham, 2008]. The total computational complexity for computing the solution of (5b) using Theorem 2 is $(\log_2(m) + p + 43)\, n_x^3$, where $(\log_2(m) + p)\, n_x^3$ comes from the matrix exponential, and $43n_x^3$ comes from solving the Lyapunov equation ($35n_x^3$) and taking four matrix products giving $2n_x^3$ each time.

The algebraic Lyapunov function (35b) has a unique solution only if the eigenvalues of $\mathbf{A}$ are not mirrored in the imaginary axis [Gajić and Qureshi, 1995], as a result of the assumption that $\mathbf{A_P}$ is non-singular, and this is the main drawback with using Theorem 2 rather than using Lemma 1. In the case of integrators, the method presented in Wahlström et al. [2014] can be used. To be able to calculate $\mathbf{Q}_d(t)$, the method transforms the system such that the Lyapunov equation (35b) is used for the subspace without the integrators, and the integral in (3) is used for the subspace containing the integrators.

Discrete-time Recursion The recursive solution to the differential equations in (5) describing the first and second order moments of the SDE (1) can now be written as

$$\hat{\mathbf{x}}((k+1)h) = e^{\mathbf{A}h}\hat{\mathbf{x}}(kh), \tag{44a}$$

$$\mathbf{P}((k+1)h) = e^{\mathbf{A}h}\mathbf{P}(kh)e^{\mathbf{A}^\mathsf{T}h} + \mathbf{Q}_d(h), \tag{44b}$$

$$\mathbf{A}\mathbf{Q}_d(h) + \mathbf{Q}_d(h)\mathbf{A}^\mathsf{T} + \mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T} - e^{\mathbf{A}h}\mathbf{G}\mathbf{Q}\mathbf{G}^\mathsf{T}e^{\mathbf{A}^\mathsf{T}h} = \mathbf{0}, \tag{44c}$$

Equations (44b) to (44c) are derived using $t = kh$ and $t = (k+1)h$ in (35).

The method presented in Theorem 2 is derived straightforwardly from Lemma 1. A similar solution that also solves an algebraic Lyapunov function is presented in Davison [1975]. The main difference is that Theorem 2 gives a value of the covariance matrix $\mathbf{Q}_d(t)$ for the discrete-time noise explicitly, as opposed to the

Table 2: *Six variants to calculate* $\mathbf{P}(t)$*. The last column shows the computational complexity* p *in* $\mathcal{O}(n_x^p)$ *which is described in detail in Section 5.1.*

Method	Description	p in $\mathcal{O}(n_x^p)$
I	$\mathbf{P}(t)$ from Lemma 1.	6
II	$\mathbf{P}(t)$ from Theorem 2.	3
III	$\mathbf{P}(t)$ from (4b) with $\mathbf{Q}_d$ calculated using equation (8).	3
IV	$\mathbf{P}(t)$ from (4b) with $\mathbf{Q}_d$ calculated using an eigenvalue decomposition for diagonalising the integrand in (3).	3
V	$\mathbf{P}(t)$ from (4b) with $\mathbf{Q}_d$ calculated using numerical integration of the integral in (3) using `quadv` in Matlab.	3
VI	$\mathbf{P}(t)$ from the matrix fraction decomposition in (9).	3

solution in Davison [1975]. Moreover, the algebraic Lyapunov function in Davison [1975] is independent of time, which is not the case here since $e^{\mathbf{A}t}\mathbf{GQG}^{\mathsf{T}}e^{\mathbf{A}^{\mathsf{T}}t}$ changes with time. This is not an issue for the recursive time update due to the fact that $e^{\mathbf{A}h}\mathbf{GQG}^{\mathsf{T}}e^{\mathbf{A}^{\mathsf{T}}h}$ is only dependent on h, hence the algebraic Lyapunov equation (44c) has to be solved only once.

5 Comparison of Solutions for the CDLE

This section will provide rough estimates of the computational complexity of the different approaches to compute the CDLE, by counting the number of flops. Numerical properties are also discussed. Table 2 summarises six variants of the methods presented in Section 2 of how to calculate $\mathbf{P}(t)$.

5.1 Computational Complexity

Rough estimates of the computational complexity can be given by counting the number of operations that is required. From Section 4 it is given that the computational complexity for Method I is $\mathcal{O}(n_x^6)$ and for Method II it is $(\log_2(m) + p + 43)n_x^3$. The total computational complexity of Method III is roughly $(8(\log_2(m) + p) + 6)n_x^3$, where $(\log_2(m) + p)(2n_x)^3$ comes from $e^{\mathbf{H}t}$ and $6n_x^3$ from the remaining three matrix products. Using e.g. an eigenvalue decomposition to calculate the integral, i.e., Method IV, gives a computational complexity of $\mathcal{O}(n_x^3)$. For numerical integration, i.e., Method V, the computational complexity will be $\mathcal{O}(n_x^3)$ due to the matrix exponential and matrix products. The constant in front of n_x^3 will be larger than for Method III and Method IV because of element-wise integration of the $n_x \times n_x$ symmetric matrix integrand, which requires $n_x(n_x + 1)/2$ number of integrations. For the last method Method VI, the same matrix expo-

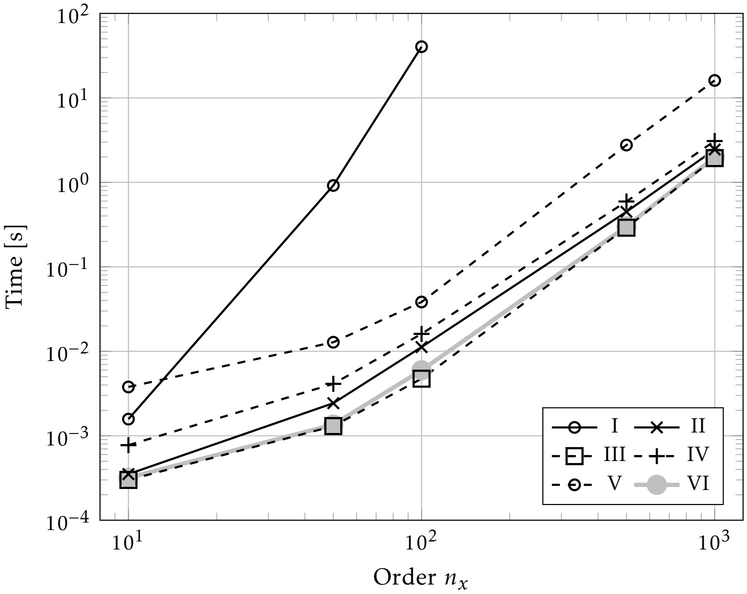


Figure 2: *Mean execution time for calculating* $\mathbf{P}(t)$ *for randomly generated state matrices with order* $n_x \times n_x$ *over 1000 MC simulations.*

nential as in METHOD III is calculated which gives $(\log_2(m) + p)8n_x^3$ operations. In addition, $2n_x^3$ operations for the matrix inverse and $4n_x^3$ operations for the two remaining matrix products are required. In total, the computational complexity is $(8(\log_2(m) + p) + 6)n_x^3$. The product $\mathbf{C}(t)\mathbf{M}_3(t)^{-1}$ can of course be calculated without first calculate the inverse and then perform the multiplication, but it is a rough estimate presented here.

The computational complexity is also analysed by performing Monte Carlo simulations over 1000 randomly chosen stable systems. The order of the systems are $n_x = 10, 50, 100, 500, 1000$. As expected, the solution using METHOD I takes very long time as can be seen in Figure 2. For METHOD I the size has been restricted to $n_x \leq 100$ since $\mathbf{A_P}$ grows to much for larger values. However, the computational time using METHOD I compared to the other methods is clear. The computational time for the other methods are in the same order, which is also expected. As discussed previously, the numerical integration will give a computational complexity that has the same slope but with a higher offset than METHOD II–IV, and METHOD VI, which is seen in Figure 2. It can also be noted that the numerical integration for $n_x = 10$ is slower than the METHOD I.

Note that not only the number of operations of performing e.g. the matrix exponential and the matrix multiplications affect the total time. Also, time for memory management is included. However, the slope of the lines for large n_x is approximately six for METHOD I and three for the other methods, which agree

***Table 3:** Standard deviation of the execution time for calculating* $\mathbf{P}(t)$.

	10	50	100	500	1000
I	$9.63 \cdot 10^{-5}$	$7.85 \cdot 10^{-2}$	1.38	–	–
II	$1.88 \cdot 10^{-5}$	$1.46 \cdot 10^{-4}$	$8.20 \cdot 10^{-4}$	$4.37 \cdot 10^{-2}$	$1.03 \cdot 10^{-1}$
III	$6.09 \cdot 10^{-6}$	$8.84 \cdot 10^{-5}$	$3.71 \cdot 10^{-4}$	$3.89 \cdot 10^{-2}$	$2.41 \cdot 10^{-1}$
IV	$7.67 \cdot 10^{-5}$	$1.72 \cdot 10^{-4}$	$7.42 \cdot 10^{-4}$	$5.14 \cdot 10^{-2}$	$2.15 \cdot 10^{-1}$
V	$1.26 \cdot 10^{-4}$	$4.96 \cdot 10^{-4}$	$1.68 \cdot 10^{-3}$	$2.12 \cdot 10^{-1}$	1.39
VI	$1.95 \cdot 10^{-5}$	$5.35 \cdot 10^{-5}$	$3.89 \cdot 10^{-4}$	$3.69 \cdot 10^{-2}$	$2.06 \cdot 10^{-1}$

with the computational complexity discussed above. The standard deviation for the computation time for the different methods is at least one order of magnitude less than the mean value, see Table 3.

5.2 Numerical Properties

Here, the numerical properties will be analysed. First, the solution $\mathbf{P}(t)$ should hold for any value of t. It means that a large enough value of t should give that $\mathbf{P}(t)$ equals the stationary solution given from the stationary Lyapunov equation

$$\mathbf{A}\mathbf{P}^{\text{stat}} + \mathbf{P}^{\text{stat}}\mathbf{A}^{\mathsf{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}} = \mathbf{0} \tag{45}$$

Second, the recursive updates should approach $\mathbf{P}^{\text{stat}}$ and then stay there when $k \to \infty$.

Randomly generated stable system matrices, over 100 MC simulations[2], of order $n_x = 2$ will be used with $\mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}} = \mathbf{I}$ to show how the methods perform. For the first case the value $t = 100$ has been used and for the second case the sample time $h = 0.01$ s has been used and a total of 10,000 samples.

The stationary matrix $\mathbf{P}^{\text{stat}}$ is not obtained for METHOD III–IV, and METHOD VI for all MC simulations. However, methods METHOD I–II and METHOD V gives $\mathbf{P}^{\text{stat}}$ as the solution. The reason that METHOD III and METHOD VI cannot give the correct stationary matrix is that they have to calculate the ill-conditioned matrix $e^{\mathbf{H}t}$.

For the second case, where the recursive update is used, the norm of difference $\left\|\mathbf{P}(t) - \mathbf{P}^{\text{stat}}\right\|$ for the methods are shown in Figure 3 for the first 1000 samples. It can be seen that METHOD I–V converge to the stationary solution. METHOD VI is not able to converge to the stationary solution when the time goes by, instead numerical problems occur, giving Inf or NaN (Not-a-Number) as solution.

6 Linear Spring-Damper Example

The different solutions and approximations described above will be investigated for a linear model of a mass M hanging in a spring and damper, see Figure 4. The

[2] Here, only 100 MC simulations are used to be able to visualise the result, otherwise too many samples with Inf or NaN as solution, which cannot be displayed in the figure.

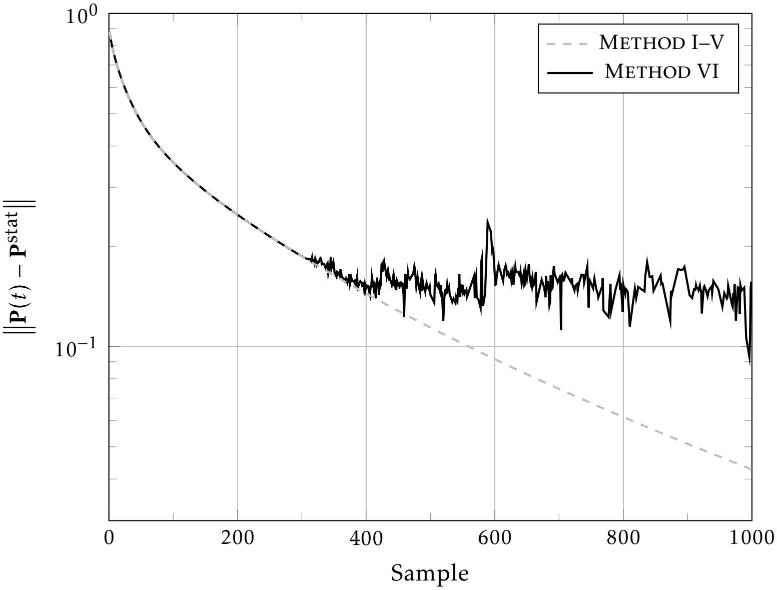


Figure 3: *The mean norm over 100 MC simulations of the error $\mathbf{P}(t) - \mathbf{P}^{stat}$ for the recursive updates. METHOD I–V gives the same behaviour whereas METHOD VI diverges*

equation of motion is

$$M\ddot{q} + d\dot{q} + kq - Mg = 0 \tag{46}$$

where q is the distance from where the spring/damper is unstretched and $g = 9.81$ is the gravity constant. A linear state space model, using $M = 1$, with $\mathbf{x} = \begin{pmatrix} q & \dot{q} \end{pmatrix}^{\mathsf{T}}$ is given by

$$\dot{\mathbf{x}}(t) = \underbrace{\begin{pmatrix} 0 & 1 \\ -k & -d \end{pmatrix}}_{\mathbf{A}} \mathbf{x}(t) + \underbrace{\begin{pmatrix} 0 \\ g \end{pmatrix}}_{\mathbf{B}} . \tag{47}$$

6.1 Stability Bound on the Sample Time

The bound on the sample time that makes the solution to (47) stable, when Euler sampling $e_{1,m}(\mathbf{A}h)$ is used, can be calculated using Theorem 1. The eigenvalues for $\mathbf{A}$ are

$$\lambda_{1,2} = -\frac{d}{2} \pm \frac{1}{2}\sqrt{d^2 - 4k}. \tag{48}$$

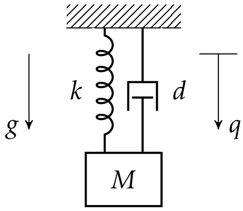


Figure 4: *A mass hanging in a spring and damper.*

If $d^2 - 4k \geq 0$ the system is well damped and the eigenvalues are real, hence

$$h < \min\left\{\frac{2m}{d+\sqrt{d^2-4k}}, \frac{2m}{d-\sqrt{d^2-4k}}, \frac{2m}{d}\right\} = \frac{2m}{d+\sqrt{d^2-4k}}, \tag{49}$$

If instead $d^2 - 4k < 0$, the system is oscillating and the eigenvalues are complex, giving

$$h < \min\left\{\frac{dm}{2k}, \frac{2m}{d}, \frac{dm}{2k}\right\} = \frac{dm}{2k}, \tag{50}$$

where we have used the fact that $d^2 - 4k < 0$ to get the minimum value.

The values on the parameters have been chosen as $d = 2$ and $k = 10$ giving an oscillating system. The stability bound is therefore $h < 0.1m$ s.

6.2 Kalman Filtering

We will now focus on Kalman filtering of the spring-damper example.

The continuous-time model (47) is augmented with process noise giving the model

$$d\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)dt + \mathbf{B} + \mathbf{G}d\beta(t), \tag{51}$$

where $\mathbf{A}$ and $\mathbf{B}$ are given by (47), $\mathbf{G} = \begin{pmatrix}0 & 1\end{pmatrix}^{\mathsf{T}}$ and $d\beta(t)$ is a scalar Wiener process with $\mathrm{E}\left[d\beta(t)d\beta(t)^{\mathsf{T}}\right] = \mathbf{Q}dt$. Here it is used that $\mathbf{Q} = 5 \cdot 10^{-3}$. It is assumed that the velocity $\dot{q}$ is measured with a sample rate T_s. The measurement equation can be written as

$$\mathbf{y}_k = \begin{pmatrix}0 & 1\end{pmatrix}\mathbf{x}_k + \mathbf{e}_k = \mathbf{C}\mathbf{x}_k + \mathbf{e}_k, \tag{52}$$

where $\mathbf{e}_k \in \mathbb{R}$ is discrete-time normal distributed white noise with zero mean and a standard deviation of $\sigma = 0.05$. Here, $\mathbf{y}_k \triangleq \mathbf{y}(kT_s)$ has been used for notational convenience. It is easy to show that the system is observable with this measurement. The stability condition for the first order approximation $e_{1,m}(\mathbf{A}h)$ was calculated to be $h < 0.1m$ seconds in Section 6.1. We chose therefore $T_s = h = 0.09$ s.

The simulation represents free motion of the mass when starting at $\mathbf{x}_0 = \begin{pmatrix}0 & 0\end{pmatrix}^{\mathsf{T}}$. The ground truth data is obtained by simulating the continuous-time SDE over

$t_{\max} = 20\,\mathrm{s}$ with a sample time h_S that is 100 times shorter than T_s. In that case, the Wiener process $d\boldsymbol{\beta}(t)$ can at each sample instance be approximated by a normal distributed zero mean white noise process with covariance matrix $\mathbf{Q}h_S$.

Four Kalman filters are compared where $e^{\mathbf{A}h}$ is approximated either by $e_{1,m}(\mathbf{A}h)$ or by the MATLAB-function `expm`. The function `expm` uses scaling and squaring techniques with a Padé approximation to compute the matrix exponential, see Moler and Van Loan [2003]; Higham [2005]. Moreover, the update of the covariance matrix $\mathbf{P}(t)$ is according to the discrete filter (19b) or according to one of the solutions presented in Sections 2.1 to 2.3. Here, the solution to the CDLE given by Theorem 2 has been used, but the other methods would give the same results. Remember though that the matrix fraction method can have numerical problems. In summary, the Kalman filters are:

1. $\mathbf{F}_h = e_{1,m}(\mathbf{A}h)$ and $\mathbf{P}(k+1) = \mathbf{F}_h\mathbf{P}(k)\mathbf{F}_h^{\mathsf{T}} + \mathbf{G}_h\mathbf{Q}_h\mathbf{G}_h^{\mathsf{T}}$,
2. $\mathbf{F}_h$ is given by the MATLAB-function `expm` and $\mathbf{P}(k+1) = \mathbf{F}_h\mathbf{P}(k)\mathbf{F}_h^{\mathsf{T}} + \mathbf{G}_h\mathbf{Q}_h\mathbf{G}_h^{\mathsf{T}}$,
3. $\mathbf{F}_h = e_{1,m}(\mathbf{A}h)$ and $\mathbf{P}(k+1) = \mathbf{F}_h\mathbf{P}(k)\mathbf{F}_h^{\mathsf{T}} + \mathbf{Q}_d(h)$,
4. $\mathbf{F}_h$ is given by the MATLAB-function `expm` and $\mathbf{P}(k+1) = \mathbf{F}_h\mathbf{P}(k)\mathbf{F}_h^{\mathsf{T}} + \mathbf{Q}_d(h)$,

where $\mathbf{Q}_d(h)$ is the solution to the Lyapunov equation in (44c).

The Kalman filters are initialised with the true $\mathbf{x}_0$, used for ground truth data, plus a normal distributed random term with zero mean and standard deviation 0.1. The state covariance is initialised by $\mathbf{P}(0) = \mathbf{I}$. The covariance matrix for the measurement noise is the true one, i.e., $\mathbf{R} = \sigma^2$. The covariance matrix for the process noise is different for the filters. For filter 1 and 2 the covariance matrix $\mathbf{Q}h/m$ is used whereas for filter 3 and 4 the true covariance matrix $\mathbf{Q}$ is used.

The filters are compared to each other using $N_{MC} = 1000$ Monte Carlo simulations for different values of m. The oversampling constant m takes the following values:

$$\{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ 20,\ 30,\ 40,\ 50\} \tag{53}$$

Figure 5 shows the *root mean square error* (RMSE) defined according to

$$\rho_i = \sqrt{\frac{1}{N}\sum_{t=t_0}^{t_{\max}} \left(\rho_i^{MC}(t)\right)^2} \tag{54a}$$

where $t_0 = t_{\max}/2$ in order to remove the transients, N is the number of samples in $[t_0, t_{\max}]$, and

$$\rho_i^{MC}(t) = \sqrt{\frac{1}{N_{MC}}\sum_{j=1}^{N_{MC}} \left(\mathbf{x}_i^{(j)}(t) - \hat{\mathbf{x}}_i^{(j)}(t)\right)^2}, \tag{54b}$$

where $\mathbf{x}_i^{(j)}(t)$ is the true ith state and $\hat{\mathbf{x}}_i^{(j)}(t)$ is the estimated ith state for Monte Carlo simulation number j. The two filters 1 and 3 give almost identical results for the RMSE, therefore only filter 1 is shown in Figure 5, see the solid line. The dashed lines are the RMSE for filter 4 (filter 2 gives the same result). We can see that a factor of $m = 20$ or higher is required to get the same result for Euler sampling as for the continuous-time solution[3]. The execution time is similar for all four filters and increases with the same amount when m increases, hence a large enough oversampling can be difficult to achieve for systems with hard real-time requirements. In that case, the continuous-time solution is to prefer.

Remark 2. The maximum sample time, derived according to Theorem 1, is restricted by the CDLE as is described in the proof. It means that we can use a larger sample time for the ODE describing the states, in this particular case a twice as large sample time. Based on this, we already have oversampling by a factor of at least two, for the ODE describing the states, when the sample time is chosen according to Theorem 1.

In Figure 6 we can see how the norm of the stationary covariance matrix[4] for the estimation error changes when oversampling is used. The four filters converge to the same value when m increases. For the discrete-time update in (19b), i.e., filter 1 and 2, the stationary value is too large for small values of m. For the continuous-time update in Theorem 2, it can be seen that a first order Taylor approximation of the exponential function, i.e., filter 3, gives a too small covariance matrix which increases when m increases.

A too small or too large covariance matrix for the estimation error can be crucial for different applications, such as target tracking, where the covariance matrix is used for data association.

7 Extensions to Non-linear Systems

We will in this section adapt the results for linear systems to non-linear systems. Inevitably, some approximations have to be done, and the most fundamental one is to assume that the state is constant during the small time steps h/m. This approximation becomes better the larger oversampling factor m is chosen.

7.1 EKF Time Update

Let the dynamics be given by the non-linear SDE

$$d\mathbf{x}(t) = f(\mathbf{x}(t))dt + G(\mathbf{x}(t))d\boldsymbol{\beta}(t), \tag{55}$$

for $t \geq 0$, where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$, $f(\mathbf{x}(t)) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$, $G(\mathbf{x}(t)) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x \times n_\beta}$, and $d\boldsymbol{\beta}(t) \in \mathbb{R}^{n_\beta}$ is a vector of Wiener processes with $\mathrm{E}\left[d\boldsymbol{\beta}(t)d\boldsymbol{\beta}(t)^{\mathsf{T}}\right] = \mathbf{Q}dt$. For simplicity, it is assumed that $G(\mathbf{x}(t)) \triangleq \mathbf{G}$. The propagation of the first and second order moments for the *extended Kalman filter* (EKF) can, as in the linear case, be

[3] It is wise to choose m to be a power of 2, as explained in Section 2.5

[4] The covariance matrix at time $t_{\max}$ is used as the stationary covariance matrix, i.e., $P(t_{\max})$.

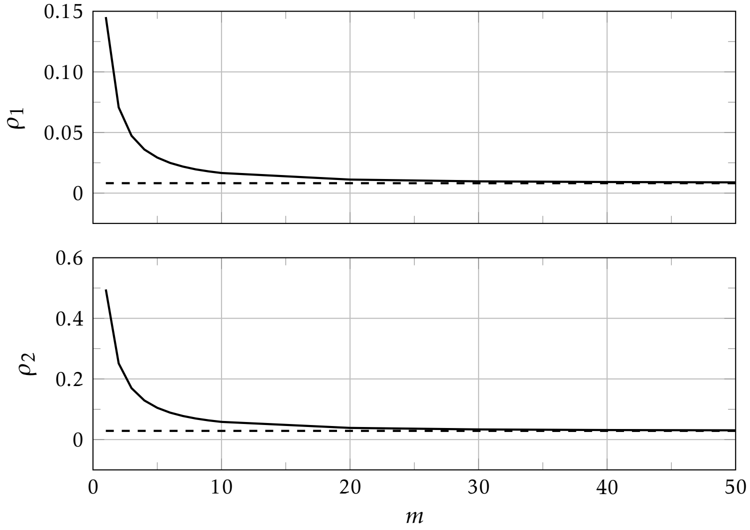


Figure 5: *RMSE according to* (54) *as a function of the degree of oversampling, where the solid line is filter 1 (filter 3 gives the same) and the dashed line is filter 4 (filter 2 gives the same).*

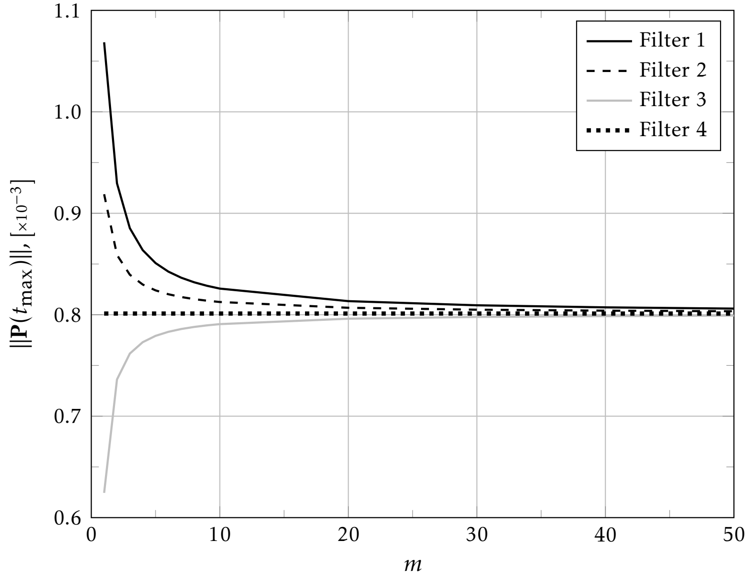


Figure 6: *The norm of the stationary covariance matrix for the estimation error for the four filters, as a function of the degree of oversampling.*

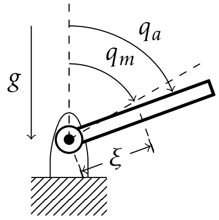


Figure 7: *A single flexible joint.*

written as [Jazwinski, 1970]

$$\dot{\hat{\mathbf{x}}}(t) = f(\hat{\mathbf{x}}(t)), \tag{56a}$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}(\hat{\mathbf{x}}(t))\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}(\hat{\mathbf{x}}(t))^{\mathsf{T}} + \mathbf{GQG}^{\mathsf{T}}, \tag{56b}$$

where $\mathbf{F}(\hat{\mathbf{x}}(t))$ is the Jacobian of $f(\mathbf{x}(t))$ evaluated at $\hat{\mathbf{x}}(t)$. The main differences to (5) are that a linear approximation of $f(\mathbf{x})$ is used in the CDLE as well as the CDLE is dependent on the state vector $\mathbf{x}$. Without any assumptions, the two equations in (56) have to be solved simultaneously. The easiest way is to vectorise (56b) similar to what is described in Appendix A and then solve the non-linear ODE

$$\frac{d}{dt}\begin{pmatrix}\hat{\mathbf{x}}(t)\\ \operatorname{vech}\mathbf{P}(t)\end{pmatrix} = \begin{pmatrix} f(\hat{\mathbf{x}}(t)),\\ \mathbf{A_P}(\hat{\mathbf{x}}(t))\operatorname{vech}\mathbf{P} + \operatorname{vech}\mathbf{GQG}^{\mathsf{T}}\end{pmatrix}, \tag{57}$$

where $\mathbf{A_P}(\hat{\mathbf{x}}(t)) = \mathbf{D}^{\dagger}(\mathbf{I}\otimes\mathbf{F}(\hat{\mathbf{x}}(t)) + \mathbf{F}(\hat{\mathbf{x}}(t))\otimes\mathbf{I})\mathbf{D}$. The non-linear ODE can be solved using a numerical solver such as Runge-Kutta methods [Hairer et al., 1987]. If it is assumed that $\hat{\mathbf{x}}(t)$ is constant over an interval of length h/m, then the two ODEs describing $\hat{\mathbf{x}}(t)$ and $\operatorname{vech}\mathbf{P}(t)$ can be solved separately. The ODE for $\hat{\mathbf{x}}(t)$ is solved using a numerical solver and the ODE for $\operatorname{vech}\mathbf{P}(t)$ becomes a linear ODE which can be solved using Theorem 2, where $\mathbf{A} \triangleq \mathbf{F}(\hat{\mathbf{x}}(t))$.

Remark 3. When m increases, the length of the interval, where $\hat{\mathbf{x}}(t)$ has to be constant, decreases. In that case, the assumption of constant $\hat{\mathbf{x}}(t)$ is more valid, hence the two ODEs can be solved separately without introducing too much errors.

Similar extensions for the method using matrix fraction are straightforward to derive. The advantage with the vectorised solution is that it is easy to solve the combined ODE for $\mathbf{x}(t)$ and $\operatorname{vech}\mathbf{P}(t)$ using a Runge-Kutta solver. This can be compared to the method using matrix fraction, which becomes a coupled differential equation with both vector and matrix variables.

7.2 Simulations of a Flexible Joint

A non-linear model for a single flexible joint is investigated in this section, see Figure 7. The equations of motion are given by

$$J_a\ddot{q}_a + G(q_a) + D(\dot{q}_a, \dot{q}_m) + T(q_a, q_m) = 0, \tag{58a}$$

$$J_m\ddot{q}_m + F(\dot{q}_m) - D(\dot{q}_a, \dot{q}_m) - T(q_a, q_m) = u, \tag{58b}$$

Table 4: *Model parameters for the non-linear model.*

J_a	J_m	M	ξ	d	k_1	k_2	f_d	g
1	1	1	1	1	10	100	1	9.81

where the gravity, damping, spring, and friction torques are modelled as

$$G(q_a) = -gM\xi\sin(q_a), \tag{59a}$$

$$D(\dot{q}_a, \dot{q}_m) = d(\dot{q}_a - \dot{q}_m), \tag{59b}$$

$$T(q_a, q_m) = k_2(q_a - q_m) + k_1(q_a - q_m)^3, \tag{59c}$$

$$F(\dot{q}_m) = f_d\dot{q}_m, \tag{59d}$$

Numerical values of the parameters, used for simulation, are given in Table 4. The parameters are chosen to get a good system without unnecessary large oscillations. With the state vector $\mathbf{x} = \begin{pmatrix} q_a & q_m & \dot{q}_a & \dot{q}_m \end{pmatrix}^\mathsf{T}$ a non-linear system of continuous-time ODEs can be written as

$$\dot{\mathbf{x}} = \underbrace{\begin{pmatrix} x_3 \\ x_4 \\ \frac{1}{J_a}\Big(gM\xi\sin(x_1) - d\Delta_{34} - k_2\Delta_{12} - k_1\Delta_{12}^3\Big) \\ \frac{1}{J_m}\Big(d\Delta_{34} + k_2\Delta_{12} + k_1\Delta_{12}^3 - f_d x_4 + u\Big) \end{pmatrix}}_{f(x,u)}, \tag{60}$$

where $\Delta_{ij} = x_i - x_j$. The state space model (60) is also augmented with a noise model according to (55) with

$$\mathbf{G} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ J_a^{-1} & 0 \\ 0 & J_m^{-1} \end{pmatrix}. \tag{61}$$

For the simulation, the arm is released from rest in the position $q_a = q_m = \pi/2$ and moves freely, i.e., $u(t) = 0$, to the stationary point $\mathbf{x} = \begin{pmatrix} \pi & \pi & 0 & 0 \end{pmatrix}^\mathsf{T}$. The ground truth data is obtained using a standard fourth order Runge-Kutta method with a sample time $h_S = 1 \cdot 10^{-6}$ s, which is much smaller than the sample time T_s for the measurements. In the same way as for the linear example in Section 6, the Wiener process $d\beta(t)$ can be approximated at each discrete-time instant by a zero mean white noise process with a covariance matrix $\mathbf{Q}h_S$, where $\mathbf{Q} = 1 \cdot 10^{-3} I_2$. It is assumed that the motor position q_m and velocity $\dot{q}_m$ are measured, with additive zero mean Gaussian measurement noise $\mathbf{e}(kT_s) \in \mathbb{R}^2$ with a standard deviation $\sigma = 0.05\mathbf{I}_2$. The sample time for the measurements is chosen to be $T_s = h = 0.1$ s.

Two *extended Kalman filters* (EKF) are compared. The first filter uses the discrete-time update (19) where Euler sampling

$$\mathbf{x}((k+1)h) = \mathbf{x}(kh) + hf(\mathbf{x}(kh), \mathbf{u}(kh)) \triangleq F(\mathbf{x}(kh)) \tag{62}$$

has been used for discretisation. The second filter solves the continuous-time ODE (57) using a standard fourth order Runge-Kutta method. The filters are initialised with the true $\mathbf{x}(0)$ used for simulating ground truth data plus a random term with zero mean and standard deviation 0.1. The covariance matrix for the estimation error is initialised by $\mathbf{P}(0) = 1 \cdot 10^{-4}\mathbf{I}_4$. The results are evaluated over 1000 Monte Carlo simulations using the different values of m listed in (53).

Figure 8 shows the RMSE, defined in (54), for the four states. The discrete-time filter using Euler sampling requires an oversampling of approximately $m = 10$ in order to get the same performance as the continuous-time filter, which is not affected by m that much. In Figure 9, the norm of the stationary covariance matrix of the estimation error, i.e., $\|\mathbf{P}(t_{\text{max}})\|$, is shown. Increasing m, the value $\|\mathbf{P}(t_{\text{max}})\|$ decreases and approaches the corresponding value for the continuous-time filter. The result is in accordance with the linear model described in Section 6.2. The standard deviation for $\|\mathbf{P}(t_{\text{max}})\|$ is several orders of magnitude less than the mean value and decreases as m increases with a similar rate as the mean value in Figure 9.

The execution time for the two filters differs a lot. They both increase linearly with m and the continuous-time filter is approximately 4-5 times slower than the discrete-time filter. This is because of that the Runge-Kutta solver evaluates the function $f(\mathbf{x}(t))$ four times for each time instant whereas the discrete-time filter evaluates the function $F(\mathbf{x}(kh))$ only once. However, the time it takes for the discrete-time filter using $m = 10$ is approximately 1.6 times slower than using $m = 1$ for the continuous-time filter.

8 Conclusions

This paper investigates the continuous-discrete filtering problem for Kalman filters and extended Kalman filters. The critical time update consists of solving one ODE and one *continuous-time differential Lyapunov equation* (CDLE). The problem can be rewritten as one ODE by vectorisation of the CDLE. The main contributions of the paper are:

1. A survey of different ways to solve the linear SDE is presented. The different methods, presented in Table 2, are compared to each other with respect to stability, computational complexity and numerical properties.

2. Stability condition for Euler sampling of the linear ODE which describes the first and second order moments of the SDE. An explicit upper bound on the sample time is derived such that a stable continuous-time system remains stable after discretisation. The stability condition for higher order of approximations, such as the Runga-Kutta method, is also briefly investigated.

3. A numerical stable and time efficient solution to the CDLE that does not

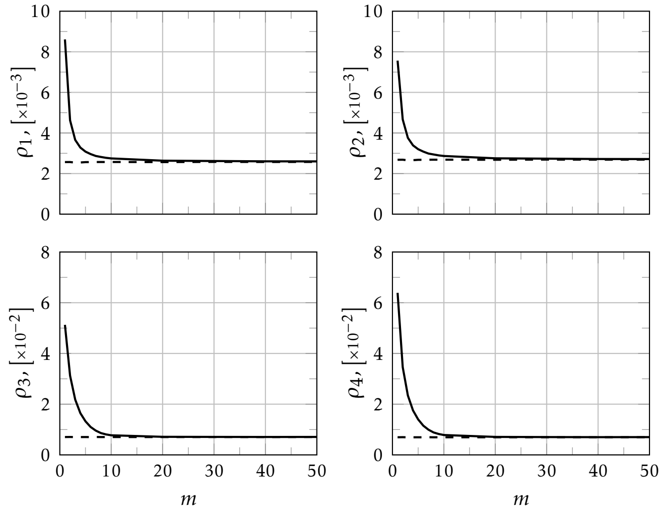


Figure 8: *RMSE according to* (54)*, where the solid line is the discrete-time filter using Euler sampling and the dashed line is the continuous-time filter using a Runge-Kutta solver.*

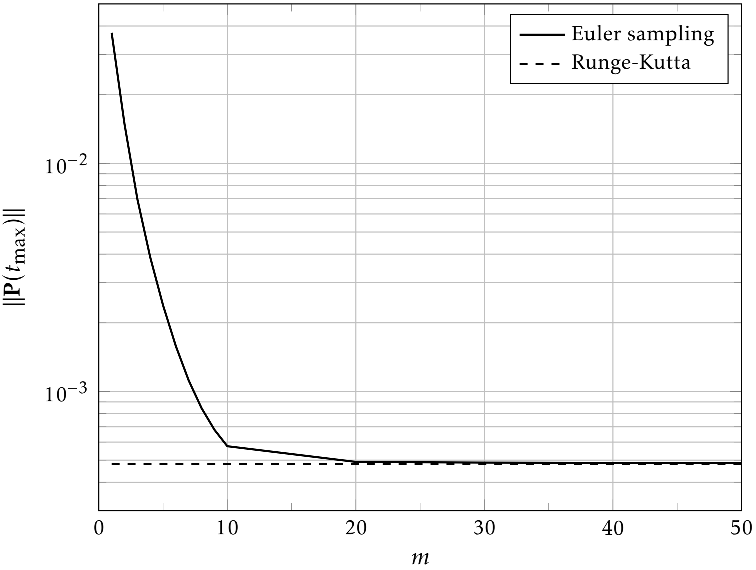


Figure 9: *The norm of the stationary covariance matrix for the estimation error for the EKF using Euler sampling (solid) and a fourth order Runge-Kutta method (dashed).*

require any vectorisation. The computational complexity for the straightforward solution, using vectorisation, of the CDLE is $\mathcal{O}(n_x^6)$, whereas the proposed solution, and the methods proposed in the literature, have a complexity of only $\mathcal{O}(n_x^3)$.

The continuous-discrete filtering problem, using the proposed methods, is evaluated on a linear model describing a mass hanging in a spring-damper pair. It is shown that the standard use of the discrete-time Kalman filter requires a much higher sample rate in order to achieve the same performance as the proposed solution.

The continuous-discrete filtering problem is also extended to non-linear systems and evaluated on a non-linear model describing a single flexible joint of an industrial manipulator. The proposed solution requires the solution from a Runge-Kutta method and without any assumptions, vectorisation has to be used for the CDLE. Simulations of the non-linear joint model show also that a much higher sample time is required for the standard discrete-time Kalman filter to be comparable to the proposed solution.

Appendix

A Vectorisation of the CDLE

The matrix valued CDLE

$$\dot{\mathbf{P}}(t) = \mathbf{A}\mathbf{P}(t) + \mathbf{P}(t)\mathbf{A}^{\mathsf{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}}, \tag{63}$$

can be converted to a vector valued ODE using vectorisation of the matrix $\mathbf{P}(t)$. $\mathbf{P}(t) \in \mathbb{R}^{n_x \times n_x}$ is symmetric so the half-vectorisation is used. The relationship between vectorisation, denoted by vec, and half-vectorisation, denoted by vech, is

$$\operatorname{vec}\mathbf{P}(t) = \mathbf{D}\operatorname{vech}\mathbf{P}(t), \tag{64}$$

where $\mathbf{D}$ is a $n_x^2 \times n_x(n_x+1)/2$ duplication matrix. Let $n_P = n_x(n_x+1)/2$ and $\widetilde{\mathbf{Q}} = \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}}$. Vectorisation of (63) gives

$$\begin{aligned}\operatorname{vech}\dot{\mathbf{P}}(t) &= \operatorname{vech}(\mathbf{A}\mathbf{P}(t) + \mathbf{P}(t)\mathbf{A}^{\mathsf{T}} + \widetilde{\mathbf{Q}}) \\ &= \operatorname{vech}\mathbf{A}\mathbf{P}(t) + \operatorname{vech}\mathbf{P}(t)\mathbf{A}^{\mathsf{T}} + \operatorname{vech}\widetilde{\mathbf{Q}} \\ &= \mathbf{D}^{\dagger}(\operatorname{vec}\mathbf{A}\mathbf{P}(t) + \operatorname{vec}\mathbf{P}(t)\mathbf{A}^{\mathsf{T}}) + \operatorname{vech}\widetilde{\mathbf{Q}} \\ &\overset{(69)}{=} \mathbf{D}^{\dagger}[(\mathbf{I}\otimes\mathbf{A}) + (\mathbf{A}\otimes\mathbf{I})]\mathbf{D}\operatorname{vech}\mathbf{P}(t) + \operatorname{vech}\widetilde{\mathbf{Q}} \\ &= \mathbf{A}_{\mathbf{P}}\operatorname{vech}\mathbf{P}(t) + \operatorname{vech}\widetilde{\mathbf{Q}}\end{aligned} \tag{65}$$

where $\otimes$ is the Kronecker product and $\mathbf{D}^{\dagger} = (\mathbf{D}^{\mathsf{T}}\mathbf{D})^{-1}\mathbf{D}^{\mathsf{T}}$ is the pseudo inverse of $\mathbf{D}$. Note that $\mathbf{D}^{\dagger}\mathbf{D} = \mathbf{I}$ and $\mathbf{D}\mathbf{D}^{\dagger} \neq \mathbf{I}$. The solution of the ODE (65) is given by [Rugh, 1996]

$$\operatorname{vech}\mathbf{P}(t) = e^{\mathbf{A_P}t}\operatorname{vech}\mathbf{P}(0) + \int_0^t e^{\mathbf{A_P}(t-s)}\,\mathrm{d}s\,\operatorname{vech}\widetilde{\mathbf{Q}}. \tag{66}$$

Assume that $\mathbf{A_P}$ is invertible, then the integral can be computed as

$$\begin{aligned}\int_0^t e^{\mathbf{A_P}(t-s)}\,\mathrm{d}s &= e^{\mathbf{A_P}t}\int_0^t e^{-\mathbf{A_P}s}\,\mathrm{d}s = \Big/ \frac{d}{ds}\Big(-\mathbf{A}_\mathbf{P}^{-1}e^{-\mathbf{A_P}s}\Big) = e^{-\mathbf{A_P}s}\Big/ \\ &= e^{\mathbf{A_P}t}\mathbf{A}_\mathbf{P}^{-1}\left(\mathbf{I} - e^{-\mathbf{A_P}t}\right) = \mathbf{A}_\mathbf{P}^{-1}\left(e^{\mathbf{A_P}t} - I\right).\end{aligned} \tag{67}$$

B Eigenvalues of the Approximated Exponential Function

The eigenvalues of $e_{p,m}(\mathbf{A}h)$ as a function of the eigenvalues of $\mathbf{A}$ are given in Lemma 2 and Lemma 3 presents the result when $p \to \infty$ if $\mathbf{A}$ is Hurwitz.

Lemma 2. *Let λ_i and v_i be the eigenvalues and eigenvectors, respectively, of $\mathbf{A} \in \mathbb{R}^{n\times n}$. Then the p'th order Taylor expansion $e_{p,m}(\mathbf{A}h)$ of $e^{\mathbf{A}h}$ is given by*

$$e_{p,m}(\mathbf{A}h) = \left(\mathbf{I} + \frac{\mathbf{A}h}{m} + \ldots + \frac{1}{p!}\left(\frac{\mathbf{A}h}{m}\right)^p\right)^m$$

which has the eigenvectors v_i and the eigenvalues

$$\left(1 + \frac{h\lambda_i}{m} + \frac{h^2\lambda_i^2}{2!m^2} + \frac{h^3\lambda_i^3}{3!m^3} + \ldots + \frac{h^p\lambda_i^p}{p!m^p}\right)^m \tag{68}$$

for $i = 1, \ldots, n$.

Proof: The result follows from an eigenvalue decomposition of the matrix $\mathbf{A}$. □

Lemma 3. *In the limit $p \to \infty$, the eigenvalues of $e_{p,m}(\mathbf{A}h)$ converge to $e^{h\lambda_i}$, $i = 1, \ldots, n$. If $\mathbf{A}$ is Hurwitz ($\mathfrak{Re}\{\lambda_i\} < 0$), then the eigenvalues are inside the unit circle.*

Proof: When $p \to \infty$ the sum in (68) converges to the exponential function $e^{h\lambda_i}$, $i = 1, \ldots, n$. The exponential function can be written as

$$e^{\lambda_i h} = e^{\mathfrak{Re}\{\lambda_i\}h}(\cos \mathfrak{Im}\{\lambda_i\} + i \sin \mathfrak{Im}\{\lambda_i\})$$

which for $\mathfrak{Re}\{\lambda_i\} < 0$ has an absolute value less than 1, hence $e^{\lambda_i h}$ is inside the unit circle. □

C Rules for Vectorisation and the Kronecker Product

The rules for vectorisation and the Kronecker product are from Higham [2008] and Lütkepohl [1996].

$$\operatorname{vec} \mathbf{AB} = (\mathbf{I} \otimes \mathbf{A}) \operatorname{vec} \mathbf{B} = (\mathbf{B}^\mathsf{T} \otimes \mathbf{I}) \operatorname{vec} \mathbf{A} \tag{69}$$

$$(\mathbf{C}^\mathsf{T} \otimes \mathbf{A}) \operatorname{vec} \mathbf{B} = \operatorname{vec} \mathbf{ABC} \tag{70}$$

$$\mathbf{I} \otimes \mathbf{A} + \mathbf{B} \otimes \mathbf{I} = \mathbf{A} \oplus \mathbf{B} \tag{71}$$

$$e^{\mathbf{A} \oplus \mathbf{B}} = e^{\mathbf{A}} \otimes e^{\mathbf{B}} \tag{72}$$

$$\mathbf{DD}^\dagger (\mathbf{I} \otimes \mathbf{A} + \mathbf{A} \otimes \mathbf{I}) \mathbf{D} = (\mathbf{I} \otimes \mathbf{A} + \mathbf{A} \otimes \mathbf{I}) \mathbf{D} \tag{73}$$

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC. The authors would like to thank Dr. Daniel Petersson, Linköping University, Sweden, for valuable comments regarding matrix algebra.

Bibliography

I. Arasaratnam and S. Haykin. Cubature Kalman filtering: A powerful tool for aerospace applications. In *Proceedings of the International Radar Conference*, Bordeaux, France, October 2009.

Patrik Axelsson and Fredrik Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *Submitted to IEEE Transactions on Automatic Control*, 2012.

Johan Bagterp Jörgensen, Per Grove Thomsen, Henrik Madsen, and Morten Rode Kristensen. A computational efficient and robust implementation of the continuous-discrete extended Kalman filter. In *Proceedings of the American Control Conference*, pages 3706–3712, New York City, USA, July 2007.

E. J. Davison. The numerical solution of $\dot{X} = A_1 X + X A_2 + D$, $X(0) = C$. *IEEE Transactions on Automatic Control*, 20(4):566–567, August 1975.

Paul Frogerais, Jean-Jacques Bellanger, and Lofti Senhadji. Various ways to compute the continuous-discrete extended Kalman filter. *IEEE Transactions on Automatic Control*, 57(4):1000–1004, April 2012.

Zoran Gajić and Muhammad Tahir Javed Qureshi. *Lyapunov Matrix Equation in System Stability and Control*, volume 195 of *Mathematics in Science and Engineering*. Academic Press, San Diego, CA, USA, 1995.

Mohinder S. Grewal and Angus P. Andrews. *Kalman Filtering. Theory and Practice Using MATLAB*. John Wiley & Sons, Hoboken, NJ, USA, third edition, 2008.

Ernst Hairer, Syvert Paul Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I – Nonstiff Problems*. Springer Series in Computational Mathematics. Springer-Verlag, Berlin, Heidelberg, Germany, 1987.

Nicholas J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Journal on Matrix Analysis and Applications*, 26(4):1179–1193, June 2005.

Nicholas J. Higham. *Functions of Matrices – Theory and Computation*. SIAM, Philadelphia, PA, USA, 2008.

Diederich Hinrichsen and Anthony J. Pritchard. *Mathematical Systems Theory I – Modelling, State Space Analysis, Stability and Robustness*. Texts in Applied Mathematics. Springer-Verlag, Berlin, Heidelberg, Germany, 2005.

Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64. Academic Press, New York, NY, USA, 1970.

Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, 2000.

J.J. LaViola. A comparison of unscented and extended Kalman filtering for estimating quaternion motion. In *Proceedings of the American Control Conference*, pages 2435–2440, Denver, CO, USA, June 2003.

Helmut Lütkepohl. *Handbook of Matrices*. John Wiley & Sons, Chichester, West Sussex, England, 1996.

Mahendra Mallick, Mark Morelande, and Lyudmila Mihaylova. Continuous-discrete filtering using EKF, UKF, and PF. In *Proceedings of the 15th International Conference on Information Fusion*, pages 1087–1094, Singapore, July 2012.

Thomas Mazzoni. Computational aspects of continuous-discrete extended Kalman-filtering. *Computational Statistics*, 23(4):519–539, 2008.

Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):1–46, February 2003.

Bin Rao, Shunping Xiao, Xuesong Wang, and Yongzhen Li. Nonlinear Kalman filtering with numerical integration. *Chinese Journal of Electronics*, 20(3):452–456, July 2011.

H. J. Rome. A direct solution to the linear variance equation of a time-invariant linear system. *IEEE Transactions on Automatic Control*, 14(5):592–593, October 1969.

Wilson J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1996.

Simo Särkkä. *Recursive Bayesian Inference on Stochastic Differential Equations*. PhD thesis, Helsinki University of Technology, Finland, April 2006. ISBN 9-512-28127-9.

Simo Särkkä. On unscented Kalman filtering for state estimation of continuous-time nonlinear systems. *IEEE Transactions on Automatic Control*, 52(9):1631–1641, September 2007.

Charles F. Van Loan. Computing integrals involving the matrix exponential. *IEEE Transactions on Automatic Control*, 23(3):395–404, June 1978.

Niklas Wahlström, Patrik Axelsson, and Fredrik Gustafsson. Discretizing stochastic dynamical systems using lyapunov equations. arXiv:1402.1358., 2014.

P. Zhang, J. Gu, E.E. Milios, and P. Huynh. Navigation with IMU/ GPS/digital compass with unscented Kalman filter. In *Proceedings of the IEEE International Conference on Mechatronics and Automation*, pages 1497–1502, Niagara Falls, Ontario, Canada, July–August 2005.

Paper D

ML Estimation of Process Noise Variance in Dynamic Systems

Authors: Patrik Axelsson, Umut Orguner, Fredrik Gustafsson, and Mikael Norrlöf

Edited version of the paper:

> 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/September 2011.

ML Estimation of Process Noise Variance in Dynamic Systems

Patrik Axelsson*, Umut Orguner**, Fredrik Gustafsson*, and Mikael Norrlöf*

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
fredrik.gustafsson@liu.se,
mikael.norrlof@liu.se

**Dept. of Electrical &
Electronics Engineering,
Middle East Technical University,
06531 Ankara, Turkey
umut@metu.edu.tr

Abstract

The performance of a non-linear filter hinges in the end on the accuracy of the assumed non-linear model of the process. In particular, the process noise covariance $\mathbf{Q}$ is hard to get by physical modelling and dedicated system identification experiments. We propose a variant of the *expectation maximisation* (EM) algorithm which iteratively estimates the unobserved state sequence and $\mathbf{Q}$ based on the observations of the process. The *extended Kalman smoother* (EKS) is the instrument to find the unobserved state sequence. Our contribution fills a gap in literature, where previously only the linear Kalman smoother and particle smoother have been applied. The algorithm will be important for future industrial robots with more flexible structures, where the particle smoother cannot be applied due to the high state dimension. The proposed method is compared to two alternative methods on a simulated robot.

1 Introduction

Joint parameter identification and state estimation in state space model is an important branch of system identification [Ljung, 1999]. During the last decade, subspace based approaches for estimating fully parametrised linear state space models (so called black box models) have been well explored [Ljung, 1999]. At the same time, the theory of grey-box identification of uncertain parameters in physical models has been developed [Bohlin, 2006]. The model is here a non-linear state space model without process noise. The basic idea is that the system can be simulated for each value of the parameter vector, and the simulated output can be compared to the observed measurements, where for instance the *maximum likelihood estimate* (MLE) is computed. The situation with process noise is considerably harder, since the simulated output has to be replaced with a smoothing filter for the state sequence. A further problem is that the likelihood function becomes rather complicated. The EM algorithm in Dempster et al. [1977] provides a method to compute the MLE by separating the smoothing and parameter estimation problems. It has been explored for linear Gaussian models, where the system matrices $(\mathbf{A}, \mathbf{C}, \mathbf{Q}, \mathbf{R})$ are estimated using the Kalman smoother as the state estimator [Cappé et al., 2005]. For non-linear models, there is ongoing research on using the particle smoother to estimate the parameters in a non-linear dynamic model [Schön et al., 2011]. However, the particle smoother is not applicable for models with high state dimension.

Here we propose to use a linearised model for state estimation, leading to an *extended Kalman smoother* (EKS). The EM algorithm will thus be approximate in the same way as the EKS. We focus on the process noise covariance matrix, which is the hardest one to assess in the modelling phase. Our application in mind is industrial robots, where inertia, flexibilities and friction parameters in each joint are all rather straightforwardly identified by dedicated experiments, see Wernholt [2007] and Carvalho Bittencourt et al. [2010]. The sensor noise covariance is also quite easy to get. Process noise, on the other hand, models quite complex phenomena as well as model uncertainties.

The motivation to do this for industrial robots is the development of new robots with increased elasticity and larger individual variations, such as variation of gearbox stiffness or in the parameters describing the mechanical arm. To maintain or improve the robot performance, the motion control must be improved for this new generation of robots. For robots with traditional measurement systems, where only the motor angular position is measured, this can be obtained by improving the model-based control as described in Björkman et al. [2008]. Another option is to use inertial sensors to improve the estimation of the robot tool position and velocity, which requires good knowledge about the noise. The estimated state trajectories can be used for on-line feedback control as a mean of increasing both the robust and the nominal performance of the robot. Another possible use of tool estimation is *iterative learning control* (ILC) [Wallén et al., 2009].

Section 2 gives a short introduction to the problem and the EM algorithm. The calculations for the EM algorithm are then given in Section 3. Two alternative methods are presented in Section 4. Section 5 describes a robot model which is used in Section 6 to compare the three methods. Finally, Section 7 concludes the paper.

2 Problem Formulation

This paper is focused on a model structure given by

$$\mathbf{x}_{k+1} = F_1(\mathbf{x}_k, \mathbf{u}_k) + F_2(\mathbf{x}_k)\mathbf{w}_k, \tag{1a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k, \tag{1b}$$

where $\mathbf{x}_k \in \mathbb{R}^{n_x}$, $\mathbf{y}_k \in \mathbb{R}^{n_y}$, $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$. All model parameters are assumed to be known except for $\mathbf{Q} \in \mathbb{S}_{+}^{n_v}$ and $\mathbf{R} \in \mathbb{S}_{++}^{n_y}$[1]. Assume also that $F_2(\mathbf{x}_k)$ has the following structure

$$F_2(\mathbf{x}_k) = \begin{pmatrix} \mathbf{0} \\ \widetilde{F}_2(\mathbf{x}_k) \end{pmatrix}. \tag{2}$$

This type of model structure is common for mechanical systems derived by Newton's law or Lagrange's equation.

One approach to find the covariance matrices $\mathbf{R}$ and $\mathbf{Q}$ is to use the well known *Maximum likelihood* (ML) method. The idea with the ML method is to find the unknown parameters $\boldsymbol{\theta}$ such that the measurements $\mathbf{y}_{1:N} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ become as likely as possible. In other words,

$$\hat{\boldsymbol{\theta}}^{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta} \in \Theta} \; p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}), \tag{3}$$

where $p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N})$ is the *probability density function* (PDF) of the observations parametrised with the parameter $\boldsymbol{\theta}$. It is common to take the logarithm of the PDF,

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}) = \log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}), \tag{4}$$

and find the parameter $\boldsymbol{\theta}$ that maximises (4), i.e.,

$$\hat{\boldsymbol{\theta}}^{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta} \in \Theta} \; L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}). \tag{5}$$

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. Boyd and Vandenberghe [2009]. The solution can however be hard to find which has lead to the development of the *expectation maximisation* (EM) algorithm.

The EM algorithm was originally invented by Dempster et al. [1977]. Theory and examples for the EM algorithm can be found in McLachlan and Krishnan [2008], see also Gibson and Ninness [2005] for robust estimation of LTI state space models. In Schön et al. [2011], a solution of the more complicated problem to estimate non-linear state space models is given, using a particle smoother. As mentioned before, the particle smoother cannot be applied if the state dimension is too high.

2.1 The Expectation Maximisation Algorithm

The principal idea with the EM algorithm is to introduce variables $\mathbf{x}_{1:N}$ which are not observed directly. The variables $\mathbf{x}_{1:N}$ can instead be observed indirectly from $\mathbf{y}_{1:N}$. Take now the joint log-likelihood function

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) = \log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) \tag{6}$$

[1] $\mathbb{S}_{++}^{p}$ $\left(\mathbb{S}_{+}^{p}\right)$ is the set of all symmetric positive definite (semi-definite) $p \times p$ matrices

of the observed variables $\mathbf{y}_{1:N}$ and the variables $\mathbf{x}_{1:N}$. Equation (6) cannot be used directly since $\mathbf{x}_{1:N}$ are unknown. Instead, calculate

$$\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) = \mathrm{E}_{\boldsymbol{\theta}_l}\left[\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) | \mathbf{y}_{1:N}\right], \tag{7}$$

where $\mathrm{E}_{\boldsymbol{\theta}_l}[\,\cdot\,|\,\cdot\,]$ is the conditional mean with respect to a PDF defined by the parameter $\boldsymbol{\theta}_l$. It can now be shown, see e.g. Schön et al. [2011], that any $\boldsymbol{\theta}$, such that

$$\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) > \Gamma(\boldsymbol{\theta}_l; \boldsymbol{\theta}_l), \tag{8}$$

implies that

$$L_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}) > L_{\boldsymbol{\theta}_l}(\mathbf{y}_{1:N}). \tag{9}$$

Hence, the EM algorithm provides a sequence $\boldsymbol{\theta}_l$, $l = 1, 2, \ldots$, which approximates $\hat{\boldsymbol{\theta}}^{\mathrm{ML}}$ better and better for every iteration. The EM algorithm is summarised in Algorithm 1. A possible stop criterion for the EM algorithm could be when the change in $\boldsymbol{\theta}$, between two iterations, is small enough.

Algorithm 1 The Expectation Maximisation (EM) Algorithm

```
1: Select an initial value $\boldsymbol{\theta}_0$ and set $l = 0$.
2: Expectation Step (E-step): Calculate
       $\Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l) = \mathrm{E}_{\boldsymbol{\theta}_l}\left[\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) | \mathbf{y}_{1:N}\right].$
3: Maximisation Step (M-step): Compute
       $\boldsymbol{\theta}_{l+1} = \arg\max_{\boldsymbol{\theta} \in \Theta} \Gamma(\boldsymbol{\theta}; \boldsymbol{\theta}_l).$
4: If converged, stop. If not, set $l = l + 1$ and go to step 2.
```

3 EM for Covariance Estimation

This section describes how the covariance matrices for the process and measurement noise in (1) can be estimated using the EM algorithm. It is not possible to simultaneously estimate both the covariance matrix $\mathbf{Q}$ for the process noise and the covariance matrix $\mathbf{R}$ for the measurement noise, since it is the scaling between them that affects the estimate when an *extended Kalman filter* (EKF) [Anderson and Moore, 1979] is used. Therefore, estimate first $\mathbf{R}$ with dedicated experiments and then $\mathbf{Q}$ with the EM algorithm. The matrix $\mathbf{R}$ can be estimated according to the following steps.

1. Perform experiments and select a constant segment $\widetilde{\mathbf{y}}$ of the measured signal $\mathbf{y}$.

2. Calculate $\mathbf{v}_k = \widetilde{\mathbf{y}}_k - \bar{\mathbf{y}}$, where $\bar{\mathbf{y}}$ is the mean of $\widetilde{\mathbf{y}}$.

3. Finally,

$$\mathbf{R} = \frac{1}{N} \sum_{k=1}^{N} \mathbf{v}_k \mathbf{v}_k^{\mathrm{T}}. \tag{10}$$

The matrix $\mathbf{R}$ can now be used in the EM algorithm to estimate $\mathbf{Q}$.

Equation (1) can also be expressed in the more general conditional densities as

$$\mathbf{x}_{k+1} \sim p(\mathbf{x}_{k+1}|\mathbf{x}_k) = \mathcal{N}\left(\mathbf{x}_{k+1}; F_{1,k}, F_{2,k}\mathbf{Q}F_{2,k}^{\mathsf{T}}\right), \tag{11a}$$

$$\mathbf{y}_k \sim p(\mathbf{y}_k|\mathbf{x}_k) = \mathcal{N}\left(\mathbf{y}_k; h_k, \mathbf{R}\right), \tag{11b}$$

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

$$\mathcal{N}(\boldsymbol{\mu}; \bar{\boldsymbol{\mu}}, \Sigma) \triangleq \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mu-\bar{\mu})^{\mathsf{T}}\Sigma^{-1}(\mu-\bar{\mu})}. \tag{12}$$

The short notation

$$F_{1,k} \triangleq F_1(\mathbf{x}_k, \mathbf{u}_k), \quad F_{2,k} \triangleq F_2(\mathbf{x}_k), \quad h_k \triangleq h(\mathbf{x}_k, \mathbf{u}_k),$$

is sometimes used for simplicity in the sequel. Proceed now with the derivation of the expectation and maximisation steps in Algorithm 1, where $\boldsymbol{\theta} = \mathbf{Q}$ is the sought parameter.

3.1 Expectation step

The joint likelihood can easily be written as

$$p_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) = p_{\mathbf{Q}}(\mathbf{x}_1, \mathbf{y}_1) \prod_{i=2}^{N} p_{\mathbf{Q}}(\mathbf{y}_i|\mathbf{x}_i) p_{\mathbf{Q}}(\mathbf{x}_i|\mathbf{x}_{i-1}), \tag{13}$$

where

$$p(\mathbf{x}_k, \mathbf{y}_k|\mathbf{x}_{1:k-1}, \mathbf{y}_{1,k-1}) = p(\mathbf{x}_k, \mathbf{y}_k|\mathbf{x}_{k-1}) = p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}) \tag{14}$$

has been used repeatedly. Taking the logarithm of (13) and using (11) together with (12) give

$$\begin{aligned} L_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) = \log p_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) &= \widetilde{L} + \frac{1}{2}\sum_{i=2}^{N} \log\left(\prod_{\lambda_j \neq 0} \lambda_j\left(\left(F_{2,i-1}\mathbf{Q}F_{2,i-1}^{\mathsf{T}}\right)^{\dagger}\right)\right) \\ &\quad - \frac{1}{2}\sum_{i=2}^{N} \widetilde{F}_{1,i}^{\mathsf{T}}\left(F_{2,i-1}\mathbf{Q}F_{2,i-1}^{\mathsf{T}}\right)^{\dagger}\widetilde{F}_{1,i}, \end{aligned} \tag{15}$$

where $\widetilde{L}$ is an expression independent of $\mathbf{Q}$, $\dagger$ is the Moore-Penrose inverse [Mitra and Rao, 1971],

$$\widetilde{F}_{1,i} \triangleq \mathbf{x}_i - F_{1,i-1}, \tag{16}$$

and $\prod_{\lambda_j \neq 0} \lambda_j(\,\cdot\,)$ means to take the product of all non-zero eigenvalues. The structure of F_2 in (2) gives

$$\left(F_{2,i-1}\mathbf{Q}F_{2,i-1}^{\mathsf{T}}\right)^{\dagger} = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \widetilde{F}_{2,i-1}\mathbf{Q}\widetilde{F}_{2,i-1}^{\mathsf{T}} \end{pmatrix}^{\dagger}, \tag{17}$$

which leads to

$$\prod_{\lambda_j \neq 0} \lambda_j \left(\left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right) = \left| \left(\widetilde{F}_{2,i-1} \mathbf{Q} \widetilde{F}_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right|. \tag{18}$$

The joint log-likelihood function (15) can now be written as

$$\begin{aligned} L_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) =& \widetilde{L} + \frac{1}{2} \sum_{i=2}^{N} \log \left(\left| \left(\widetilde{F}_{2,i-1} \mathbf{Q} \widetilde{F}_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right| \right) \\ & - \frac{1}{2} \sum_{i=2}^{N} \widetilde{F}_{1,i}^{\mathsf{T}} \left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \widetilde{F}_{1,i}. \end{aligned} \tag{19}$$

Next step is to calculate the expectation of $L_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N})$ to obtain $\Gamma(\mathbf{Q}; \mathbf{Q}_l)$.

$$\begin{aligned} \Gamma(\mathbf{Q}; \mathbf{Q}_l) =& \operatorname{E}_{\mathbf{Q}_l} \left[L_{\mathbf{Q}}(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}) | \mathbf{y}_{1:N} \right] = \bar{L} + \frac{1}{2} \sum_{i=2}^{N} \operatorname{E}_{\mathbf{Q}_l} \left[\log \left(\left| \left(\widetilde{F}_{2,i-1} \mathbf{Q} \widetilde{F}_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right| \right) \middle| \mathbf{y}_{1:N} \right] \\ & - \frac{1}{2} \operatorname{tr} \sum_{i=2}^{N} \operatorname{E}_{\mathbf{Q}_l} \left[\left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \widetilde{F}_{1,i} \widetilde{F}_{1,i}^{T} \middle| \mathbf{y}_{1:N} \right] \end{aligned} \tag{20}$$

where $\bar{L}$ is independent of $\mathbf{Q}$. The trace operator comes from the fact that

$$\widetilde{F}_{1,i}^{\mathsf{T}} \left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \widetilde{F}_{1,i} = \operatorname{tr} \left(\widetilde{F}_{1,i}^{\mathsf{T}} \left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \widetilde{F}_{1,i} \right). \tag{21}$$

Start with the calculations of the first expectation in (20).

$$\begin{aligned} & \operatorname{E}_{\mathbf{Q}_l} \left[\log \left(\left| \left(\widetilde{F}_{2,i-1} \mathbf{Q} \widetilde{F}_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right| \right) \middle| \mathbf{y}_{1:N} \right] \\ & \quad = \int \log \left(\left| \left(\widetilde{F}_{2}(\mathbf{x}_{i-1}) \mathbf{Q} \widetilde{F}_{2}^{\mathsf{T}}(\mathbf{x}_{i-1}) \right)^{\dagger} \right| \right) p_{\mathbf{Q}_l}(\mathbf{x}_{i-1} | \mathbf{y}_{1:N}) \, \mathrm{d}\mathbf{x}_{i-1}. \end{aligned} \tag{22}$$

The integral cannot be solved analytically. Instead, an approximation has to be made. The smoothed density of $\mathbf{x}_{i-1}$ has a high peak around the smoothed estimate if the sampling frequency and the SNR are high. Here, we use the *extended Kalman smoother* (EKS), see Yu et al. [2004]. We can therefore approximate $\mathbf{x}_{i-1}$ with the smoothed states $\hat{\mathbf{x}}_{i-1|N}^{s}$, in other words,

$$\operatorname{E}_{\mathbf{Q}_l} \left[\log \left(\left| \left(\widetilde{F}_{2,i-1} \mathbf{Q} \widetilde{F}_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \right| \right) \middle| \mathbf{y}_{1:N} \right] \approx \log \left(\left| \left(\widetilde{F}_{2}(\hat{\mathbf{x}}_{i-1|N}^{s}) \mathbf{Q} \widetilde{F}_{2}^{\mathsf{T}}(\hat{\mathbf{x}}_{i-1|N}^{s}) \right)^{\dagger} \right| \right). \tag{23}$$

The second expectation in (20) can be written as

$$\begin{aligned} & \operatorname{E}_{\mathbf{Q}_l} \left[\left(F_{2,i-1} \mathbf{Q} F_{2,i-1}^{\mathsf{T}} \right)^{\dagger} \widetilde{F}_{1,i} \widetilde{F}_{1,i}^{\mathsf{T}} \middle| \mathbf{y}_{1:N} \right] \\ & \quad = \int \left(F_{2}(\mathbf{x}_{i-1}) \mathbf{Q} F_{2}^{\mathsf{T}}(\mathbf{x}_{i-1}) \right)^{\dagger} \widetilde{F}_{1,i} \widetilde{F}_{1,i}^{\mathsf{T}} p_{\mathbf{Q}_l}(\mathbf{x}_i, \mathbf{x}_{i-1} | \mathbf{y}_{1:N}) \, \mathrm{d}\mathbf{x}_i \, \mathrm{d}\mathbf{x}_{i-1}. \end{aligned} \tag{24}$$

Now use the smoothed density again and let

$$\left(F_2(\mathbf{x}_{i-1})\mathbf{Q}F_2^\mathsf{T}(\mathbf{x}_{i-1})\right)^\dagger \approx \left(F_2(\hat{\mathbf{x}}^s_{i-1|N})\mathbf{Q}F_2^\mathsf{T}(\hat{\mathbf{x}}^s_{i-1|N})\right)^\dagger. \tag{25}$$

We have now

$$\begin{aligned}
&\mathrm{E}_{\mathbf{Q}_l}\left[\left(F_{2,i-1}\mathbf{Q}F^\mathsf{T}_{2,i-1}\right)^\dagger \widetilde{F}_{1,i}\widetilde{F}^\mathsf{T}_{1,i}\middle|\mathbf{y}_{1:N}\right] \\
&\quad\approx \left(F_2(\hat{\mathbf{x}}^s_{i-1|N})\mathbf{Q}F_2^\mathsf{T}(\hat{\mathbf{x}}^s_{i-1|N})\right)^\dagger \int \widetilde{F}_{1,i}\widetilde{F}^\mathsf{T}_{1,i}\, p_{\mathbf{Q}_l}(\mathbf{x}_i, \mathbf{x}_{i-1}|\mathbf{y}_{1:N})\,\mathrm{d}\mathbf{x}_i\,\mathrm{d}\mathbf{x}_{i-1},
\end{aligned} \tag{26}$$

where $p_{\mathbf{Q}_l}(\mathbf{x}_i, \mathbf{x}_{i-1}|\mathbf{y}_{1:N})$ can be seen as the smoothed density of the augmented state vector $\boldsymbol{\xi}_i = \begin{pmatrix}\mathbf{x}^\mathsf{T}_{i-1} & \mathbf{x}^\mathsf{T}_i\end{pmatrix}^\mathsf{T}$, i.e.,

$$p_{\mathbf{Q}_l}(\mathbf{x}_i, \mathbf{x}_{i-1}|\mathbf{y}_{1:N}) = p_{\mathbf{Q}_l}(\boldsymbol{\xi}_i|\mathbf{y}_{1:N}) = \mathcal{N}\left(\boldsymbol{\xi}_i; \hat{\boldsymbol{\xi}}^s_{i|N}, \mathbf{P}^{\xi,s}_{i|N}\right). \tag{27}$$

The first and second order moments of the smoothed $\boldsymbol{\xi}_i$ can be expressed as

$$\hat{\boldsymbol{\xi}}^s_{i|N} = \begin{pmatrix}\hat{\mathbf{x}}^s_{i-1|N} \\ \hat{\mathbf{x}}^s_{i|N}\end{pmatrix}, \quad \mathbf{P}^{\xi,s}_{i|N} = \begin{pmatrix}\mathbf{P}^s_{i-1|N} & \mathbf{P}^s_{i-1,i|N} \\ \left(\mathbf{P}^s_{i-1,i|N}\right)^\mathsf{T} & \mathbf{P}^s_{i|N}\end{pmatrix}, \tag{28}$$

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

$$\boldsymbol{\xi}_{k+1} = \begin{pmatrix}\mathbf{x}_k \\ F_1(\mathbf{x}_k, \mathbf{u}_k)\end{pmatrix} \tag{29}$$

is used in the EKS. The integral in (26) cannot be solved analytically. Instead, a first order Taylor expansion of $F_1(\mathbf{x}_{i-1})$ around $\hat{\mathbf{x}}^s_{i-1|N}$ is used. The expression $\widetilde{F}_{1,i}\widetilde{F}^\mathsf{T}_{1,i}$ in (26) can now be written as

$$\begin{aligned}
\widetilde{F}_{1,i}\widetilde{F}^T_{1,i} =& \left(\mathbf{x}_i - F_1(\mathbf{x}_{i-1})\right)\left(\mathbf{x}_i - F_1(\mathbf{x}_{i-1})\right)^\mathsf{T} \\
\approx& \left(\mathbf{x}_i - F_1(\hat{\mathbf{x}}^s_{i-1|N}) - \mathbf{J}_{1,i-1}\cdot(\mathbf{x}_{i-1} - \hat{\mathbf{x}}^s_{i-1|N})\right) \\
&\times\left(\mathbf{x}_i - F_1(\hat{\mathbf{x}}^s_{i-1|N}) - \mathbf{J}_{1,i-1}\cdot(\mathbf{x}_{i-1} - \hat{\mathbf{x}}^s_{i-1|N})\right)^\mathsf{T} \\
=& \begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}\left(\boldsymbol{\xi}_i - \hat{\boldsymbol{\xi}}^s_{i|N}\right)\left(\boldsymbol{\xi}_i - \hat{\boldsymbol{\xi}}^s_{i|N}\right)^T\begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}^\mathsf{T} \\
&+\begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}\left(\boldsymbol{\xi}_i - \hat{\boldsymbol{\xi}}^s_{i|N}\right)\left(\hat{\mathbf{x}}^s_{i|N} - F_1(\hat{\mathbf{x}}^s_{i-1|N})\right)^\mathsf{T} \\
&+\left(\hat{\mathbf{x}}^s_{i|N} - F_1(\hat{\mathbf{x}}^s_{i-1|N})\right)\left(\boldsymbol{\xi}_i - \hat{\boldsymbol{\xi}}^s_{i|N}\right)^\mathsf{T}\begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}^\mathsf{T} \\
&+\left(\hat{\mathbf{x}}^s_{i|N} - F_1(\hat{\mathbf{x}}^s_{i-1|N})\right)\left(\hat{\mathbf{x}}^s_{i|N} - F_1(\hat{\mathbf{x}}^s_{i-1|N})\right)^\mathsf{T},
\end{aligned} \tag{30}$$

where the Taylor expansion

$$F_1(\mathbf{x}_{i-1}) \approx F_1(\hat{\mathbf{x}}^s_{i-1|N}) + \mathbf{J}_{1,i-1}\cdot(\mathbf{x}_{i-1} - \hat{\mathbf{x}}^s_{i-1|N}), \tag{31a}$$

$$\mathbf{J}_{1,i-1} = \left.\frac{\partial F_1(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}^s_{i-1|N}}, \tag{31b}$$

has been used.

The integral in (26) now becomes

$$\mathbf{M}_i \triangleq \int \widetilde{F}_{1,i}\widetilde{F}_{1,i}^{\mathsf{T}} p_{\mathbf{Q}_l}(\mathbf{x}_i, \mathbf{x}_{i-1}|\mathbf{y}_{1:N})\,\mathrm{d}\mathbf{x}_i\,\mathrm{d}\mathbf{x}_{i-1} = \begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}\mathbf{P}_{i|N}^{\xi,s}\begin{pmatrix}-\mathbf{J}_{1,i-1} & \mathbf{I}\end{pmatrix}^{\mathsf{T}} \\ + \left(\hat{\mathbf{x}}_{i|N}^{s} - F_1(\hat{\mathbf{x}}_{i-1|N}^{s})\right)\left(\hat{\mathbf{x}}_{i|N}^{s} - F_1(\hat{\mathbf{x}}_{i-1|N}^{s})\right)^{\mathsf{T}}. \tag{32}$$

It is thus possible to calculate $\Gamma(\mathbf{Q};\mathbf{Q}_l)$ according to

$$\Gamma(\mathbf{Q};\mathbf{Q}_l) = \bar{L} + \frac{1}{2}\sum_{i=2}^{N}\log\left(\left|\left(\widetilde{F}_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\right)^{\mathsf{T}}\right|\right) + \frac{1}{2}\sum_{i=2}^{N}\left[\log\left|\mathbf{Q}^{\dagger}\right| + \log\left(\left|\widetilde{F}_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\right|\right)\right] \\ - \frac{1}{2}\operatorname{tr}\mathbf{Q}^{\dagger}\sum_{i=2}^{N}F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\mathbf{M}_i\left(F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\right)^{\mathsf{T}}, \tag{33}$$

where we have used the fact that

$$\left(\widetilde{F}_{2,i-1}\mathbf{Q}\widetilde{F}_{2,i-1}^{\mathsf{T}}\right)^{\dagger} = \left(\widetilde{F}_{2,i-1}^{\dagger}\right)^{\mathsf{T}}\mathbf{Q}^{\dagger}\widetilde{F}_{2,i-1}^{\dagger}. \tag{34}$$

In (34) it is used that $\widetilde{F}_{2,i-1}^{\mathsf{T}}$ and $\mathbf{Q}$ have full row rank, and $\widetilde{F}_{2,i-1}$ and $\widetilde{F}_{2,i-1}\mathbf{Q}$ have full column rank, see Mitra and Rao [1971].

3.2 Maximisation step

Maximisation with respect to $\mathbf{Q}$ is the same as maximisation with respect to $\mathbf{Q}^{\dagger} = \mathbf{Q}^{-1}$. Take the derivative of (33) with respect to $\mathbf{Q}^{-1}$ and let the result be equal to zero gives

$$\frac{\partial\Gamma(\mathbf{Q};\mathbf{Q}_l)}{\partial\mathbf{Q}^{-1}} = \frac{N-1}{2}\mathbf{Q} - \frac{1}{2}\sum_{i=2}^{N}F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\mathbf{M}_i\left(F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\right)^{\mathsf{T}} = 0. \tag{35}$$

The solution of the maximisation step is now obtained as

$$\mathbf{Q}_{l+1} = \frac{1}{N-1}\sum_{i=2}^{N}F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\mathbf{M}_i\left(F_2^{\dagger}(\hat{\mathbf{x}}_{i-1|N}^{s})\right)^{\mathsf{T}}. \tag{36}$$

3.3 Stop Criterion

The stop criterion can be chosen in different ways. Section 2.1 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 $L_{\mathbf{Q}}(\mathbf{y}_{1:N})$ in (4). The main problem is to maximise $L_{\mathbf{Q}}(\mathbf{y}_{1:N})$, therefore stop the algorithm when no increase in $L_{\mathbf{Q}}(\mathbf{y}_{1:N})$ can be observed. Equation (4) can be written as

$$L_{\mathbf{Q}}(\mathbf{y}_{1:N}) = \log p_{\mathbf{Q}}(\mathbf{y}_{1:N}) = \log\left(p(\mathbf{y}_1)\prod_{i=1}^{N-1}p_{\mathbf{Q}}(\mathbf{y}_{i+1}|\mathbf{y}_{1:i})\right)$$

$$= \log p(\mathbf{y}_1) + \sum_{i=1}^{N-1} \log p_{\mathbf{Q}}(\mathbf{y}_{i+1}|\mathbf{y}_{1:i}), \tag{37}$$

where Bayes' rule has been used repeatedly. Here, $\log p(\mathbf{y}_1)$ is a constant and can be omitted in the sequel for simplicity. The PDF $p_{\mathbf{Q}}(\mathbf{y}_{i+1}|\mathbf{y}_{1:i})$ is identified as the PDF for the innovations which can be calculated as

$$p_{\mathbf{Q}}(\mathbf{y}_{i+1}|\mathbf{y}_{1:i}) = \mathcal{N}\left(\mathbf{y}_{i+1}; h(\hat{\mathbf{x}}_{i+1|i}), \mathbf{H}_{i+1}\mathbf{P}_{i+1|i}\mathbf{H}_{i+1}^{\mathsf{T}} + \mathbf{R}\right), \tag{38}$$

$$\mathbf{H}_{i+1} = \left.\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{i+1|i}}, \tag{39}$$

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

$$\left|L_{\mathbf{Q}_l}(\mathbf{y}_{1:N}) - L_{\mathbf{Q}_{l-m}}(\mathbf{y}_{1:N})\right| \le \gamma, \tag{40}$$

where m and γ are parameters to choose.

4 Alternative Ways to Find the Covariance Matrix of the Process Noise

There are many alternative ways of estimating the covariance matrix for the process noise and here two examples will be described. These two alternatives, which are less complicated than the EM algorithm, will be compared to the result of the EM algorithm in Section 6.

The first method, presented in Algorithm 2, minimises the path error

$$\mathsf{e}_k = \sqrt{|\mathsf{x}_k - \hat{\mathsf{x}}_k|^2 + |\mathsf{y}_k - \hat{\mathsf{y}}_k|^2}, \tag{41}$$

where x_k and y_k are the true coordinates for the tool, and $\hat{\mathsf{x}}_k$ and $\hat{\mathsf{y}}_k$ are the estimated coordinates for the tool. To simplify the problem, the covariance matrix is parametrised as a diagonal matrix. The notation $(\hat{\mathsf{x}}, \hat{\mathsf{y}}) = \text{EKF}(\mathbf{Q})$ in Algorithm 2 denotes that the estimated position is a function of $\mathbf{Q}$. A standard optimisation method can be used to solve step 2 in Algorithm 2, see e.g. Boyd and Vandenberghe [2009]. The problem is not convex, i.e., the solution is not guaranteed to give a global optimum. However, the method is straightforward and has been used before, see Henriksson et al. [2009] and Axelsson [2009]. One disadvantage with the method is that the true tool position is required.

The second method starts with an initial guess $\mathbf{Q}_0$. The smoothed states are then calculated using $\mathbf{Q}_0$. After that, equation (1a) and the smoothed states are used in order to derive the noise $\mathbf{w}_k$, $k = 1, \ldots, N$. The covariance matrix is finally obtained from the vector $\mathbf{w}_{1:N} = \{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$. The method is repeated with the new $\mathbf{Q}$-matrix until convergence is obtained. The method is summarised Algorithm 3.

Algorithm 2 Minimisation of the path error

```
1: Select an initial diagonal matrix $\mathbf{Q}_0 \in \mathbb{R}^{4\times 4}$.
2: Minimise $\sqrt{\sum_{k=1}^{N} |\mathrm{e}_k|^2}$ subject to $\lambda_j > 0$, $j = 1,\ldots,4$, $(\hat{\mathrm{x}}, \hat{\mathrm{y}}) = \textsc{ekf}(\mathbf{Q})$, and
   $\mathbf{Q} = \mathrm{diag}\,(\lambda_1, \lambda_2, \lambda_3, \lambda_4)\,\mathbf{Q}_0$.
3: The sought covariance matrix $\mathbf{Q}$ is obtained as $\mathbf{Q} = \mathrm{diag}\left(\lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*\right)\mathbf{Q}_0$
   where $\lambda_j^*$, $j = 1,\ldots,4$, are the optimal values from step 2.
```

Algorithm 3 Iterative covariance estimation with EKS

```
1: Select an initial value $\mathbf{Q}_0$ and set $l = 0$.
2: Use the EKS with $\mathbf{Q}_l$.
3: Calculate the noise according to
   $$\mathbf{w}_k = F_2^{\dagger}(\hat{\mathbf{x}}^s_{k|N})\left(\hat{\mathbf{x}}^s_{k+1|N} - F_1(\hat{\mathbf{x}}^s_{k|N}, \mathbf{u}_k)\right).$$
4: Let $\mathbf{Q}_{l+1}$ be the covariance matrix for $\mathbf{w}_k$ according to
   $$\mathbf{Q}_{l+1} = \frac{1}{N}\sum_{k=1}^{N} \mathbf{w}_k \mathbf{w}_k^{\mathsf{T}}.$$
5: If converged, stop, If not, set $l = l + 1$ and go to step 2.
```

5 Application to Industrial Robots

The robot model is a joint flexible two axes model from Moberg et al. [2008]. The model assumes rigid links and flexible joints. Each joint is a two mass system consisting of two angles, the arm angle q_{ai}, and the motor angle q_{mi}. Let the state vector be given by

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1^{\mathsf{T}} & \mathbf{x}_2^{\mathsf{T}} & \mathbf{x}_3^{\mathsf{T}} & \mathbf{x}_4^{\mathsf{T}} \end{pmatrix}^T = \begin{pmatrix} \mathbf{q}_a^{\mathsf{T}} & \mathbf{q}_m^{\mathsf{T}} & \dot{\mathbf{q}}_a^{\mathsf{T}} & \dot{\mathbf{q}}_m^{\mathsf{T}} \end{pmatrix}^T, \tag{42}$$

where $\mathbf{q}_a = \begin{pmatrix} q_{a1} & q_{a2} \end{pmatrix}^{\mathsf{T}}$, $\mathbf{q}_m = \begin{pmatrix} q_{m1} & q_{m2} \end{pmatrix}^{\mathsf{T}}$, contain the arm angles $\mathbf{q}_a$ and the motor angles $\mathbf{q}_m$ of both joints. The model accounts for flexibilities in the joints via non-linear stiffness and linear viscous damping. The model also includes non-linear friction. A continuous-time state space model of the system is given by,

$$\dot{\mathbf{x}} = \begin{pmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \\ M_a^{-1}(\mathbf{x}_1)\Big(- C(\mathbf{x}_1, \mathbf{x}_3) - G(\mathbf{x}_1) - N(\mathbf{x}) + \mathbf{w}_a\Big) \\ \mathbf{M}_m^{-1}\Big(N(\mathbf{x}) + F(\mathbf{x}_4) + \mathbf{u} + \mathbf{w}_m\Big) \end{pmatrix}, \tag{43}$$

where $N(\mathbf{x}) = \mathbf{D}\cdot(\mathbf{x}_3 - \mathbf{x}_4) + T(\mathbf{x}_1, \mathbf{x}_2)$. $N(\mathbf{x})$ accounts for the flexibilities in the joints, via the linear viscous damping $\mathbf{D}\cdot(\mathbf{x}_3 - \mathbf{x}_4)$ and the non-linear stiffness $T(\mathbf{x}_1, \mathbf{x}_2)$. In other words, if we dispense with $N(\mathbf{x})$, we are back at a standard rigid robot model. Furthermore, $M_a(\mathbf{x}_1)$ and $\mathbf{M}_m$ are the mass matrices for the arm and motor, $C(\mathbf{x}_1, \mathbf{x}_3)$ accounts for the centrifugal and centripetal torques, and $G(\mathbf{x}_1)$ accounts for the effect of gravity on the links. The non-linear friction is described by $F(\mathbf{x}_3)$, $\mathbf{u}$ represents the motor torque applied to the robot and $\mathbf{w} = \begin{pmatrix} \mathbf{w}_a^{\mathsf{T}} & \mathbf{w}_m^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}}$ is the process noise. An Euler forward approximation of (43) gives a discretised model according to (1) and (2). The rank conditions in order to use (34) are also satisfied.

6 Simulation Results

The method in Section 3 is evaluated and compared to the two alternative methods described in Section 4. The model given in Section 2 is first simulated, according to Axelsson [2009], to get all the required quantities, i.e., $\mathbf{u}_k$, $\mathbf{y}_k$, x_k and y_k. 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. Instead, the estimated covariance matrices, for different initial values, are used to calculate the path error according to (41). When the path error differs a lot with different initial values it means that the method converges to different local optima. 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 2-norm of the path error 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.

Table 1 shows that the maximal and minimal path errors for the EM algorithm are more or less the same. The same concerns Algorithm 3. The EM algorithm gives however a lower path error. Algorithm 2 gives, on the other hand, path errors that differs considerably. This can also be seen in Figure 1. This means that Algorithm 2 gets stuck in different local optima. A comparison between the path errors for the EM algorithm, Algorithm 3 and the best solution of Algorithm 2 is shown in Figure 2. The EM algorithm is clearly much better than the two alternatives.

It is also interesting to see how (37) looks like for $\mathbf{Q}_l$, $l = 0, \ldots,$ both for the EM algorithm and Algorithm 3. The EM algorithm and Algorithm 3 were therefore forced to take more iterations than necessary. The log-likelihood function (37) can be seen in Figure 3 for 100 iterations. We see 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

Table 1: *Max and min of the 2-norm of the path error in mm for the three different methods.*

	Max	Min
EM	0.2999	0.2996
Alg. 2	3.3769	1.5867
Alg. 3	2.6814	2.6814

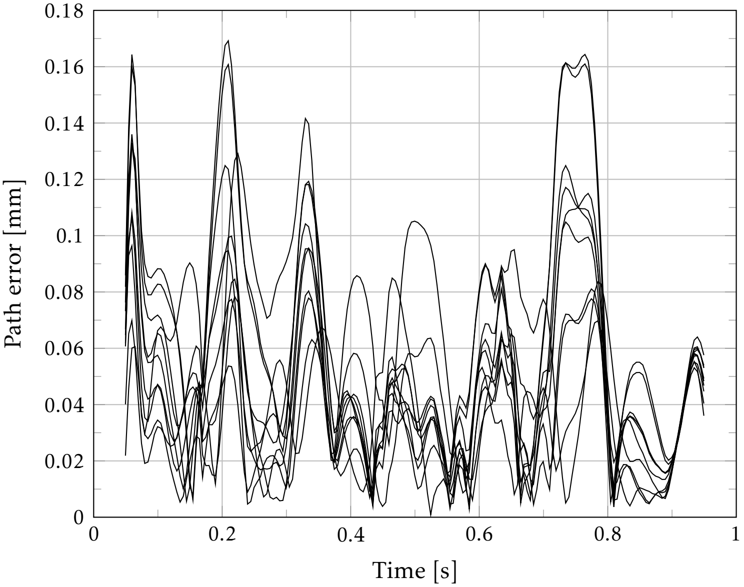


Figure 1: *The path error for 10 Monte Carlo simulations of Algorithm 2.*

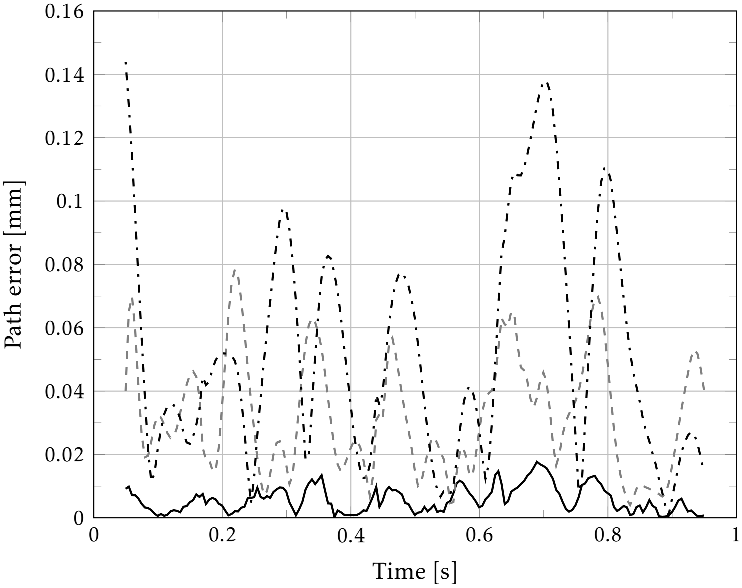


Figure 2: *The best path error for the* EM *algorithm (solid), Algorithm 2 (dashed) and Algorithm 3 (dash-dot).*

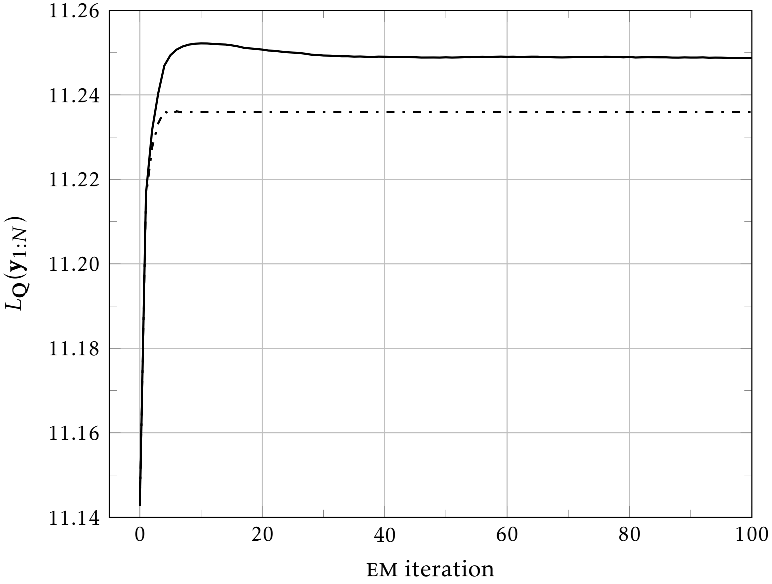


Figure 3: *The log-likelihood function for the first 100 iterations in the* EM *algorithm (solid) and Algorithm 3 (dash-dot).*

thing to comment is the peak around 10 iterations in the curve. This contradicts the property of the EM algorithm that the sequence $\mathbf{Q}_l$, $l = 0, \ldots,$ approximates $\hat{\mathbf{Q}}^{\text{ML}}$ better and better. This can be explained by the approximations that have been made during the expectation step and that the calculation of (37) in the EKF is approximately. The curve for Algorithm 3 flattens out after 10 iterations and stays constant after that. Algorithm 3 is also without any peak and the stationary value is lower than for the EM algorithm.

7 Conclusions and Future Work

Three different methods to estimate the covariance matrices have been compared. The EM algorithm derived in Section 3 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 2. A next step is to use the EM algorithm on experimental data to estimate the covariance matrices.

Acknowledgement

This work was supported by the Swedish Research Council under the Linnaeus Center CADICS and by the Vinnova Excellence Center LINK-SIC.

Bibliography

Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Information and System Sciences Series. Prentice Hall Inc., Englewood Cliffs, NJ, USA, 1979.

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 Enginering, Linköping University, SE-581 83 Linköping, Sweden, December 2009.

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/September 2011.

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.

Torsten Bohlin. *Practical Grey-box Process Identification, Theory and Applications*. Advances in Industrial Control. Springer, London, UK, 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, United Kingdom, first edition, 2009.

Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer, New York, NY, USA, 2005.

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.

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.

Stuart Gibson and Brett Ninness. Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682, October 2005.

Robert Henriksson, Mikael Norrlöf, 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.

Lennart Ljung. *System Identification, Theory for the User*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1999.

Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, NJ, USA, second edition, 2008.

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.

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.

Thomas B. Schön, Adrian Wills, and Brett Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, January 2011.

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.

Erik Wernholt. *Multivariable Frequency-Domain Identification of Industrial Robots*. Linköping Studies in Science and Technology. Dissertations No. 1138, Linköping University, SE-581 83 Linköping, Sweden, November 2007. http://www.control.isy.liu.se/publications/.

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`, 19 October 2004.

Paper E

$\mathcal{H}_\infty$-Controller Design Methods Applied to One Joint of a Flexible Industrial Manipulator

Authors: Patrik Axelsson, Anders Helmersson and Mikael Norrlöf

Edited version of the paper:

Patrik Axelsson, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-controller design methods applied to one joint of a flexible industrial manipulator. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014b.

$\mathcal{H}_\infty$-Controller Design Methods Applied to One Joint of a Flexible Industrial Manipulator

Patrik Axelsson, Anders Helmersson and Mikael Norrlöf

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
anders.helmersson@liu.se,
mikael.norrlof@liu.se

Abstract

Control of a flexible joint of an industrial manipulator using $\mathcal{H}_\infty$-design methods is presented. The considered design methods are i) mixed-$\mathcal{H}_\infty$ design, and ii) $\mathcal{H}_\infty$ loop shaping design. Two different controller configurations are examined: one uses only the actuator position, while the other uses the actuator position and the acceleration of the end-effector. The four resulting controllers are compared to a standard PID controller where only the actuator position is measured. The choices of the weighting functions are discussed in details. For the loop shaping design method, the acceleration measurement is required to improve the performance compared to the PID controller. For the mixed-$\mathcal{H}_\infty$ method it is enough to have only the actuator position to get an improved performance. Model order reduction of the controllers is briefly discussed, which is important for implementation of the controllers in the robot control system.

1 Introduction

The requirements for controllers in modern industrial manipulators are that they should provide high performance, at the same time, robustness to model uncertainty. In the typical standard control configuration the actuator positions are the only measurements used in the higher level control loop. At a lower level, in the drive system, the currents and voltages in the motor are measured to provide torque control for the motors. In this contribution different $\mathcal{H}_\infty$-controller design schemes are compared when using two different sensor configurations. First, the standard case where only the position of the actuator rotation is used, and second a configuration where, in addition, the acceleration of the tool tip is measured. Two different $\mathcal{H}_\infty$ methods are investigated: i) loop shaping [McFarlane and Glover, 1992], and ii) multi-$\mathcal{H}_\infty$ design [Pipeleers and Swevers, 2013; Zavari et al., 2012].

Motivated by the conclusions from Sage et al. [1999] regarding the area of robust control applied to industrial manipulators, this contribution includes:

- results presented using realistic models,
- a comparison with a standard PID control structure,
- model reduction of the controllers to get a result that more easily can be implemented in practice.

The model used in this contribution represents one joint of a typical modern industrial robot [Moberg et al., 2009]. It is a physical model consisting of four masses, which should be compared to the typical two-mass model used in many previous contributions, see Sage et al. [1999] and the references therein. The joint model represents the first joint of a serial 6-DOF industrial manipulator, where the remaining five axes have been configured to minimise the couplings to the first axis. To handle changes in the configuration of the remaining axes, gain scheduling techniques can be used based on the results in this paper.

An important part of the design is the choice of the weighting functions, which is an essential task to get a satisfactory performance. The work of choosing the weights is difficult, tedious and time consuming. This can be be the reasons for why $\mathcal{H}_\infty$ methods are not used that often in practice even though the performance and robustness can be increased. In particular, the use of two measurements for control of one variable requires special treatment. The development of the weighting functions for the four controllers is discussed in details, and provides a significant part of the contributions in the paper.

Controller synthesis using $\mathcal{H}_\infty$ methods has been proposed in Song et al. [1992]; Stout and Sawan [1992], where the complete non-linear robot model first is linearised using exact linearisation, second an $\mathcal{H}_\infty$ controller is designed using the linearised model. The remaining non-linearities due to model errors are seen as uncertainties and/or disturbances. In both papers, the model is rigid and the $\mathcal{H}_\infty$ controller, using only joint positions, is designed using the mixed-sensitivity method. In Sage et al. [1997] $\mathcal{H}_\infty$ loop shaping with measurements of the actuator positions is applied to a robot. The authors use a flexible joint model which has been linearised. The linearised model makes it possible to use decentralised control, hence $\mathcal{H}_\infty$ loop shaping is applied to n SISO-systems instead of the complete MIMO-system.

Explicit use of acceleration measurements for control in robotic applications has been reported in, for example, de Jager [1993]; Dumetz et al. [2006]; Kosuge et al. [1989]; Readman and Bélanger [1991] and Xu and Han [2000]. In Dumetz et al. [2006], a control law using motor position and acceleration of the load in the feedback loop is proposed for a Cartesian robot[1]. The robot is assumed to be

[1] For a Cartesian robot the joint acceleration is measured directly by an accelerometer, while for a serial type robot there is a non-linear mapping depending on the states.

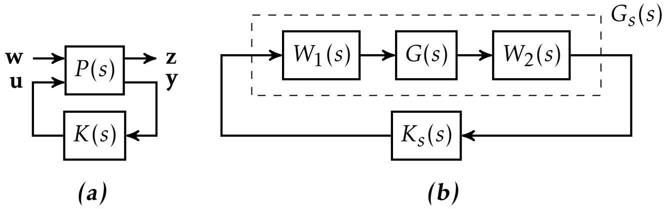


Figure 1: *System description for general $\mathcal{H}_\infty$ synthesis (a) and loop shaping (b).*

flexible and modelled as a two-mass system, where the masses are connected by a linear spring-damper pair. Another control law of a Cartesian robot using acceleration measurements is presented in de Jager [1993]. The model is a rigid joint model and the evaluation is made both in simulation and experiments.

In Kosuge et al. [1989] a 2-*degrees-of-freedom* (DOF) manipulator is controlled using acceleration measurements of the end-effector. The model is assumed to be rigid and it is exactly linearised. The joint angular acceleration used in the nonlinear feedback loop is calculated using the inverse kinematic acceleration model and the measured acceleration. The use of direct measurements of the angular acceleration in the feedback loop is presented in Readman and Bélanger [1991] for both rigid and flexible joint models. A more recent work is presented in Xu and Han [2000], where a 3-DOF manipulator is controlled using only measurements of the end-effector acceleration.

The theory for synthesis of $\mathcal{H}_\infty$ controllers is presented in Section 2. The model describing the robot joint is explained in Section 3. In Section 4, the requirements of the system as well as the design of four controllers are described, and in Section 5 the simulation results are shown. Finally, Section 6 discuss low order controller synthesis and Section 7 concludes the work.

2 Controller Design Methods

In this section, a brief introduction to mixed-$\mathcal{H}_\infty$ design [Pipeleers and Swevers, 2013; Zavari et al., 2012] and $\mathcal{H}_\infty$ loop shaping [McFarlane and Glover, 1992] will be presented.

2.1 Mixed-$\mathcal{H}_\infty$ Controller Design

A common design method is to construct the system $P(s)$ in

$$\begin{pmatrix} \mathbf{z} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{u} \end{pmatrix} = P(s) \begin{pmatrix} \mathbf{w} \\ \mathbf{u} \end{pmatrix} \tag{1}$$

by augmenting the original system $\mathbf{y} = G(s)\mathbf{u}$ with the weights $W_{\mathbf{u}}(s)$, $W_S(s)$, and $W_T(s)$, such that the system $\mathbf{z} = F_l(P, K)\mathbf{w}$, depicted in Figure 1a, can be written as

$$F_l(P,K) = \begin{pmatrix} W_{\mathbf{u}}(s)G_{\mathbf{wu}}(s) \\ -W_T(s)T(s) \\ W_S(s)S(s) \end{pmatrix}, \tag{2}$$

where $S(s) = (I + G(s)K(s))^{-1}$ is the sensitivity function, $T(s) = I - S(s)$ is the complementary sensitivity function, and $G_{\mathbf{wu}}(s) = -K(s)(I + G(s)K(s))^{-1}$ is the transfer function from **w** to **u**. The $\mathcal{H}_\infty$ controller is then obtained by minimising the $\mathcal{H}_\infty$-norm of the system $F_l(P,K)$, i.e., minimise γ such that $\|F_l(P,K)\|_\infty < \gamma$. Using (2) gives

$$\bar{\sigma}(W_{\mathbf{u}}(i\omega)G_{\mathbf{wu}}(i\omega)) < \gamma,\ \forall\omega, \tag{3a}$$

$$\bar{\sigma}(W_T(i\omega)T(i\omega)) < \gamma,\ \forall\omega, \tag{3b}$$

$$\bar{\sigma}(W_S(i\omega)S(i\omega)) < \gamma,\ \forall\omega. \tag{3c}$$

The transfer functions $G_{\mathbf{wu}}(s)$, $S(s)$, and $T(s)$ can now be shaped to satisfy the requirements by choosing the weights $W_{\mathbf{u}}(s)$, $W_S(s)$, and $W_T(s)$. The aim is to get a value of γ close to 1, which in general is hard to achieve and it requires insight in the deign method as well as the system dynamics. For more details about the design method, see e.g. Skogestad and Postletwaite [2005]; Zhou et al. [1996].

The mixed-$\mathcal{H}_\infty$ controller design [Pipeleers and Swevers, 2013; Zavari et al., 2012] is a modification of the standard $\mathcal{H}_\infty$-design method. Instead of choosing the weights in (2) such that the norm of all weighted transfer functions satisfies (3), the modified method divides the problem into design constraints and design objectives. The controller can now be found as the solution to

$$\underset{K(s)}{\text{minimise}} \quad \gamma \tag{4a}$$

$$\text{subject to} \quad \|W_P(s)S(s)\|_\infty < \gamma \tag{4b}$$

$$\|M_S(s)S(s)\|_\infty < 1 \tag{4c}$$

$$\|W_{\mathbf{u}}(s)G_{\mathbf{wu}}(s)\|_\infty < 1 \tag{4d}$$

$$\|W_T(s)T(s)\|_\infty < 1 \tag{4e}$$

where γ not necessarily has to be close to 1. Here, the weight $W_S(s)$ has been replaced by the weights $M_S(s)$ and $W_P(s)$. The method can be generalised to other control structures and in its general form it is formulated as a multi-objective optimisation problem. More details about the general form and how to solve the optimisation problem are presented in Pipeleers and Swevers [2013]; Zavari et al. [2012].

2.2 Loop Shaping using $\mathcal{H}_\infty$ Synthesis

For loop shaping [McFarlane and Glover, 1992], the system $G(s)$ is pre- and post-multiplied with weights $W_1(s)$ and $W_2(s)$, see Figure 1b, such that the shaped system $G_s(s) = W_2(s)G(s)W_1(s)$ has the desired properties. The controller $K_s(s)$ is then obtained using the method described in Glover and McFarlane [1989] applied on the system $G_s(s)$, giving the controller $K_s(s)$. Finally, the controller

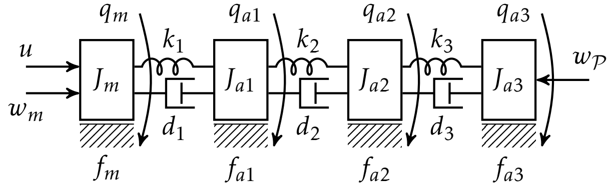


Figure 2: *A four-mass flexible joint model, where J_m is the motor inertia and J_{a1}, J_{a2}, and J_{a3} are the distributed arm inertias.*

$K(s)$ is given by

$$K(s) = W_1(s)K_s(s)W_2(s). \tag{5}$$

Note that the structure in Figure 1b for loop shaping can be rewritten as a standard $\mathcal{H}_\infty$ problem according to Figure 1a, see Zhou et al. [1996] for details. It will be used in Section 6 for synthesis of low order controllers.

The MATLAB function `ncfsyn`, included in the Robust Control Toolbox, is used in this paper for synthesis of $\mathcal{H}_\infty$ controllers using loop shaping.

3 Flexible Joint Model

The model considered in this paper is a four-mass benchmark model of a single flexible joint, see Figure 2, presented in Moberg et al. [2009]. The model corresponds to joint 1 of a serial 6-DOF industrial manipulator, where the five remaining axes are configured such that the couplings to joint 1 are minimised, see Moberg et al. [2009] for more details about the operating point where the model has been linearised.

Input to the system is the motor torque u, the motor disturbance w_m and the end-effector disturbance $w_{\mathcal{P}}$. The four masses are connected by spring-damper pairs, where the first mass corresponds to the motor. The other masses represents distributed masses placed along the arm. The first spring-damper pair is modelled by a linear damper and non-linear spring, whereas the other spring-damper pairs are modelled as linear springs and dampers. The non-linear spring is characterised by a low stiffness for low deflections and a high stiffness for high deflections. This behaviour is typical for compact gear boxes, such as harmonic drive [Ruderman and Bertram, 2012]. For design of the $\mathcal{H}_\infty$ controllers, the non-linear model is linearised in one operating point in the high stiffness region, motivated by that a constant torque, e.g. gravity, is acting on the joint. Moreover, the friction torques are assumed to be linear and the input torque u is limited to ±20 Nm. The outputs of the system are the motor position q_m and the end-effector acceleration $\ddot{\mathcal{P}}$, where

$$\mathcal{P} = \frac{l_1 q_{a1} + l_2 q_{a2} + l_3 q_{a3}}{\eta}. \tag{6}$$

In (6), η is the gear ratio and l_1, l_2, and l_3 are the respective link lengths.

Using Lagrange's equation, the linearised flexible joint model can be described by a set of four ODEs, which can be reformulated as a linear state space model according to

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B_u}\mathbf{u} + \mathbf{B_w}\mathbf{u}, \tag{7a}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D_u}\mathbf{u} + \mathbf{D_w}\mathbf{w}. \tag{7b}$$

where the state vector and disturbance vector are given by

$$\mathbf{x} = \begin{pmatrix} q_m & q_{a1} & q_{a2} & q_{a3} & \dot{q}_m & \dot{q}_{a1} & \dot{q}_{a2} & \dot{q}_{a3} \end{pmatrix}^{\mathsf{T}}, \tag{8a}$$

$$\mathbf{w} = \begin{pmatrix} w_m & w_{\mathcal{P}} \end{pmatrix}^{\mathsf{T}}. \tag{8b}$$

The linear state space model is used for design of the $\mathcal{H}_\infty$ controllers. Note that the matrices $\mathbf{C}$, $\mathbf{D_u}$, and $\mathbf{D_w}$ are different depending on which sensor configuration that is used, whereas the matrices $\mathbf{A}$, $\mathbf{B_u}$, and $\mathbf{B_w}$ stay the same.

4 Design of Controllers

In this section, four controllers based on the methods in Sections 2.1 and 2.2 are considered, using only the motor angle q_m or both q_m and the acceleration of the end-effector $\ddot{\mathcal{P}}$. The controllers are

1. $\mathcal{H}_\infty^{ls}(q_m)$: Loop shaping controller using q_m.
2. $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$: Loop shaping controller using q_m and $\ddot{\mathcal{P}}$.
3. $\mathcal{H}_\infty^{m}(q_m)$: Mixed-$\mathcal{H}_\infty$ controller using q_m.
4. $\mathcal{H}_\infty^{m}(q_m, \ddot{\mathcal{P}})$: Mixed-$\mathcal{H}_\infty$ controller using q_m and $\ddot{\mathcal{P}}$.

The four controllers are compared to a PID controller where only q_m is used. The PID controller is tuned to give the same performance as the best controller presented in Moberg et al. [2009].

To get high enough gain for low frequencies, without having the pole exactly in 0, the break-point of the magnitude function has to be very small, around 10^{-5} rad/s. From Figure 3 it can be seen that the main dynamics of the system is present in the frequency interval 30-110 rad/s. The large frequency span from 10^{-5} rad/s to 110 rad/s makes it numerically difficult to solve for the controller using the standard iterative methods described in Skogestad and Postletwaite [2005]; Zhou et al. [1996]. The mixed-$\mathcal{H}_\infty$ method does not suffer from this, since the design objectives (choice of W_P) for low frequencies is separated from the design constraints (choice of M_S).

4.1 Requirements

The controllers using $\mathcal{H}_\infty$ methods are designed to give better performance than the PID controller. In practice it means that the $\mathcal{H}_\infty$ controllers should attenuate

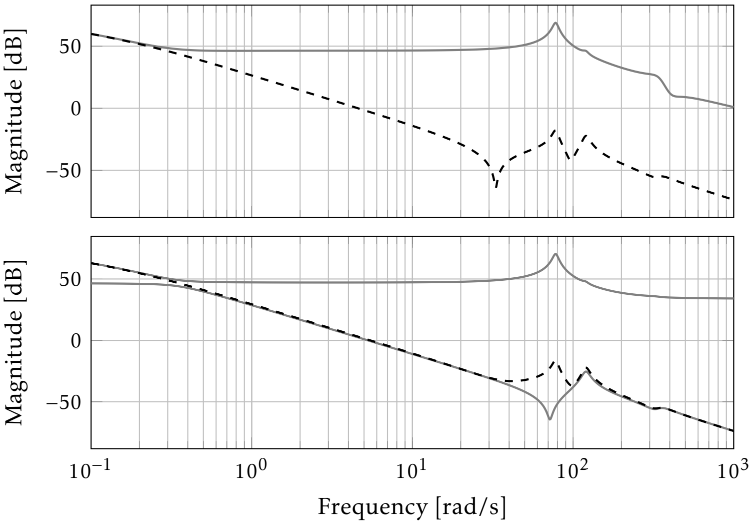


Figure 3: *Singular values for the system from* $\mathbf{u}$ *to* $\mathbf{y}$ *(top) and* $\mathbf{w}$ *to* $\mathbf{y}$ *(bottom), for* $\mathbf{y} = q_m$ *(dashed) and* $\mathbf{y} = (q_m \quad \ddot{\mathcal{P}})^{\mathsf{T}}$ *(solid).*

the disturbances at least as much as the PID controller and the cut-off frequency should be approximately the same.

In Figure 3, the singular values of the systems from $\mathbf{w}$ to $\mathbf{y} = q_m$ and $\mathbf{w}$ to $\mathbf{y} = \begin{pmatrix} q_m & \ddot{\mathcal{P}} \end{pmatrix}^{\mathsf{T}}$ show that an integrator is present. It means that in order to attenuate piecewise constant disturbances, it is required to have at least two integrators in the open loop GK. Since G already has one integrator, the other integrators have to be included in the controller K. For controllers 2 and 4, an integrator will be present if W_1 or W_2 include an integrator, recall (5). The requirements for controllers 1 and 3 become that $|S(i\omega)| \to 0$ for $\omega \to 0$. Note that it is not possible to stabilise the plant $P(s)$ with marginally stable weights. Instead the pole has to be moved into the left half plan a small distance.

4.2 Choice of Weights

$\boldsymbol{\mathcal{H}_\infty^{ls}(q_m)}$**:** Using only q_m as a measurement gives a SISO-system, hence W_1 and W_2 are scalar transfer functions. For a linear SISO-system it is possible to use one of W_1 and W_2 since the transfer functions commute with the system $G(s)$. Therefore, $W_1(s) = 1$ and $W_2(s)$ is chosen such that the desired loop shape is obtained. First of all, it is necessary to have an integrator as discussed above. Having a pure integrator will lead to that the phase margin will be decreased, a zero in $s = -10$ is therefore added in order not to change the loop gain for frequencies above 10 rad/s. Next, the gain is increased to get the desired cut-off frequency. The result using the weight is that the loop shape has peaks above

30 rad/s. To reduce the magnitude of the peaks a modified elliptic filter[2]

$$H(s) = \frac{0.5227s^2 + 3.266s + 1406}{s^2 + 5.808s + 2324} \tag{9}$$

is introduced in W_2. The weights are finally given as

$$W_1(s) = 1, \quad W_2(s) = 100\frac{s+10}{s}H(s). \tag{10}$$

Using `ncfsyn` a controller of order 13 is obtained.

$\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$: Adding an extra measurement signal in terms of the acceleration of the end-effector gives a system with one input and two outputs. For stability reasons, it is not possible to have an integrator in both control channels. Therefore, the integrator is placed in the channel for q_m since the accelerometer measurement has low frequency noise, such as drift. For the same reason as for the other controller, a zero in $s = -3$ is introduced. The transfer function from input torque to acceleration of the end-effector has a high gain in the frequency range of interest. To decrease the gain such that it is comparable with the motor angle measurement, a low pass filter is added in the acceleration channel. The final weights are

$$W_1(s) = 50, \quad W_2(s) = \operatorname{diag}\left(\frac{s+3}{s}, \frac{0.2}{(s+5)^2}\right), \tag{11}$$

giving a controller of order 13. Introducing an elliptic filter to attenuate the peaks in the open loop did not give the same results as for the $\mathcal{H}_\infty^{ls}(q_m)$-controller. Instead of improving the loop gain, the elliptic filter made it worse.

$\mathcal{H}_\infty^{m}(q_m)$: For this controller, four different weights have to be chosen, recall (4). The weight M_S should limit the peak of S and is therefore chosen to be a constant[3]. The peak of $G_{\mathbf{wu}}$ is also important to reduce in order to keep the control signal bounded, especially for high frequencies. A constant value of $W_{\mathbf{u}}$ is therefore also chosen. In the spirit of try simplest thing first, the weight W_T is also chosen to be a constant

In order to attenuate the disturbances it is, as was mentioned above, necessary to have at least one integrator in the controller. Forcing S to 0 is the same as letting W_P approach ∞ when $\omega \to 0$. To get a proper inverse, a zero is also included in the weight. Since a pure integrator is not used, the slope of the weight has to be higher than 20 dB per decade frequency, in order to force S to be low enough. This was accomplished by taking the weight to the power of 3 (2 was not enough). The numerical values of the weights are chosen as

$$W_{\mathbf{u}} = 10^{-50/20}, \quad W_T = 10^{-10/20}, \tag{12a}$$

$$M_S = 10^{-10/20}, \quad W_P = \left(\frac{s+100.1}{s+0.1}\right)^3. \tag{12b}$$

[2] The filter is designed to have a magnitude of 0 dB up to 50 rad/s, after that -10 dB, but due to ripple, the real magnitude will differ from that.

[3] More complicated weights can be used but here we try simple things first.

The constant weights in the form $10^{-\alpha/20}$ can be interpret as a maximum value, for the corresponding transfer function, of α dB. The resulting controller is of order 10.

$\mathcal{H}_\infty^m(q_m, \ddot{\mathcal{P}})$: Like for the controller $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$, designing the weights for the mixed-$\mathcal{H}_\infty$ method becomes somewhat more involved with two measurements and one control signal. The aim is to attenuate the disturbances influence on the end-effector position. A variant is to find a rough estimate of the end-effector position and then choosing the weights from that. A straightforward estimate of $\mathcal{P}$ using $\ddot{\mathcal{P}}$ is

$$\hat{\mathcal{P}} = \frac{1}{s^2}\ddot{\mathcal{P}}. \tag{13}$$

Due to low frequency drift and bias in an accelerometer, this estimate is only useful for high frequencies. A high pass filter is therefore used according to

$$\hat{\mathcal{P}}_{\text{high}} = c_2 \frac{s^2}{(p+s)^2}\frac{1}{s^2}\ddot{\mathcal{P}} = c_2 \frac{1}{(p+s)^2}\ddot{\mathcal{P}} \tag{14}$$

where c_2 and p are constants to choose. Another straightforward estimate of $\mathcal{P}$ is to use the motor angle q_m according to $\hat{\mathcal{P}} = lq_m$ where l is the length of the arm. Compared to the estimated position using the acceleration, this new estimate is only valid for low frequencies. Using a low pass filter gives an estimate for low frequencies. It is important that the two estimates do not overlap each other, hence the low pass filter is chosen as the complementarity to the previous used high pass filter. The low frequency estimate is now given by

$$\hat{\mathcal{P}}_{\text{low}} = c_1\left(1 - \frac{s^2}{(p+s)^2}\right) lq_m = c_1 \frac{2s+p}{(p+s)^2} plq_m \tag{15}$$

where c_1 is a design variable. The final estimate of $\mathcal{P}$ is the sum of the two estimates above, hence

$$\hat{\mathcal{P}} = \underbrace{\begin{pmatrix} c_1 \frac{2s+p}{(p+s)^2} pl & c_2 \frac{1}{(p+s)^2} \end{pmatrix}}_{W} \begin{pmatrix} q_m \\ \ddot{\mathcal{P}} \end{pmatrix} \tag{16}$$

Using the weights

$$M_S = \tilde{M}_S W, \quad W_P = \tilde{W}_P W, \quad W_T = \tilde{W}_T W, \tag{17}$$

where $\tilde{M}_S$, $\tilde{W}_P$, and $\tilde{W}_T$ can be designed in a similar way as in Section 4.2, makes it possible to use more than one output together with one input. The last weight $W_{\mathbf{u}}$ can be chosen similar as in Section 4.2. The numerical values of the weights are

$$W_{\mathbf{u}} = 10^{-40/20}, \quad W = \begin{pmatrix} \frac{30s+75}{(s+5)^2} & \frac{0.1}{(s+5)^2} \end{pmatrix}, \tag{18a}$$

$$\tilde{M}_S = 10^{-2/20}, \quad \tilde{W}_P = \left(\frac{s+80}{s+0.15}\right)^3, \tag{18b}$$

and it turns out that W_T is not needed for the performance. Using these weights results in a controller of order 13.

4.3 Controller Characteristics

The resulting loop gains for the five controllers are shown in Figure 4. The four controllers using $\mathcal{H}_\infty$ methods do not give as high peaks as the PID-controller around 100 rad/s. It can also be seen that introducing $\ddot{\mathcal{P}}$ as a measurement eliminates the notch at 30 rad/s.

In Figure 4, the magnitudes of the five controllers are presented. The PID controller is smoother than the other controllers. The reason is that a part of the system dynamics is included in the $\mathcal{H}_\infty$ controllers. As a result, they try to remove the resonance peaks from the system, which can be seen in Figure 3, hence the peaks in the amplitude function of the $\mathcal{H}_\infty$ controllers. The weights $W_\mathbf{u}$ for the controllers $\mathcal{H}_\infty^m(q_m)$ and $\mathcal{H}_\infty^m(q_m, \ddot{\mathcal{P}})$ are different which can be seen in Figure 4 for high frequencies. Comparing the two controllers $\mathcal{H}_\infty^{ls}(q_m)$ and $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$ for high frequencies it can be seen that they behave similar. The PID-controller has the highest magnitude for high frequencies, which implies that the measurement noise will be amplified more than for the $\mathcal{H}_\infty$ controllers.

5 Simulation Results

The five controllers are evaluated using a simulation model. The simulation model consists of the flexible joint model described in Section 3, a measurement system, and a controller. The robot joint model is implemented in continuous time whereas the controllers operate in discrete time. The continuous-time controllers developed in Section 4, are therefore discretised using Tustin's formula. The measurements are affected by a time delay of one sample as well as zero mean normal distributed measurement noise. The sample time is $T_s = 0.5\,\text{ms}$.

The system is excited by a disturbance signal $\mathbf{w}$ containing steps and chirp signals on both the motor and end-effector. The performance is evaluated using a performance index, which is a weighted sum of peak-to-peak errors and settling times in the simulated end-effector position and the maximum torque and the torque noise in the simulated motor torque. The reader is referred to Moberg et al. [2009] for complete details about the disturbance signals and the performance index.

Figure 5 shows how the motor torque differs between the five controllers. In the upper diagram it can be see that $\mathcal{H}_\infty^{ls}(q_m)$ gives higher torques than the PID and the $\mathcal{H}_\infty^m(q_m)$ controllers. The PID gives higher torque noise during steady state due to the gain of the controller for high frequencies, recall Figure 4. In the lower diagram in Figure 5 it is shown that the controllers $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$ and $\mathcal{H}_\infty^m(q_m, \ddot{\mathcal{P}})$ give similar torque signals, and lower compared to the PID controller. A low torque signal is preferred to reduce the energy consumption and to decrease the wear in the motor and gear.

The end-effector position is presented in Figure 6. In the top graph it is seen that

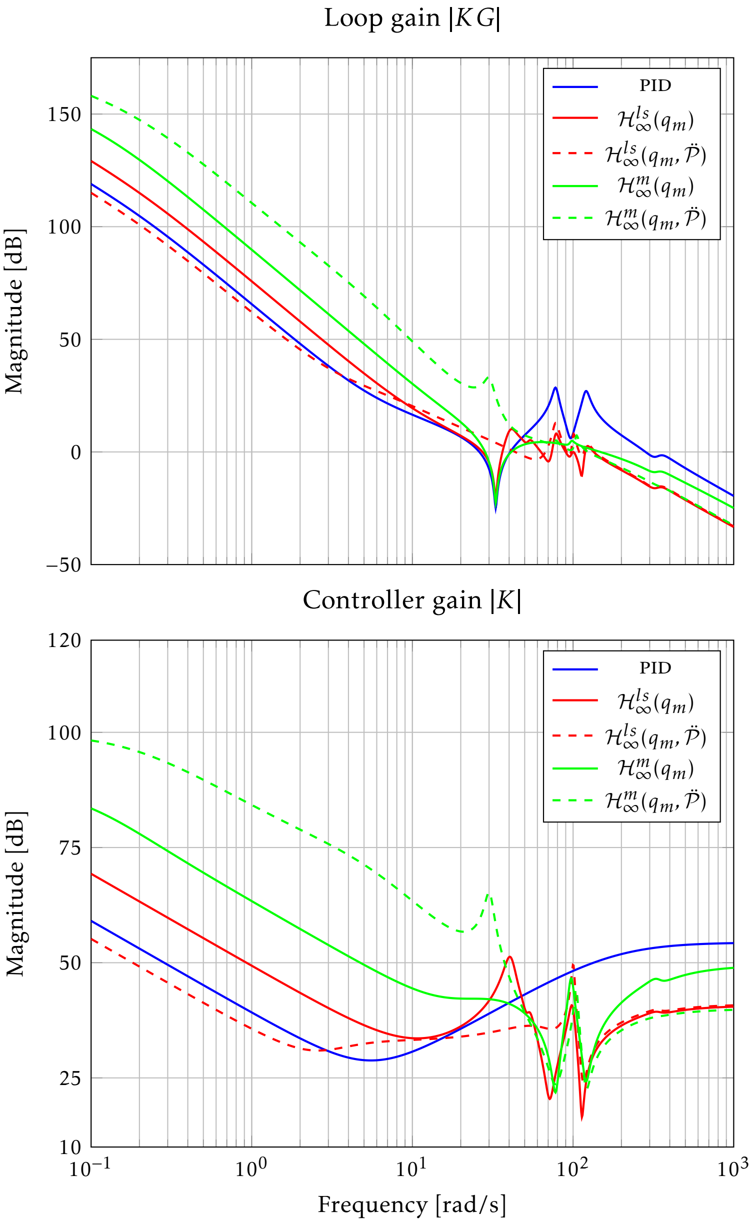


Figure 4: *Loop gain $|KG|$ and controller gain $|K|$ for the five controllers.*

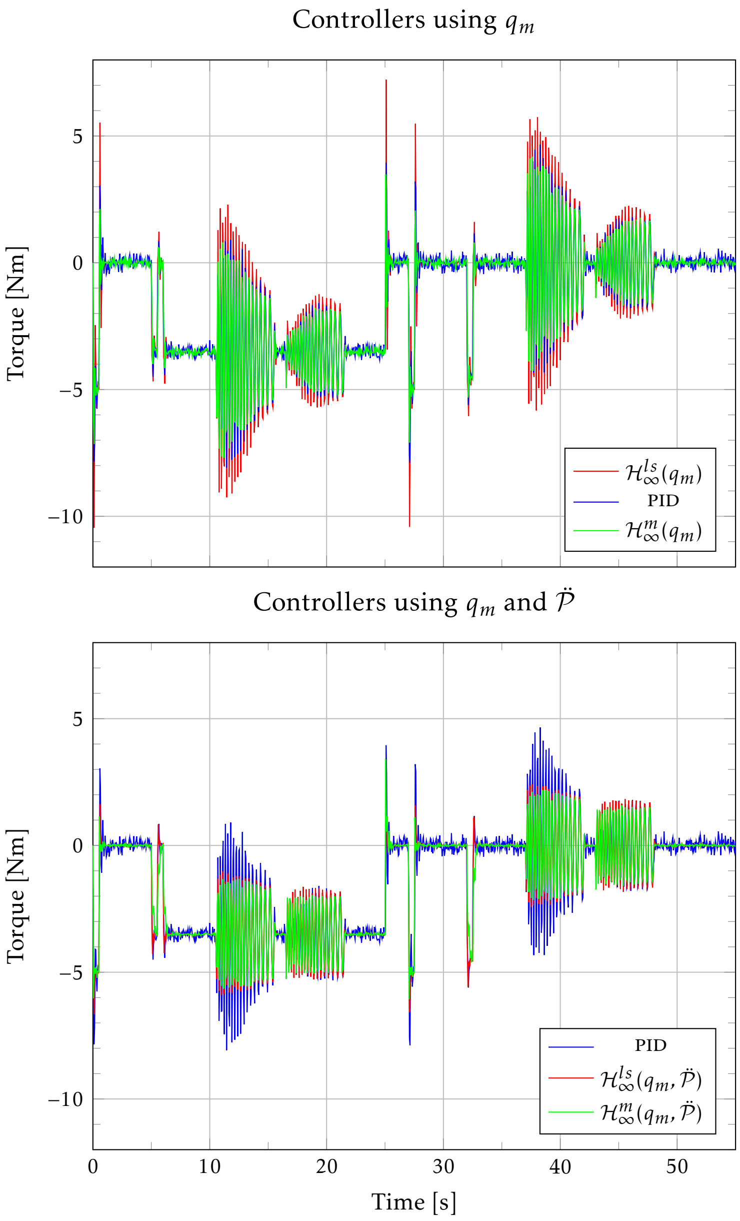


Figure 5: *Applied motor torque from the five controllers. The top graph shows the controllers using only q_m. The bottom graph shows the PID and the controllers using q_m and $\ddot{\mathcal{P}}$.*

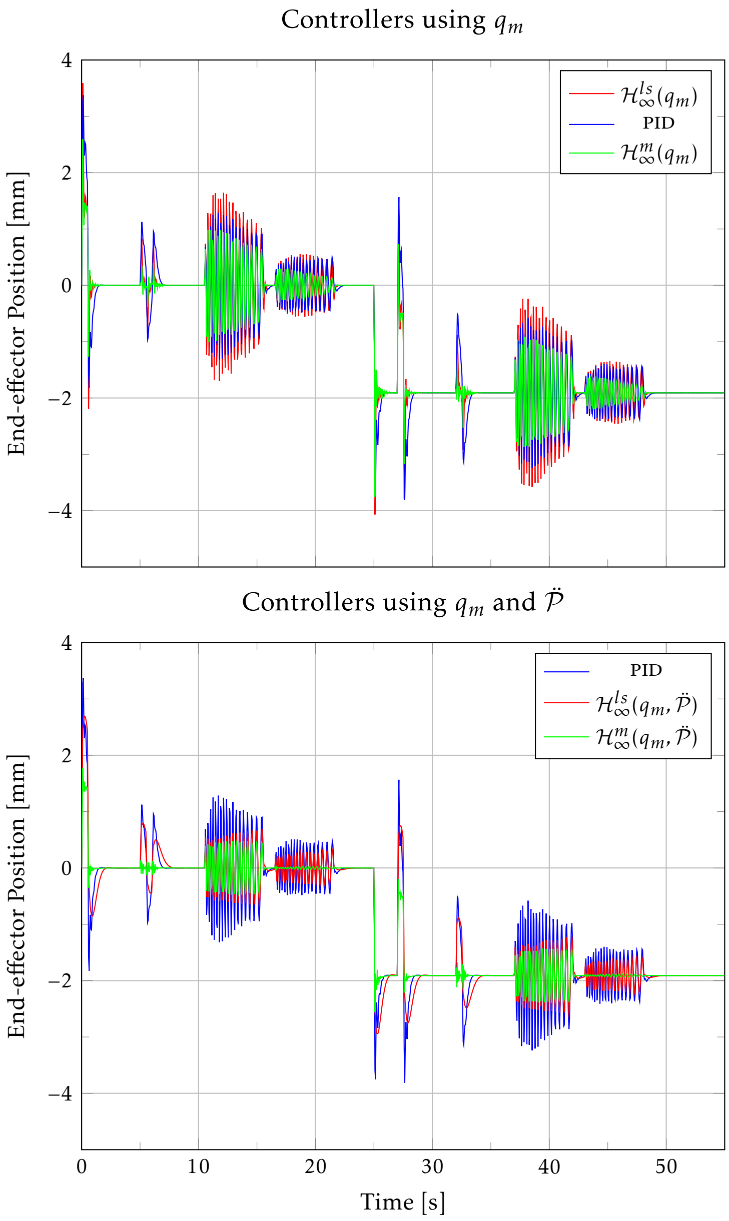


Figure 6: *Simulated end-effector position for the five controllers. The top graph shows the controllers using only q_m. The bottom graph shows the PID and the controllers using q_m and $\ddot{\mathcal{P}}$.*

Table 1: *Performance index for the five controllers, where lower value is better.*

PID	$\mathcal{H}_\infty^{ls}(q_m)$	$\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$	$\mathcal{H}_\infty^{m}(q_m)$	$\mathcal{H}_\infty^{m}(q_m, \ddot{\mathcal{P}})$
55.7	55.4	45.8	42.4	28.8

$\mathcal{H}_\infty^{ls}(q_m)$ gives, compared to the PID, higher oscillations during the time intervals 10-15 s and 37-42 s, which corresponds to a chirp disturbance at the end-effector. For step disturbances and chirp disturbances on the motor (time intervals 16-21 s and 43-58 s) $\mathcal{H}_\infty^{ls}(q_m)$ and the PID are more similar. The controller $\mathcal{H}_\infty^{m}(q_m)$ is better than the other two controllers in the simulation. The bottom graph shows that $\mathcal{H}_\infty^{m}(q_m, \ddot{\mathcal{P}})$ can handle the chirp disturbances on the motor (time intervals 16-21 s and 43-58 s) and step disturbances very good. For a chirp disturbance on the end-effector, the two $\mathcal{H}_\infty$ controllers give similar results. For step disturbances, the controller $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$ gives lower peaks than the PID controller, however the settling time is longer. The steady state error of approximately 2 mm after 25 s is a result of a constant torque disturbance on the end-effector. The size of the error will depend on the size of the disturbance and the stiffness of the joint. The motor position, which is measured, is controlled to zero for all five controllers.

The performance index for the five controllers is presented in Table 1. It shows, as discussed above, that $\mathcal{H}_\infty^{ls}(q_m)$ and the PID controller behave similar and that $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$ and $\mathcal{H}_\infty^{m}(q_m)$ give similar behaviour. The $\mathcal{H}_\infty^{m}(q_m, \ddot{\mathcal{P}})$-controller gives the best result.

6 Low Order Synthesis

For implementation of the controller in the robot control system it is important to have a low order controller. A controller in state space form requires $\mathcal{O}(n_x^2)$ operations to calculate the control signal, where n_x is the dimension of the state vector in the controller.

The low order controllers are here obtained using the MATLAB-function `hinfstruct`, which is included in Robust Control Toolbox and it is based on techniques from Apkarian and Noll [2006].

To find controllers with low orders using `hinfstruct` requires a model description, including the weights, in the form of (1). This structure is already used for the controllers $\mathcal{H}_\infty^{m}(q_m)$ and $\mathcal{H}_\infty^{m}(q_m, \ddot{\mathcal{P}})$, hence it is straightforward to synthesis low order controllers using the weights presented in Sections 4.2 and 4.2. For the loop shaping design method, the structure in Figure 1b can be rewritten in the form of (1) including the weights $W_1(s)$ and $W_2(s)$, explained in e.g. Zhou et al. [1996]. Using the rewritten structure, the low order controllers based on $\mathcal{H}_\infty^{ls}(q_m)$ and $\mathcal{H}_\infty^{ls}(q_m, \ddot{\mathcal{P}})$ can be obtained using the weights from Sections 4.2 and 4.2.

Table 2 shows the lowest order for the respective controllers, that can be achieved without changing the closed-loop performance too much. The table also shows

Table 2: Lowest order of the controllers obtained using `hinfstruct`*, with the order before reduction in brackets. The corresponding performance index from simulations is also shown.*

	$\mathcal{H}_\infty^{ls}(q_m)$	$\mathcal{H}_\infty^{ls}(q_m,\ddot{\mathcal{P}})$	$\mathcal{H}_\infty^{m}(q_m)$	$\mathcal{H}_\infty^{m}(q_m,\ddot{\mathcal{P}})$
Order	5 (13)	4 (13)	5 (10)	7 (13)
Perf. ind.	60.8	49.4	48.3	40.8

the performance index obtained when the controllers are used in the simulation environment. The orders can be reduced by a factor of two to three but the performance of the reduced order controllers is worse than the full order controllers. Since the controller based on loop shaping with only q_m as measurement has the same performance for the full order controller as the PID controller, the low order controller gives a worse performance than the PID controller. The other full order controllers are much better than the PID controller and afford to get a reduced performance for the low order controllers without getting worse than the PID controller.

Finally, note that the controllers only represents local minima solutions, hence rerunning `hinfstruct` with other initial values can give a better, or worse, controller. To handle this, several initial values have been used in `hinfstruct`.

7 Conclusions and Future Work

Four different $\mathcal{H}_\infty$ controllers for a flexible joint of an industrial manipulator are designed using mixed-$\mathcal{H}_\infty$ controller design and the loop shaping method. The model, on which the controllers are based, is a four-mass model. As input, the controllers use either the motor angle only or both the motor angle and the acceleration of the end-effector. Tuning of the controllers requires understanding of both the synthesis method and how the system behaves. For example, the measurements for the mixed-$\mathcal{H}_\infty$ controller are first pre-filtered to give an estimate of the tool position. The weighting functions for the resulting SISO system, from input torque to the estimated tool position, are then chosen similar to the case where only the motor position is used.

The controllers are compared to a PID controller and it is shown that if only the motor angle is measured it is much better to use the mixed-$\mathcal{H}_\infty$ design method compared to loop shaping. If instead the end-effector acceleration is added then the performance is improved significantly for both methods. The steady state error for the end-effector position is unaffected since the accelerometer does not provide low frequency measurements. Using a low order controller synthesis method, it is possible to reduce the order of the controllers by a factor of two to three but at the same time a decrease in the performance index of 10–30 % can be observed.

Investigation of robustness for stability with respect to model errors is one of several future directions of research. The mixed-$\mathcal{H}_\infty$ method has an advantage compared to the loop shaping method since a model of the error is possible to incorporate in the augmented plant $P(s)$.

Another continuation is to investigate the improvement for other types of sensors. One possibility is to have an encoder measuring the position directly after the gearbox, i.e., q_{a1}. This will improve the stiffness of the system, although it will not eliminate the stationary error for the end-effector position. The ultimate solution is to measure the end-effector position, but for practical reasons this is in general not possible, instead the end-effector position can be estimated, as described in Axelsson [2012]; Axelsson et al. [2012]; Chen and Tomizuka [2014], and used in the feedback loop.

Extending the system to several joints giving a non-linear model, which has to be linearised in several points, is also a future problem to investigate. A controller, using the results from this paper, is designed in each point and for example gain scheduling can be used when the robot moves between different points.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC and by the Excellence Center at Linköping-Lund in Information Technology, ELLIIT.

Bibliography

Pierre Apkarian and Dominikus Noll. Nonsmooth $\mathcal{H}_\infty$ synthesis. *IEEE Transactions on Automatic Control*, 51(1):71–86, January 2006.

Patrik Axelsson. Evaluation of six different sensor fusion methods for an industrial robot using experimental data. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 126–132, Dubrovnik, Croatia, September 2012.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012.

Patrik Axelsson, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-controller design methods applied to one joint of a flexible industrial manipulator. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014.

Wenjie Chen and Masayoshi Tomizuka. Direct joint space state estimation in robots with multiple elastic joints. *IEEE/ASME Transactions on Mechatronics*, 19(2):697–706, April 2014. DOI: 10.1109/TMECH.2013.2255308.

Bram de Jager. The use of acceleration measurements to improve the tracking control of robots. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 647–652, Le Touquet, France, October 1993.

Eric Dumetz, Jean-Yves Dieulot, Pierre-Jean Barre, Frédéric Colas, and Thomas Delplace. Control of an industrial robot using acceleration feedback. *Journal of Intelligent & Robotic Systems*, 46(2):111–128, 2006.

Keith Glover and Duncan McFarlane. Robust stabilization of normalized coprime factor plant descriptions with $\mathcal{H}_\infty$-bounded uncertainty. *IEEE Transactions on Automatic Control*, 34(8):821–830, August 1989.

K. Kosuge, M. Umetsu, and K. Furuta. Robust linearization and control of robot arm using acceleration feedback. In *Proceedings of the IEEE International Conference on Control and Applications*, pages 161–165, Jerusalem, Israel, April 1989.

Duncan McFarlane and Keith Glover. A loop shaping design procedure using $\mathcal{H}_\infty$ synthesis. *IEEE Transactions on Automatic Control*, 37(6):759–769, June 1992.

Stig Moberg, Jonas Öhr, and Svante Gunnarsson. A benchmark problem for robust feedback control of a flexible manipulator. *IEEE Transactions on Control Systems Technology*, 17(6):1398–1405, November 2009.

Goele Pipeleers and Jan Swevers. MATLAB-software `mixedHinfsyn`, 2013. Available at `http://set.kuleuven.be/optec/Software/mixedhinfsyn`.

Mark Readman and Pierre Bélanger. Acceleration feedback for flexible joint robots. In *Proceedings of the 30th IEEE Conference on Decision and Control*, pages 1385–1390, Brighton, England, December 1991.

Michael Ruderman and Torsten Bertram. Modeling and observation of hysteresis lost motion in elastic robot joints. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 13–18, Dubrovnik, Croatia, September 2012.

Hansjörg G. Sage, Michel F. de Mathelin, Gabriel Abba, Jacques A. Gangloff, and Eric Ostertag. Nonlinear optimization of robust $\mathcal{H}_\infty$ controllers for industrial robot manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2352–2357, Albuquerque, NM, USA, April 1997.

Hansjörg G. Sage, Michel F. de Mathelin, and Eric Ostertag. Robust control of robot manipulators: A survey. *International Journal of Control*, 72(16):1498–1522, 1999.

Sigurd Skogestad and Ian Postletwaite. *Multivariable Feedback Control, Analysis and Design*. John Wiley & Sons, Chichester, West Sussex, England, second edition, 2005.

Y. D. Song, A. T. Alouani, and J. N. Anderson. Robust path tracking control of industrial robots: An $\mathcal{H}_\infty$ approach. In *Proceedings of the IEEE Conference on Control Applications*, pages 25–30, Dayton, OH, USA, September 1992.

Wayne L. Stout and M. Edwin Sawan. Application of H-infinity theory to robot manipulator control. In *Proceedings of the IEEE Conference on Control Applications*, pages 148–153, Dayton, OH, USA, September 1992.

W. L. Xu and J. D. Han. Joint acceleration feedback control for robots: Analysis, sensing and experiments. *Robotics and Computer-Integrated Manufacturing*, 16(5):307–320, October 2000.

Keivan Zavari, Goele Pipeleers, and Jan Swevers. Multi-$\mathcal{H}_\infty$ controller design and illustration on an overhead crane. In *Proceedings of the IEEE Conference on Control Applications. Part of the IEEE Multi-Conference on Systems and Control*, pages 670–674, Dubrovnik, Croatia, October 2012.

Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall Inc., Upper Saddle River, NJ, USA, 1996.

Paper F

$\mathcal{H}_\infty$-Synthesis Method for Control of Non-linear Flexible Joint Models

Authors: Patrik Axelsson, Goele Pipeleers, Anders Helmersson and Mikael Norrlöf

Edited version of the paper:

Patrik Axelsson, Goele Pipeleers, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-synthesis method for control of non-linear flexible joint models. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014d.

$\mathcal{H}_\infty$-Synthesis Method for Control of Non-linear Flexible Joint Models

Patrik Axelsson*, Goele Pipeleers**, Anders Helmersson* and Mikael Norrlöf*

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
anders.helmersson@liu.se,
mikael.norrlof@liu.se

**Department of Mechanical
Engineering, Katholieke Universiteit
Leuven, Celestijnenlaan 300B, B-3001
Heverlee, Belgium
goele.pipeleers
@mech.kuleuven.be

Abstract

An $\mathcal{H}_\infty$-synthesis method for control of a flexible joint, with non-linear spring characteristic, is proposed. The first step of the synthesis method is to extend the joint model with an uncertainty description of the stiffness parameter. In the second step, a non-linear optimisation problem, based on nominal performance and robust stability requirements, has to be solved. Using the Lyapunov shaping paradigm and a change of variables, the non-linear optimisation problem can be rewritten as a convex, yet conservative, LMI problem. The method is motivated by the assumption that the joint operates in a specific stiffness region of the non-linear spring most of the time, hence the performance requirements are only valid in that region. However, the controller must stabilise the system in all stiffness regions. The method is validated in simulations on a non-linear flexible joint model originating from an industrial robot.

1 Introduction

The demand and the requirements for high precision control in electro mechanical systems have been increasing over time. At the same time cost reduction and more developed mechanical design criteria, with lower margins in the design, reduces the size of the components involved. One such example is the speed reducers used in many electro mechanical systems where the size and cost have become increasingly important. The harmonic drive, sometimes referred to as "strain-wave gearing", is a very common example of a gear type that can deliver high gear reduction ratio in a small device [Tuttle and Seering, 1996]. Characteristic to compact gear boxes, such as harmonic drives, are that they have a relatively small backlash, a highly non-linear friction behaviour, and in addition a very non-linear stiffness [Tjahjowidodo et al., 2006]. One typical application of electromechanical systems, where harmonic drive gearboxes are used, is in industrial robots where the motivation for the work presented in this paper also comes from. In this paper the control design for the electromechanical system, motor-gearbox joint, hereafter referred to as the *flexible joint system*, is considered. In general, robots are strongly coupled multivariate systems with non-linear dynamics and in previous research on control of robots linear spring stiffness has been considered, see e.g. Sage et al. [1999] and the references therein. When the speed reducers are of harmonic drive type, linear models are however not sufficient for the control as will be shown in the paper. Several non-linear models of the gearbox have been presented in the literature, see Tuttle and Seering [1996]; Tjahjowidodo et al. [2006]; Ruderman and Bertram [2012] among others.

What characterises the $\mathcal{H}_\infty$-controller synthesis method presented in this work is that it can facilitate in designing a controller which gives performance in one region of parameter values, while for another region the performance requirement is lower and only stability is sufficient. The proposed method is motivated by the fact that the flexible joint operates in specific regions most of the time. For example, a joint which is affected by gravity operates most of the time in the high stiffness region, hence it is more important to have a controller with good performance in the high stiffness region. However, the controller must stabilise the system in all stiffness regions.

This paper is organised as follows. Section 2 presents the problem and how it will be solved and the proposed method is outlined in Section 3. In Section 4, the non-linear joint model used to test the proposed method is presented. The design of the controller and the results are given in Sections 5 and 6. Finally, Section 7 concludes the paper.

2 Problem Formulation

The problem is to design a linear $\mathcal{H}_\infty$ controller that can stabilise a non-linear flexible joint model, for example a motor-harmonic drive-joint system, using only the primary position, the motor position q_m. There are a number of non-linearities that characterise the gearbox in the flexible joint. Here, the spring stiffness of the joint is considered and it is described by the function $\tau_s(\Delta_q)$, where $\Delta_q = q_m - q_a$ is the deflection between the motor position q_m and the secondary position, the arm position q_a. The non-linear spring is characterised by a low stiffness for small deflections and a high stiffness for large deflections, which is typical for compact gear boxes, such as harmonic drive [Tuttle and Seering, 1996; Ruderman and Bertram, 2012]. Linearising the stiffness function would give a linear expression $k \cdot (q_m - q_a)$, where the gain k depends on the deflection $q_m - q_a$ of the joint. The lowest and maximal values of k are k^{low} and k^{high}, respectively. The complete model and an explicit expression for $\tau_s(\Delta_q)$ are presented in Section 4.

Control of non-linear systems using linear $\mathcal{H}_\infty$ methods is usually done by first linearising the model in several operating points, e.g. gain scheduling techniques, or using exact linearisation. Gain scheduling requires to know the operating point of the spring and exact linearisation requires the full state vector, hence only the motor position is not enough to measure. A common solution is to introduce an observer for estimating the state vector. However, it is not certain that the estimated state vector is accurate enough to use due to model errors and disturbances. The estimation problem has been investigated in e.g. Axelsson et al. [2012]; Chen and Tomizuka [2014].

Instead, the problem considered in this paper is managed using an uncertainty description of the stiffness parameter k to obtain a controller over the whole interval for k. In general, the interval has to be relative short in order to obtain a controller using regular methods. The reason for this is that the methods try to be both robustly stable and have robust performance over the whole interval. The uncertainty description of the linearised spring stiffness can give a long interval of the parameter k that has to be covered. Instead of having a controller that satisfies the requirements of both robust stability and performance over the whole interval, the aim is to find a controller that is stable for all values of k but only satisfies the performance requirements in the high stiffness region. The reason for this is that in practice, the joint operates only in the high stiffness region most of the time, e.g. an industrial manipulator affected by the gravity force. In reality as low as zero stiffness must be handled due to backlash, but that is omitted here.

3 Proposed $\mathcal{H}_\infty$-Synthesis Method

This section presents the proposed $\mathcal{H}_\infty$-synthesis method. First, the uncertainty description is given. After that, the requirement for nominal stability and performance together with robust stability is discussed, and the final optimisation problem is presented.

3.1 Uncertainty Description

Let k be modelled as an uncertainty according to

$$k(\delta) = k + \bar{k}\delta, \tag{1a}$$

$$k = \alpha k^{\text{high}} + \beta k^{\text{low}}, \tag{1b}$$

$$\bar{k} = \alpha k^{\text{high}} - \beta k^{\text{low}}, \tag{1c}$$

where α, β are scaling parameters such that $\beta \leq \alpha$. The uncertain parameter δ is contained in $\boldsymbol{\delta} = \begin{bmatrix}-1 & 1\end{bmatrix} \subset \mathbb{R}$ and may change arbitrarily fast. For $\delta = \pm 1$ it holds that the stiffness parameter

$$k(\delta) \in \begin{bmatrix}2\beta k^{\text{low}} & 2\alpha k^{\text{high}}\end{bmatrix}. \tag{2}$$

Since the aim is to have a controller that has good performance in the high stiffness region, but only stable in the low stiffness region, it is desirably to have k close to k^{high} and the lower bound of $k(\delta)$ not larger than k^{low}.

The stiffness parameter enters only in the A-matrix of the linearised system and

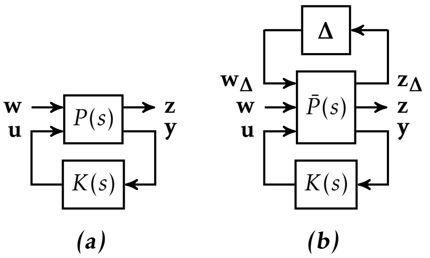


Figure 1: *Closed-loop system from* $\mathbf{w}$ *to* $\mathbf{z}$*, without and with an uncertainty description in (a) and (b) respectively.*

assume that the part containing δ is of rank one, then

$$\mathbf{A}(\delta) = \mathbf{A} + \mathbf{L}\delta\mathbf{R} \tag{3}$$

with $\mathbf{A} \in \mathbb{R}^{n_x \times n_x}$, $\mathbf{L} \in \mathbb{R}^{n_x, 1}$ and $\mathbf{R} \in \mathbb{R}^{1, n_x}$. For the forthcoming calculations, it is important to have $\mathbf{L}$ and $\mathbf{R}$ as a column and row matrix, respectively. The augmented system in Figure 1b can now be constructed according to

$$\bar{P} = \left(\begin{array}{c|ccc} \mathbf{A} & \mathbf{L} & \mathbf{B_w} & \mathbf{B_u} \\ \hline \mathbf{R} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{C_z} & \mathbf{0} & \mathbf{D_{zw}} & \mathbf{D_{zu}} \\ \mathbf{C_y} & \mathbf{0} & \mathbf{D_{yw}} & \mathbf{0} \end{array}\right), \tag{4}$$

with $\mathbf{\Delta} = \delta$.

3.2 Nominal Stability and Performance

Let P represent an LTI system, see Figure 1a,

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B_w}\mathbf{w} + \mathbf{B_u}\mathbf{u}, \tag{5a}$$
$$\mathbf{z} = \mathbf{C_z}\mathbf{x} + \mathbf{D_{zw}}\mathbf{w} + \mathbf{D_{zu}}\mathbf{u}, \tag{5b}$$
$$\mathbf{y} = \mathbf{C_y}\mathbf{x} + \mathbf{D_{yw}}\mathbf{w}, \tag{5c}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state vector, $\mathbf{w} \in \mathbb{R}^{n_w}$ is the disturbance vector, $\mathbf{u} \in \mathbb{R}^{n_u}$ is the control signal, $\mathbf{y} \in \mathbb{R}^{n_y}$ is the measurement signal, and $\mathbf{z} \in \mathbb{R}^{n_z}$ is the output signal that reflects our specifications. The matrices in (5) have dimensions corresponding to the vectors $\mathbf{x}$, $\mathbf{w}$, $\mathbf{u}$, $\mathbf{y}$, and $\mathbf{z}$. Note that it is the nominal system, i.e., $\delta = 0$ that is used here. Let the controller K in Figure 1a be given by

$$\dot{\mathbf{x}}_K = \mathbf{A}_K\mathbf{x}_K + \mathbf{B}_K\mathbf{y}, \tag{6a}$$
$$\mathbf{u} = \mathbf{C}_K\mathbf{x}_K + \mathbf{D}_K\mathbf{y}. \tag{6b}$$

Using the lower fractional transformation $F_l(P,K)$ gives the closed loop system from $\mathbf{w}$ to $\mathbf{z}$ according to

$$F_l(P,K) = \left(\begin{array}{c|c} \mathbf{A}_{CL} & \mathbf{B}_{CL} \\ \hline \mathbf{C}_{CL} & \mathbf{D}_{CL} \end{array}\right)$$
$$= \left(\begin{array}{cc|c} \mathbf{A} + \mathbf{B_u}\mathbf{D}_K\mathbf{C_y} & \mathbf{B_u}\mathbf{C}_K & \mathbf{B_w} + \mathbf{B_u}\mathbf{D}_K\mathbf{D_{yw}} \\ \mathbf{B}_K\mathbf{C_y} & \mathbf{A}_K & \mathbf{B}_K\mathbf{D_{yw}} \\ \hline \mathbf{C_z} + \mathbf{D_{zu}}\mathbf{D}_K\mathbf{C_y} & \mathbf{D_{zu}}\mathbf{C}_K & \mathbf{D_{zw}} + \mathbf{D_{zu}}\mathbf{D}_K\mathbf{D_{yw}} \end{array}\right) \tag{7}$$

From Gahinet and Apkarian [1994] it holds that the $\mathcal{H}_\infty$-norm of $F_l(P,K)$ is less than γ, i.e., $\|F_l(P,K)\|_\infty < \gamma$, and the closed loop system is stable, i.e., $\mathbf{A}_{CL}$ has all eigenvalues in the left half plane, if and only if there exists a positive definite matrix $\mathcal{P}$ such that

$$\begin{pmatrix} \mathbf{A}_{CL}^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A}_{CL} & \mathcal{P}\mathbf{B}_{CL} & \mathbf{C}_{CL}^\mathsf{T} \\ \mathbf{B}_{CL}^\mathsf{T}\mathcal{P} & -\gamma\mathbf{I} & \mathbf{D}_{CL}^\mathsf{T} \\ \mathbf{C}_{CL} & \mathbf{D}_{CL} & -\gamma\mathbf{I} \end{pmatrix} \prec 0. \tag{8}$$

3.3 Robust Stability

To guarantee robust stability of the uncertain system for arbitrarily fast changes in δ, quadratic stability is enforced which is given by

$$\exists \mathcal{P} \in \mathbb{S}^{n_x} : \mathcal{P} \succ 0 \text{ and} \tag{9a}$$
$$\mathbf{A}(\delta)^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A}(\delta) \prec 0,\ \forall \delta \in \boldsymbol{\delta}. \tag{9b}$$

From Scherer [2006], the robust LMI (9b) holds if and only if there exist $p, q \in \mathbb{R}$ with $p > 0$ such that

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A} & \mathbf{L} \end{pmatrix}^\mathsf{T} \begin{pmatrix} \mathbf{0} & \mathcal{P} \\ \mathcal{P} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A} & \mathbf{L} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{R} & \mathbf{0} \end{pmatrix}^\mathsf{T} \begin{pmatrix} -p & iq \\ -iq & p \end{pmatrix} \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{R} & \mathbf{0} \end{pmatrix} \prec 0 \tag{10}$$

Note that $p, q \in \mathbb{R}$ with $p > 0$ parametrise all multipliers that satisfy

$$\begin{pmatrix} \delta \\ 1 \end{pmatrix}^\mathsf{T} \begin{pmatrix} -p & iq \\ -iq & p \end{pmatrix} \begin{pmatrix} \delta \\ 1 \end{pmatrix} \geq 0,\ \forall \delta \in \boldsymbol{\delta}. \tag{11}$$

Since the negative definiteness of a Hermitian matrix implies that its real part is negative definite, $q = 0$ can be enforced in (10) without loss of generality. It follows from the fact that $\mathbf{L}$ and $\mathbf{R}$ are rank one matrices[1]. In addition, $p = 1$ can be enforced since the LMI is homogeneous in $\mathcal{P}$ and p. By elaborating the left-hand side of (10) gives that (9) is equivalent to

$$\exists \mathcal{P} \in \mathbb{S}^{n_x} :\ \mathcal{P} \succ 0 \text{ and } \begin{pmatrix} \mathbf{A}^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A} + \mathbf{R}^\mathsf{T}\mathbf{R} & \mathcal{P}\mathbf{L} \\ \mathbf{L}^\mathsf{T}\mathcal{P} & -1 \end{pmatrix} \prec 0. \tag{12}$$

[1] If $\operatorname{rank}\mathbf{L} = \operatorname{rank}\mathbf{R} = r$, than q should be a $r \times r$ skew symmetric matrix and the involved computations are much more complex.

The last LMI in (12) can be rewritten, similar to what is given by (8), using the Schur complement, according to

$$\begin{pmatrix} \mathbf{A}^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A} & \mathcal{P}\mathbf{L} & \mathbf{R}^\mathsf{T} \\ \mathbf{L}^\mathsf{T}\mathcal{P} & -1 & 0 \\ \mathbf{R} & 0 & -1 \end{pmatrix} \prec 0. \tag{13}$$

The LMI in (12) can also be obtained using IQC-based robust stability analysis with frequency independent multipliers [Megretski and Rantzer, 1997], which guarantees stability for arbitrarily fast changes in δ.

3.4 Controller Synthesis

The controller is now obtained from the following optimisation problem

$$\underset{A_K,B_K,C_K,D_K}{\text{minimise}} \quad \gamma \tag{14a}$$

$$\text{subject to} \quad \mathcal{P} \succ 0 \tag{14b}$$

$$\begin{pmatrix} \mathbf{A}_{CL}^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A}_{CL} & \mathcal{P}\mathbf{B}_{CL} & \mathbf{C}_{CL}^\mathsf{T} \\ \mathbf{B}_{CL}^\mathsf{T}\mathcal{P} & -\gamma\mathbf{I} & \mathbf{D}_{CL}^\mathsf{T} \\ \mathbf{C}_{CL} & \mathbf{D}_{CL} & -\gamma\mathbf{I} \end{pmatrix} \prec 0 \tag{14c}$$

$$\begin{pmatrix} \mathbf{A}_{CL}^\mathsf{T}\mathcal{P} + \mathcal{P}\mathbf{A}_{CL} & \mathcal{P}\mathbf{L}_{CL} & \mathbf{R}_{CL}^\mathsf{T} \\ \mathbf{L}_{CL}^\mathsf{T}\mathcal{P} & -1 & 0 \\ \mathbf{R}_{CL} & 0 & -1 \end{pmatrix} \prec 0 \tag{14d}$$

where $\mathbf{L}_{CL}$ and $\mathbf{R}_{CL}$ are the matrices $\mathbf{L}$ and $\mathbf{R}$ augmented with zeros in order for the dimensions to satisfy the closed-loop state vector $\mathbf{x}_{CL} = \begin{pmatrix} \mathbf{x}^\mathsf{T} & \mathbf{x}_K^\mathsf{T} \end{pmatrix}^\mathsf{T}$.

The minimisation problem in (14) gives a conservative solution because of the same Lyapunov matrix $\mathcal{P}$ is used in both (8) and (13). For the approach not to be conservative, different Lyapunov matrices should be used in (8) and (13). However, this multi-objective controller design is non-convex. To obtain a convex, yet conservative, approximation, the Lyapunov shaping paradigm, as introduced by Scherer et al. [1997], is used. Moreover, the minimisation problem in (14) is non-linear due to products of $\mathcal{P}$ and the controller parameters $\mathbf{A}_K$, $\mathbf{B}_K$, $\mathbf{C}_K$, and $\mathbf{D}_K$. However, a change of variables [Scherer et al., 1997] makes the constraints linear and the resulting minimisation problem can be solved using LMI optimisation, e.g. using Yalmip [Löfberg, 2004].

The optimisation problem (14) will be easier to solve the smaller the perturbation is. It can therefore be useful to introduce a scaling parameter $0 < \kappa \leq 1$ such that $\delta \in [-\kappa \quad \kappa]$. Decreasing $\boldsymbol{\delta}$ to $[-\kappa \quad \kappa]$ is equivalent to preserving $\boldsymbol{\delta} = [-1 \quad 1]$ but rescaling $\mathbf{L}$, according to $\mathbf{L} \to \kappa\mathbf{L}$. The good thing is that it can still be possible to stabilise the system for $\delta \in [-1 \quad 1]$ due to the conservatism of the proposed method. Note that κ is a tuning parameter that affect the solution to (14). A too large value can make the problem impossible to solve whereas a too small value gives a controller that is not able to stabilise the non-linear system.

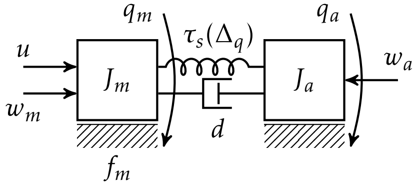


Figure 2: *A two-mass flexible joint model, where J_m is the motor and J_a the arm.*

4 Non-linear Flexible Joint Model

The flexible joint model considered in this paper is of two-mass model type, see Figure 2, where q_m is the motor position and q_a the arm position. Here, both q_m and q_a are described on the motor side of the gearbox. Input to the system is the motor torque u, the motor disturbance w_m and the arm disturbance w_a. The two masses are connected by a spring-damper pair, where the first mass corresponds to the motor and the second mass corresponds to the arm. The spring-damper pair is modelled by a linear damper, described by the parameter d, and the non-linear spring is described by the function $\tau_s(\Delta_q)$ which is a piecewise affine function with five segments, i.e.,

$$\tau_s(\Delta_q) = \begin{cases} k_{\text{low}}\Delta_q, & & & |\Delta_q| \le \frac{\Psi}{4} \\ (k_{\text{low}} + m_k)\,\Delta_q & -\text{sign}(\Delta_q)\frac{m_k\Psi}{4}, & \frac{\Psi}{4} < & |\Delta_q| \le \frac{\Psi}{2} \\ (k_{\text{low}} + 2m_k)\,\Delta_q & -\text{sign}(\Delta_q)\frac{3m_k\Psi}{4}, & \frac{\Psi}{2} < & |\Delta_q| \le \frac{3\Psi}{4} \\ (k_{\text{low}} + 3m_k)\,\Delta_q & -\text{sign}(\Delta_q)\frac{3m_k\Psi}{2}, & \frac{3\Psi}{4} < & |\Delta_q| \le \Psi \\ k_{\text{high}}\Delta_q & -\text{sign}(\Delta_q)\frac{5m_k\Psi}{2}, & \Psi < & |\Delta_q| \end{cases}$$

where $m_k = (k_{\text{high}} - k_{\text{low}})/4$, and Ψ a model parameter describing the transition to the high stiffness region. Moreover, the friction torque is assumed to be linear, described by the parameter f_m, and the input torque u is limited to ±20 Nm. The measurement of the system is the motor position q_m, and q_a is the variable that is to be controlled. The model is a simplification of the experimental results achieved in, e.g. Tjahjowidodo et al. [2006], where the non-linear torsional stiffness also shows hysteresis behaviour.

The dynamical model of the flexible joint is given by

$$J_a\ddot{q}_a - \tau_s(\Delta_q) - d(\dot{q}_m - \dot{q}_a) = w_a, \tag{15a}$$

$$J_m\ddot{q}_m + \tau_s(\Delta_q) + d(\dot{q}_m - \dot{q}_a) + f_m\dot{q}_m = u + w_m, \tag{15b}$$

where the model parameters are presented in Table 1. Using a state vector x according to

$$\mathbf{x} = \begin{pmatrix} q_a & q_m & \dot{q}_a & \dot{q}_m \end{pmatrix}^{\mathsf{T}}, \tag{16}$$

***Table 1:** Numerical values of the model parameters.*

J_a	J_m	Ψ	k_{high}	k_{low}	d	f_m
0.042	0.005	$220\pi/60/180$	100	100/6	0.08	0.006

gives the non-linear state space model

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{q}_a \\ \dot{q}_m \\ \frac{1}{J_a}\left(\tau_s(\Delta_q) + d(\dot{q}_m - \dot{q}_a) + w_a\right) \\ \frac{1}{J_m}\left(u - \tau_s(\Delta_q) - d(\dot{q}_m - \dot{q}_a) - f_m\dot{q}_m + w_m\right) \end{pmatrix} \tag{17}$$

Linearising the non-linear flexible joint model (17) gives a linear state space model $\dot{\widetilde{\mathbf{x}}} = \widetilde{\mathbf{A}}\widetilde{\mathbf{x}} + \widetilde{\mathbf{B}}_\mathbf{u}\mathbf{u} + \widetilde{\mathbf{B}}_\mathbf{w}\mathbf{w}$, where $\mathbf{w} = \begin{pmatrix} w_a & w_m \end{pmatrix}^\mathsf{T}$ and

$$\widetilde{\mathbf{A}} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k}{J_a} & \frac{k}{J_a} & -\frac{d}{J_a} & \frac{d}{J_a} \\ \frac{k}{J_m} & -\frac{k}{J_m} & \frac{d}{J_m} & -\frac{d+f_m}{J_m} \end{pmatrix}, \tag{18a}$$

$$\widetilde{\mathbf{B}}_\mathbf{u} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{J_m} \end{pmatrix}, \quad \widetilde{\mathbf{B}}_\mathbf{w} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{J_a} & 0 \\ 0 & \frac{1}{J_m} \end{pmatrix} \tag{18b}$$

$$\widetilde{\mathbf{C}} = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}. \tag{18c}$$

Here, k is the stiffness parameter given by (1b). The uncertainty description (1) gives

$$\widetilde{\mathbf{A}}(\delta) = \widetilde{\mathbf{A}} + \widetilde{\mathbf{L}}\delta\widetilde{\mathbf{R}}, \tag{19a}$$

$$\widetilde{\mathbf{L}} = \begin{pmatrix} 0 & 0 & \frac{\bar{k}}{J_a} & -\frac{\bar{k}}{J_m} \end{pmatrix}^\mathsf{T}, \tag{19b}$$

$$\widetilde{\mathbf{R}} = \begin{pmatrix} -1 & 1 & 0 & 0 \end{pmatrix}. \tag{19c}$$

The notation $\widetilde{\ }$ indicates that the weighting functions in the system $P(s)$ are not included here. Section 5 presents the weighting functions and how they are included in the state space model to give the system $P(s)$ in (5).

5 Controller Design

A common design method is to construct the system P in (5) by augmenting the original system $\mathbf{y} = G(s)\mathbf{u}$ with the weights $W_\mathbf{u}(s)$, $W_S(s)$, and $W_T(s)$, such that the system $\mathbf{z} = F_l(P, K)\mathbf{w}$, depicted in Figure 1a, can be written as

$$F_l(P, K) = \begin{pmatrix} W_{\mathbf{u}}(s)G_{\mathbf{wu}}(s) \\ -W_T(s)T(s) \\ W_S(s)S(s) \end{pmatrix}, \tag{20}$$

where $S(s) = (I + G(s)K(s))^{-1}$ is the sensitivity function, $T(s) = I - S(s)$ is the complementary sensitivity function, and $G_{\mathbf{wu}}(s) = -K(s)(I + G(s)K(s))^{-1}$ is the transfer function from $\mathbf{w}$ to $\mathbf{u}$. The $\mathcal{H}_\infty$ controller is obtained by minimising the $\mathcal{H}_\infty$-norm of the system $F_l(P, K)$, i.e., minimise γ such that $\|F_l(P, K)\|_\infty < \gamma$. Using (20) gives

$$\bar{\sigma}(W_{\mathbf{u}}(i\omega)G_{\mathbf{wu}}(i\omega)) < \gamma, \ \forall\omega, \tag{21a}$$

$$\bar{\sigma}(W_T(i\omega)T(i\omega)) < \gamma, \ \forall\omega, \tag{21b}$$

$$\bar{\sigma}(W_S(i\omega)S(i\omega)) < \gamma, \ \forall\omega. \tag{21c}$$

The transfer functions $G_{\mathbf{wu}}(s)$, $S(s)$, and $T(s)$ can now be shaped to satisfy the requirements by choosing the weights $W_{\mathbf{u}}(s)$, $W_S(s)$, and $W_T(s)$. In general this is a quite difficult task. See e.g. Skogestad and Postletwaite [2005]; Zhou et al. [1996] for details.

The mixed-$\mathcal{H}_\infty$ controller design [Pipeleers and Swevers, 2013; Zavari et al., 2012] is a modification of the standard $\mathcal{H}_\infty$-design method. Instead of choosing the weights in (20) such that the norm of all weighted transfer functions satisfies (21), the modified method divides the problem into design constraints and design objectives. The controller can now be found as the solution to

$$\underset{K(s)}{\text{minimise}} \quad \gamma \tag{22a}$$

$$\text{subject to} \quad \|W_P S\|_\infty < \gamma \tag{22b}$$

$$\|M_S S\|_\infty < 1 \tag{22c}$$

$$\|W_{\mathbf{u}} G_{\mathbf{wu}}\|_\infty < 1 \tag{22d}$$

$$\|W_T T\|_\infty < 1 \tag{22e}$$

where γ not necessarily has to be close to 1. The LMI in (8) can be modified to fit into the optimisation problem (22), see Zavari et al. [2012].

The weight M_S should limit the peak of S and is therefore chosen to be a constant. The peak of $G_{\mathbf{wu}}$ is also important to reduce in order to keep the control signal bounded, especially for high frequencies. A constant value of $W_{\mathbf{u}}$ is therefore also chosen. Finally, the weight W_T is also chosen to be a constant for simplicity.

The system from $\mathbf{w}$ to the output includes an integrator, hence it is necessary to have at least two integrators in the open loop GK in order to attenuate piecewise constant disturbances. The system G has one integrator hence at least one integrator must be included in the controller. Including an integrator in the controller is the same as letting $|S(i\omega)| \to 0$, for $\omega \to 0$. Forcing S to 0 is the same as letting W_P approach ∞ when $\omega \to 0$. However, it is not possible to force pure integrators in the design since the generalised plant $P(s)$ is not possible to stabilise with

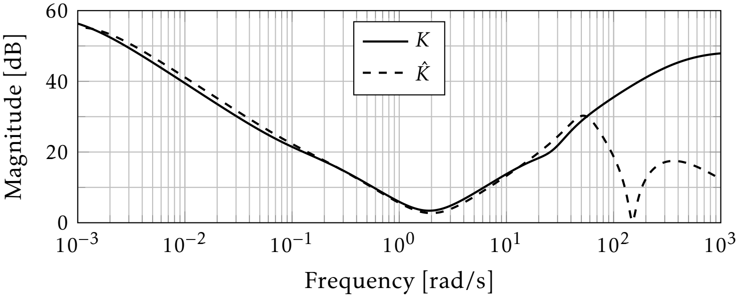


Figure 3: *Controller gains $|K|$ and $|\hat{K}|$.*

marginally stable weights. Instead the pole is placed in the left half plane close to the origin. Zeros must be included in the design as well to get a proper inverse. The following weights have been proven to work

$$W_{\mathbf{u}} = 10^{-50/20}, \quad W_T = 10^{-8/20}, \tag{23a}$$

$$M_S = 10^{-8/20}, \quad W_P = \frac{(s+50)(s+15)(s+5)}{500(s+0.2)(s+0.001)^2}. \tag{23b}$$

The constant weights in the form $10^{-\lambda/20}$ can be interpret as a maximum value, for the corresponding transfer function, of λ dB.

The augmented system P is obtained using the command `augw(G, [Wp;MS], WU,WT)` in MATLAB, where G is the system described by $\widetilde{\mathbf{A}}$, $\widetilde{\mathbf{B}}_{\mathbf{u}}$, and $\widetilde{\mathbf{C}}$ in (18). The uncertainty description of $k(\delta)$ in (1) is used with $\alpha = 0.9167$, and $\beta = 0.5$, i.e., the nominal value is $k = k^{\text{high}}$, $\bar{k} = 83.33$, and $k(\delta) \in [16.67 \quad 183.33]$. The scaling parameter is chosen as $\kappa = 0.75$. Finally, the uncertainty model is updated according to

$$\mathbf{L} = \begin{pmatrix} \mathbf{0} & \widetilde{\mathbf{L}}^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}}, \quad \mathbf{R} = \begin{pmatrix} \mathbf{0} & \widetilde{\mathbf{R}} \end{pmatrix} \tag{24}$$

where $\mathbf{0}$ is a zero matrix with suitable dimensions.

6 Results

The optimisation problem in (14) is solved using Yalmip [Löfberg, 2004] and a controller K of order $n_K = 6$ is obtained. A controller $\hat{K}$, with the same weights as for the robust stabilising controller K, using the optimisation problem in (14) without the LMI (14d) is derived to show the importance of the extra LMI for robust stability and also what is lost in terms of performance. The controller gains $|K|$ and $|\hat{K}|$ are shown in Figure 3, and they have a constant gain before 10^{-3} rad/s due to the pole at $s = -0.001$. The major difference is the notch for $\hat{K}$ around 100 rad/s and the high gain for K for high frequencies.

The robust stability can be analysed using the *structured singular value* (SSV).

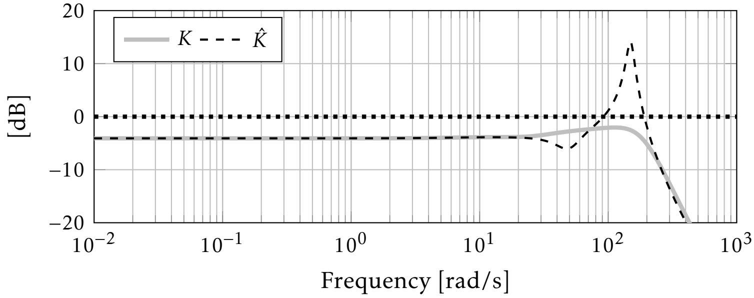


Figure 4: *Structured singular values for robust stability using the controllers K and $\hat{K}$.*

The system is robustly stable if the SSV for the closed loop system from $\mathbf{w}_\Delta$ to $\mathbf{z}_\Delta$ with respect to the uncertainty $\boldsymbol{\Delta}$ is less than one for all frequencies. A thorough description of the SSV can be found in Skogestad and Postletwaite [2005]. Figure 4 shows the SSV for the closed loop system using K and $\hat{K}$ and it can be seen that the SSV using K is less than one (0 dB) for all frequencies whereas the SSV using $\hat{K}$ has a peak of approximately 15 dB. As a result $\hat{K}$ cannot stabilise the system for all perturbations, as expected.

The step responses for the controller $\hat{K}$ using the linearised system in $k = k^{\text{high}}$ and the controller K using the linearised systems in $k = k^{\text{high}}$ and $k = k^{\text{low}}$ are shown in Figure 5. It can be seen that $\hat{K}$ is better than K for the linearised system in the high stiffness region. It means that in order to get a controller that is robustly stable for $k(\delta)$, the performance has been impaired. It can also be seen that the performance for K is better in the high stiffness region than in the low stiffness region, since the nominal value $k = k^{\text{high}}$.

The SSV can also be used for analysing nominal performance and robust performance, see Skogestad and Postletwaite [2005]. The requirement is that the SSV should be less than one for different systems with respect to some perturbations. Figure 6 shows the SSV for robust stability, nominal performance, and robust performance using K. It can be seen that the requirements for robust stability and nominal performance are satisfied. However, the requirement for robust performance is not satisfied. The reason is that the optimisation problem (14) does not take robust performance into account.

Finally, simulation of the non-linear model using K is performed to show that the controller can handle dynamic changes in the stiffness parameter and not only stabilising the system for fixed values of the parameter. The non-linear model is simulated in Simulink using the disturbance signal in Figure 7a, which is composed by steps and chirp signals. Figures 7b and 7c show the arm angle $q_a(t)$ on the arm side of the gearbox and the motor torque $\tau(t)$. From 0 s up to 25 s the system operates in the region $0 < \left|\Delta_q\right| < \Psi$ most of the time except for short periods of time when the step disturbances occur. The arm disturbance after 25 s keeps

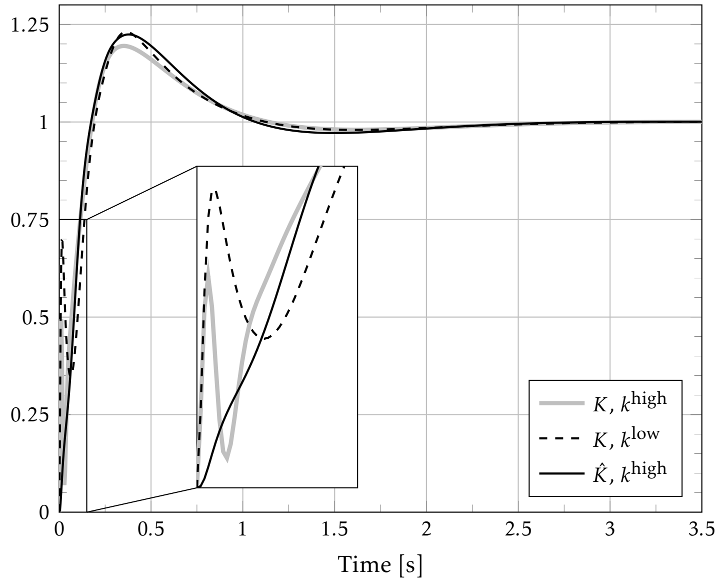


Figure 5: *Step response of the controllers K and $\hat{K}$ using system linearised in k^{low} and k^{high}. The first 0.15 s are magnified to show the initial transients.*

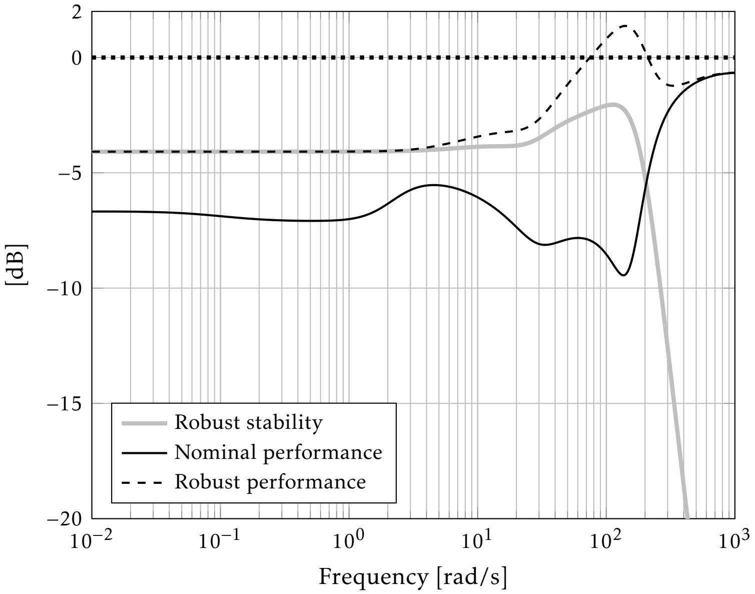
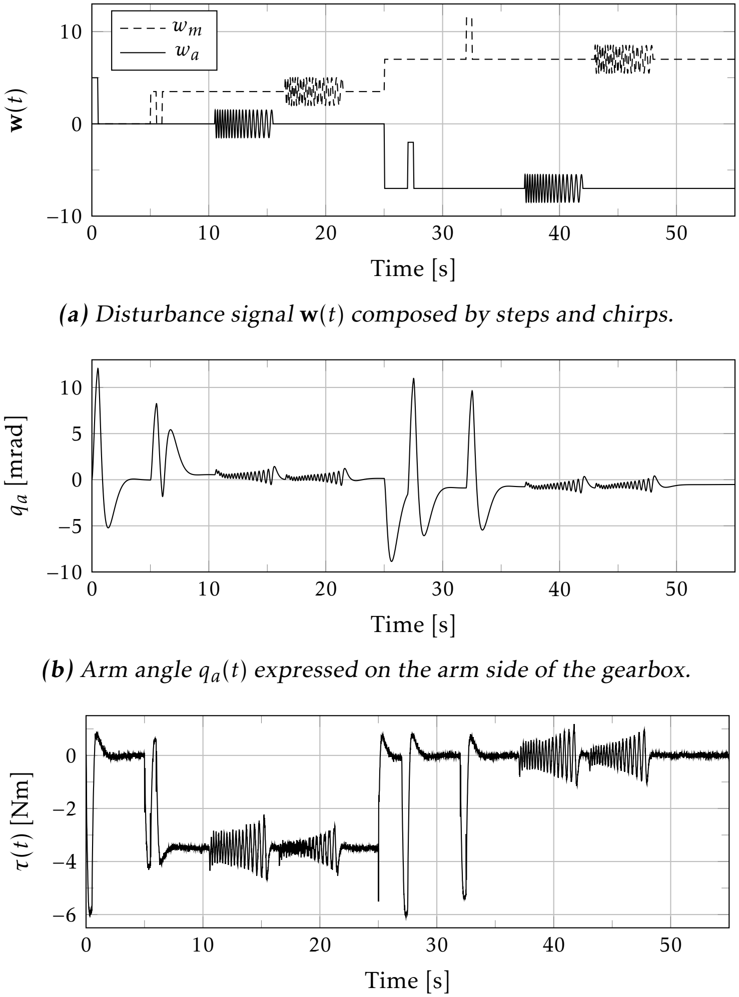


Figure 6: *Structured singular values for robust stability, nominal performance, and robust performance for the controller K.*



***(a)** Disturbance signal $\mathbf{w}(t)$ composed by steps and chirps.*

***(b)** Arm angle $q_a(t)$ expressed on the arm side of the gearbox.*

***(c)** Motor torque $\tau(t)$.*

***Figure 7:** Disturbance signal for the simulation experiment and the obtained arm angle and motor torque.*

the system in the high stiffness region except for a few seconds in connection with the step disturbances.

The stationary error for $q_a(t)$ at the end is due to the fact that the controller only uses $q_m(t)$ and the constant $w_a(t)$ is not observable in $q_m(t)$ hence $q_a(t)$ cannot be controlled to zero. The primary position $q_m(t)$ does not have any stationary error.

7 Conclusions

A method to synthesise and design $\mathcal{H}_\infty$ controllers for flexible joints, with non-linear spring characteristics, is presented. The non-linear model is linearised in a specific operating point, where the performance requirements should be full filled. Moreover, an uncertainty description of the stiffness parameter is utilised to get robust stability for the non-linear system in all operating points. The resulting non-linear and non-convex optimisation problem can be rewritten as a convex, yet conservative, LMI problem using the Lyapunov shaping paradigm and a change of variables, and efficient solutions can be obtained using standard solvers.

Using the proposed synthesis method an $\mathcal{H}_\infty$ controller is computed for a specific model, where good performance can be achieved for high stiffness values while stability is achieved in the complete range of the stiffness parameter. A controller derived with the same performance requirement but without the additional stability constraint is included for comparison. By analysing the structured singular values for robust stability for the two controllers it becomes clear that the controller without the extra stability constraint will not be stable for the parameter variations introduced by the non-linear stiffness parameter.

In the synthesis method it is assumed that the parameter δ changes arbitrarily fast, which is a conservative assumption for real systems. It would therefore be a good idea to have a limited change in δ in the future development of the method. Moreover, the use of a common Lyapunov matrix $\mathcal{P}$ must be relaxed to reduce the conservatism further. Here, the path following method from Hassibi et al. [1999]; Ostertag [2008] can be further investigated.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC and by the Excellence Center at Linköping-Lund in Information Technology, ELLIIT.

Bibliography

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Bayesian state estimation of a flexible industrial robot. *Control Engineering Practice*, 20(11):1220–1228, November 2012.

Patrik Axelsson, Goele Pipeleers, Anders Helmersson, and Mikael Norrlöf. $\mathcal{H}_\infty$-synthesis method for control of non-linear flexible joint models. *Accepted to the 19th IFAC World Congress, Cape Town, South Africa*, 2014.

Wenjie Chen and Masayoshi Tomizuka. Direct joint space state estimation in robots with multiple elastic joints. *IEEE/ASME Transactions on Mechatronics*, 19(2):697–706, April 2014. DOI: 10.1109/TMECH.2013.2255308.

Pascal Gahinet and Pierre Apkarian. A linear matrix inequality approach to $\mathcal{H}_\infty$ control. *International Journal of Robust and Nonlinear Control*, 4(1):421–448, 1994.

Arash Hassibi, Jonathan How, and Stephen Boyd. A path-following method for solving BMI problems in control. In *Proceedings of the American Control Conference*, pages 1385–1389, San Diego, CA, USA, June 1999.

Johan Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the IEEE International Symposium on Computer Aided Control Systems Design*, pages 248–289, Taipei, Taiwan, September 2004. URL `http://users.isy.liu.se/johanl/yalmip`.

Alexandre Megretski and Anders Rantzer. System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, 42(6):819–830, June 1997.

Eric Ostertag. An improved path-following method for mixed $\mathcal{H}_2/\mathcal{H}_\infty$ controller design. *IEEE Transactions on Automatic Control*, 53(8):1967–1971, September 2008.

Goele Pipeleers and Jan Swevers. MATLAB-software `mixedHinfsyn`, 2013. Available at `http://set.kuleuven.be/optec/Software/mixedhinfsyn`.

Michael Ruderman and Torsten Bertram. Modeling and observation of hysteresis lost motion in elastic robot joints. In *Proceedings of the 10th International IFAC Symposium on Robot Control*, pages 13–18, Dubrovnik, Croatia, September 2012.

Hansjörg G. Sage, Michel F. de Mathelin, and Eric Ostertag. Robust control of robot manipulators: A survey. *International Journal of Control*, 72(16):1498–1522, 1999.

Carsten Scherer. LMI relaxations in robust control. *European Journal of Control*, 12(1):3–29, 2006.

Carsten Scherer, Pascal Gahinet, and Mahmoud Chilali. Multiobjective output-feedback control via LMI optimization. *IEEE Transactions on Automatic Control*, 42(7):896–911, July 1997.

Sigurd Skogestad and Ian Postletwaite. *Multivariable Feedback Control, Analysis and Design*. John Wiley & Sons, Chichester, West Sussex, England, second edition, 2005.

Tegoeh Tjahjowidodo, Farid Al-Bender, and Hendrik Van Brussel. Nonlinear modelling and identification of torsional behaviour in harmonic drives. In *Proceedings of the International Conference on Noise and Vibration Engineering*, pages 2785–2796, Leuven, Belgium, September 2006.

Timothy D. Tuttle and Warren P. Seering. A nonlinear model of a harmonic drive gear transmission. *IEEE Transactions on Robotics and Automation*, 12(3):368–374, June 1996.

Keivan Zavari, Goele Pipeleers, and Jan Swevers. Multi-$\mathcal{H}_\infty$ controller design and illustration on an overhead crane. In *Proceedings of the IEEE Conference on Control Applications. Part of the IEEE Multi-Conference on Systems and Control*, pages 670–674, Dubrovnik, Croatia, October 2012.

Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall Inc., Upper Saddle River, NJ, USA, 1996.

Paper G

Estimation-based Norm-optimal Iterative Learning Control

Authors: Patrik Axelsson, Rickard Karlsson and Mikael Norrlöf

Edited version of the paper:

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based norm-optimal iterative learning control. *Submitted to Systems & Control Letters*, 2014c.

Estimation-based Norm-optimal Iterative Learning Control

Patrik Axelsson*, Rickard Karlsson** and Mikael Norrlöf*

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
mikael.norrlof@liu.se

**Nira Dynamics
Teknikringen 6
SE-583 30 Linköping, Sweden
rickard.karlsson
@niradynamics.se

Abstract

In this paper the norm-optimal *iterative learning control* (ILC) algorithm for linear systems is extended to an estimation-based norm-optimal ILC algorithm where the controlled variables are not directly available as measurements. The objective function in the optimisation problem is modified to incorporate not only the mean value of the estimated variable, but also information about the uncertainty of the estimate. It is further shown that if a stationary Kalman filter is used for linear time-invariant systems the ILC design is independent of the estimation method. Finally, the concept is extended to nonlinear state space models using linearisation techniques, where it is assumed that the full state vector is estimated and used in the ILC algorithm. Stability and convergence properties are also derived.

1 Introduction

The *iterative learning control* (ILC) method [Arimoto et al., 1984; Moore, 1993] improves performance, for instance trajectory tracking accuracy, for systems that repeat the same task several times. Traditionally, a successful solution to such problems base the ILC control law on direct measurements of the control quantity. When this quantity is not directly available as a measurement, the controller must rely on measurements that indirectly relate to the control quantity, or the control quantity has to be estimated from other measurements.

In Ahn et al. [2006]; Lee and Lee [1998] a state space model in the iteration domain is formulated for the error signal, and a KF is used for estimation. The difference to this paper is that in Ahn et al. [2006]; Lee and Lee [1998] it is assumed that the control error is measured directly, hence the KF is merely a low-pass filter, with smoothing properties, for reducing the measurement noise. In Wallén et al. [2009] it is shown that the performance of an industrial robot is significantly increased if an estimate of the control quantity is used instead of measurements of a related quantity. Still, the concept of ILC in combination with estimation of the control quantity, has not been given much attention in the literature.

In this paper the estimation-based ILC framework, where the control quantity has to be estimated, is combined with an ILC design based on an optimisation approach, referred to as norm-optimal ILC [Amann et al., 1996]. The estimation problem is here formulated using recursive Bayesian methods. Extensions to non-linear systems, utilising linearisation techniques, are also presented. The contributions are summarised as

1. Extension of the objective function to include the full *probability density function* (PDF) of the estimated control quantity, utilising the Kullback-Leibler divergence.
2. A separation lemma, stating that the extra dynamics introduced by the stationary KF is not necessary to include in the design procedure of the ILC algorithm.
3. Extensions to non-linear systems, including stability and convergence properties as well as additional appropriate approximations to simplify the ILC algorithm.

2 Iterative Learning Control (ILC)

The ILC-method improves the performance of systems that repeat the same task multiple times. The systems can be open loop as well as closed loop, with internal feedback. The ILC control signal $\mathbf{u}_{k+1}(t) \in \mathbb{R}^{n_u}$ for the next iteration $k+1$ at discrete time t is calculated using the error signal and the ILC control signal, both from the current iteration k. Different types of update laws can be found in e.g. Moore [1993].

One design method is the *norm-optimal ILC* algorithm [Amann et al., 1996; Gunnarsson and Norrlöf, 2001]. The method solves

$$\begin{aligned} \underset{\mathbf{u}_{k+1}(\cdot)}{\text{minimise}} \quad & \sum_{t=0}^{N-1} \|\mathbf{e}_{k+1}(t)\|^2_{\mathbf{W_e}} + \|\mathbf{u}_{k+1}(t)\|^2_{\mathbf{W_u}} \\ \text{subject to} \quad & \sum_{t=0}^{N-1} \|\mathbf{u}_{k+1}(t) - \mathbf{u}_k(t)\|^2 \leq \delta, \end{aligned} \tag{1}$$

where $\mathbf{e}_{k+1}(t) = \mathbf{r}(t) - \mathbf{z}_{k+1}(t)$ is the error, $\mathbf{r}(t)$ the reference signal, and $\mathbf{z}_{k+1}(t)$ the variable to be controlled. The matrices $\mathbf{W}_e \in \mathbb{S}^{n_z}_{++}$, and $\mathbf{W}_u \in \mathbb{S}^{n_u}_{++}$ are weight matrices, used as design parameters, for the error and the ILC control signal, respectively[1].

Using a Lagrange multiplier and a batch formulation (see Appendix A) of the

[1] $\mathbb{S}^n_{++}$ denotes a $n \times n$ positive definite matrix.

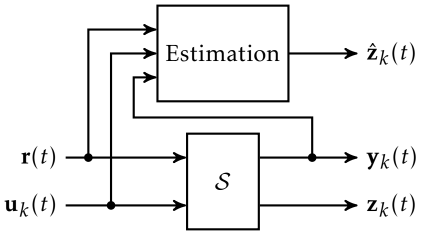


Figure 1: *System $\mathcal{S}$ with reference $\mathbf{r}(t)$, ILC input $\mathbf{u}_k(t)$, measured variable $\mathbf{y}_k(t)$ and controlled variable $\mathbf{z}_k(t)$ at ILC iteration k and time t.*

system from $\mathbf{u}_{k+1}(t)$ and $\mathbf{r}(t)$ to $\mathbf{z}_{k+1}(t)$ gives the solution

$$\overline{\mathbf{u}}_{k+1} = \mathcal{Q} \cdot (\overline{\mathbf{u}}_k + \mathcal{L} \cdot \overline{\mathbf{e}}_k) \tag{2a}$$

$$\mathcal{Q} = (\mathbf{S}_{\mathbf{zu}}^{\mathsf{T}} \mathcal{W}_{\mathbf{e}} \mathbf{S}_{\mathbf{zu}} + \mathcal{W}_{\mathbf{u}} + \lambda \mathbf{I})^{-1} (\lambda \mathbf{I} + \mathbf{S}_{\mathbf{zu}}^{\mathsf{T}} \mathcal{W}_{\mathbf{e}} \mathbf{S}_{\mathbf{zu}}) \tag{2b}$$

$$\mathcal{L} = (\lambda \mathbf{I} + \mathbf{S}_{\mathbf{zu}}^{\mathsf{T}} \mathcal{W}_{\mathbf{e}} \mathbf{S}_{\mathbf{zu}})^{-1} \mathbf{S}_{\mathbf{zu}}^{\mathsf{T}} \mathcal{W}_{\mathbf{e}}, \tag{2c}$$

where λ is a design parameter and

$$\mathcal{W}_{\mathbf{e}} = \mathbf{I}_N \otimes \mathbf{W}_e \in \mathbb{S}_{++}^{Nn_z}, \quad \mathcal{W}_{\mathbf{u}} = \mathbf{I}_N \otimes \mathbf{W}_u \in \mathbb{S}_{++}^{Nn_u}, \tag{3}$$

$\mathbf{I}_N$ is the $N \times N$ identity matrix, $\otimes$ denotes the Kronecker product, $\mathbf{S}_{\mathbf{zu}}$ is the batch model from $\mathbf{u}$ to $\mathbf{z}$, and $\overline{\mathbf{e}}_k = \overline{\mathbf{r}} - \overline{\mathbf{z}}_k$. The reader is referred to Amann et al. [1996]; Gunnarsson and Norrlöf [2001] for details of the derivation.

The update equation (2a) is stable and monotone for all system descriptions $\mathbf{S}_{\mathbf{zu}}$, i.e., the batch signal $\overline{\mathbf{u}}$ converges to a constant value monotonically, see e.g. Barton et al. [2008]; Gunnarsson and Norrlöf [2001].

3 Estimation-based ILC for Linear Systems

The error $\mathbf{e}_k(t)$ used in the ILC algorithm should be the difference between the reference $\mathbf{r}(t)$ and the controlled variable $\mathbf{z}_k(t)$ at iteration k. In general the controlled variable $\mathbf{z}_k(t)$ is not directly measurable, therefore an estimation-based ILC algorithm must be used, i.e., the ILC algorithm has to rely on estimates based on measurements of related quantities. The situation is schematically described in Figure 1.

3.1 Estimation-based Norm-optimal ILC

A straightforward extension to the standard norm-optimal ILC method is to use the error $\hat{\mathbf{e}}_k(t) = \mathbf{r}(t) - \hat{\mathbf{z}}_k(t)$ in the equations from Section 2, where $\hat{\mathbf{z}}_k(t)$ is an estimate of the controlled variable. The estimate $\hat{\mathbf{z}}_k(t)$ can be obtained using e.g., a *Kalman filter* (KF) for the linear case, or an *extended Kalman filter* (EKF) for the non-linear case [Kailath et al., 2000]. Linear systems are covered in this section while Section 4 extends the ideas to non-linear systems. In both cases it must be assumed that i) the system is observable, and ii) the filter, used for estimation, converges.

The solution to the optimisation problem can be obtained in a similar way as for the standard norm-optimal ILC problem in Section 2, where the detailed derivation is presented in Amann et al. [1996]; Gunnarsson and Norrlöf [2001]. An important distinction, compared to standard norm-optimal ILC, relates to what models are used in the design. In the estimation-based approach, the model from $\mathbf{u}_{k+1}(t)$ and $\mathbf{r}(t)$ to $\hat{\mathbf{z}}_{k+1}(t)$ is used, i.e., the dynamics from the KF must be included in the design, while in the standard norm-optimal design, the model from $\mathbf{u}_{k+1}(t)$ and $\mathbf{r}(t)$ to $\mathbf{z}_{k+1}(t)$ is used.

Let the discrete-time state space model be given by

$$\mathbf{x}_k(t+1) = \mathbf{A}(t)\mathbf{x}_k(t) + \mathbf{B}_{\mathbf{u}}(t)\mathbf{u}_k(t) + \mathbf{B}_{\mathbf{r}}(t)\mathbf{r}(t) + \mathbf{G}(t)\mathbf{w}_k(t), \tag{4a}$$

$$\mathbf{y}_k(t) = \mathbf{C}_{\mathbf{y}}(t)\mathbf{x}_k(t) + \mathbf{D}_{\mathbf{yu}}(t)\mathbf{u}_k(t) + \mathbf{D}_{\mathbf{yr}}(t)\mathbf{r}(t) + \mathbf{v}_k(t), \tag{4b}$$

$$\mathbf{z}_k(t) = \mathbf{C}_{\mathbf{z}}(t)\mathbf{x}_k(t) + \mathbf{D}_{\mathbf{zu}}(t)\mathbf{u}_k(t) + \mathbf{D}_{\mathbf{zr}}(t)\mathbf{r}(t), \tag{4c}$$

where the process noise $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t)$, and the measurement noise $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$. A batch model (see Appendix A for definitions) for the output $\mathbf{y}_k$ and the estimate $\hat{\mathbf{z}}_k$ can be written as

$$\overline{\mathbf{y}}_k = \mathcal{C}_{\mathbf{y}}\mathbf{\Phi}\mathbf{x}_0 + \mathbf{S}_{\mathbf{yu}}\overline{\mathbf{u}}_k + \mathbf{S}_{\mathbf{yr}}\overline{\mathbf{r}}, \tag{5a}$$

$$\hat{\overline{\mathbf{z}}}_k = \mathcal{C}_{\mathbf{z}}\widetilde{\mathbf{\Phi}}\hat{\mathbf{x}}_0 + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{u}}\overline{\mathbf{u}}_k + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{r}}\overline{\mathbf{r}} + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y}}\overline{\mathbf{y}}_k, \tag{5b}$$

where $\mathbf{w}(t)$ and $\mathbf{v}(t)$ have been replaced by their expected values, which are both equal to zero, in the output model (5a). The KF batch formulation has been used in the model of the estimate in (5b). The optimal solution is now given by

$$\overline{\mathbf{u}}_{k+1} = \mathcal{Q} \cdot (\overline{\mathbf{u}}_k + \mathcal{L} \cdot \hat{\overline{\mathbf{e}}}_k) \tag{6a}$$

$$\mathcal{Q} = (\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\mathcal{W}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}} + \mathcal{W}_{\mathbf{u}} + \lambda\mathbf{I})^{-1}(\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\mathcal{W}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}}) \tag{6b}$$

$$\mathcal{L} = (\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\mathcal{W}_{\mathbf{e}}\mathbf{S}_{\mathbf{u}})^{-1}\mathbf{S}_{\mathbf{u}}^{\mathsf{T}}\mathcal{W}_{\mathbf{e}}, \tag{6c}$$

where $\mathbf{S}_{\mathbf{u}} = \mathbf{S}_{\hat{\mathbf{z}}\mathbf{u}} + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y}}\mathbf{S}_{\mathbf{yu}}$ (see (26), (30) for details), and $\hat{\overline{\mathbf{e}}}_k = \overline{\mathbf{r}} - \hat{\overline{\mathbf{z}}}_k$. The expressions for $\mathcal{Q}$ and $\mathcal{L}$ in (6) are similar to (2). The only difference is that $\mathbf{S}_{\mathbf{u}}$ is used instead of $\mathbf{S}_{\mathbf{zu}}$. Theorem 1 presents a result for the special case of LTI-systems using the stationary KF[2].

Theorem 1. *(Separation lemma for estimation-based ILC): Given an LTI-system and the stationary KF with constant gain matrix* $\mathbf{K}$*, then the matrices* $\mathbf{S}_{\mathbf{u}}$ *and* $\mathbf{S}_{\mathbf{zu}}$ *are equal, hence the ILC algorithm* (2) *can be used for the estimation-based norm-optimal ILC.*

Proof: The result follows from the fact that it can be shown algebraically that $\mathbf{S}_{\mathbf{u}}$ and $\mathbf{S}_{\mathbf{zu}}$ are equal. The proof involves several algebraical manipulations, and

[2] The stationary Kalman filter uses a constant gain $\mathbf{K}$ which is the solution to an algebraic Riccati equation, see Kailath et al. [2000].

only the first steps of the calculations, in the case of $\mathbf{D_{yu}} = \mathbf{0}$ and $\mathbf{D_{zu}} = \mathbf{0}$, are presented here. From Appendix A it follows that $\mathbf{S_{zu}} = \mathcal{C}_\mathbf{z}\boldsymbol{\Psi}\mathcal{B}_\mathbf{u}$ and $\mathbf{S_u} = \mathcal{C}_\mathbf{z}\widetilde{\boldsymbol{\Psi}}\widetilde{\mathcal{B}}_\mathbf{u} + \mathcal{C}_\mathbf{z}\widetilde{\boldsymbol{\Psi}}_2\mathcal{K}\mathcal{C}_\mathbf{y}\boldsymbol{\Psi}\mathcal{B}_\mathbf{u}$. The structure of $\widetilde{\mathcal{B}}_\mathbf{u}$ gives $\mathbf{S_u} = \mathcal{C}_\mathbf{z}\left(\widetilde{\boldsymbol{\Psi}}\boldsymbol{\Gamma} + \widetilde{\boldsymbol{\Psi}}_2\mathcal{K}\mathcal{C}_\mathbf{y}\boldsymbol{\Psi}\right)\mathcal{B}_\mathbf{u}$, where $\boldsymbol{\Gamma} = \operatorname{diag}\left(\mathbf{I} - \mathbf{K}\mathbf{C_y}, \quad \ldots, \quad \mathbf{I} - \mathbf{K}\mathbf{C_y}, \quad \mathbf{0}\right)$. After some manipulations it can be shown that $\widetilde{\boldsymbol{\Psi}}\boldsymbol{\Gamma} + \widetilde{\boldsymbol{\Psi}}_2\mathcal{K}\mathcal{C}_\mathbf{y}\boldsymbol{\Psi} = \boldsymbol{\Psi}$, hence $\mathbf{S_{zu}} = \mathbf{S_u}$. The case $\mathbf{D_{yu}} \neq \mathbf{0}$ and $\mathbf{D_{zu}} \neq \mathbf{0}$ gives similar, but more involved, calculations. □

The result from Theorem 1 makes it computationally more efficient to compute the matrices $\mathcal{Q}$ and $\mathcal{L}$, since the matrix $\mathbf{S_{zu}}$ requires fewer calculations to obtain, compared to the matrix $\mathbf{S_u}$. Even if the iterative KF update is used, the Kalman gain $\mathbf{K}$ converges eventually to the stationary value for LTI systems. If this is done reasonably fast, then $\mathbf{S_u} \approx \mathbf{S_{zu}}$ can be a good enough approximation to use.

The stability result for the ILC algorithm in (6) is given in Theorem 2.

Theorem 2. *(Stability and monotonic convergence): The estimation-based ILC algorithm* (6) *is stable and monotonically convergent for all systems given by* (5).

Proof: Assuming that the KF is initialised with the same covariance matrix $\mathbf{P}_0$ at each iteration, makes the sequence of Kalman gains $\mathbf{K}(t)$, $t = 0, \ldots, N-1$ the same for different ILC iterations since $\mathbf{P}$ and $\mathbf{K}$ are independent of the control signal $\mathbf{u}$. The system matrices $\mathbf{S_{\hat{z}u}}$ and $\mathbf{S_{\hat{z}y}}$ are therefore constant over the ILC iterations, hence the same analysis for stability and monotone convergence as for the standard norm-optimal ILC, presented in Barton et al. [2008]; Gunnarsson and Norrlöf [2001], can be used. □

3.2 Utilising the Complete PDF for the ILC Error

Usually, only the expected value of the *probability density function* (PDF) $p(\mathbf{z}(t))$ of the estimated control quantity, which can be obtained by the KF or the EKF, is used. However, since the KF and EKF provides both the mean estimate and the covariance matrix for the estimation error, it is possible to use the full PDF in the design. The norm-optimal ILC is therefore extended to handle both the expected value and the covariance matrix of the estimated control error.

The objective of ILC is to achieve an error close to zero. Only considering the mean value is insufficient since a large variance means that there is a high probability that the actual error is not close to zero. Hence, the mean of the distribution should be close to zero and at the same time the variance should be small. To achieve this the PDF of the error is compared with a desired PDF with zero mean and small variance, using the *Kullback-Leibler* (KL) divergence [Kullback and Leibler, 1951]. The objective function (1) is modified by replacing the term $\|\mathbf{e}_{k+1}(t)\|^2_{\mathbf{W_e}}$ with the KL-divergence $D_{\mathrm{KL}}\left(q(\mathbf{e}_{k+1}(t))\|p(\mathbf{e}_{k+1}(t))\right)$, where $p(\mathbf{e}_{k+1}(t))$ is the actual distribution of the error given by the estimator, and $q(\mathbf{e}_{k+1}(t))$ is the desired distribution for the error, which becomes a design parameter.

Using a KF to estimate the state vector and calculating the control error according to $\hat{\mathbf{e}}(t) = \mathbf{r}(t) - \hat{\mathbf{z}}(t)$ gives that $\hat{\mathbf{e}}(t)$ has a Gaussian distribution [Kailath et al., 2000]

$$\hat{\mathbf{x}}(t) \sim \mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}_{t|t}, \mathbf{P}_{t|t}), \tag{7}$$

where $\hat{\mathbf{x}}_{t|t}$ denotes the mean value and $\mathbf{P}_{t|t}$ denotes the covariance matrix for the estimation error. In both cases the value is valid at time t given measurements up to time t. Moreover, let the estimated control quantity be given by

$$\hat{\mathbf{z}}(t) = \mathbf{C}_{\mathbf{z}}(t)\hat{\mathbf{x}}(t) + \mathbf{D}_{\mathbf{zu}}(t)\mathbf{u}(t) + \mathbf{D}_{\mathbf{zr}}(t)\mathbf{r}(t). \tag{8}$$

Since (8) is an affine transformation of a Gaussian distribution

$$\hat{\mathbf{z}}(t) \sim \mathcal{N}\left(\mathbf{z}; \hat{\mathbf{z}}(t|t), \Sigma_{\mathbf{z}}(t|t)\right), \tag{9a}$$

$$\hat{\mathbf{z}}(t|t) = \mathbf{C}_{\mathbf{z}}(t)\hat{\mathbf{x}}_{t|t} + \mathbf{D}_{\mathbf{zu}}(t)\mathbf{u}(t) + \mathbf{D}_{\mathbf{zr}}(t)\mathbf{r}(t), \tag{9b}$$

$$\Sigma_{\mathbf{z}}(t|t) = \mathbf{C}_{\mathbf{z}}(t)\mathbf{P}_{t|t}\mathbf{C}_{\mathbf{z}}(t)^{\mathsf{T}}. \tag{9c}$$

Finally, the error $\hat{\mathbf{e}}(t) = \mathbf{r}(t) - \hat{\mathbf{z}}(t)$ has the distribution

$$\hat{\mathbf{e}}(t) \sim p(\mathbf{e}(t)) = \mathcal{N}\left(\mathbf{e}; \hat{\mathbf{e}}(t|t), \Sigma_{\mathbf{e}}(t|t)\right), \tag{10a}$$

$$\hat{\mathbf{e}}(t|t) = \mathbf{r}(t) - \hat{\mathbf{z}}(t|t), \tag{10b}$$

$$\Sigma_{\mathbf{e}}(t|t) = \mathbf{C}_{\mathbf{z}}(t)\mathbf{P}_{t|t}\mathbf{C}_{\mathbf{z}}(t)^{\mathsf{T}}. \tag{10c}$$

For the linear case where $\hat{\mathbf{e}}_k(t)$ is Gaussian distributed according to (10) it is assumed that the desired distribution should resemble $q(\mathbf{e}(t)) = \mathcal{N}(\mathbf{e}; \mathbf{0}, \Sigma)$, where Σ is a design parameter. Straightforward calculations utilising two Gaussian distributions give the KL-divergence [Arndt, 2001]

$$D_{\mathrm{KL}}(\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_0, \Sigma_0) \| \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_1, \Sigma_1)) = \frac{1}{2}\Bigg(\operatorname{tr} \Sigma_2^{-1}\Sigma_1$$

$$+ (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^{\mathsf{T}}\Sigma_2^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \log \frac{|\Sigma_2|}{|\Sigma_1|} - n_x \Bigg), \tag{11}$$

where tr is the trace operator and $|\cdot|$ the determinant of a matrix. Therefore, the KL-divergence using $p(\mathbf{e}(t))$ from (10) and $q(\mathbf{e}(t)) = \mathcal{N}(\mathbf{e}; \mathbf{0}, \Sigma)$ gives

$$D_{\mathrm{KL}}(q(\mathbf{e}(t)) \| p(\mathbf{e}(t))) = \frac{1}{2}\hat{\mathbf{e}}(t|t)^{\mathsf{T}}\Sigma_{\mathbf{e}}^{-1}(t|t)\hat{\mathbf{e}}(t|t) + \xi, \tag{12}$$

where ξ is a term which does not depend on $\mathbf{x}$, $\mathbf{u}$, and $\mathbf{y}$. Here we used that the covariance matrix for the estimation error is independent of the observed data, as a consequence of the KF update equations. It only depends on the model parameters and the initial value $\mathbf{P}_0$. The objective function is finally modified by replacing the term $\|\mathbf{e}_{k+1}(t)\|^2_{\mathbf{W}_e}$ in (1) with

$$\|\hat{\mathbf{e}}_{k+1}(t|t)\|^2_{\Sigma_{\mathbf{e}}^{-1}(t|t)}. \tag{13}$$

The interpretation of the result is that, if the estimated quantity is certain it will affect the objective function more than if it is less certain, i.e., an estimated error signal with large uncertainty has a low weight. The optimisation problem is solved in the same way as in Section 3.1. The only difference is that the weight matrix for the error in batch form, now is given by

$$\mathcal{W}_{\mathbf{e}} = \operatorname{diag}\left(\Sigma_{\mathbf{e}}^{-1}(0|0), \quad \ldots, \quad \Sigma_{\mathbf{e}}^{-1}(N-1|N-1)\right) \in \mathbb{S}_{++}^{Nn_z}.$$

Remark 1. The separation lemma in Theorem 1 and the stability result in Theorem 2 are not affected when the full PDF is included in the objective function.

Remark 2. By interchanging the distributions $p(\cdot)$ and $q(\cdot)$ the result will be the norm of the mean error but now weighted with Σ^{-1}, which is a tuning parameter, hence no information of the covariance matrix of the control error is used in the design.

3.3 Structure of the Systems used for ILC and Estimation

In the derivation of the estimation-based norm-optimal ILC algorithm in Section 3.1 it is assumed that the KF takes the signals $\mathbf{r}(t)$ and $\mathbf{u}_k(t)$ as inputs. However, in general the estimation algorithm does not always use $\mathbf{r}(t)$ and $\mathbf{u}_k(t)$ as input. As an example, a system with a feedback loop usually uses the input to the controlled system for estimation, not the input to the controller. More general, the estimation algorithm only uses a part of the full system for estimation, whereas the other part is completely known or not interesting to use for estimation. Nevertheless, the system (4) is a valid description of the estimation-based ILC since the internal signal used for estimation can be written as an output from a system with $\mathbf{r}(t)$, $\mathbf{u}_k(t)$, and $\mathbf{y}_k(t)$ as inputs.

Let $\boldsymbol{\tau}_k(t)$ be the known signal used for estimation, hence the estimated variable can be written as

$$\hat{\mathbf{z}}_k(t) = F_{\hat{\mathbf{z}}\boldsymbol{\tau}}(q)\boldsymbol{\tau}_k(t) + F_{\hat{\mathbf{z}}\mathbf{y}}(q)\mathbf{y}_k(t). \tag{14}$$

Moreover, the signal $\boldsymbol{\tau}_k(t)$ can be seen as an output from a system with $\mathbf{r}(t)$, $\mathbf{u}_k(t)$, and $\mathbf{y}_k(t)$ as inputs, hence

$$\boldsymbol{\tau}_k(t) = F_{\boldsymbol{\tau}\mathbf{r}}(q)\mathbf{r}(t) + F_{\boldsymbol{\tau}\mathbf{u}}(q)\mathbf{u}_k(t) + F_{\boldsymbol{\tau}\mathbf{y}}(q)\mathbf{y}_k(t). \tag{15}$$

Combining (14) and (15) gives

$$\hat{\mathbf{z}}_k(t) = S_{\hat{\mathbf{z}}\mathbf{r}}(q)\mathbf{r}(t) + S_{\hat{\mathbf{z}}\mathbf{u}}(q)\mathbf{u}_k(t) + S_{\hat{\mathbf{z}}\mathbf{y}}(q)\mathbf{y}_k(t), \tag{16}$$

where

$$\begin{aligned}
&S_{\hat{\mathbf{z}}\mathbf{r}}(q) = F_{\hat{\mathbf{z}}\boldsymbol{\tau}}(q)F_{\boldsymbol{\tau}\mathbf{r}}(q), \;\; S_{\hat{\mathbf{z}}\mathbf{u}}(q) = F_{\hat{\mathbf{z}}\boldsymbol{\tau}}(q)F_{\boldsymbol{\tau}\mathbf{u}}(q),\\
&S_{\hat{\mathbf{z}}\mathbf{y}}(q) = F_{\hat{\mathbf{z}}\boldsymbol{\tau}}(q)F_{\boldsymbol{\tau}\mathbf{y}}(q) + F_{\hat{\mathbf{z}}\mathbf{y}}(q),
\end{aligned}$$

which is in the form described in Figure 1. It is straightforward to take this into consideration when deriving the ILC update control algorithm since it only changes the definitions of $\mathbf{S}_{\hat{\mathbf{z}}\mathbf{u}}$, $\mathbf{S}_{\hat{\mathbf{z}}\mathbf{r}}$, and $\mathbf{S}_{\hat{\mathbf{z}}\mathbf{y}}$. Note that the dimension of the state vector describing (5a) and (5b) can differ since (5b) is constructed using only a part of the complete system.

4 Estimation-based ILC for Non-linear Systems

Iterative learning control for non-linear systems has been considered in e.g. Avrachenkov [1998]; Lin et al. [2006]; Xiong and Zhang [2004]. Using a batch formulation of a non-linear state space model gives the following set of non-linear equations $\overline{\mathbf{y}}_k = \mathcal{F}(\overline{\mathbf{u}}_k)$. The goal for the control is to solve for $\overline{\mathbf{u}}$ such that $\overline{\mathbf{r}} = \mathcal{F}(\overline{\mathbf{u}})$, which has an iterative solution according to $\overline{\mathbf{u}}_{k+1} = \overline{\mathbf{u}}_k + \mathbf{L}_k \overline{\mathbf{e}}_k$. The proposed solutions in Avrachenkov [1998]; Lin et al. [2006]; Xiong and Zhang [2004] all have in common that they calculate the gain $\mathbf{L}_k$ using different approximations of the Newton method, which is a common method for solving non-linear equations. In Lin et al. [2006] the solution is rewritten, giving first a linearised system, where standard linear ILC methods can be applied, and then a final update of the ILC signal for the non-linear system. Nothing is stated about how to generally obtain the state trajectory for the linearisation step.

The method proposed here is to directly transform the non-linear system to a linear time-varying system, and then use the standard norm-optimal method. The linear state space model is obtained by linearising around an estimate of the complete state trajectory obtained from the previous ILC iteration. A necessary assumption is to have $\overline{\mathbf{u}}_{k+1}$ close to $\overline{\mathbf{u}}_k$, in order to get accurate models for the optimisation. It is possible to achieve $\overline{\mathbf{u}}_{k+1}$ close to $\overline{\mathbf{u}}_k$ by assigning λ a large enough value. The drawback is that the convergence rate can become slow.

4.1 Solution using Linearised Model

The non-linear model for iteration k and discrete time t

$$\mathbf{x}_k(t+1) = f(\mathbf{x}_k(t), \mathbf{u}_k(t), \mathbf{r}(t)) \tag{17a}$$

$$\mathbf{y}_k(t) = h_{\mathbf{y}}(\mathbf{x}_k(t), \mathbf{u}_k(t), \mathbf{r}(t)) \tag{17b}$$

$$\mathbf{z}_k(t) = h_{\mathbf{z}}(\mathbf{x}_k(t), \mathbf{u}_k(t), \mathbf{r}(t)) \tag{17c}$$

is linearised around $\hat{\mathbf{x}}_{k-1}(t|t)$, $\mathbf{u}_{k-1}(t)$, and $\mathbf{r}(t)$, yielding the following linear time-varying model

$$\begin{aligned}
\mathbf{x}_k(t+1) &= \mathbf{A}_{k-1}(t)\mathbf{x}_k(t) + \mathbf{B}_{k-1}(t)\mathbf{u}_k(t) + \mathbf{s}_{\mathbf{x},k-1}(t) \\
\mathbf{y}_k(t) &= \mathbf{C}_{\mathbf{y},k-1}\mathbf{x}_k(t) + \mathbf{D}_{\mathbf{y},k-1}(t)\mathbf{u}_k(t) + \mathbf{s}_{\mathbf{y},k-1}(t) \\
\mathbf{z}_k(t) &= \mathbf{C}_{\mathbf{z},k-1}\mathbf{x}_k(t) + \mathbf{D}_{\mathbf{z},k-1}(t)\mathbf{u}_k(t) + \mathbf{s}_{\mathbf{z},k-1}(t)
\end{aligned}$$

where

$$\mathbf{A}_{k-1}(t) = \frac{\partial f}{\partial \mathbf{x}}, \qquad \mathbf{B}_{k-1}(t) = \frac{\partial f}{\partial \mathbf{u}}, \quad \mathbf{C}_{\mathbf{y},k-1}(t) = \frac{\partial h_{\mathbf{y}}}{\partial \mathbf{x}},$$

$$\mathbf{C}_{\mathbf{z},k-1}(t) = \frac{\partial h_{\mathbf{z}}}{\partial \mathbf{x}}, \quad \mathbf{D}_{\mathbf{y},k-1}(t) = \frac{\partial h_{\mathbf{y}}}{\partial \mathbf{u}}, \quad \mathbf{D}_{\mathbf{z},k-1}(t) = \frac{\partial h_{\mathbf{z}}}{\partial \mathbf{u}},$$

all evaluated at the point $\mathbf{x} = \hat{\mathbf{x}}_{k-1}(t|t)$, $\mathbf{u} = \mathbf{u}_{k-1}(t)$, and $\mathbf{r} = \mathbf{r}(t)$, with

$$\begin{aligned}
\mathbf{s}_{\mathbf{x},k-1}(t) &= f(\hat{\mathbf{x}}_{k-1}(t|t), \mathbf{u}_{k-1}(t), \mathbf{r}(t)) - \mathbf{A}_{k-1}(t)\hat{\mathbf{x}}_{k-1}(t|t) - \mathbf{B}_{k-1}(t)\mathbf{u}_{k-1}(t)\\
\mathbf{s}_{\mathbf{y},k-1}(t) &= h_y(\hat{\mathbf{x}}_{k-1}(t|t), \mathbf{u}_{k-1}(t), \mathbf{r}(t)) - \mathbf{C}_{\mathbf{y},k-1}(t)\hat{\mathbf{x}}_{k-1}(t|t) - \mathbf{D}_{\mathbf{y},k-1}(t)\mathbf{u}_{k-1}(t)\\
\mathbf{s}_{\mathbf{z},k-1}(t) &= h_z(\hat{\mathbf{x}}_{k-1}(t|t), \mathbf{u}_{k-1}(t), \mathbf{r}(t)) - \mathbf{C}_{\mathbf{z},k-1}(t)\hat{\mathbf{x}}_{k-1}(t|t) - \mathbf{D}_{\mathbf{z},k-1}(t)\mathbf{u}_{k-1}(t).
\end{aligned}$$

The linearised quantities only depend on the previous ILC iteration, hence they are known at the current iteration. Using the linearised state space model gives, as before, the estimate $\hat{\mathbf{z}}_k(t)$ and the output $\mathbf{y}_k(t)$ in batch form as

$$\begin{aligned}
\bar{\mathbf{y}}_k &= \mathcal{C}_{\mathbf{y},k-1}\mathbf{\Phi}_{k-1}\mathbf{x}_0 + \mathbf{S}_{\mathbf{yu},k-1}\bar{\mathbf{u}}_k + \mathbf{S}_{\mathbf{ys_x},k-1}\bar{\mathbf{s}}_{\mathbf{x},k-1} + \bar{\mathbf{s}}_{\mathbf{y},k-1},\\
\hat{\bar{\mathbf{z}}}_k &= \mathcal{C}_{\mathbf{z},k-1}\widetilde{\mathbf{\Phi}}_{k-1}\hat{\mathbf{x}}_0 + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{u},k-1}\bar{\mathbf{u}}_k + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y},k-1}\bar{\mathbf{y}}_k\\
&\quad + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{s_x},k-1}\bar{\mathbf{s}}_{\mathbf{x},k-1} + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{s_y},k-1}\bar{\mathbf{s}}_{\mathbf{y},k-1} + \bar{\mathbf{s}}_{\mathbf{z},k-1},
\end{aligned}$$

where all the matrices in the right hand side are dependent of k, and the vectors $\bar{\mathbf{s}}_{\mathbf{x},k}$, $\bar{\mathbf{s}}_{\mathbf{y},k}$, and $\bar{\mathbf{s}}_{\mathbf{z},k}$ are stacked versions of $\mathbf{s}_{\mathbf{x},k}(t)$, $\mathbf{s}_{\mathbf{y},k}(t)$, and $\mathbf{s}_{\mathbf{z},k}(t)$.

The norm-optimal ILC problem can be formulated and solved in the same way as before. Unfortunately, since the batch form matrices are dependent of k, the terms including $\bar{\mathbf{r}}$, $\hat{\mathbf{x}}_0$, $\mathbf{x}_0$, $\bar{\mathbf{s}}_{\mathbf{x},k}$, $\bar{\mathbf{s}}_{\mathbf{y},k}$, and $\bar{\mathbf{s}}_{\mathbf{z},k}$ cannot be removed similar to what has been done before. Note that the weight matrix for the error is also dependent of k, since it consists of the covariance matrices for the control error. The solution is therefore given by

$$\begin{aligned}
\bar{\mathbf{u}}_{k+1} &= \left(\mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\mathbf{S}_{\mathbf{u},k} + \mathcal{W}_{\mathbf{u}} + \lambda\mathbf{I}\right)^{-1}\Big[\lambda\bar{\mathbf{u}}_k + \mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\\
&\quad \times \Big(\bar{\mathbf{r}} - \mathcal{C}_{\mathbf{z},k}\widetilde{\mathbf{\Phi}}_k\hat{\mathbf{x}}_0 - \mathbf{S}_{\hat{\mathbf{z}}\mathbf{s_x},k}\bar{\mathbf{s}}_{\mathbf{x},k} - \mathbf{S}_{\hat{\mathbf{z}}\mathbf{s_y},k}\bar{\mathbf{s}}_{\mathbf{y},k}\\
&\quad - \bar{\mathbf{s}}_{\mathbf{z},k} - \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y},k}(\mathcal{C}_{\mathbf{y},k}\mathbf{\Phi}_k\mathbf{x}_0 + \mathbf{S}_{\mathbf{ys_x},k}\bar{\mathbf{s}}_{\mathbf{x},k} + \bar{\mathbf{s}}_{\mathbf{y},k})\Big)\Big],
\end{aligned} \tag{18}$$

where $\mathbf{S}_{\mathbf{u},k} = \mathbf{S}_{\hat{\mathbf{z}}\mathbf{u},k} + \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y},k}\mathbf{S}_{\mathbf{yu},k}$ (see (26), (30) for details). The initial condition $\mathbf{x}_0$ is of course not known, hence $\hat{\mathbf{x}}_0$ must be used instead.

Remark 3. If the change in $\|\bar{\mathbf{u}}_{k+1} - \bar{\mathbf{u}}_k\|$ is sufficiently small, then the approximation $\mathbf{S}_{\mathbf{yu},k-1} \approx \mathbf{S}_{\mathbf{yu},k}$ and similar for the rest of the quantities is appropriate. Given the approximation the ILC algorithm (18) is simplified to

$$\bar{\mathbf{u}}_{k+1} = \mathcal{Q}_k \cdot (\bar{\mathbf{u}}_k + \mathcal{L}_k \cdot \hat{\bar{\mathbf{e}}}_k) \tag{19a}$$

$$\mathcal{Q}_k = (\mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\mathbf{S}_{\mathbf{u},k} + \mathcal{W}_{\mathbf{u}} + \lambda\mathbf{I})^{-1}(\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\mathbf{S}_{\mathbf{u},k}) \tag{19b}$$

$$\mathcal{L}_k = (\lambda\mathbf{I} + \mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\mathbf{S}_{\mathbf{u},k})^{-1}\mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}. \tag{19c}$$

Note that the change in $\bar{\mathbf{u}}$ depends strongly on the choice of λ.

4.2 Stability Analysis for the Linearised Solution

To analyse the stability of non-linear systems, utilising the above described linearisation technique, it is necessary to use the convergence results from Norrlöf and Gunnarsson [2002], which are based on theory for discrete time-variant systems. The stability property for the iteration-variant ILC system (18) is presented in Theorem 3.

Theorem 3. *(Iteration-variant stability): If $f(\cdot)$, $h_{\mathbf{y}}(\cdot)$ and $h_{\mathbf{z}}(\cdot)$ are differentiable, then the system using the* ILC *algorithm* (18) *is stable.*

Proof: From [Norrlöf and Gunnarsson, 2002, Corollary 3] it follows that (18) is stable if and only if

$$\bar{\sigma}\left(\left(\mathbf{S}_{\mathbf{u},k}^{\mathsf{T}}\mathcal{W}_{\mathbf{e},k}\mathbf{S}_{\mathbf{u},k} + \mathcal{W}_{\mathbf{u}} + \lambda\mathbf{I}\right)^{-1}\lambda\right) < 1, \tag{20}$$

for all k. The construction of $\mathcal{W}_{\mathbf{e},k}$ from the covariance matrices, gives that $\mathcal{W}_{\mathbf{e},k} \in \mathbb{S}_{++}$ for all k. This fact, together with the fact that $\mathcal{W}_{\mathbf{u}} \in \mathbb{S}_{++}$, and $\lambda \in \mathbb{R}_{+}$ guarantee that (20) is fulfilled for all k, hence the system with the ILC algorithm is stable. □

5 Conclusions and Future Work

An estimation-based norm-optimal ILC control algorithm was derived where the regular optimisation criteria for norm-optimal ILC was extended to an optimisation criteria including the first and second order moments of the posterior PDF, enabling better performance. For LTI-systems it was shown that the control law can be separated from the estimator dynamics. Extensions of the results to non-linear systems using linearisation were also presented together with stability and convergence results. The estimation-based norm-optimal ILC framework enables a systematic model-based design of ILC algorithms for linear as well as non-linear systems, where the controlled variables are not directly available as measurements.

Future work includes smoothing instead of filtering, to obtain the estimates, and to include the smoother dynamics in the ILC design. Also, investigating the use of the KL-divergence when the estimates are obtained using a particle filter or particle smoother is a possible extension.

Appendix

A State Space Model and Kalman Filter in Batch Form

For the norm-optimal ILC algorithm it is convenient to describe the state space model over the whole time horizon in batch form. The discrete-time state space model

$$\mathbf{x}(t+1) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}_{\mathbf{u}}(t)\mathbf{u}(t), \tag{21a}$$

$$\mathbf{y}(t) = \mathbf{C}_{\mathbf{y}}(t)\mathbf{x}(t) + \mathbf{D}_{\mathbf{yu}}(t)\mathbf{u}(t), \tag{21b}$$

has the following update formula for the state vector [Rugh, 1996]

$$\mathbf{x}(t) = \phi(t, t_0)\mathbf{x}(t_0) + \sum_{j=t_0}^{t-1} \phi(t, j+1)\mathbf{B_u}(j)\mathbf{u}(j), \tag{22}$$

for $t \geq t_0 + 1$, where the discrete-time transition matrix $\phi(t, j)$ is

$$\phi(t, j) = \begin{cases} \mathbf{A}(t-1) \cdot \ldots \cdot \mathbf{A}(j), & t \geq j+1 \\ \mathbf{I}, & t = j \end{cases} . \tag{23}$$

Using (21b) and (22), the output is given by

$$\mathbf{y}(t) = \mathbf{C_y}(t)\phi(t, t_0)\mathbf{x}(t_0) + \mathbf{D_{yu}}(t)\mathbf{u}(t) + \sum_{j=t_0}^{t-1} \mathbf{C_y}(t)\phi(t, j+1)\mathbf{B_u}(j)\mathbf{u}(j). \tag{24}$$

Introducing the vectors

$$\bar{\mathbf{x}} = \begin{pmatrix} \mathbf{x}(t_0)^\mathsf{T} & \ldots & \mathbf{x}(t_0+N)^\mathsf{T} \end{pmatrix}^\mathsf{T} \tag{25a}$$

$$\bar{\mathbf{u}} = \begin{pmatrix} \mathbf{u}(t_0)^\mathsf{T} & \ldots & \mathbf{u}(t_0+N)^\mathsf{T} \end{pmatrix}^\mathsf{T} \tag{25b}$$

$$\bar{\mathbf{y}} = \begin{pmatrix} \mathbf{y}(t_0)^\mathsf{T} & \ldots & \mathbf{y}(t_0+N)^\mathsf{T} \end{pmatrix}^\mathsf{T} \tag{25c}$$

gives the solution from $t = t_0$ to $t = t_0 + N$ as $\bar{\mathbf{x}} = \mathbf{\Phi}\mathbf{x}(t_0) + \mathbf{\Psi}\boldsymbol{\mathcal{B}}_\mathbf{u}\bar{\mathbf{u}}$ and for the output as

$$\bar{\mathbf{y}} = \boldsymbol{\mathcal{C}}_\mathbf{y}\mathbf{\Phi}\mathbf{x}(t_0) + \underbrace{(\boldsymbol{\mathcal{C}}_\mathbf{y}\mathbf{\Psi}\boldsymbol{\mathcal{B}}_\mathbf{u} + \boldsymbol{\mathcal{D}}_\mathbf{yu})}_{\triangleq \mathbf{S_{yu}}}\bar{\mathbf{u}}. \tag{26}$$

Here $\mathbf{x}(t_0)$ is the initial value, and

$$\boldsymbol{\mathcal{B}}_\mathbf{u} = \operatorname{diag}\begin{pmatrix} \mathbf{B_u}(t_0), & \ldots, & \mathbf{B_u}(t_0+N-1), & \mathbf{0} \end{pmatrix} \tag{27a}$$

$$\boldsymbol{\mathcal{C}}_\mathbf{y} = \operatorname{diag}\begin{pmatrix} \mathbf{C_y}(t_0), & \ldots, & \mathbf{C_y}(t_0+N) \end{pmatrix} \tag{27b}$$

$$\boldsymbol{\mathcal{D}}_\mathbf{yu} = \operatorname{diag}\begin{pmatrix} \mathbf{D_{yu}}(t_0), & \ldots, & \mathbf{D_{yu}}(t_0+N) \end{pmatrix} \tag{27c}$$

$$\mathbf{\Phi} = \begin{pmatrix} \mathbf{I} \\ \phi(t_0+1, t_0) \\ \phi(t_0+2, t_0) \\ \vdots \\ \phi(t_0+N, t_0) \end{pmatrix} \tag{27d}$$

$$\mathbf{\Psi} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \phi(t_0+2, t_0+1) & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \phi(t_0+N, t_0+1) & \phi(t_0+N, t_0+2) & \cdots & \mathbf{I} & \mathbf{0} \end{pmatrix} \tag{27e}$$

The Kalman filter can be written in a similar batch form as described above. Let the state space model be given by

$$\mathbf{x}(t+1) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}_{\mathbf{u}}(t)\mathbf{u}(t) + \mathbf{G}(t)\mathbf{w}(t), \tag{28}$$

$$\mathbf{y}(t) = \mathbf{C}_{\mathbf{y}}(t)\mathbf{x}(t) + \mathbf{D}_{\mathbf{yu}}(t)\mathbf{u}(t) + \mathbf{v}(t), \tag{29}$$

where $\mathbf{w}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}(t))$, and $\mathbf{v}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{R}(t))$. From the filter recursions [Kailath et al., 2000] we get

$$\begin{aligned}\hat{\mathbf{x}}(t+1|t+1) =& \big(\mathbf{I} - \mathbf{K}(t+1)\mathbf{C}_{\mathbf{y}}(t+1)\big)\mathbf{A}(t)\hat{\mathbf{x}}(t|t) + \big(\mathbf{I} - \mathbf{K}(t+1)\mathbf{C}_{\mathbf{y}}(t+1)\big)\mathbf{B}_{\mathbf{u}}(t)\mathbf{u}(t)\\ &- \mathbf{K}(t+1)\mathbf{D}_{\mathbf{yu}}(t+1)\mathbf{u}(t+1) + \mathbf{K}(t+1)\mathbf{y}(t+1)\\ =& \widetilde{\mathbf{A}}(t)\hat{\mathbf{x}}(t|t) + \widetilde{\mathbf{B}}_{\mathbf{u}}(t)\mathbf{u}(t) - \widetilde{\mathbf{D}}_{\mathbf{yu}}(t+1)\mathbf{u}(t+1) + \mathbf{K}(t+1)\mathbf{y}(t+1),\end{aligned}$$

where, $\mathbf{K}(t)$ is the Kalman gain given by the recursion

$$\begin{aligned}\mathbf{P}(t|t-1) =& \mathbf{A}(t-1)\mathbf{P}(t-1|t-1)\mathbf{A}(t-1)^{\mathsf{T}} + \mathbf{G}(t-1)\mathbf{Q}(t-1)\mathbf{G}(t-1)^{\mathsf{T}}\\ \mathbf{K}(t) =& \mathbf{P}(t|t-1)\mathbf{C}_{\mathbf{y}}(t)^{\mathsf{T}}\left(\mathbf{C}_{\mathbf{y}}(t)\mathbf{P}(t|t-1)\mathbf{C}_{\mathbf{y}}(t)^{\mathsf{T}} + \mathbf{R}(t)\right)^{-1}\\ \mathbf{P}(t|t) =& \big(\mathbf{I} - \mathbf{K}(t)\mathbf{C}_{\mathbf{y}}(t)\big)\mathbf{P}(t|t-1).\end{aligned}$$

The update formula for the estimated state vector is finally given by

$$\begin{aligned}\hat{\mathbf{x}}(t|t) = \widetilde{\phi}(t, t_0)\hat{\mathbf{x}}(t_0|t_0) + \sum_{j=t_0}^{t-1} \widetilde{\phi}(t, j+1)\widetilde{\mathbf{B}}_{\mathbf{u}}(j)\mathbf{u}(j)\\ - \sum_{j=t_0+1}^{t} \widetilde{\phi}(t, j)\widetilde{\mathbf{D}}_{\mathbf{yu}}(j)\mathbf{u}(j) + \sum_{j=t_0+1}^{t} \widetilde{\phi}(t, j)\mathbf{K}(j)\mathbf{y}(j),\end{aligned}$$

where $\widetilde{\phi}$ is defined as in (23), with $\widetilde{\mathbf{A}}(t)$ instead of $\mathbf{A}(t)$. The KF recursion in batch form becomes

$$\hat{\bar{\mathbf{x}}} = \widetilde{\mathbf{\Phi}}\hat{\mathbf{x}}(t_0|t_0) + (\widetilde{\mathbf{\Psi}}\widetilde{\boldsymbol{\mathcal{B}}}_{\mathbf{u}} - \widetilde{\mathbf{\Psi}}_2\widetilde{\boldsymbol{\mathcal{D}}}_{\mathbf{yu}})\bar{\mathbf{u}} + \widetilde{\mathbf{\Psi}}_2\boldsymbol{\mathcal{K}}\bar{\mathbf{y}},$$

where $\widetilde{\mathbf{\Phi}}$, $\widetilde{\mathbf{\Psi}}$, and $\widetilde{\boldsymbol{\mathcal{B}}}_{\mathbf{u}}$ are given in (27) with $\widetilde{\mathbf{A}}(t)$ and $\widetilde{\mathbf{B}}_{\mathbf{u}}(t)$ instead of $\mathbf{A}(t)$ and $\mathbf{B}_{\mathbf{u}}(t)$. The remaining matrices are defined as

$$\begin{aligned}\widetilde{\mathbf{\Psi}}_2 &= \begin{pmatrix}\mathbf{0}_{(N+1)n_x\times n_x} & \widetilde{\mathbf{\Psi}}\begin{pmatrix}\mathbf{I}_{Nn_x}\\ \mathbf{0}_{n_x\times Nn_x}\end{pmatrix}\end{pmatrix}\\ \widetilde{\boldsymbol{\mathcal{D}}}_{\mathbf{yu}} &= \operatorname{diag}\left(\mathbf{0}, \quad \widetilde{\mathbf{D}}_{\mathbf{yu}}(t_0+1), \quad \ldots, \quad \widetilde{\mathbf{D}}_{\mathbf{yu}}(t_0+N)\right)\\ \boldsymbol{\mathcal{K}} &= \operatorname{diag}\left(\mathbf{0}, \quad \mathbf{K}(t_0+1), \quad \ldots, \quad \mathbf{K}(t_0+N)\right).\end{aligned}$$

Finally, the batch formulation for the variable

$$\mathbf{z}(t) = \mathbf{C}_{\mathbf{z}}(t)\mathbf{x}_t + \mathbf{D}_{\mathbf{zu}}(t)\mathbf{u}_t,$$

is given by

$$\hat{\bar{\mathbf{z}}} = \mathcal{C}_{\mathbf{z}}\widetilde{\boldsymbol{\Phi}}\hat{\mathbf{x}}(t_0|t_0) + \underbrace{\Big(\mathcal{C}_{\mathbf{z}}(\widetilde{\boldsymbol{\Psi}}\widetilde{\boldsymbol{\mathcal{B}}}_{\mathbf{u}} - \widetilde{\boldsymbol{\Psi}}_2\widetilde{\boldsymbol{\mathcal{D}}}_{\mathbf{yu}}) + \boldsymbol{\mathcal{D}}_{\mathbf{zu}}\Big)}_{\triangleq \mathbf{S}_{\hat{\mathbf{z}}\mathbf{u}}}\bar{\mathbf{u}} + \underbrace{\mathcal{C}_{\mathbf{z}}\widetilde{\boldsymbol{\Psi}}_2\mathcal{K}}_{\triangleq \mathbf{S}_{\hat{\mathbf{z}}\mathbf{y}}}\bar{\mathbf{y}}, \tag{30}$$

where $\mathcal{C}_{\mathbf{z}}$ and $\boldsymbol{\mathcal{D}}_{\mathbf{zu}}$ are given in (27) using $\mathbf{C}_{\mathbf{z}}(t)$ and $\mathbf{D}_{\mathbf{zu}}(t)$.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC, by the Excellence Center at Linköping-Lund in Information Technology, ELLIIT, and by the SSF project Collaborative Localization.

Bibliography

Hyo-Sung Ahn, Kevin L. Moore, and YangQuan Chen. Kalman filter-augmented iterative learning control on the iteration domain. In *Proceedings of the American Control Conference*, pages 250–255, Minneapolis, MN, USA, June 2006.

Notker Amann, David H. Owens, and Eric Rogers. Iterative learning control for discrete-time systems with exponential rate of convergence. *IEE Proceedings Control Theory and Applications*, 143(2):217–224, March 1996.

Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2):123–140, 1984.

Christoph Arndt. *Information Measures*. Springer-Verlag, Berlin, Heidelberg, Germany, 2001.

Konstantin E. Avrachenkov. Iterative learning control based on quasi-Newton methods. In *Proceedings of the 37th IEEE Conference on Decision and Control*, pages 170–174, Tampa, FL, USA, December 1998.

Patrik Axelsson, Rickard Karlsson, and Mikael Norrlöf. Estimation-based norm-optimal iterative learning control. *Submitted to Systems & Control Letters*, 2014.

Kira Barton, Jeroen van de Wijdeven, Andrew Alleyne, Okko Bosgra, and Maarten Steinbuch. Norm optimal cross-coupled iterative learning control. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 3020–3025, Cancun, Mexico, December 2008.

Svante Gunnarsson and Mikael Norrlöf. On the design of ILC algorithms using optimization. *Automatica*, 37(12):2011–2016, December 2001.

Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, 2000.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, March 1951.

Kwang Soon Lee and Jay H. Lee. *Iterative Learning Control: Analysis, Design, Integration and Application*, chapter Design of Quadratic Criterion-based Iterative Learning Control, pages 165–192. Kluwer Academic Publishers, Boston, MA, USA, 1998.

T. Lin, David H. Owens, and Jari Hätönen. Newton method based iterative learning control for discrete non-linear systems. *International Journal of Control*, 79(10):1263–1276, 2006.

Kevin L. Moore. *Iterative Learning Control for Deterministic Systems*. Advances in Industrial Control. Springer-Verlag, London, UK, 1993.

Mikael Norrlöf and Svante Gunnarsson. Time and frequency domain convergence properties in iterative learning control. *International Journal of Control*, 75(14):1114–1126, 2002.

Wilson J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1996.

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.

Z. Xiong and J. Zhang. Batch-to-batch optimal control of nonlinear batch processes based on incrementally updated models. *IEE Proceedings Control Theory and Applications*, 151(2):158–165, March 2004.

Paper H

Controllability Aspects for Iterative Learning Control

Authors: Patrik Axelsson, Daniel Axehill, Torkel Glad and Mikael Norrlöf

Edited version of the paper:

Patrik Axelsson, Daniel Axehill, Torkel Glad, and Mikael Norrlöf. Controllability aspects for iterative learning control. *Submitted to International Journal of Control*, 2014a.

Controllability Aspects for Iterative Learning Control

Patrik Axelsson, Daniel Axehill, Torkel Glad and Mikael Norrlöf

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
patrik.axelsson@liu.se,
daniel.axehill@liu.se,
torkel.glad@liu.se,
mikael.norrlof@liu.se

Abstract

This paper considers the aspects of controllability in the iteration domain for systems that are controlled using *iterative learning control* (ILC). The focus is on controllability for a proposed state space model in the iteration domain and it relates to an assumption often used to prove convergence of ILC algorithms. It is shown that instead of investigating controllability it is more suitable to use the concept of *target path controllability* (TPC), where it is investigated if the output of a system can follow a trajectory instead of the ability to control the system to an arbitrary point in the state space. Finally, a simulation study is performed to show how the ILC algorithm can be designed using the LQ-method, if the state space model in the iteration domain is output controllable. The LQ-method is compared to the standard norm-optimal ILC algorithm, where it is shown that the control error can be reduced significantly using the LQ-method compared to the norm-optimal approach.

1 Introduction

Iterative learning control (ILC) is a method to improve the control of processes that perform the same task repeatedly [Arimoto et al., 1984; Moore, 1993]. A good example of such a process is an industrial robot performing arc welding or laser cutting in a general production situation. The system used for ILC can be both an open loop system as well as a closed loop system, therefore let the system be described by Figure 1. Usually, the ILC control signal $\mathbf{u}_k(t) \in \mathbb{R}^{n_u}$ is updated according to

$$\mathbf{u}_{k+1}(t) = \mathcal{F}\left(\{\mathbf{u}_k(i)\}_{i=0}^{N-1}, \{\mathbf{e}_k(i)\}_{i=0}^{N-1}\right), \quad t = 0, \ldots, N-1,$$

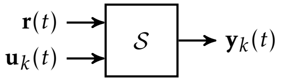


***Figure 1:** System used for ILC. It can be either an open or closed loop system. The reference signal is denoted by $\mathbf{r}(t)$, the ILC control signal by $\mathbf{u}_k(t)$ and the output signal by $\mathbf{y}_k(t)$, where k is the ILC iteration index and t the time index.*

where $\mathbf{e}_k(t) = \mathbf{r}(t) - \mathbf{y}_k(t)$ is the control error, $\mathbf{r}(t) \in \mathbb{R}^{n_y}$ the reference signal, $\mathbf{y}_k(t) \in \mathbb{R}^{n_y}$ the measurement signal, k the iteration index, t the time index and $\mathcal{F}(\,\cdot\,)$ is an update function. The main task is to find an update function that is able to drive the error to zero as the number of iterations tends to infinity, i.e.,

$$\|\mathbf{e}_k(t)\| \to 0, \quad k \to \infty, \forall t. \tag{1}$$

For the convergence proof it is usually suitable to use a batch description of the system,

$$\overline{\mathbf{y}}_k = \mathbf{S}_\mathbf{u} \overline{\mathbf{u}}_k + \mathbf{S}_\mathbf{r} \overline{\mathbf{r}}. \tag{2}$$

Here, the vectors $\overline{\mathbf{y}}_k$, $\overline{\mathbf{u}}_k$, and $\overline{\mathbf{r}}$ are composed by the vectors $\mathbf{y}_k(t)$, $\mathbf{u}_k(t)$, and $\mathbf{r}(t)$ stacked on each other for $t = 0, \ldots, N-1$, and the matrices $\mathbf{S}_\mathbf{u}$ and $\mathbf{S}_\mathbf{r}$ are lower triangular Toeplitz matrices containing the Markov parameters for the system, see Appendix A for details.

In Lee and Lee [1998, 2000]; Lee et al. [2000] it is proven that (1) holds under the assumption that $\mathbf{S}_\mathbf{u}$ has full row rank. Moreover, in Amann et al. [1996] it is assumed that $\ker \mathbf{S}_\mathbf{u}^\mathsf{T} = \emptyset$ which is equivalent to $\mathbf{S}_\mathbf{u}$ having full row rank. An important implication from this assumption is that it is necessary to have at least as many control signals as measurement signals. Even if the numbers of measurement signals and control signals are the same it can not be guaranteed that the full rank requirement is fulfilled.

This paper investigates what it means that $\mathbf{S}_\mathbf{u}$ has full row rank, based on a state space model in the iteration domain for which different types of controllability properties are considered. The result shows that the requirement of full row rank of $\mathbf{S}_\mathbf{u}$ is equivalent to the proposed state space model being output controllable. Using the definition of controllability, the interpretation of having a rank deficient matrix $\mathbf{S}_\mathbf{u}$, for a general system, is discussed. The aspects of controllability are then extended to *target path controllability* (TPC) [Engwerda, 1988] which is shown to be a more suitable requirement for ILC. TPC naturally leads to the concept of "lead-in", which is about extending the trajectory with a part in the beginning, where it is not important to have perfect trajectory tracking, see e.g. Wallén et al. [2011]. It is also important to realise that TPC can help in the design of the reference trajectory to achieve tracking with as short lead-in as possible.

In this paper the focus is on time-invariant systems. However, extensions to time-varying systems are straightforward using standard results for time-varying systems. In general, non-linear phenomena, such as control signal limitations, must also be considered, but that has been omitted in this work.

2 State Space Model in the Iteration Domain

The state space model in the iteration domain uses the batch formulation of the system dynamics. The batch formulation is known as the lifted system representation in the ILC community and is used both for design of ILC update laws, e.g. norm-optimal ILC [Amann et al., 1996; Gunnarsson and Norrlöf, 2001], as well as for analysis of stability and convergence. In this work, the following linear time-invariant state space model in discrete-time

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B_u}\mathbf{u}(t) + \mathbf{B_r}\mathbf{r}(t) + \mathbf{B_w}\mathbf{w}(t), \tag{3a}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{v}(t), \tag{3b}$$

where $\mathbf{w}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the process noise and $\mathbf{v}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the measurement noise, is considered. It is henceforth assumed that the system in (3) is both controllable and observable. The following batch formulation of the system

$$\overline{\mathbf{x}} = \mathbf{\Phi}\mathbf{x}(0) + \mathbf{S_{xu}}\overline{\mathbf{u}} + \mathbf{S_{xr}}\overline{\mathbf{r}}, \tag{4a}$$

$$\overline{\mathbf{y}} = \mathcal{C}\overline{\mathbf{x}}, \tag{4b}$$

where $\mathbf{w}(t)$ and $\mathbf{v}(t)$ have been replaced with their expected values which are equal to zero, can now be obtained according to Appendix A.

At ILC iteration k and $k+1$ it holds that

$$\overline{\mathbf{x}}_k = \mathbf{\Phi}\mathbf{x}(0) + \mathbf{S_{xu}}\overline{\mathbf{u}}_k + \mathbf{S_{xr}}\overline{\mathbf{r}}, \tag{5}$$

$$\overline{\mathbf{x}}_{k+1} = \mathbf{\Phi}\mathbf{x}(0) + \mathbf{S_{xu}}\overline{\mathbf{u}}_{k+1} + \mathbf{S_{xr}}\overline{\mathbf{r}}, \tag{6}$$

where the initial state $\mathbf{x}(0)$ and the reference $\overline{\mathbf{r}}$ repeats for all iterations, which are two common assumptions when applying ILC [Arimoto, 1990]. Subtracting (5) from (6) gives the following expression

$$\overline{\mathbf{x}}_{k+1} = \overline{\mathbf{x}}_k + \mathbf{S_{xu}}(\overline{\mathbf{u}}_{k+1} - \overline{\mathbf{u}}_k) = \overline{\mathbf{x}}_k + \mathbf{S_{xu}}\Delta_{\overline{\mathbf{u}}_k}, \tag{7}$$

where $\Delta_{\overline{\mathbf{u}}_k} \triangleq \overline{\mathbf{u}}_{k+1} - \overline{\mathbf{u}}_k$ is considered as a new control signal. The state space model in the iteration domain is therefore given by

$$\overline{\mathbf{x}}_{k+1} = \overline{\mathbf{x}}_k + \mathbf{S_{xu}}\Delta_{\overline{\mathbf{u}}_k}, \tag{8a}$$

$$\overline{\mathbf{y}}_k = \mathcal{C}\overline{\mathbf{x}}_k. \tag{8b}$$

3 Controllability

An important property for state space models is controllability, which considers the ability to control the system to a predefined state or output. This section considers necessary and sufficient conditions for the system in (8) to be controllable. First, state controllability is investigated and second, output controllability is investigated.

3.1 State Controllability

Controllability can formally be stated according to Definition 1.

Definition 1 (Controllability [Rugh, 1996]). A linear time-invariant state space model is called controllable if given any state $\mathbf{x}_f$ there is a positive integer t_f and an input signal $\mathbf{u}(t)$ such that the corresponding response of the system, beginning at $\mathbf{x}(0) = \mathbf{0}$, satisfies $\mathbf{x}(t_f) = \mathbf{x}_f$.

Theorem 1 (Controllability [Rugh, 1996]). *An LTI system is controllable if and only if the rank of the controllability matrix $\mathscr{C}$ is equal to the state dimension.*

From Theorem 1 it follows that the system in (8) is controllable if and only if $\operatorname{rank}\mathscr{C} = Nn_x$. Since the dynamics for (8) is an integrator, the controllability matrix $\mathscr{C}$ is given by $\mathbf{S_{xu}}$ repeated N times, i.e.,

$$\mathscr{C} = \begin{pmatrix} \mathbf{S_{xu}} & \cdots & \mathbf{S_{xu}} \end{pmatrix} \tag{9}$$

and the rank is simply given by

$$\operatorname{rank}\mathscr{C} = \operatorname{rank}\mathbf{S_{xu}}. \tag{10}$$

The system is therefore controllable if and only if $\operatorname{rank}\mathbf{S_{xu}} = Nn_x$. A necessary and sufficient condition for controllability of (8) is presented in Theorem 2.

Theorem 2. *System* (8) *in the iteration domain is controllable according to Definition 1 if and only if* $\operatorname{rank}\mathbf{B_u} = n_x$.

Proof: Exploiting the structure of $\mathbf{S_{xu}}$ gives

$$\mathbf{S_{xu}} = \underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{N-1} & \mathbf{A}^{N-2} & \cdots & \mathbf{I} \end{pmatrix}}_{\triangleq \boldsymbol{\Psi}} \underbrace{\begin{pmatrix} \mathbf{B_u} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{B_u} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B_u} \end{pmatrix}}_{\triangleq \boldsymbol{\mathcal{B}}}, \tag{11}$$

where it can be noted that the matrix $\boldsymbol{\Psi}$ is square and triangular with all diagonal elements equal to 1. The determinant of a square triangular matrix is equal to the product of the diagonal elements [Lütkepohl, 1996], hence $\det\boldsymbol{\Psi} = 1$ giving that $\boldsymbol{\Psi}$ is non-singular. It now follows that

$$\operatorname{rank}\mathbf{S_{xu}} = \operatorname{rank}\boldsymbol{\Psi}\boldsymbol{\mathcal{B}} = \operatorname{rank}\boldsymbol{\mathcal{B}} = N\operatorname{rank}\mathbf{B_u}. \tag{12}$$

The system is therefore controllable if and only if $\operatorname{rank}\mathbf{B_u} = n_x$. □

Corollary 1. *A necessary condition for system* (8) *to be controllable is that $n_u \geq n_x$.*

Proof: It is given that $\mathbf{B_u} \in \mathbb{R}^{n_x \times n_u}$, hence $\operatorname{rank} \mathbf{B_u} \leq \min\{n_x, n_u\}$. It is therefore necessary to have $n_u \geq n_x$ to be able to obtain $\operatorname{rank} \mathbf{B_u} = n_x$. □

Remark 1. Controllability of the time domain system (3), i.e.,

$$\operatorname{rank}\begin{pmatrix}\mathbf{B_u} & \mathbf{AB_u} & \cdots & \mathbf{A}^{n_x-1}\mathbf{B_u}\end{pmatrix} = n_x, \tag{13}$$

does not imply that the iteration domain system in (8) is controllable.

3.2 Output Controllability

The system in (8) is, as shown above, not necessarily controllable. However, the requirement of controllability is very strict. Often, it is not of interest in ILC to control all the states but only the output. Therefore, it is more relevant to consider output controllability of the system. A formal definition of output controllability follows from Definition 1 with $\mathbf{x}$ replaced by $\mathbf{y}$. The requirement for output controllability is that the output controllability matrix, denoted by $\mathscr{C}^o$, has full rank [Ogata, 2002], where

$$\mathscr{C}^o = \begin{pmatrix}\mathbf{CB_u} & \mathbf{CAB_u} & \cdots & \mathbf{CA}^{n_x-1}\mathbf{B_u} & \mathbf{D}\end{pmatrix} \tag{14}$$

for a general state space model parametrised by $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$. A condition for output controllability for the system in (8) is presented in Theorem 3.

Theorem 3. *The system in* (8) *is output controllable if and only if*

$$\operatorname{rank} \boldsymbol{\mathcal{C}}\mathbf{S_{xu}} = N n_y. \tag{15}$$

Proof: From (9) and (14) it can be concluded that $\mathscr{C}^o = \boldsymbol{\mathcal{C}}\mathscr{C}$ for the system in (8). Hence, the system is output controllable if and only if

$$\mathscr{C}^o = \begin{pmatrix}\boldsymbol{\mathcal{C}}\mathbf{S_{xu}} & \cdots & \boldsymbol{\mathcal{C}}\mathbf{S_{xu}}\end{pmatrix} \tag{16}$$

has full rank, i.e., $\operatorname{rank} \mathscr{C}^o = N n_y$. The result follows directly from the fact that $\operatorname{rank} \mathscr{C}^o = \operatorname{rank} \boldsymbol{\mathcal{C}}\mathbf{S_{xu}}$. □

Note that a general controllable LTI system is not necessarily output controllable, and a general output controllable system is not necessarily controllable.

It follows from (15) that if the system is output controllable the matrix $\boldsymbol{\mathcal{C}}\mathbf{S_{xu}}$ must have $N n_y$ independent rows. Hence, the measurements in the time domain model (3) must be independent which holds if $\operatorname{rank} \mathbf{C} = n_y$. Theorem 4 presents a necessary, but not sufficient, condition for output controllability.

Theorem 4. *Assume* $\operatorname{rank} \mathbf{C} = n_y$. *A necessary condition for system* (8) *to be output controllable is that* $\operatorname{rank} \mathbf{B_u} \geq n_y$.

Proof: The rank of a product of two matrices is less than or equal to the minimum of the rank of each matrix [Lütkepohl, 1996], hence

$$\operatorname{rank} \boldsymbol{\mathcal{C}}\mathbf{S_{xu}} \leq \min\{\operatorname{rank} \boldsymbol{\mathcal{C}}, \operatorname{rank} \mathbf{S_{xu}}\} = \min\{N \operatorname{rank} \mathbf{C}, N \operatorname{rank} \mathbf{B_u}\} \tag{17}$$

From the assumption rank $\mathbf{C} = n_y$ it follows that a necessary condition to have rank $\mathcal{C}\mathbf{S}_{\mathbf{xu}} = Nn_y$ is to have rank $\mathbf{B}_{\mathbf{u}} \geq n_y$ □

In Section 1 it was mentioned that in Lee and Lee [1998, 2000]; Lee et al. [2000]; Amann et al. [1996] $\mathbf{S}_{\mathbf{u}}$ was assumed to have full row rank. In this work, $\mathbf{S}_{\mathbf{u}} = \mathcal{C}\mathbf{S}_{\mathbf{xu}}$, hence the property (1) holds if the state space model (8) is output controllable. Section 4 will discuss the difficulties of having the system (8) controllable and output controllable.

3.3 Stabilisability

When a system is uncontrollable it is important to make sure that the uncontrollable modes are asymptotically stable, referred to as stabilisability, see Definition 2.

Definition 2 (Stabilisability [Rugh, 1996]). A linear system is stabilisable if there exists a state feedback gain $\mathbf{L}$ such that the closed-loop system

$$\mathbf{x}(t+1) = (\mathbf{A} - \mathbf{B}\mathbf{L})\mathbf{x}(t) \tag{18}$$

is exponentially stable.

It follows from Definition 2 that a linear system is stabilisable if the uncontrollable modes are asymptomatically stable, i.e., the eigenvalues of $\mathbf{A}$, that are associated with the uncontrollable modes, lie inside the unit circle. The uncontrollable modes can be obtained using a variable transformation, see e.g. Rugh [1996]. For a general state space system, the transformed system can be cast in the form

$$\begin{pmatrix} \widetilde{\mathbf{x}}^1_{k+1} \\ \widetilde{\mathbf{x}}^2_{k+1} \end{pmatrix} = \begin{pmatrix} \widetilde{\mathbf{A}}_{11} & \widetilde{\mathbf{A}}_{12} \\ \mathbf{0} & \widetilde{\mathbf{A}}_{22} \end{pmatrix} \begin{pmatrix} \widetilde{\mathbf{x}}^1_{k} \\ \widetilde{\mathbf{x}}^2_{k} \end{pmatrix} + \begin{pmatrix} \widetilde{\mathbf{B}}_1 \\ \mathbf{0} \end{pmatrix} \mathbf{u}_k \tag{19}$$

where $\widetilde{\mathbf{x}}^1$ is the controllable part, i.e., $(\widetilde{\mathbf{A}}_{11}, \widetilde{\mathbf{B}}_1)$ is controllable, and $\widetilde{\mathbf{x}}^2$ is the uncontrollable part. The eigenvalues of the uncontrollable part are the eigenvalues of $\widetilde{\mathbf{A}}_{22}$ which must be inside the unit circle in order for the total system to be stabilisable.

Considering the system in (8), it holds that all eigenvalues are equal to 1. Since a similarity transformation does not change the eigenvalues, the uncontrollable modes of (8) are only marginally stable, hence the system is not stabilisable.

4 Interpretation of the Controllability Requirements

A discussion about the possibilities for the system in (8) to be (output) controllable is given in this section. It is shown that (output) controllability of system (8) is difficult to achieve, both from theoretical and practical perspectives.

In Rugh [1996] it is stated that controllability of the system (3) does not consider what happens after time t_f. It only considers if it is possible to reach the desired state $\mathbf{x}_f$ within the desired time interval. Moreover, a single input system with state dimension n_x can require n_x time steps to be able to reach the desired state $\mathbf{x}_f$ or the desired output $\mathbf{y}_f$. It means that it can take up to n_x time steps before the state space model (3) reaches the reference trajectory. This practically means that the first part of $\overline{\mathbf{x}}$, and the corresponding part of $\overline{\mathbf{y}}$, cannot be defined by an arbitrary reference.

Another reason for the system not being (output) controllable is the construction of the vectors $\overline{\mathbf{x}}$ and $\overline{\mathbf{y}}$. It will be physically impossible to achieve any given $\overline{\mathbf{x}}_f$ or $\overline{\mathbf{y}}_f$. A simple example with $n_x = 2$, where the states are position and velocity, and the input is the acceleration, will be used to illustrate this.

Let $p(t)$ be the position, $v(t)$ the velocity, and let the state vector be

$$\mathbf{x}(t) = \begin{pmatrix} p(t) & v(t) \end{pmatrix}^{\mathsf{T}},$$

then the continuous-time model becomes

$$\dot{\mathbf{x}}(t) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \mathbf{x}(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u(t). \tag{20}$$

Discretisation of (20) using zero order hold gives the discrete-time model

$$\mathbf{x}(t+1) = \begin{pmatrix} 1 & T_s \\ 0 & 1 \end{pmatrix} \mathbf{x}(t) + \begin{pmatrix} T_s^2/2 \\ T_s \end{pmatrix} u(t), \tag{21}$$

where T_s is the sample time. The batch vector $\overline{\mathbf{x}}$ becomes

$$\overline{\mathbf{x}} = \begin{pmatrix} p(1) & v(1) & p(2) & v(2) & \cdots & p(N) & v(N) \end{pmatrix}^{\mathsf{T}}. \tag{22}$$

From Theorem 2 it follows that the system in (8) is controllable if and only rank $\mathbf{B}_{\mathbf{u}} = n_x = 2$. Here, rank $\mathbf{B}_{\mathbf{u}} = 1$ hence the system is not controllable. To explain this, consider the dynamics and assume that at time t the position $p(t) = a$ and the velocity $v(t) = b$ for some constants a and b. It should be possible to choose the position and velocity at the next time step $t+1$ arbitrarily to have controllability of the state space model in the iteration domain, according to Definition 1. It can be noticed from (21) that it is impossible to go from $p(t) = a$ and $v(t) = b$ to an arbitrary point at time $t+1$. It is therefore not possible to have any value of $\overline{\mathbf{x}}_f$ as Definition 1 requires.

If the position or the velocity is considered as the output, i.e., $n_y = 1$, then the necessary condition for output controllability from Theorem 4 is satisfied, hence the system can be output controllable. In order to establish if the system is output controllable it is necessary to check the rank of the matrix $\mathcal{C}\mathbf{S}_{\mathbf{xu}}$. It turns out that the system is output controllable in both cases.

If instead Euler sampling is used to discretise (20), then the discrete-time model becomes

$$\mathbf{x}(t+1) = \begin{pmatrix} 1 & T_s \\ 0 & 1 \end{pmatrix} \mathbf{x}(t) + \begin{pmatrix} 0 \\ T_s \end{pmatrix} u(t). \tag{23}$$

If the position or the velocity is measured, then the necessary condition for output controllability from Theorem 4 is still satisfied. However, considering the position as output gives that the first row in $\mathcal{C}\mathbf{S}_{\mathbf{xu}}$ is equal to zero because of the zero element in $\mathbf{B}_{\mathbf{u}}$, hence the rank condition for $\mathcal{C}\mathbf{S}_{\mathbf{xu}}$ is not satisfied. It means that the control signal does not affect the position directly. Instead the position is affected indirectly via the velocity. Hence, from the explanation above, it follows that the state space model in the time domain can require up to n_x time steps before it can reach the reference trajectory, which is the reason for the system in the iteration domain not being output controllable.

For many practical applications, such as an industrial robot, it is not only the position that is of importance but also the velocity needs to follow a predefined trajectory in order to achieve a satisfactory result.

From this discussion, it follows that the assumption of $\mathbf{S}_{\mathbf{u}}$ having full row rank is too restrictive to be of practical value. Instead of requiring standard controllability it is necessary to check if the system can follow a trajectory, which is discussed in Section 5.

5 Target Path Controllability

Output controllability concerns the possibility to reach a desired output at a specific time. For ILC it is of interest to reach a desired trajectory, in as few steps as possible, and then be able to follow that trajectory, hence it is more interesting to use the concept of *target path controllability* (TPC) [Engwerda, 1988]. Target path controllability is used to investigate if it is possible to track any given reference trajectory over some time interval for any initial state. A formal definition can be found in Definition 3, where $\overline{\mathbf{r}}(l, m)$ symbolises the vectors $\mathbf{r}(t)$ for $t = l, \dots, m$ stacked on top of each other. The same notation holds for $\overline{\mathbf{u}}(l, m)$.

Definition 3 (Target path controllability [Engwerda, 1988]). Let p and q be positive integers. Then a linear time-varying system is said to be target path controllable at t_0 with lead p and lag q, if for any initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ and for any reference output trajectory $\overline{\mathbf{r}}(t_0 + p, t_0 + p + q - 1)$, there exists a control sequence $\overline{\mathbf{u}}(t_0, t_0 + p + q - 2)$ such that $\mathbf{y}(t) = \mathbf{r}(t)$ for all $t_0 + p \leq t \leq t_0 + p + q - 1.$

The target path controllability will be abbreviated as TPC$(t_0; p, q)$. For $q = \infty$ the system is said to be globally TPC at t_0 with lead p. In this paper only LTI systems are considered, therefore the starting time $t_0 = 0$ is used without loss of generality, and TPC$(p, q) \overset{\Delta}{=}$ TPC$(0; p, q)$.

In Engwerda [1988] several results are presented to guarantee a system to be TPC. These results are based on different subspaces defined using the system matrices and will not be presented here. However, a rank condition of a certain matrix will be presented. The condition is similar to what is used for standard output controllability.

Theorem 5. *A linear time-invariant system is* TPC(p, q) *if and only if*

$$\operatorname{rank} \mathbf{S_{yu}}(p, q) = q n_y,$$

where

$$\mathbf{S_{yu}}(p, q) = \begin{pmatrix} \mathbf{C}\mathbf{A}^{p-1}\mathbf{B_u} & \cdots & \mathbf{C}\mathbf{B_u} & \cdots & \mathbf{0} \\ \vdots & \ddots & & \ddots & \vdots \\ \mathbf{C}\mathbf{A}^{p+q-2}\mathbf{B_u} & & \cdots & & \mathbf{C}\mathbf{B_u} \end{pmatrix}.$$

Proof: The result follows directly from [Engwerda, 1988, Lemma 8] using the LTI model in (3). □

Remark 2. Let $p = n_x$ and $q = 1$, then $\mathbf{S_{yu}}(n_x, 1) = \mathscr{C}^o$ for the system in (3), i.e., the standard output controllability matrix is obtained.

The connection between TPC and output controllability is presented in Theorem 6.

Theorem 6. *Output controllability of the system in* (8) *is equivalent to the system in* (3) *being* TPC$(1, N)$.

Proof: From Theorem 3 it holds that the system in (8) is output controllable if and only if $\operatorname{rank} \mathcal{C}\mathbf{S_{xu}} = N n_y$. Using $p = 1$ and $q = N$ gives $\mathbf{S_{yu}}(1, N) = \mathcal{C}\mathbf{S_{xu}}$ and from Theorem 5 it follows that the system is TPC$(1, N)$ if and only if $\operatorname{rank} \mathbf{S_{yu}}(1, N) = N n_y$. Hence, the two properties are equivalent. □

Return to the example in Section 4 for the case where Euler sampling has been used and the position is the output. It was shown that $\mathcal{C}\mathbf{S_{xu}}$ did not have full row rank due to the zero element in $\mathbf{B_u}$. However, removing the first row with only zeros in $\mathcal{C}\mathbf{S_{xu}}$ gives the matrix $\mathbf{S_{yu}}(2, N-1)$. The conditions in Theorem 5 are now satisfied, hence the system is TPC with lead 2.

Theorem 5 states a requirement for the system to be TPC. Another important question, is it possible to track a given predefined reference trajectory? Even if the system is TPC it can exist reference trajectories that cannot be tracked. Theorem 7 presents a necessary and sufficient condition for reference trajectories that are possible to track, which basically states that the reference should lie in the range space of $\mathbf{S_{yu}}(p, q)$.

Theorem 7 (Strongly admissible reference trajectory [Engwerda, 1988, Theorem 8]). *A reference trajectory* $\bar{\mathbf{r}}(p, p+q-1)$ *is strongly admissible if and only if*

$$\operatorname{rank} \begin{pmatrix} \mathbf{S_{yu}}(p, q) & \bar{\mathbf{z}}(p, p+q-1) \end{pmatrix} = \operatorname{rank} \mathbf{S_{yu}}(p, q) \tag{24}$$

where $\mathbf{z}(i) = \mathbf{r}(i) - \mathbf{C}\mathbf{A}^i\mathbf{x}(0)$ *for* $i = p, \ldots, p+q-1$.

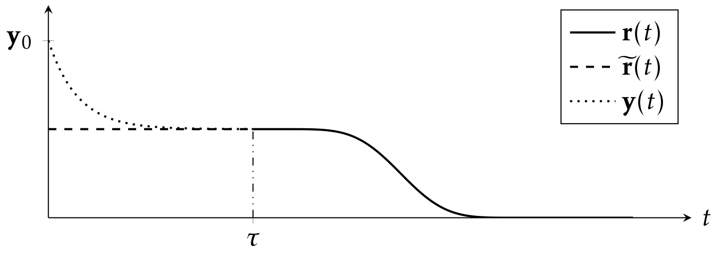


Figure 2: Augmentation of the reference trajectory to include lead-in. The original reference is denoted by $\mathbf{r}(t)$ *and* $\widetilde{\mathbf{r}}(t)$ *is the appended trajectory. The transient of the output* $\mathbf{y}(t)$ *is also shown.*

The reader is referred to Engwerda [1988] for a thorough description of admissible references and how to generate them.

6 Concept of Lead-in

Target path controllability can now be used to investigate after how many samples it is possible to track the reference, and during how many samples the reference can be tracked. It comes now naturally to define the concept of lead-in. Lead-in means that the starting point of the original reference trajectory $\mathbf{r}(t)$ is moved τ samples forward in time by appending the reference with a new initial part $\widetilde{\mathbf{r}}(t)$, see Figure 2. Note that $\tau \geq p$ in order to fulfil the requirements for TPC. The output now follows the new reference signal, see Figure 2. The assumption of the system being TPC with lead $p \leq \tau$ means that the system should be able to follow the original reference $\mathbf{r}(t)$. The error in the beginning, i.e., $\widetilde{\mathbf{r}}(t) - \mathbf{y}(t)$ for $t \leq \tau$, does not matter since the aim is to follow $\mathbf{r}(t)$. The new initial part $\widetilde{\mathbf{r}}(t)$ of the reference trajectory must of course be chosen carefully, e.g. using Theorem 7, to be able to track the remaining trajectory $\mathbf{r}(t)$.

Lead-in may not always be possible to use in practice. If the application and the trajectory do not permit to append $\widetilde{\mathbf{r}}(t)$, then lead-in cannot be used. In that case, an update of the initial state must be used to get closer to the reference signal. Being able to change the initial state while applying the philosophy of ILC can help to decrease the initial error in an iterative manner.

7 Observability

Given a state space description of the system, such as (8), it is natural to consider estimation of the state, using for example a Kalman filter. Intuitively, this would enable smoothing in the time domain, within each batch, for a fixed iteration. Dual to the controllability property, the observability becomes important when considering state observer design.

An LTI system is observable if and only if the rank of the observability matrix $\mathscr{O}$ is equal to the state dimension [Rugh, 1996]. A condition for observability of the system in (8) is presented in Theorem 8.

Theorem 8. *System* (8) *in the iteration domain is observable if and only if*

$$\operatorname{rank} \mathbf{C} = n_x.$$

Proof: The observability matrix for the system in (8) is given by $\boldsymbol{\mathcal{C}}$ repeated N times, i.e.,

$$\mathscr{O} = \begin{pmatrix} \boldsymbol{\mathcal{C}}^{\mathsf{T}} & \cdots & \boldsymbol{\mathcal{C}}^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}}. \tag{25}$$

The system is therefore observable if and only if

$$\operatorname{rank} \mathscr{O} = \operatorname{rank} \boldsymbol{\mathcal{C}} = N n_x. \tag{26}$$

From the structure of $\boldsymbol{\mathcal{C}}$ it follows that $\operatorname{rank} \boldsymbol{\mathcal{C}} = N \operatorname{rank} \mathbf{C}$. Hence, the system is observable if and only if $\operatorname{rank} \mathbf{C} = n_x$. □

Corollary 2. *A necessary condition for system* (8) *to be observable is that* $n_y \geq n_x$.

Proof: It is given that $\mathbf{C} \in \mathbb{R}^{n_y \times n_x}$, hence $\operatorname{rank} \mathbf{C} \leq \min\left\{n_y, n_x\right\}$. It is therefore necessary to have $n_y \geq n_x$ to be able to obtain $\operatorname{rank} \mathbf{C} = n_x$. □

In practice, the number of measurements is usually less than the number of states. Therefore, the system in (8) is in practice often not observable.

Remark 3. Observability of the time domain system (3), i.e.,

$$\operatorname{rank} \begin{pmatrix} \mathbf{C}^{\mathsf{T}} & (\mathbf{CA})^{\mathsf{T}} & \cdots & (\mathbf{CA}^{n_x-1})^{\mathsf{T}} \end{pmatrix}^{\mathsf{T}} = n_x, \tag{27}$$

does not imply that system (8) is observable.

8 Output Controllable System for Design of ILC Algorithms

If system (8) is output controllable, then the variable substitution $\overline{\mathbf{z}} \triangleq \overline{\mathbf{y}} = \boldsymbol{\mathcal{C}}\overline{\mathbf{x}}$ gives the new state space model

$$\overline{\mathbf{z}}_{k+1} = \overline{\mathbf{z}}_k + \mathbf{S}_{\mathbf{u}} \Delta_{\overline{\mathbf{u}}_k}, \tag{28a}$$

$$\overline{\mathbf{y}}_k = \overline{\mathbf{z}}_k, \tag{28b}$$

which is controllable by design. The ILC control law can now be found using standard discrete-time control methods such as *linear quadratic* (LQ) control with infinite horizon, $\mathcal{H}_\infty$ control, model predictive control, etcetera. For simplicity, no noise terms are included in (28), hence, there is no need for a state observer since all states are directly measured. A small simulation study of a flexible joint model with linear spring characteristic will show the advantage of using LQ-design of the ILC control law compared to the standard norm-optimal ILC method [Amann et al., 1996].

The flexible joint model in continuous-time with the state vector

$$\mathbf{x} = \begin{pmatrix} q_a & \dot{q}_a & q_m & \dot{q}_m \end{pmatrix}^{\mathsf{T}}$$

is given by

$$\dot{\mathbf{x}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{M_a} & -\frac{d}{M_a} & \frac{k}{M_a} & \frac{d}{M_a} \\ 0 & 0 & 0 & 1 \\ \frac{k}{M_m} & \frac{d}{M_m} & -\frac{k}{M_m} & -\frac{f+d}{M_m} \end{pmatrix} \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{k_\tau}{M_m} \end{pmatrix} u, \tag{29}$$

with $k = 8$, $d = 0.0924$, $M_a = 0.0997$, $M_m = 0.0525$, $f = 1.7825$ $k_\tau = 0.61$. A discrete-time model is obtained using zero order hold sampling with a sample time $T_s = 0.1\,\mathrm{s}$. The system in the iteration domain is clearly not controllable since the state dimension is larger than the number of inputs. To satisfy the requirement for output controllability it is necessary to have at most one output. The output is chosen as q_m which gives that $\mathcal{C}\mathbf{S}_{\mathbf{xu}}$ has full row rank.

The LQ-problem for the model in (28) can be formulated as

$$\begin{aligned} \underset{\Delta_{\bar{\mathbf{u}}}}{\text{minimise}} \quad & \sum_{i=1}^{\infty} \bar{\mathbf{e}}_i^{\mathsf{T}} \mathcal{W}_{\mathbf{e}} \bar{\mathbf{e}}_i + \Delta_{\bar{\mathbf{u}}_i}^{\mathsf{T}} \mathcal{W}_{\Delta} \Delta_{\bar{\mathbf{u}}_i} \\ \text{subject to} \quad & \bar{\mathbf{e}}_i = \bar{\mathbf{r}} - \bar{\mathbf{y}}_i, \\ & (28), \end{aligned} \tag{30}$$

which can be solved using standard methods [Franklin et al., 1998].

Remark 4. Since only the output is considered in the LQ-formulation it is important to know that the remaining original states do not cause any problems. It can be concluded that the remaining states will behave properly due to observability of the original state space model in the time domain.

The norm-optimal ILC from Amann et al. [1996]; Gunnarsson and Norrlöf [2001] can be formulated using the batch vectors as

$$\underset{\bar{\mathbf{u}}_{k+1}}{\text{minimise}} \quad \frac{1}{2} \bar{\mathbf{e}}_{k+1}^{\mathsf{T}} \mathcal{W}_{\mathbf{e}} \bar{\mathbf{e}}_{k+1} + \bar{\mathbf{u}}_{k+1}^{\mathsf{T}} \mathcal{W}_{\mathbf{u}} \bar{\mathbf{u}}_{k+1} + \lambda (\bar{\mathbf{u}}_{k+1} - \bar{\mathbf{u}}_k)^{\mathsf{T}} (\bar{\mathbf{u}}_{k+1} - \bar{\mathbf{u}}_k) \tag{31}$$

and the solution can be found in Gunnarsson and Norrlöf [2001]. To obtain a fair comparison the tuning of the two design methods must be similar. Comparing (31) and (30) gives that the control weights should be chosen as $\mathcal{W}_{\mathbf{u}} = \mathbf{0}$ and $\mathcal{W}_{\Delta} = \lambda \mathbf{I}$. Numerical values of the weight matrices are $\mathcal{W}_{\mathbf{e}} = \mathbf{I}$ and $\lambda = 1$. The reference signal is a unit step filtered four times through an FIR filter of length $n = 100$ with all coefficients equal to $1/n$. The performance of the two ILC algorithms is evaluated using the relative reduction of the RMSE in percent with

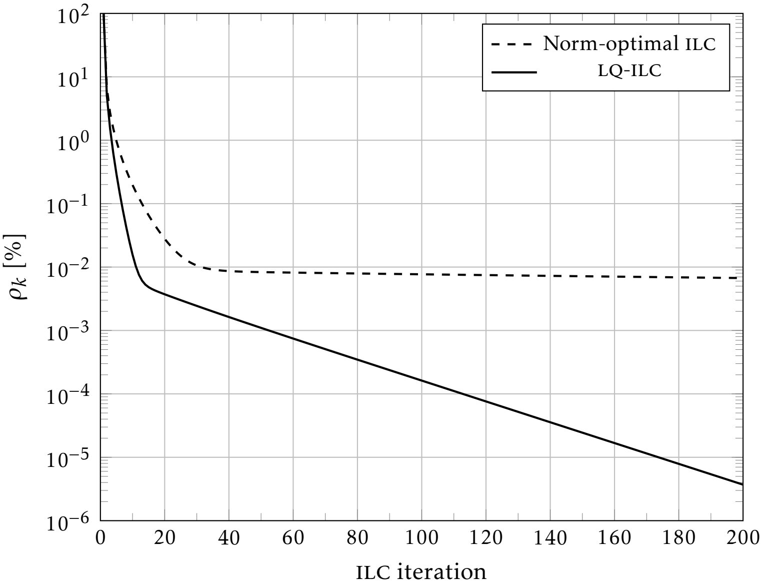


Figure 3: *Relative reduction of the RMSE for the norm-optimal ILC algorithm and the LQ-ILC algorithm.*

respect to the error when no ILC signal is applied, i.e.,

$$\rho_k = 100\sqrt{\frac{1}{N}\sum_{t=1}^{N} e_k(t)^2} \Bigg/ \sqrt{\frac{1}{N}\sum_{t=1}^{N} e_0(t)^2}. \tag{32}$$

The relative reduction of the RMSE is shown in Figure 3 for the two ILC algorithms. It can be seen that the error for the norm-optimal ILC levels out whereas the error for the LQ-ILC continues to decrease. The convergence speed is also faster for the LQ-ILC. The simulation study shows that it can be worth to check output controllability of the batch system and then use LQ-design instead of the standard norm-optimal ILC algorithm. However, if the system is not output controllable then the norm-optimal ILC control law must be used.

Remark 5. By introducing a terminal cost term $\|\bar{\mathbf{e}}_{k+2}\|_{\mathbf{P}}$ in the objective function for the norm-optimal ILC, and letting $\mathbf{P}$ be the solution to a suitable Riccati equation, the norm-optimal ILC turns out to be equivalent to the LQ-controller. This follows from the principal of optimality and dynamic programming.

9 Conclusions

This paper introduces a state space model in the iteration domain and discusses what it means that the state space model is (output) controllable. It is shown that the condition to guarantee output controllability is equivalent to a condition used in the literature to prove that the error tends to zero for the complete batch. Furthermore, it is discussed what it means to have a general system (output) controllable. A system with two states, position and velocity, is used to exemplify this. For systems that are not controllable it is more suitable to use target path controllability. This leads to the concept of lead-in where the first part of the reference trajectory is not considered for perfect tracking performance. Finally, LQ design of the ILC law for an output controllable system is compared to the standard norm-optimal ILC law. It is shown that the LQ design outperforms the norm-optimal ILC law.

Appendix

A State Space Model in Batch Form

The discrete-time state space model

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}_{\mathbf{u}}\mathbf{u}(t) + \mathbf{B}_{\mathbf{r}}\mathbf{r}(t) + \mathbf{B}_{\mathbf{w}}\mathbf{w}(t), \tag{33a}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{v}(t), \tag{33b}$$

where $\mathbf{w}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the process noise and $\mathbf{v}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the measurement noise, has the following update formula for the state vector [Rugh, 1996]

$$\mathbf{x}(t) = \mathbf{A}^t\mathbf{x}(0) + \sum_{j=0}^{t-1}\mathbf{A}^{t-j-1}\mathbf{B}_{\mathbf{u}}\mathbf{u}(j) + \sum_{j=0}^{t-1}\mathbf{A}^{t-j-1}\mathbf{B}_{\mathbf{r}}\mathbf{r}(j) + \sum_{j=0}^{t-1}\mathbf{A}^{t-j-1}\mathbf{B}_{\mathbf{w}}\mathbf{w}(j), \tag{34}$$

for $t \geq 1$. After introducing the vectors

$$\begin{aligned}
\overline{\mathbf{x}} &= \begin{pmatrix}\mathbf{x}(1)^\mathsf{T} & \dots & \mathbf{x}(N)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_x},\\
\overline{\mathbf{u}} &= \begin{pmatrix}\mathbf{u}(0)^\mathsf{T} & \dots & \mathbf{u}(N-1)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_u},\\
\overline{\mathbf{y}} &= \begin{pmatrix}\mathbf{y}(1)^\mathsf{T} & \dots & \mathbf{y}(N)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_y},\\
\overline{\mathbf{r}} &= \begin{pmatrix}\mathbf{r}(0)^\mathsf{T} & \dots & \mathbf{r}(N-1)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_y},\\
\overline{\mathbf{w}} &= \begin{pmatrix}\mathbf{w}(0)^\mathsf{T} & \dots & \mathbf{w}(N-1)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_w},\\
\overline{\mathbf{v}} &= \begin{pmatrix}\mathbf{v}(1)^\mathsf{T} & \dots & \mathbf{v}(N)^\mathsf{T}\end{pmatrix}^\mathsf{T} \in \mathbb{R}^{Nn_v},
\end{aligned}$$

the model in (33) can be written more compactly for a batch of length N as

$$\overline{\mathbf{x}} = \mathbf{\Phi}\mathbf{x}(0) + \mathbf{S}_{\mathbf{xu}}\overline{\mathbf{u}} + \mathbf{S}_{\mathbf{xr}}\overline{\mathbf{r}} + \mathbf{S}_{\mathbf{xw}}\overline{\mathbf{w}} \tag{35a}$$

$$\overline{\mathbf{y}} = \boldsymbol{\mathcal{C}}\overline{\mathbf{x}} + \overline{\mathbf{v}} \tag{35b}$$

where $\mathbf{x}(0)$ is the initial value, $\boldsymbol{\mathcal{C}} = \mathbf{I}_N \otimes \mathbf{C}$, and

$$\boldsymbol{\Phi} = \begin{pmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^N \end{pmatrix}, \quad \mathbf{S}_{\mathbf{x}*} = \begin{pmatrix} \mathbf{B}_* & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}\mathbf{B}_* & \mathbf{B}_* & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{N-1}\mathbf{B}_* & \mathbf{A}^{N-2}\mathbf{B}_* & \cdots & \mathbf{B}_* \end{pmatrix}, \tag{36}$$

where $* = \{\mathbf{u}, \mathbf{r}, \mathbf{w}\}$. The process and measurement noise for (35) are $\overline{\mathbf{w}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{Q}})$ and $\overline{\mathbf{v}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{R}})$, where $\boldsymbol{\mathcal{Q}} = \mathbf{I}_N \otimes \mathbf{Q}$ and $\boldsymbol{\mathcal{R}} = \mathbf{I}_N \otimes \mathbf{R}$.

Acknowledgement

This work was supported by the Vinnova Excellence Center LINK-SIC and by the Excellence Center at Linköping-Lund in Information Technology, ELLIIT.

Bibliography

Notker Amann, David H. Owens, and Eric Rogers. Iterative learning control for discrete-time systems with exponential rate of convergence. *IEE Proceedings Control Theory and Applications*, 143(2):217–224, March 1996.

Suguru Arimoto. Learning control theory for robotic motion. *International Journal of Adaptive Control and Signal Processing*, 4(6):543–564, 1990.

Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2):123–140, 1984.

Patrik Axelsson, Daniel Axehill, Torkel Glad, and Mikael Norrlöf. Controllability aspects for iterative learning control. *Submitted to International Journal of Control*, 2014.

Jacob C. Engwerda. Control aspects of linear discrete time-varying systems. *International Journal of Control*, 48(4):1631–1658, 1988.

Gene F. Franklin, J. David Powell, and Michael L. Workman. *Digital Control of Dynamic Systems*. Addison Wesley, Menlo Park, CA, USA, third edition, 1998.

Svante Gunnarsson and Mikael Norrlöf. On the design of ILC algorithms using optimization. *Automatica*, 37(12):2011–2016, December 2001.

Jay H. Lee, Kwang S. Lee, and Won C. Kim. Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica*, 36(5): 641–657, May 2000.

Kwang S. Lee and Jay H. Lee. Convergence of constrained model-based predictive control for batch processes. *IEEE Transactions on Automatic Control*, 45(10): 1928–1932, October 2000.

Kwang Soon Lee and Jay H. Lee. *Iterative Learning Control: Analysis, Design, Integration and Application*, chapter Design of Quadratic Criterion-based Iterative Learning Control, pages 165–192. Kluwer Academic Publishers, Boston, MA, USA, 1998.

Helmut Lütkepohl. *Handbook of Matrices*. John Wiley & Sons, Chichester, West Sussex, England, 1996.

Kevin L. Moore. *Iterative Learning Control for Deterministic Systems*. Advances in Industrial Control. Springer-Verlag, London, UK, 1993.

Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall Inc., Upper Saddle River, NJ, USA, fourth edition, 2002.

Wilson J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1996.

Johanna Wallén, Isolde Dressler, Anders Robertsson, Mikael Norrlöf, and Svante Gunnarsson. Observer-based ILC applied to the Gantry-Tau parallel kinematic robot. In *Proceedings of the 18th IFAC World Congress*, pages 992–998, Milano, Italy, August/September 2011.

PhD Dissertations
Division of Automatic Control
Linköping University

M. Millnert: Identification and control of systems subject to abrupt changes. Thesis No. 82, 1982. ISBN 91-7372-542-0.

A. J. M. van Overbeek: On-line structure selection for the identification of multivariable systems. Thesis No. 86, 1982. ISBN 91-7372-586-2.

B. Bengtsson: On some control problems for queues. Thesis No. 87, 1982. ISBN 91-7372-593-5.

S. Ljung: Fast algorithms for integral equations and least squares identification problems. Thesis No. 93, 1983. ISBN 91-7372-641-9.

H. Jonson: A Newton method for solving non-linear optimal control problems with general constraints. Thesis No. 104, 1983. ISBN 91-7372-718-0.

E. Trulsson: Adaptive control based on explicit criterion minimization. Thesis No. 106, 1983. ISBN 91-7372-728-8.

K. Nordström: Uncertainty, robustness and sensitivity reduction in the design of single input control systems. Thesis No. 162, 1987. ISBN 91-7870-170-8.

B. Wahlberg: On the identification and approximation of linear systems. Thesis No. 163, 1987. ISBN 91-7870-175-9.

S. Gunnarsson: Frequency domain aspects of modeling and control in adaptive systems. Thesis No. 194, 1988. ISBN 91-7870-380-8.

A. Isaksson: On system identification in one and two dimensions with signal processing applications. Thesis No. 196, 1988. ISBN 91-7870-383-2.

M. Viberg: Subspace fitting concepts in sensor array processing. Thesis No. 217, 1989. ISBN 91-7870-529-0.

K. Forsman: Constructive commutative algebra in nonlinear control theory. Thesis No. 261, 1991. ISBN 91-7870-827-3.

F. Gustafsson: Estimation of discrete parameters in linear systems. Thesis No. 271, 1992. ISBN 91-7870-876-1.

P. Nagy: Tools for knowledge-based signal processing with applications to system identification. Thesis No. 280, 1992. ISBN 91-7870-962-8.

T. Svensson: Mathematical tools and software for analysis and design of nonlinear control systems. Thesis No. 285, 1992. ISBN 91-7870-989-X.

S. Andersson: On dimension reduction in sensor array signal processing. Thesis No. 290, 1992. ISBN 91-7871-015-4.

H. Hjalmarsson: Aspects on incomplete modeling in system identification. Thesis No. 298, 1993. ISBN 91-7871-070-7.

I. Klein: Automatic synthesis of sequential control schemes. Thesis No. 305, 1993. ISBN 91-7871-090-1.

J.-E. Strömberg: A mode switching modelling philosophy. Thesis No. 353, 1994. ISBN 91-7871-430-3.

K. Wang Chen: Transformation and symbolic calculations in filtering and control. Thesis No. 361, 1994. ISBN 91-7871-467-2.

T. McKelvey: Identification of state-space models from time and frequency data. Thesis No. 380, 1995. ISBN 91-7871-531-8.

J. Sjöberg: Non-linear system identification with neural networks. Thesis No. 381, 1995. ISBN 91-7871-534-2.

R. Germundsson: Symbolic systems – theory, computation and applications. Thesis No. 389, 1995. ISBN 91-7871-578-4.

P. Pucar: Modeling and segmentation using multiple models. Thesis No. 405, 1995. ISBN 91-7871-627-6.

H. Fortell: Algebraic approaches to normal forms and zero dynamics. Thesis No. 407, 1995. ISBN 91-7871-629-2.

A. Helmersson: Methods for robust gain scheduling. Thesis No. 406, 1995. ISBN 91-7871-628-4.

P. Lindskog: Methods, algorithms and tools for system identification based on prior knowledge. Thesis No. 436, 1996. ISBN 91-7871-424-8.

J. Gunnarsson: Symbolic methods and tools for discrete event dynamic systems. Thesis No. 477, 1997. ISBN 91-7871-917-8.

M. Jirstrand: Constructive methods for inequality constraints in control. Thesis No. 527, 1998. ISBN 91-7219-187-2.

U. Forssell: Closed-loop identification: Methods, theory, and applications. Thesis No. 566, 1999. ISBN 91-7219-432-4.

A. Stenman: Model on demand: Algorithms, analysis and applications. Thesis No. 571, 1999. ISBN 91-7219-450-2.

N. Bergman: Recursive Bayesian estimation: Navigation and tracking applications. Thesis No. 579, 1999. ISBN 91-7219-473-1.

K. Edström: Switched bond graphs: Simulation and analysis. Thesis No. 586, 1999. ISBN 91-7219-493-6.

M. Larsson: Behavioral and structural model based approaches to discrete diagnosis. Thesis No. 608, 1999. ISBN 91-7219-615-5.

F. Gunnarsson: Power control in cellular radio systems: Analysis, design and estimation. Thesis No. 623, 2000. ISBN 91-7219-689-0.

V. Einarsson: Model checking methods for mode switching systems. Thesis No. 652, 2000. ISBN 91-7219-836-2.

M. Norrlöf: Iterative learning control: Analysis, design, and experiments. Thesis No. 653, 2000. ISBN 91-7219-837-0.

F. Tjärnström: Variance expressions and model reduction in system identification. Thesis No. 730, 2002. ISBN 91-7373-253-2.

J. Löfberg: Minimax approaches to robust model predictive control. Thesis No. 812, 2003. ISBN 91-7373-622-8.

J. Roll: Local and piecewise affine approaches to system identification. Thesis No. 802, 2003. ISBN 91-7373-608-2.

J. Elbornsson: Analysis, estimation and compensation of mismatch effects in A/D converters. Thesis No. 811, 2003. ISBN 91-7373-621-X.

O. Härkegård: Backstepping and control allocation with applications to flight control. Thesis No. 820, 2003. ISBN 91-7373-647-3.

R. Wallin: Optimization algorithms for system analysis and identification. Thesis No. 919, 2004. ISBN 91-85297-19-4.

D. Lindgren: Projection methods for classification and identification. Thesis No. 915, 2005. ISBN 91-85297-06-2.

R. Karlsson: Particle Filtering for Positioning and Tracking Applications. Thesis No. 924, 2005. ISBN 91-85297-34-8.

J. Jansson: Collision Avoidance Theory with Applications to Automotive Collision Mitigation. Thesis No. 950, 2005. ISBN 91-85299-45-6.

E. Geijer Lundin: Uplink Load in CDMA Cellular Radio Systems. Thesis No. 977, 2005. ISBN 91-85457-49-3.

M. Enqvist: Linear Models of Nonlinear Systems. Thesis No. 985, 2005. ISBN 91-85457-64-7.

T. B. Schön: Estimation of Nonlinear Dynamic Systems — Theory and Applications. Thesis No. 998, 2006. ISBN 91-85497-03-7.

I. Lind: Regressor and Structure Selection — Uses of ANOVA in System Identification. Thesis No. 1012, 2006. ISBN 91-85523-98-4.

J. Gillberg: Frequency Domain Identification of Continuous-Time Systems Reconstruction and Robustness. Thesis No. 1031, 2006. ISBN 91-85523-34-8.

M. Gerdin: Identification and Estimation for Models Described by Differential-Algebraic Equations. Thesis No. 1046, 2006. ISBN 91-85643-87-4.

C. Grönwall: Ground Object Recognition using Laser Radar Data – Geometric Fitting, Performance Analysis, and Applications. Thesis No. 1055, 2006. ISBN 91-85643-53-X.

A. Eidehall: Tracking and threat assessment for automotive collision avoidance. Thesis No. 1066, 2007. ISBN 91-85643-10-6.

F. Eng: Non-Uniform Sampling in Statistical Signal Processing. Thesis No. 1082, 2007. ISBN 978-91-85715-49-7.

E. Wernholt: Multivariable Frequency-Domain Identification of Industrial Robots. Thesis No. 1138, 2007. ISBN 978-91-85895-72-4.

D. Axehill: Integer Quadratic Programming for Control and Communication. Thesis No. 1158, 2008. ISBN 978-91-85523-03-0.

G. Hendeby: Performance and Implementation Aspects of Nonlinear Filtering. Thesis No. 1161, 2008. ISBN 978-91-7393-979-9.

J. Sjöberg: Optimal Control and Model Reduction of Nonlinear DAE Models. Thesis No. 1166, 2008. ISBN 978-91-7393-964-5.

D. Törnqvist: Estimation and Detection with Applications to Navigation. Thesis No. 1216, 2008. ISBN 978-91-7393-785-6.

P-J. Nordlund: Efficient Estimation and Detection Methods for Airborne Applications. Thesis No. 1231, 2008. ISBN 978-91-7393-720-7.

H. Tidefelt: Differential-algebraic equations and matrix-valued singular perturbation. Thesis No. 1292, 2009. ISBN 978-91-7393-479-4.

H. Ohlsson: Regularization for Sparseness and Smoothness — Applications in System Identification and Signal Processing. Thesis No. 1351, 2010. ISBN 978-91-7393-287-5.

S. Moberg: Modeling and Control of Flexible Manipulators. Thesis No. 1349, 2010. ISBN 978-91-7393-289-9.

J. Wallén: Estimation-based iterative learning control. Thesis No. 1358, 2011. ISBN 978-91-7393-255-4.

J. Hol: Sensor Fusion and Calibration of Inertial Sensors, Vision, Ultra-Wideband and GPS. Thesis No. 1368, 2011. ISBN 978-91-7393-197-7.

D. Ankelhed: On the Design of Low Order H-infinity Controllers. Thesis No. 1371, 2011. ISBN 978-91-7393-157-1.

C. Lundquist: Sensor Fusion for Automotive Applications. Thesis No. 1409, 2011. ISBN 978-91-7393-023-9.

P. Skoglar: Tracking and Planning for Surveillance Applications. Thesis No. 1432, 2012. ISBN 978-91-7519-941-2.

K. Granström: Extended target tracking using PHD filters. Thesis No. 1476, 2012. ISBN 978-91-7519-796-8.

C. Lyzell: Structural Reformulations in System Identification. Thesis No. 1475, 2012. ISBN 978-91-7519-800-2.

J. Callmer: Autonomous Localization in Unknown Environments. Thesis No. 1520, 2013. ISBN 978-91-7519-620-6.

D. Petersson: A Nonlinear Optimization Approach to H2-Optimal Modeling and Control. Thesis No. 1528, 2013. ISBN 978-91-7519-567-4.

Z. Sjanic: Navigation and Mapping for Aerial Vehicles Based on Inertial and Imaging Sensors. Thesis No. 1533, 2013. ISBN 978-91-7519-553-7.

F. Lindsten: Particle Filters and Markov Chains for Learning of Dynamical Systems. Thesis No. 1530, 2013. ISBN 978-91-7519-559-9.