Prediction of position errors for an industrial robot, using a model of the robot with parameters acquired from the clamping procedure

Anders Söder-Hoorn

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
ISRN LUTFD2/TFRT--5991--SE
ISSN 0280-5316

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

Printed in Sweden by Tryckeriet i E-huset
Lund 2015

Abstract

A robot should follow a given path as accurately as possible. There are however almost always deviations from the desired path, and when the deviations become too large it may be a problem. Some of the deviations stem from transmissions of the robot, and the deviations become more pronounced when there are process forces and gravitational forces present, that affect the transmissions. When position is measured and controlled on the motor side, characteristics of the transmission are not accounted for in the control of the robot, resulting in deviations from the desired path. There are also deviations from the desired path that stem from links of the robot, due to process forces and gravitational forces that affect the links.

To predict the deviations that stem from the transmissions, models of the transmissions were developed and used. Models that should be able to predict the deviations that stem from the links were also developed. To acquire data about the characteristics of the transmissions of a robot, tailored experiments were performed. To acquire the data, the robot's end-effector was locked to a stiff point in space, the robot's motors were run, and the robot's sensors were used to log the data. This procedure is known as a clamping procedure. The collected data were processed to derive parameters that were used in the models of the transmissions. The robot that was used was a KR 300 R2500 ULTRA, which is an industrial robot with six degrees of freedom.

Simulations with the models of the transmissions were able to recreate the data from the clamping procedure with great accuracy. This shows that the models of the transmissions, with the parameters derived from the clamping procedure, capture characteristics of the robot, that are not taken into account in the control of the robot, which leads to deviations from the desired path.

The next step, that is not a part of this thesis, is to use a model of the whole robot, that contains the models of the transmissions and the links, and to validate that the robot model can recreate the deviations from a given path that the robot takes.

Acknowledgements

First of all, I want to thank my supervisors, Mathias Haage at Cognibotics, Klas Nilsson at Cognibotics, Anders Robertsson at the Department of Automatic Control at Lund Unversity, and my examinator Rolf Johansson at the Department of Automatic Control at Lund Unversity. Their input and guidance through this thesis has been valuable. I also want to thank Tommy Olofsson at Cognibotics, who wrote the robot program for the data acquisition, and Björn Olofsson at Cognibotics, for his input regarding the Modelica language. I wish to thank Jens Osmer and the employees at Machining Innovations Network in Varel, Germany, for the access to an industrial robot and for their help during the data acquisition. Lastly, I wish to thank the employees at Premium Aerotec in Varel, Germany, for the construction of the clamping tool that made the data acquisition possible.

Anders Söder-Hoorn
Lund, August 2015

Contents

1

Introduction

1.1 Background

Industrial robots are cheaper and more flexible compared to CNC machines that are widely used in industry. In [2] it was shown that path deviations from the desired path is a problem for industrial robots. The reason that industrial robots can not be used in certain machining tasks where CNC machines are used, is that their accuracy is limited by low stiffness. Therefore there is a need to decrease the position errors that stem from the low stiffness of industrial robots.

There are characteristics in transmissions of robots that cause deviations from a desired path, like low stiffness, friction and backlash. These characteristics are often not taken into account in the control of the robot. Previous work done that addresses modeling of stiffness is presented in [2], [3], [15] and [6]. Earlier work that has been done to model backlash is presented in [8] and [15]. In [5] a method for measuring the joint friction is presented. In [14] and [1] there are methods presented to predict or online estimate position deviations from the desired path due to process forces. A goal is to develop a model of the joints, where stiffness, backlash and friction are included. A simulation with this model of the joints would then be able to predict the effects in the transmissions that causes the deviations. Parameters for the model of the transmissions need to be derived from experimental data. To be able to recreate the deviations from the desired path for the robot's end-effector, a model of the whole robot is needed.

1.2 Previous work done in this field

There are several approaches to increase position accuracy that address low stiffness in industrial robots. One approach is to use external sensors and external actuators to compensate for deformation in real time. In [17] and [20], an external tracking system is used to detect position errors that is due to too low stiffness and process forces. In [13] position-error compensation is performed when there are process

forces present. An external actuator is used to compensate for the position errors in real time in [17], and it is shown that the accuracy is increased up to three times.

Another approach is to use a robot's own motors to compensate for deformation due to low stiffness. The deformation can either be measured with external sensors, or estimated using a model of the robot that take the stiffness into account if the process forces are known. To be able to construct a model that can predict the position errors, data from the robot needs to be acquired. In [21], data that describe the stiffness of a robot are acquired and a model is constructed that is used to compensate for robot deformations, caused by external process forces, in real time. The robot's own motors and an external force sensor are used for the compensation. In the identification of the stiffness of the robot, an air cylinder was used to achieve an external force acting on the robot, an external force sensor to measure the external force and a portable coordinate measurement arm for measuring the deformation due to the external force. It is shown that using this approach the position errors can be reduced by more than 60%.

In [4] there was a method presented that was based on the clamping of the robot's end-effector to a stiff point in space, and the use of the robot's own motors to derive information about the robot. Clamping was also used in [2], where the joints were clamped, and an external load was applied to get information about the robot's stiffness Another method to derive data about a robot was presented in [11]. There are many advantages with the approach presented in [11], that uses clamping for data acquisition. No expensive external sensors are needed as the data is logged using the robot's own sensors. The only external equipment that is needed is a device to clamp the robot's end-effector to a stiff environment. This method for acquiring data is known as a clamping procedure. A validation of the feasibility of the method is presented in [10]. Since no external sensors are used in the clamping procedure, and the robot's own motors are used in the compensation of position errors, this is the cheapest method to increase the accuracy for industrial robots and should be used whenever possible. It can furthermore take into account the actual mounting in the work-cell and compliance in tooling, which would not be covered if only the robot would be measured at the manufacturer.

1.3 Approach

The approach in this thesis is experimental. The work in this thesis extends the work done in [10], as algorithms were developed to identify parameters from clamping data. The parameters were used in models of the transmissions. The models were then used in simulations to recreate data from experiments.

1.4 Data acquisition

Data were acquired by locking the robot's end-effector to a fixed point in space. Then the robot's motors were run and the motor positions and motor torques were logged using the robot's own sensors. The procedure to lock the robot's end-effector to a stiff point in space and run the robot's own motors, to derive information about the robot, is known as a clamping procedure. The procedure is presented in [11]. Data derived in this way are referred to as clamping data.

1.5 A transmission model

The main concern was to build a model of the transmission for one joint that was able to recreate the data acquired from the clamping procedure. That was done by using parameters identified from the clamping data. The next step was to use these models of the transmissions as building blocks when a model of a whole robot was constructed.

1.6 A robot model

A robot model was developed in Dymola, using the multibody libary of Modelica, see [12]. The model of the transmissions was developed and tested in JModelica.org, see [9], and then implemented again in Dymola.

1.7 Developed algorithms

There were two algorithms developed in order to identify the parameters for the transmission model from the clamping data. The first developed algorithm, *Algorithm I*, was not robust and did not use any information about the known structure of the clamping data. The second developed algorithm, *Algorithm II*, was more robust, and used information about the known structure of the clamping data when the identification was performed, which resulted in a better result. Matlab was used to develop the algorithms.

1.8 Versions of the tools used

Version R2014b of Matlab, version 2013 of Dymola, and version 1.15b1 of JModelica.org were used, all running on Windows 7.

2

Modelica

2.1 The Modelica language

The Modelica language is declarative, as it is used to state equalities and equations relating different state variables rather than causal declarations, which is common in simulation tools like, e.g., Matlab/Simulink. Modelica was used because it separates the models from the calculations. The models are expressed in the Modelica language, and the solver of the implementation that is used calculates the results of simulations. When models are used in simulations, the simulation engine of the implementation that is used, manipulates the equations and determines the order of execution. The equations have no predefined causality. The simulation engine that manipulates and simulates the model differs in different implementations, though the language is the same. The Modelica language requires balanced models. That means that the model, with all connected components, ends up in a set of differential, algebraic and discrete equations where the number of unknowns and the number of equations are the same.

The Modelica Language is free, and is developed by the Modelica Association [19]. Modelica is an object-oriented declarative modeling language, and is used for component-oriented modeling. Models of systems are built up and the models are translated into objects, which are run in a simulation engine. The language is designed to be used in variety of fields such as mechanical systems, fluid systems and energy systems. There are several free libraries available that contain models of components in many domains. These model components can be used to build up more complex systems, such as gearboxes with friction and backlash or electrical systems such as DC motors. The libraries with components make it possible to build up complex systems without having expert knowledge of the physics behind every part of the system. The Modelica language and the libraries make it easy to build up models of systems using the available models of the components. One can also create models by stating the physics behind them using the Modelica language. These models can then be used to build up more complex systems. There are several available modeling and simulation tools, both free and commercial, that make use of the Modelica language.

2.2 The Modelica Standard Library

The Modelica Association develops a Standard Library, which is free. The Modelica Standard Library is documented in [18]. The Standard Library contains models of components and functions in various domains, such as, e.g., the mechanical library. Both 1D and 3D models can by constructed. The 3D multibody library to model 3D structures is presented in [12].

2.3 Implementations of Modelica such as Dymola and JModelica.org

There are several implementations of the Modelica language avaliable. Dymola is a commercial tool with a graphical user interface, where systems can be built up using drag and drop programming with components from the available libraries. The graphical representation of the system represents the underlying connections of the different components used. Even though there is a graphical representation, models can by built by writing the equations of the model using the Modelica language. The Modelica code that is generated from the graphical representation can also be viewed and edited.

JModelica.org is a free tool that implements Modelica. It has no graphical user interface, and the models are built up by writing Modelica code.

2.4 FMU and the reuse of models in different tools

Models can be developed in one tool and used by the several others through an interface called the functional mock-up interface (FMI), see [7]. A model developed in one tool, such as the transmission model developed in JModelica.org, can then be translated to an FMU (Functional Mock-up Unit), that implements the FMI. The FMU can then be imported and executed in an other environment. The FMU is useful because developed models can be plugged in to existing models and other tools. There are two types of FMUs, one that just contains the model, and one that contains both the model and a solver able to perform a simulation.

3

Model development

3.1 Available models used

There were several models from the Modelica Standard Library that were used when robot joint models were implemented. Below is a short description of the models from the Modelica Standard Library that were used. For a more thorough review of the models, see [18].

Modelica.Mechanics.Rotational.Sources.Torque: translates an input signal to a torque acting on a flange.

Modelica.Mechanics.Rotational.Components.IdealGear: An ideal gear (with no friction, damping, etc.), where the gear ratio can be chosen.

Modelica.Mechanics.Rotational.Components.BearingFriction: The rotation that is led through the component is damped by a torque according to a look-up table that can be chosen, i.e., the damping torque is determined by the speed of the rotation, given in the table. The table consists of angular velocities and the corresponding damping forces for each angular velocity. The damping force is interpolated linearly between the defined points in the table. If the angular speed is larger than what the table defines, an extrapolation is performed from the last two entries in the table. Static friction is also included in this component. When the angular velocity is zero, the force that tries to rotate the component must be bigger than a certain threshold force before the rotation starts. This threshold force is defined through the friction force in the table that is given when the angular velocity is zero, multiplied with a constant that can be chosen.

Modelica.Mechanics.Rotational.Components.Inertia: A moment of inertia due to a given mass and geometry.

Modelica.Mechanics.Rotational.Components.Damper: A viscous damper that

has damping forces acting on the flanges of the model. The damping forces are proportional to the speed difference between the two flanges.

Modelica.Mechanics.Rotational.Components.ElastoBacklash: This model consists of a backlash connected in series with a linear rotational spring (*Modelica.Mechanics.Rotational.Components.spring*), and a viscous damper (*Modelica.Mechanics.Rotational.Components.Damper*).

Modelica.Mechanics.Rotational.Components.Fixed: A model that has a flange with a given angle that does not change, no matter of the forces acting on it.

Modelica.Mechanics.Multibody.World: It gives a global coordinate system fixed to the ground and defines a gravity field.

Modelica.Mechanics.Multibody.Joints.Revolute: Represents a joint with a rotation around a given axis.

Modelica.Mechanics.Multibody.Parts.BodyShape: A rigid body with mass and inertia.

3.2 A one-dimensional transmission model

A one-dimensional transmission model was implemented in JModelica.org using the Modelica Standard Library. A graphical representation of the model is shown in Figure 3.1. The model contains two rotational flanges, *flange_a* and *flange_b*, that can be connected to other rotational elements in the Modelica Standard Library. The model of the transmission is constructed by the following components; *IdealGear*, *ElastoBacklash*, *BearingFriction*, *Inertia*, *Damper* and *Fixed*. All these components are found in the Modelica Standard Library (*Modelica.Mechanics.Rotational.Components*), in [18]. The developed 1D transmission model shown in Figure 3.1 can be used as a building block in 3D mechanical structures using the multibody library, without neglecting any dynamic effects, as described in [16].

The components are connected in this way to make the model capable of recreating the clamping data during a simulation. If the robot's transmissions were ideal they could be modeled with an *IdealGear*. However, that is not the case, so the rest of the components are needed to make the model capable of recreating the clamping data. The idea is to use this model to more accurately simulate deviations from a desired path. If the simulations are able to recreate position errors, they can be used to increase the accuracy of the robot.

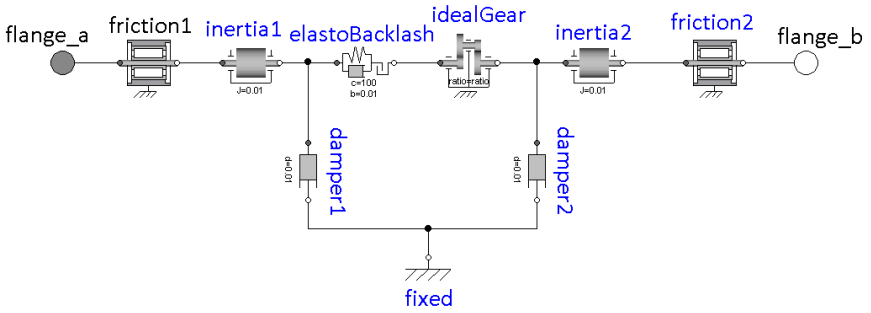

Figure 3.1 A graphical representation of the one-dimensional transmission model that was implemented in JModelica.org. This picture was generated from Dymola, since JModelica.org does not currently have any graphical user interface.

The setup for the clamping simulation

The setup for a clamping simulation with the model in Figure 3.1 is shown in Figure 3.2. In Figure 3.2 the clamping point (that was modeled as a *Fixed* element) is named *ClampingPoint*.

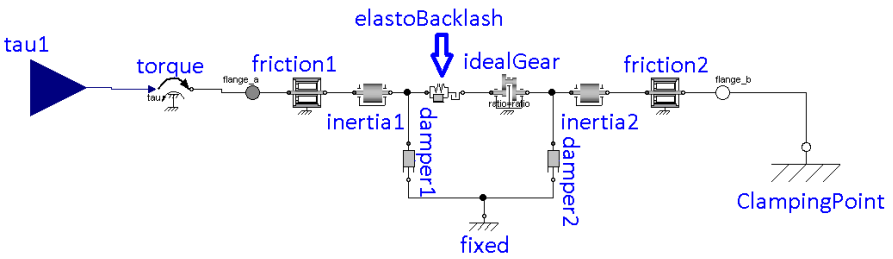

Figure 3.2 A representation of the clamped one-dimensional transmission model. It was implemented in JModelica.org, but the picture was generated from Dymola.

3.3 A multi-axes model

A model of a robot was developed. Models from the Modelica Standard Library were assembled to form models of robot components, that were used to implement

a model of the robot. The model was constructed to capture characteristics of links, such as torsion and flex. All developed models of the components are in a package named *Robot*. In Figure 3.10 a model of the robot is presented. Models presented in this chapter were implemented in Dymola, unless something else is stated.

Developed models of a mechanical structure

Robot.MechanicalStructure: A model of the mechanical structure of the robot. A graphical representation is shown in Figure 3.3. It contains models of type *Robot.Link* to model flexible links, a model of *Modelica.Mechanics.Multibody.World* to model a gravitational field and models of *Modelica.Mechanics.Multibody.Joints.Revolute* to model the robot's joints.

Robot.Link: A model of a flexible link. A graphical representation is seen in Figure 3.4. A flexible link is implemented with models from the Modelica Standard Library. It contains a model of *Robot.TorsionConnection* to model torsion, models of *Robot.FlexibleConnection* to model flex, and models of *Modelica.Mechanics.Multibody.Parts.BodyShape* to build the link.

Robot.TorsionConnection: A model was implemented to describe torsion. A graphical representation is presented in Figure 3.5. It contains the following models from the Modelica Standard Library; *Modelica.Mechanics.Multibody.Joints.Revolute* to define around which axis a rotation should by possible, *Modelica.Mechanics.Rotational.Components.Damper* to model damping in the torsion, *Modelica.Mechanics.Rotational.Components.Spring* to limit the torsion, and *Modelica.Mechanics.Rotational.Components.Fixed* to give a point for the torsion to act around.

Robot.FlexibleConnection: A model was implemented to model flex. A graphical representation is presented in Figure 3.6. It contains the following models from the Modelica Standard Library; *Modelica.Mechanics.Multibody.Joints.Revolute* to define around which axes a rotation should by possible, *Modelica.Mechanics.Rotational.Components.Damper* to model damping in the flex, *Modelica.Mechanics.Rotational.Components.Spring* to limit the flex, and *Modelica.Mechanics.Rotational.Components.Fixed* to give a point for the flex to act around.

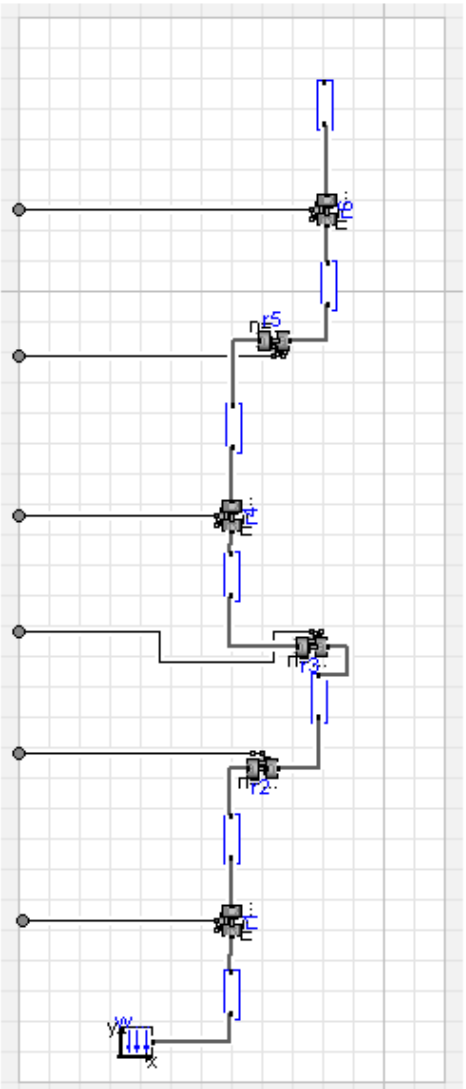

Figure 3.3 A representation of the model of *Robot.MechanicalStructure*. To the left in the Figure there are gray circular flanges that can be connected to from outside the model, to drive the joints. The blue rectangles are components of type *Robot.Link*.

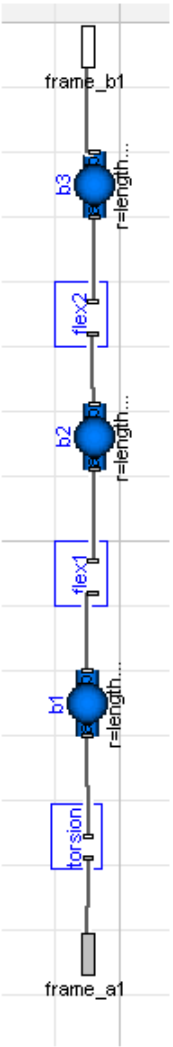

Figure 3.4 A representation of the model *Robot.Link*. The models at the top and bottom of the Figure are frames that the *Robot.Link* can be connected to from outside of the model. The blue models are of type *Modelica.Mechanics.MultiBody.Parts.-BodyShape*. The model below the bottom *BodyShape* is a model of type *Robot.-TorsionConnection*. The two models between the components of type *BodyShape* are models of type *Robot.FlexibleConnection*.

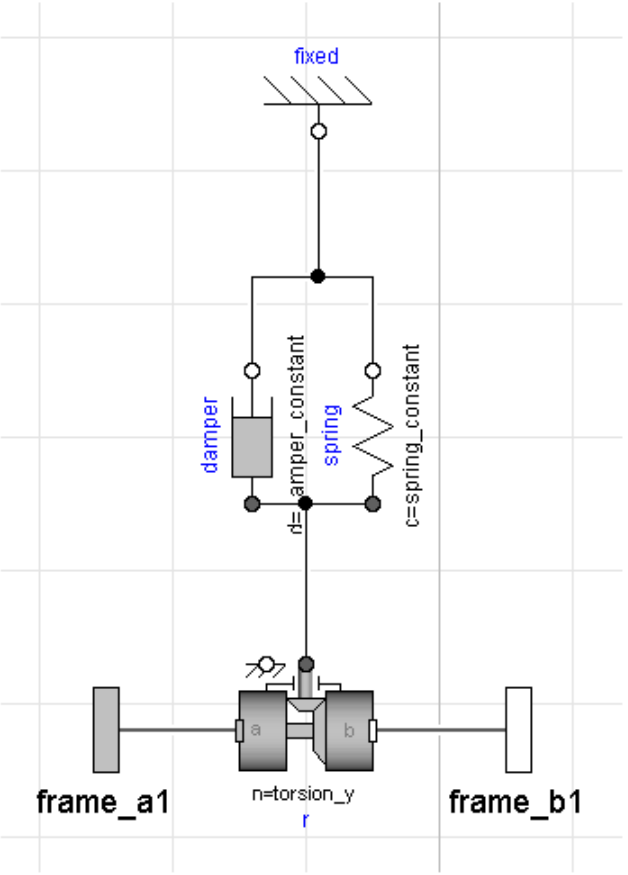

Figure 3.5 A representation of the model *Robot.TorsionConnection*. The model can be connected to the two frames from outside the model. The component named *fixed* provides a fixed point in space given in the coordinate system that *frame_a1* (of type *Modelica.Mechanics.MultiBody.Interfaces.Frame_a*) defines. *frame_a1* defines a coordinate system that is fixed to a mechanical component that is connected to from outside the model.

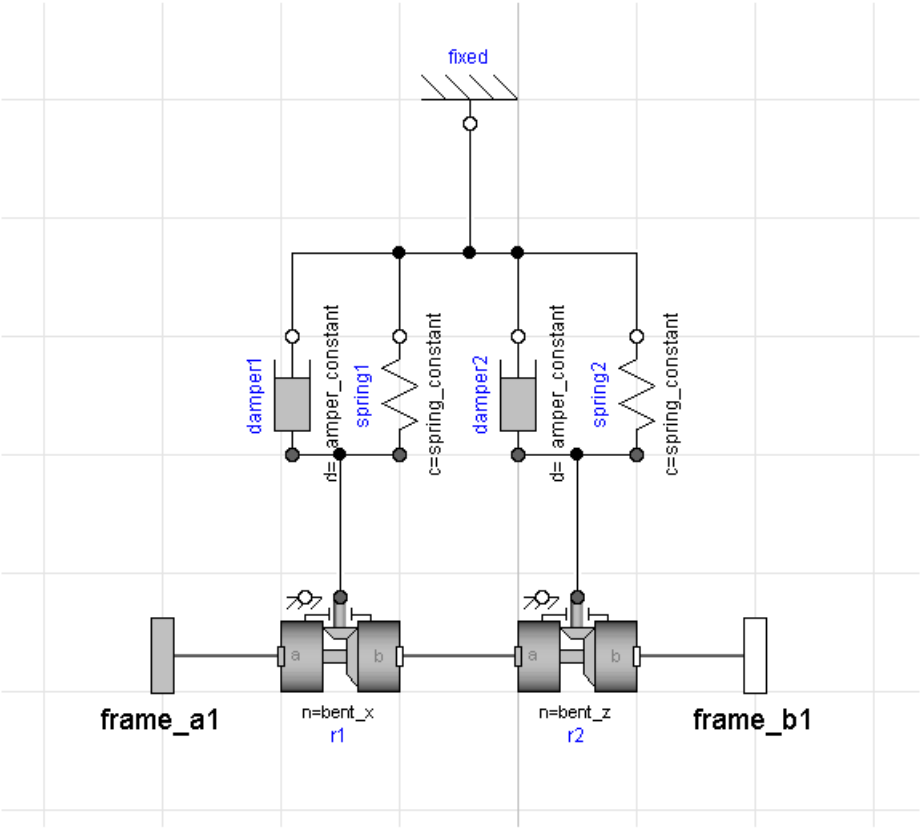

Figure 3.6 A representation of the model *Robot.FlexibleConnection*. The model can be connected to the two frames from outside the model. The component named *fixed* provides a fixed point in space given in the coordinate system that *frame_a1* (of type *Modelica.Mechanics.MultiBody.Interfaces.Frame_a*) defines. *frame_a1* defines a coordinate system that is fixed to a mechanical component that is connected to from outside the model.

A model of drive trains

Robot.Drivetrain: A model of transmissions of a robot. A graphical representation is presented in Figure 3.9. It contains the following models; *Robot.Transmission_a* to model a part of the transmission, *Robot.Multigear* to model a robot's gearbox, and a *Robot.Transmission_b* to model a part of the transmission. It has several inputs for external torques, one for each joint, as seen in Figure 3.9. These are there to be able to take known process forces into account when a simulation is performed.

Robot.Transmission_a and *Robot.Transmission_b*: Models of parts of a transmission. Graphical representation is presented in Figures 3.7 and 3.8. Together they contain all the models from the one-dimensional transmission shown in Figure 3.1, *Modelica.Mechanics.Rotational.Sensors.AngleSensor* to give angles and *Modelica.Mechanics.Rotational.Sensors.SpeedSensor* to give angular velocity. There are two models that describe different parts of the transmission so a model with a gearbox could be implemented.

Robot.Multigear: A model of a gearbox. The model contains a coupling matrix that allows cross-couplings between more than one joint and one motor, so that one motor, e.g., can affect several joints.

Robot.Drivetrain, presented in Figure 3.9, has components of type *Robot.Transmission_a, Rotot.Multigear* and *Robot.Transmission_b*. If it were not any cross-couplings in the matrix in *Robot.Multigear*, and no external torque input, the *Robot.Drivetrain* would be equivalent with six independent transmissions of the same type as shown in Figure 3.1.

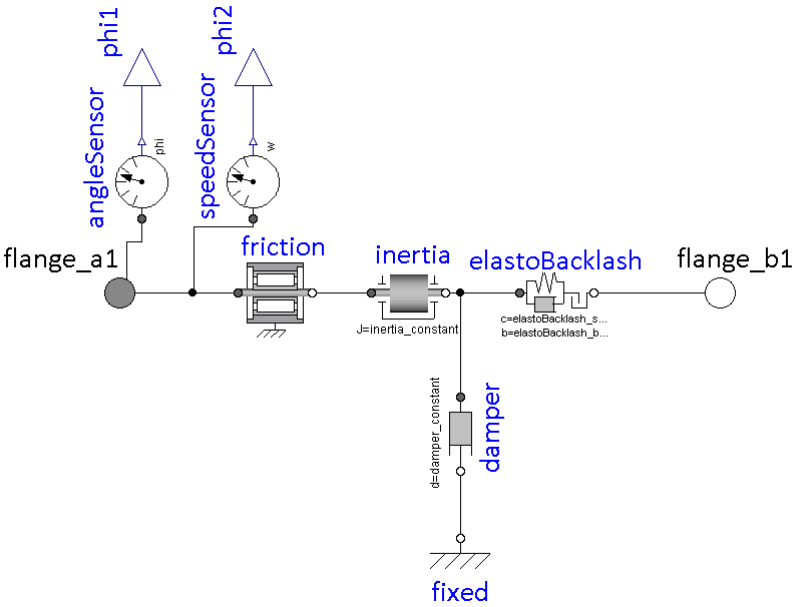

Figure 3.7 A representation of *Robot.Transmission_a* that is a model of a part of the transmission.

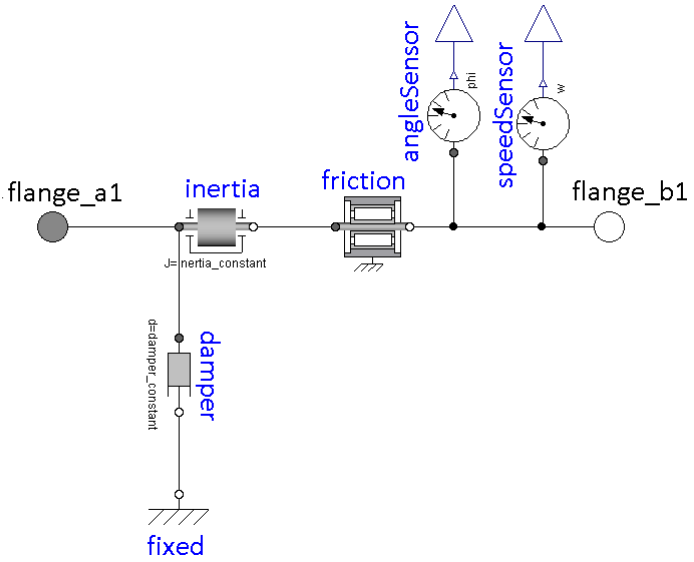

Figure 3.8 A representation of *Robot.Transmission_b* that is a model of a part of the transmission.

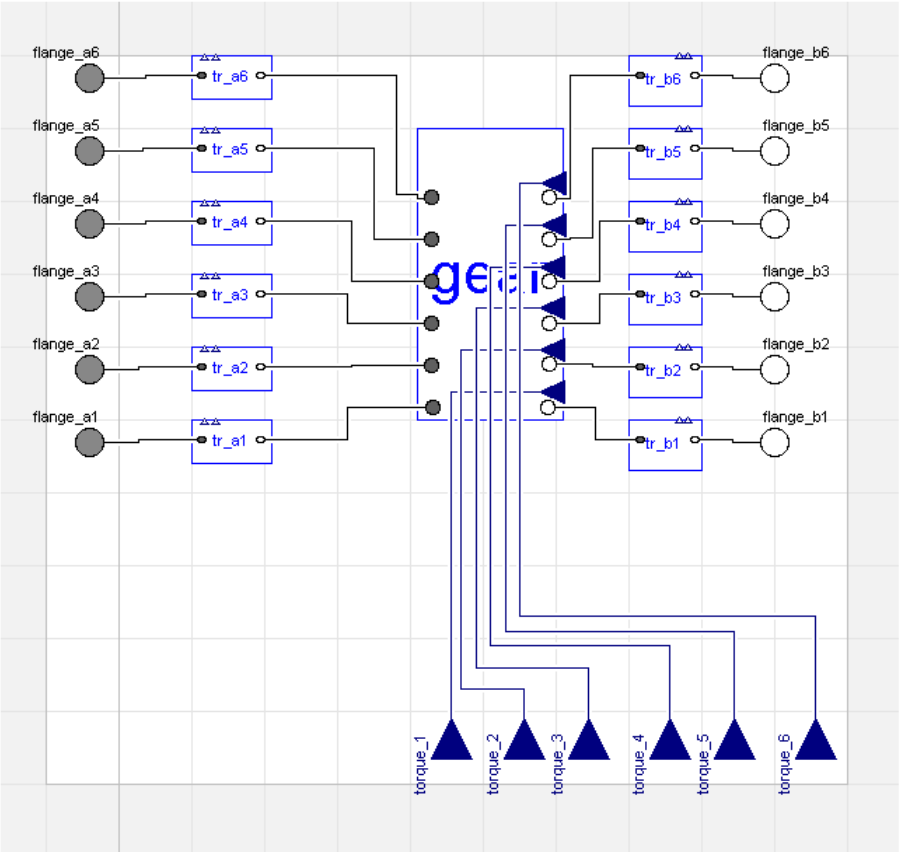

Figure 3.9 A representation of the model *Robot.Drivetrain*. The flanges on the left side of the Figure are to be connected to motors from the outside of this model. The flanges on the right side of the Figure are to be connected to the *Robot.-MechanicalStructure* from the outside of this model. The rectangles next to the left flanges are components of the type *Robot.Transmission_a*, and the rectangles next to the right flanges are components of the type *Robot.Transmission_b*. The rectangle in the middle is a component of type *Robot.Multigear*. The blue triangles at the bottom of the Figure are inputs for external torques acting on the joints.

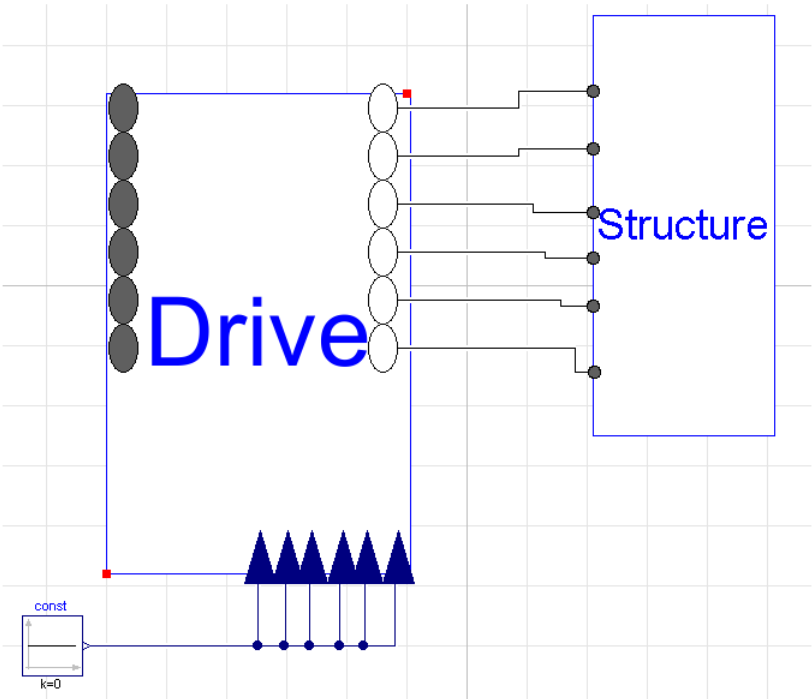

Figure 3.10 A representation of the robot. The component named *Drive* is of type *Robot.Drivetrain*, and the component named *Structure* is of type *Robot.-MechanicalStructure*. The component to the left is a constant input (*Modelica.-Blocks.Sources.Constant*) for the external torques. The gray ovals in *Drive* are inputs for torques that should act on the transmissions, but in the Figure there are no torques connected.

4

Data processing

4.1 The logged data and what it was used for

The data logged and used in the identification were the torque and the position. The torque from the motors were measured on the motor side, but they were scaled by the gear ratio to give the torque on the arm side (this was done by the KUKA system), so the given torque had to be divided by the gear ratio in order to get the torque on the motor side. In all Figures it is the torque on the motor side that is displayed. The position was measured on the motor side, and the unit was degrees, so it was transformed to radians. The time when the samples were taken were also logged. The sampling times were used in the simulations, together with the measured torque.

4.2 Characteristics of a clamping curve

There are some characteristics that make up the shape of clamping curves. These characteristics are to by described by parameters that are identified from clamping data. A clamping curve with exaggerated characteristics is shown in Figure 4.1. There is a spring constant that describes the behavior of the clamping curve in the linear regions, and this constant has to be identified. The backlash also needs to be identified, as well as the friction. There are parameters that describe the behavior of the clamping curve in the nonlinear intervals as well, but these are not identified, just the intervals are identified, because when the linear region and the backlash region are identified, the remaining intervals are the nonlinear intervals. One could identify parameters that describes the clamping curve in the nonlinear intervals as well, for instance by polynomial fitting, but then another model is needed than the model shown Figure 3.1, so this has not been done in this thesis.

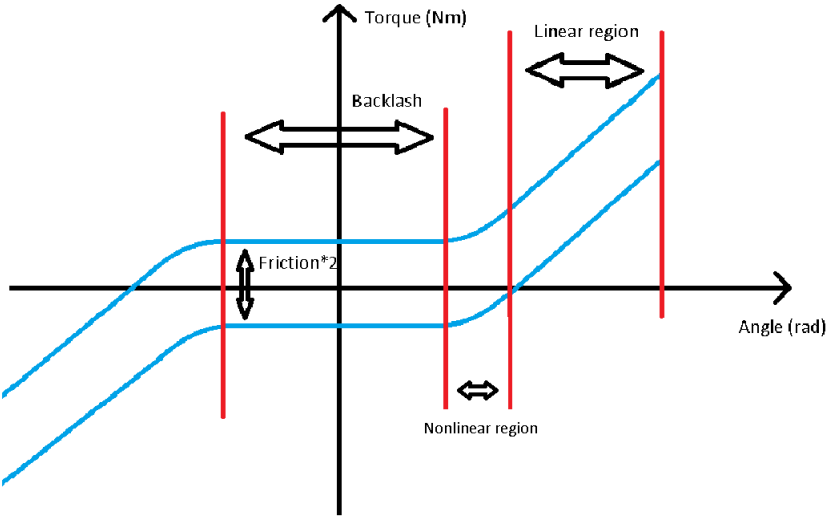

Figure 4.1 A clamping curve with exaggerated characteristics. The characteristics that make up the shape of a clamping curve have been marked. There is just one linear region and one nonlinear region marked in the picture, but there is one of each present in the left part of the picture as well even though they are not marked.

4.3 Parameter identification

The parameter identification was performed in Matlab. Two different algorithms were developed, *Algorithm I* and *Algorithm II*. To be able to identify parameters from clamping data, the data needed to be preprocessed before the identification could be performed. The two developed algorithms had different preprocessing of the data. *Algorithm I* was not as robust as *Algorithm II*, and *Algorithm II* had a better preprocessing of the data before the actual identification of the parameters was performed. The preprocessing of the data before the identification in *Algorithm II* used information about how the data was acquired, which *Algorithm I* did not. The clamping data that were processed consisted of a vector that contained positions and a vector that contained the corresponding torques. Every position and the corresponding torque in the two vectors represented a point. The elements in the vectors appeared in the order that the data was sampled. The identification was done in order to identify the parameters that can be used to more accurately simulate a robot. The simulations can then be used to predict position errors, and the accuracy can be increased if the result of these simulations are used as feed-forward.

Algorithm I

The preprocessing of the data started with the removal of the beginning of the data series. That was done in order to remove initial transient behaviour. The initial tran-

sient behaviour is seen in the middle of Figure 6.1, encircled by the cycles. Then the elements in the position vector were scaled so that every position became an integer value between one and the full desired resolution of the measurements. A higher value chosen for the resolution gave a smaller quantisation effect due to the round off that was a part of the scaling. The scaling was done in a way that the smallest value of the position in the data series got the value one after scaling, and the biggest value of the position got the value resolution after the scaling. The scaling preserved the relative distance between the position points, except for the quantisation. The next step was to separate the data in a lower and upper curve. That was done by calculating the mean of every torque in the data series that had the same scaled position. That gave a vector with mean torques that had the length resolution, where every position in the vector held the mean values of the torques that corresponded to the same scaled positions. This was basically a bin sort for the torques, based on the scaled positions. A least-squares fit for a line was then performed for a number of these mean torques and the corresponding scaled positions. That basically means that there were several least-squares fits that were performed until the whole scaled position interval from 1 to resolution was covered with lines that were able to separate the data in a lower and upper curve. These lines are seen in Figure 4.2, plotted with the separated curves. Each of these lines was calculated from a given position interval, and was used to separate the data in that interval in a lower and an upper curve. That the line was able to separate the lower and upper curve was built on the assumption that there were points in both the upper and lower curve over the specific interval.

The positions in Figure 4.2 are the scaled ones. It is therefore the label of the x-axis is "Motor positions (channel)", where channel refers to the scaling of the motor positions to discrete values between one and resolution. The resolution was set to 10000 when Figure 4.2 was created. A fraction of 0.1 of the position interval was removed from both ends of the position interval. This was done to remove the dynamics that is present in the ends of the position interval. Therefore, the x-axis only covers values between 1000 and 9000 even though the resolution was 10000. Outliers were then removed for the upper and lower curve separately. This was done by fitting many short straight lines piecewise over the interval by the least-squares methods, and calculate the standard deviation for the points over one such line. If the error that was calculated from a point relatively to the least-squares line, was bigger than a given number multiplied with the standard deviation, that point was discarded. This approach for removing outliers was based on the assumption that the clamping curve can be approximated by many short straight lines. For the torques that had the same value for the positions, the mean of them was calculated. Now the lower curve and upper curve were represented as two vectors where the indices were the same as the scaled positions, and they contained torque values. Linear interpolations were performed to fill the gaps where there were no torque values, using the torque values between the gaps. The preprocessed data for joint one and three are shown in Figures 6.7 and 6.8, where the motor angles have been scaled

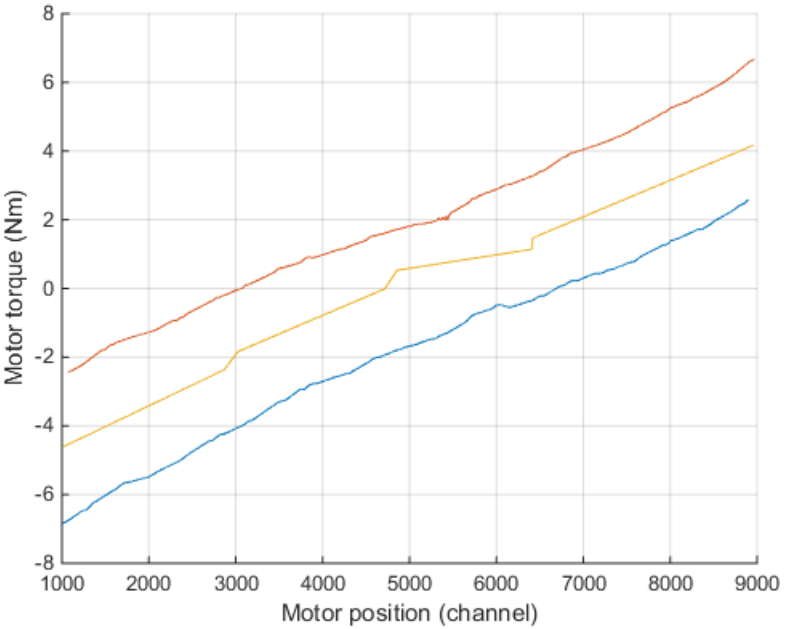

Figure 4.2 The experimental data was separated in a lower and an upper curve, based on the yellow lines. The red curve above the yellow lines was separated from the blue curve below the yellow lines.

back to motor radians to make a comparison easier.

Once the data had been preprocessed, the identification of the spring constant, the backlash and the friction should be performed. The lower curve and the upper curve were treated separately. The lower and upper curves went through the same identification procedure as will be described next. The preprocessed data, that consisted of the lower and upper curve shown in Figure 6.7, will be addressed as the curve in the following text. When it is said that some analysis was performed for one curve, it was performed on both the lower and upper curve separately.

First the curve was filtered with a moving average filter to reduce high frequency measurement noise (non-causal filtering was used in order to keep phase). The backlash was identified as the longest interval were the derivative of the curve was less than a certain threshold. The threshold was calculated as a fraction of the slope of the straight line, that was derived through a least-squares fit for all the data points of the curve. If an interval was found, a least-squares fit for a straight line for the data points in that interval on the curve was performed. That was done in order to verify that the slope of the line was zero or bigger. If the slope of that line was

less than zero, the backlash interval was incremented with one channel (or index in the vector), on both sides of the interval, and a new line was derived through a least-squares fit for the points in that new backlash interval. Then the slope of the line was checked again, and the previous increment of the interval was performed until the slope became zero or bigger.

The nonlinear intervals on the curve were then identified. The identification of the nonlinear interval was done separately, and in the same manner, for the part of the curve that was to the left side of the backlash interval, and the part of the curve that was to the right side of the backlash interval. Therefore the identification of the nonlinear interval for just the right side of the backlash interval is described. A least-squares fit for a straight line was performed for the data on the curve that was to the right side of the backlash interval. Then the derivative for the curve to the right side of the backlash interval was calculated. Then the absolute value of the derivative of the curve was compared to the absolute value of the slope of the straight line. For all the consecutive points on the curve to the right side of the backlash interval, that had an absolute value of the derivative that was less than the absolute value of the slope for the straight line, these points were considered to be in the nonlinear interval.

When the backlash interval and the nonlinear intervals had been calculated, the two intervals that were left on the curve were considered to describe the spring constants. A least-squares fit of a straight line was performed for both of these intervals, and the slope of each line was considered to be a spring constant.

The friction was calculated by taking the difference in torques for the same positions for the upper curve and the lower curve, showed in Figure 6.7, and take the mean of all these differences, and divide it by two.

In Figure 4.3 the intervals are marked with vertical lines. Figure 4.3 is derived from the experimental data for joint 1, shown in Figure 6.1. There are two backlash intervals, one for the upper part of the clamping curve, and one for the lower part of the clamping curve. When it comes to the nonlinear intervals, there are four such intervals. Two for the upper part of the clamping curve and two for the lower part. The parts of the curve that are not in these intervals are assumed to be straight lines described by the spring constants. That assumption looks more valid if Figure 4.2 is considered, where the ends of the intervals that contains the turning dynamics have been removed. So there were four identified spring constants.

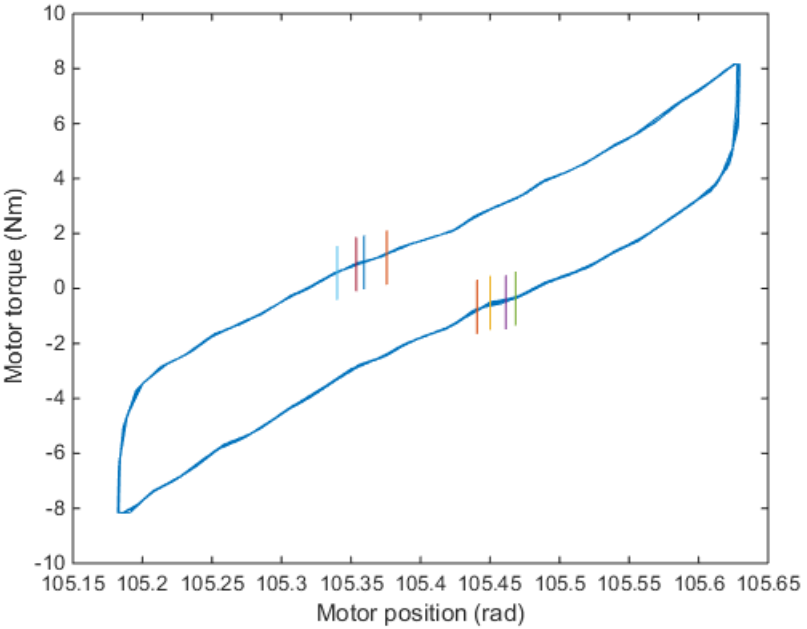

Figure 4.3 The identified intervals for the clamping curve for joint 1, shown in Figure 6.1. The intervals are marked with vertical lines on the experimental data. The beginnings of the experimental data series have been removed. The backlash interval is between the two inner lines for the upper and lower part of the clamping curve, and the nonlinear intervals are between the outer lines and the closest inner line. The parts of the curve that are outside these lines are considered to be in a linear region described by spring constants. The division of the data into different intervals was necessary in order to identify the parameters.

The identification of the friction resulted in one value, while the identification of the backlash resulted in two values, one for the lower curve and one for the upper curve. There were four identified spring constants, two for the lower curve and two for the upper curve. The final backlash constant and the spring constant that was to be used in the simulations were calculated as the mean values of the two backlash constants respective the four spring constants. The calculated constants are shown in Table 6.1.

Algorithm II

The preprocessing of the clamping data started by separating and counting the number of halfcycles in the data. A halfcycle is defined as the consecutive data points between the two turning regions of the clamping curve, as seen in Figure 4.4, where

the turning regions have been marked in the experimental data for joint one, shown in Figure 6.1. So a halfcycle is represented by consecutive points of data. The clamping curve is built up by many halfcycles. The upper part of the curve in Figure 4.4 consists of data points where the torque and position were increased, and the lower part of the curve consists of data points where the torque and position were decreased. A halfcycle was identified as the consecutive points of elements that had the same sign in the derivative vector. The derivative vector was calculated from the position vector as the difference between consecutive elements. When the sign in the derivative vector was used to separate the halfcycles, information about how the experiment was performed was taken into account. Because that separation uses the fact that the positions increase and decrease a number of times in the data series, and the increase or decrease in positions can be used to separate the data into halfcycles.

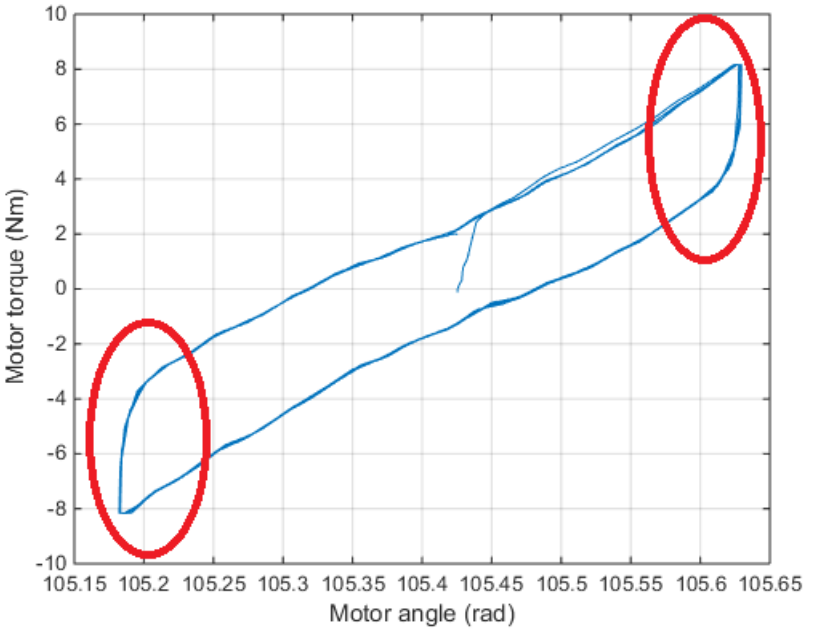

Figure 4.4 The turning regions have been marked. In these regions the velocity changes direction. A halfcycle is defined at the consecutive points of data between these two regions. Because the velocity was close to zero or zero in the turning regions, there are much more data points in these regions.

The data points are much more dense in the turning regions of the clamping curve. That is because the velocity was close to zero or zero in these regions, while the sampling rate of the data points was constant. This became a problem when

the halfcycles were to be separated based on when the sign in the derivative vector changed, as the sign in the derivative vector may change rapidly when the turning regions are considered. This leads to that many halfcycles are detected that consist of points in just the turning regions, which is wrong, as a halfcycle shall go between one turning region and the other. This was solved by taking the distances between the positions in the position vector into account. The mean distance between all position points was calculated, and if the distance between two position points were smaller than the mean distance, the corresponding element in the derivative vector was set to zero. This made that the changes of signs in the turning regions were removed, as the distance between these points is relatively small when all data is considered. The derivative vector was also filtered with a moving average filter to filter out measurement noise. Non-causal filtering was used in order to keep phase. After that the halfcycles could be identified as sequences in the derivative vector that had the same sign. The corresponding elements in the position vector and the torque vector to these sequence points built up the halfcycles.

After that the halfcycles were treated separately in order to remove outliers and the turning dynamics. A certain fraction of the data from each halfcycle was removed. The data that was removed consisted of the points that had the smallest and biggest values for the positions. This was done in order to remove outliers. The turning dynamics were then removed by removing a certain fraction of the position interval from both ends for each halfcycle. Then outliers were removed from the halfcycles by fitting a straight line by the least-squares method for each halfcycle. The standard deviation for all points in one halfcycle relative to this line was calculated, and every point that had a distance relative to this line that was bigger than the standard deviation multiplied with a constant was removed. The halfcycles were also checked if they contained enough data points, and if that was not the case that halfcycle was removed.

Then a straight line was fitted by the least-squares method for every halfcycle (here twelve halfcycles were used). The two parameters that described every derived line were then used to remove halfcycles if the parameters stood out among the others. That was done to remove outliers, e.g., halfcycles that were different due to initial conditions. Halfcycles were removed by calculating the mean of the the parameters, and then the standard deviation towards these two means. If the absolute difference between a parameter and the corresponding mean was bigger than the standard deviation for that parameter, multiplied with a constant parameter (the value 2 was used here), that halfcycle was removed.

The next step was to separate the halfcycles that made up the lower part of the curve from the halfcycles that made up the upper part of the curve from each other. The parameters for the fitted straight lines could not by used to separate the halfcycles directly, because the slope for each halfcycle is almost the same as for all halfcycles. The other parameter that gives the intersection with the y-axis could be used if the interval of the points in all halfcycles were centered around the y-axis. However, that was not the case, so that approach could not be used either. Instead

a representative torque for each halfcycle based on the parameters for each least-squares fit was calculated. The representative torque was calculated using the mean position for all data points. This gave a representative point for each halfcycle in the interval where there were much data, and where the parameters for the straight lines were valid. The halfcycles were then separated by calculating the mean torque for all representative points, and if the torque in a representative point was bigger than the mean torque the halfcycle was considered to belonging to the upper curve, and otherwise to the lower curve.

Then halfcycles were to be removed based on the parameters from the least-squares fitted straight lines. The procedure was the same as described earlier in this section, with the difference that the halfcycles that belonged to the lower curve were treated separately from the halfcycles that belonged to the upper curve. This was done because outliers that were not detected earlier could be detected now, when the data was divided into two sets.

All the information in the halfcycles could now be used to build a lower curve and an upper curve, that could be used to identify the parameters. All the upper halfcycles were used to build the upper curve, and the lower halfcycles were used to build the lower curve, and the procedure was the same for both the lower curve and the upper curve. The procedure described next will later be referred to as to merge curves. Every halfcycle covered a position interval, and a position interval that had at least a given number of halfcycles that covered that interval was chosen. The position interval was represented as a vector, with a given number of indices, that contained evenly spaced positions in the chosen interval. A torque vector that had the same size as the position vector was created. For every one of these positions in the position vector, a torque had to be calculated from the halfcycles that had an interval that contained that position. The mean of these torques from the halfcycles should then be calculated and inserted in the torque vector, at the same index as the position had in the position vector. If the position in the position vector had the same value in a halfcycle, the corresponding torque in that halfcycle was used to build the mean torque. If the position in the position vector had a value that was between two points in a halfcycle, a weighted mean of the two corresponding torques for the two points in the halfcycle was calculated. The weights for the two torques corresponded to a linear interpolation between the two points.

In Figures 6.9 and 6.10 the preprocessed data are shown for joint one and three. The preprocessed data from *Algorithm II* is plotted on the experimental data in Figures 6.11, 6.12, 6.13, 6.14 and 6.15. The data could be rejected in the preprocessing if the fraction of halfcycles that was removed was to big. That was the case for joint two. The friction was identified as half the mean distance between the two curves in the preprocessed data.

In *Algorithm II* there was a different approach to the identification. Both the lower and upper curve were transformed to a single curve, that was used to identify the parameters. The identification procedure is shown for the data from joint one (seen in Figure 6.1), in Figures 4.5, 4.6, 4.7 and 4.8. The backlash was identified as

the longest interval were the derivative of the curve was less than a certain threshold for both the lower and upper curve. The threshold was calculated as a fraction of the slope of the straight line, that was derived through a least-squares fit for all the data points on the curve (the same as for *Algorithm I*). Then the lower curve was displaced so the middle of the backlash intervals for the lower and upper curve had the same position. If no backlash were found for either of the curves, the middle of the backlash interval was considered to be at the point where the torque in that curve was closest to zero. The curves are shown in Figure 4.5 after that the lower curve has been displaced. Then the lower curve was displaced in the vertical direction by moving the lower curve upward the mean distance between the two curves. The two curves are seen in Figure 4.6 after that the lower curve has been displaced. The two curves were then merged, in the same way as it was done in the preprocessing of the data. The one curve that was the result of the merge is seen in Figure 4.7.

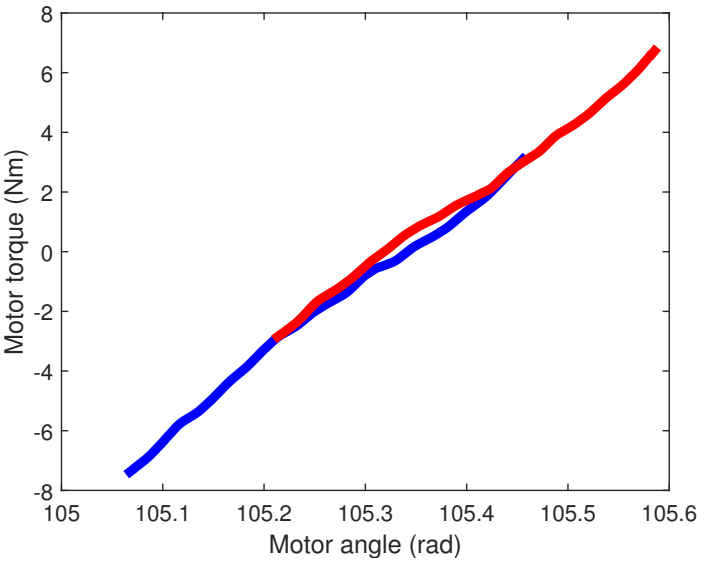

Figure 4.5 The lower curve has been displaced so the middle of the backlash intervals for both curves coincide.

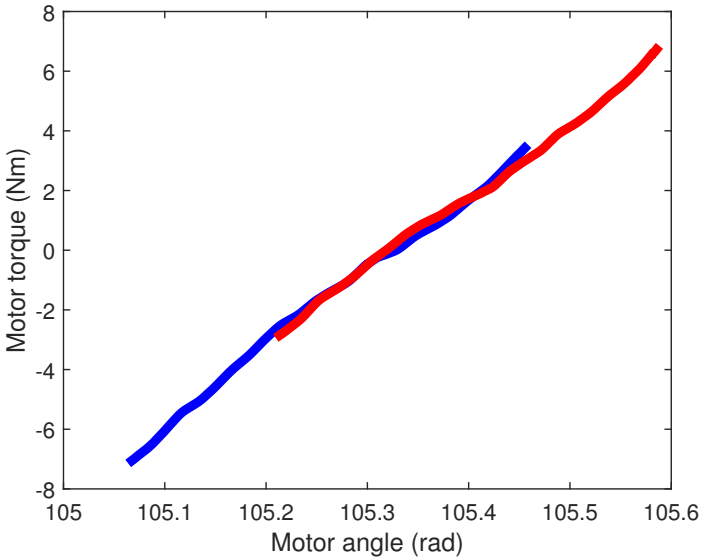

Figure 4.6 The lower curve has been displaced by moving the lower curve upward the mean distance between the two curves.

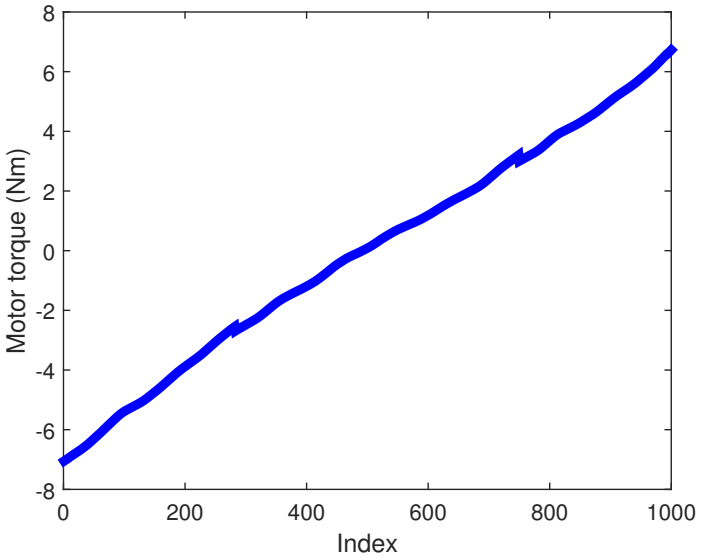

Figure 4.7 The lower curve and upper curve have been merged to one curve.

The middle of the merged curve was chosen to be in the middle of the interval where the slope was less than a certain threshold. If no such interval was found the middle was chosen to be the point on the curve that was closest to zero torque. The merged curve was then cut in the middle. The elements in the left piece were then rearranged so that elements at the first index and at the last index changed place, and the other indices changed places in the same manner. Then a new curve was formed by taking the torque value of the right curve minus the torque value at the left curve, and dividing the result by two. The operations were performed for the overlapping indices for the two curves, starting at the first index for the both curves. The result is seen in Figure 4.8. This derived curve has been constructed from the two separate curves from the preprocessing. It contains the information needed for the parameters of interest to be identified except for the friction, that was identified directly from the preprocessed data, as half the mean distance between the curves. The scale for the x-axis in Figure 4.8 is now indices, but since the difference between any consecutive indices corresponds to a given angle, it is not a problem.

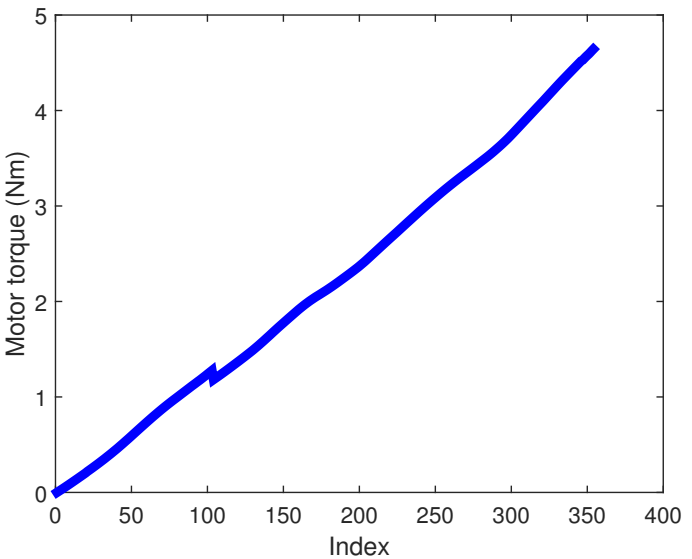

Figure 4.8 The merged curve has been transformed.

The curve in Figure 4.8 was used for the identification. The backlash interval was identified as the length of the interval, starting from the first index, where the the derivative was less than a certain threshold. The threshold was calculated as a fraction of the slope for a straight line, that was derived by a least-squares fit for the whole curve. The spring constant was calculated as the slope of a straight

line derived by a least-squares fit for the curve, where the beginning of the curve corresponding to the backlash interval had been removed.

If the transformation to the curve shown in Figure 4.8 was not possible, the identification of the parameters was done in the same way as in *Algorithm I*. The merge of the curves was not performed if the middles for the lower and upper curve were not found, or if the merged curve had a too uneven distribution of data points to the left of the middle compared to the right of the middle. In either case, the identification of the parameters was performed according to *Algorithm I* (with the preprocessed data from the second algorithm).

5

The experimental setup

5.1 The clamping procedure

To acquire data the robot's end-effector was locked (clamped) to a point in space, and in the ideal case this point would be completely stiff. Then the robot's motors were run, so the torque over each axis was slowly built up and decreased, while the robot's motor positions and the torques were logged, by using the robot's own sensors. The clamping procedure was presented in [11]. If the clamping point was stiff enough, the logged data then contained information about the compliance, backlash and friction of the transmissions. It was these data that were used to derive the parameters for the models of the transmissions.

The idea to lock the robot's end-effector to a stiff point in space (relatively to the stiffness of the robot's parts) and run the motors for each joint separately, and log the torques and motor positions is called the clamping procedure. When the torques are plotted as a function of the positions, this gives a clamping curve, which contains information about the transmission. A clamping curve for joint one is shown in Figure 6.1.

The procedure to use the robot's own sensors to log the data during a clamping experiment is a huge advantage because no additional equipment except a clamping point is needed. It is easy to perform a clamping experiment once the code for the specific robot type has been written. This makes it fast and easy to calibrate the robots individually, and they can maintain accuracy despite wear such as increased backlash.

5.2 What can be identified from a clamping experiment

Ideally the clamping experiment should generate a clamping curve as shown in Figure 6.1. The spring constants can then be identified as the slope of the straight lines in the intervals shown Figure 4.3. The backlash can be identified as an interval close to zero torque, were the slope is less than for the rest of the curve. The interval where there was a backlash detected, for the data from joint one, that is shown in

Figure 6.1, is marked out in Figure 4.3. The friction is the torque corresponding to half the vertical distance between the two lines shown in Figure 6.1.

5.3 A non-stiff clamping point

The measured positions and torques during a clamping experiment give information about the transmissions. Ideally the clamping point would be completely stiff, and the measured positions would all stem from the flex in one of the the transmissions, and not from the clamping point, other transmissions or the robot's arm structure. This is not true, however, since nothing is infinitely stiff. An assumption is that the transmission is the least stiff part of these, and that the measured clamping curve mostly corresponds to the flex in the transmission, and that the clamping curve then contains information about that transmission.

5.4 The experimental setup for the data collection

The experimental setup consisted of a stiff clamping point that the robot's end-effector was locked to. The tool that connected the end-effector to the stiff clamping point is shown in Figures 5.1 and 5.2. The experimental setup is shown in Figures 5.3 and 5.4. The stiff clamping point was the metal block showed in Figures 5.3 and 5.4.

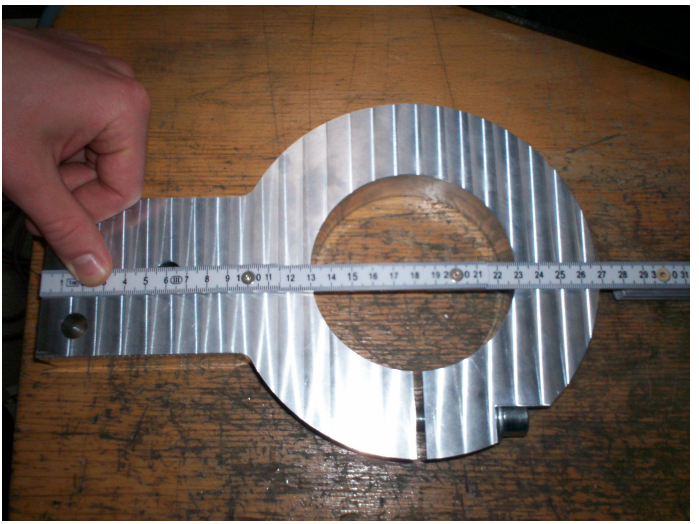

Figure 5.1 The device that was used to to connect the robot's end-effector to the clamping point.

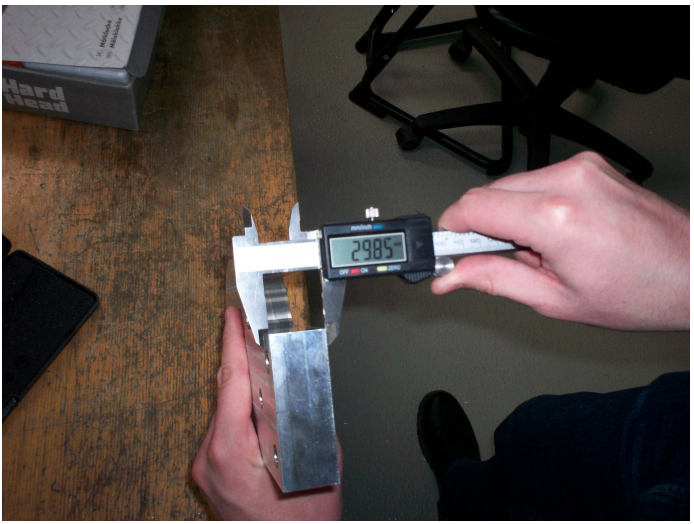

Figure 5.2 The device that was used to to connect the robot's end-effector to the clamping point.

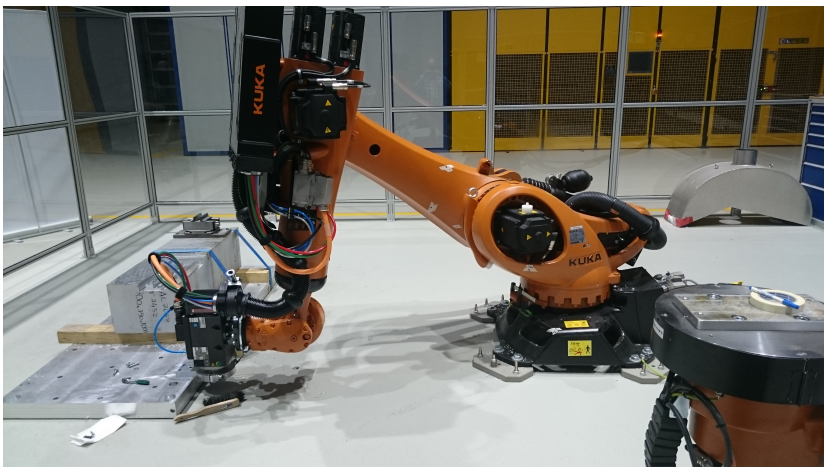

Figure 5.3 The experimental setup for the clamping experiment.

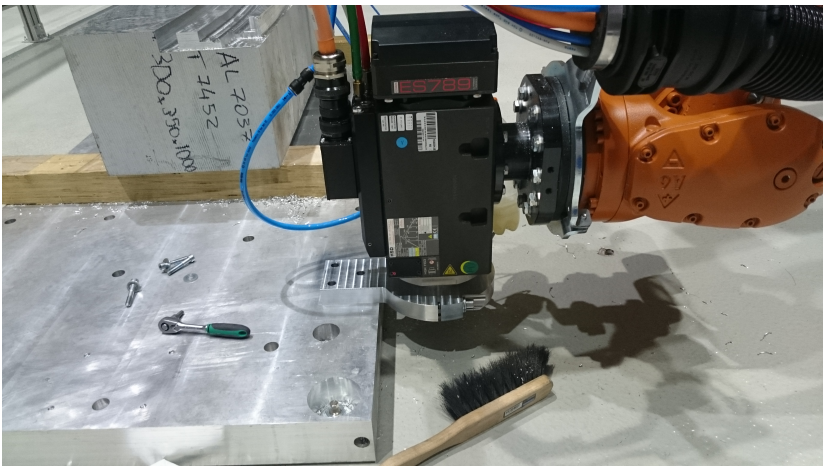

Figure 5.4 The experimental setup for the clamping experiment. The device connecting the end-effector to the clamping point is to be secured with the screws shown in the picture. The clamping point is the metal block.

6

Results

6.1 The experimental data from the robot

Data was acquired from the robot during a set of clamping experiments where positions and torques of the motors were logged. The experimental data from the clamping experiments are shown in Figures 6.1, 6.2, 6.3, 6.4, 6.5 and 6.6. The experimental data for joint 2 presented in Figure 6.2 is not centered around zero torque, and there are wavelike forms in the experimental data for joint 4, 5 and 6 presented in Figures 6.4, 6.5 and 6.6. This is discussed in Chapter 7 (Discussion).

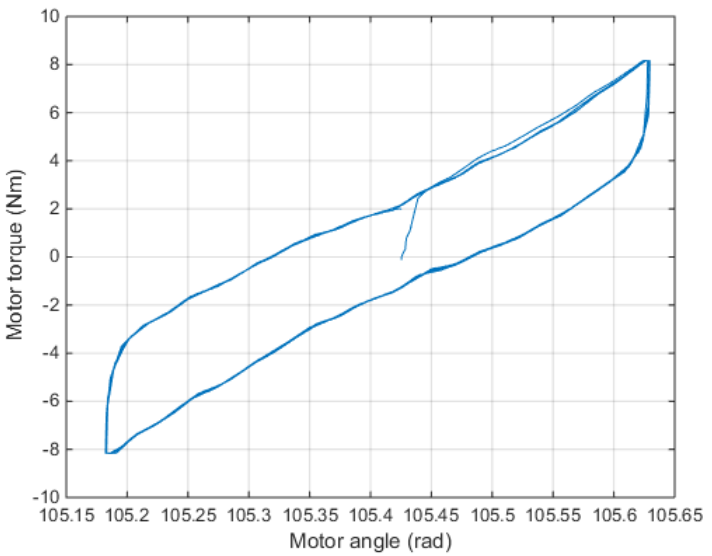

Figure 6.1 Experimental data for joint 1.

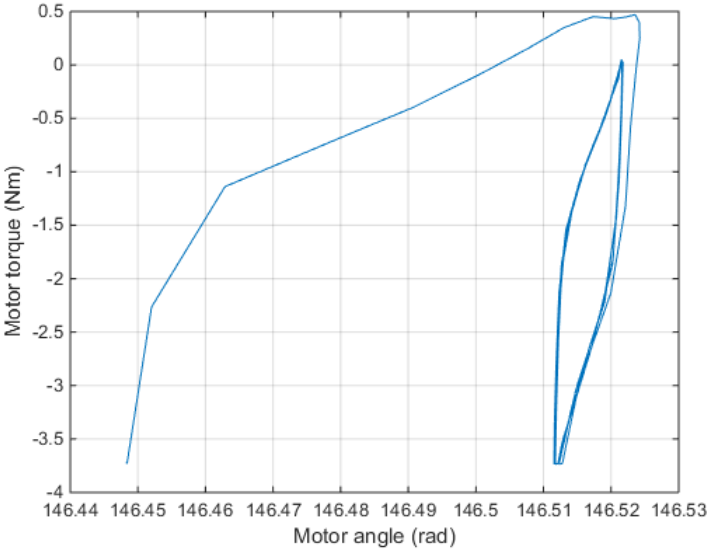

Figure 6.2 Experimental data for joint 2.

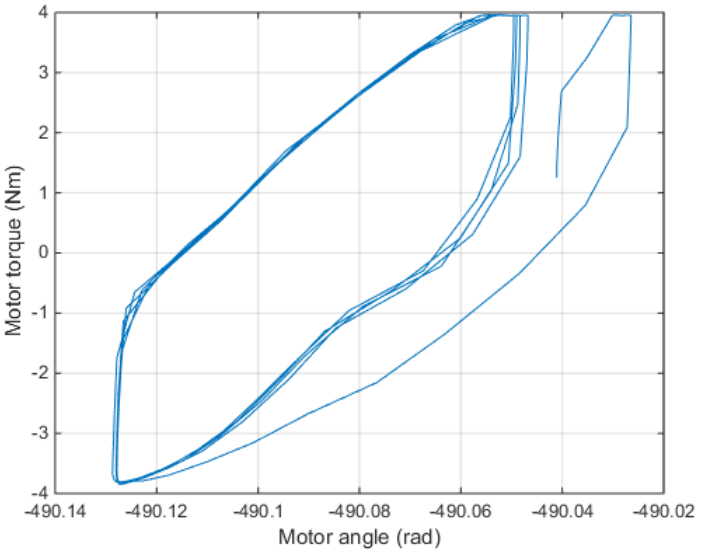

Figure 6.3 Experimental data for joint 3.

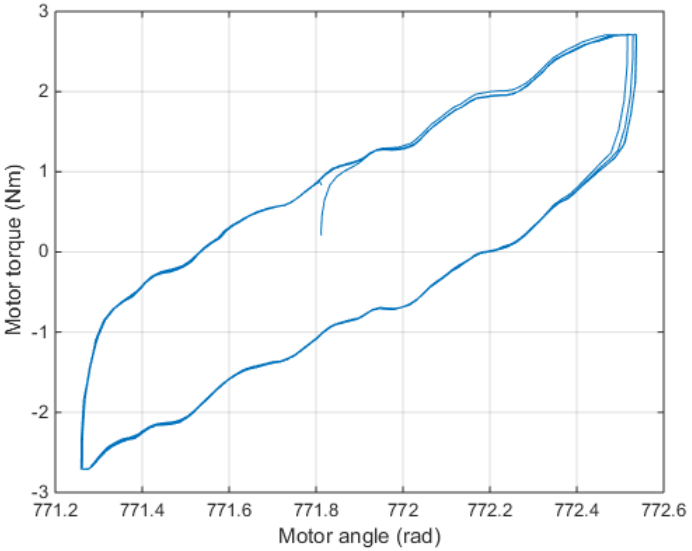

Figure 6.4 Experimental data for joint 4.

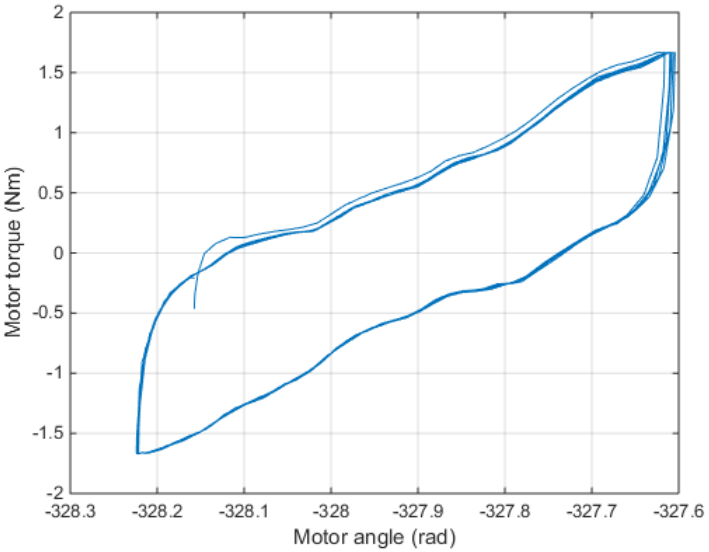

Figure 6.5 Experimental data for joint 5.

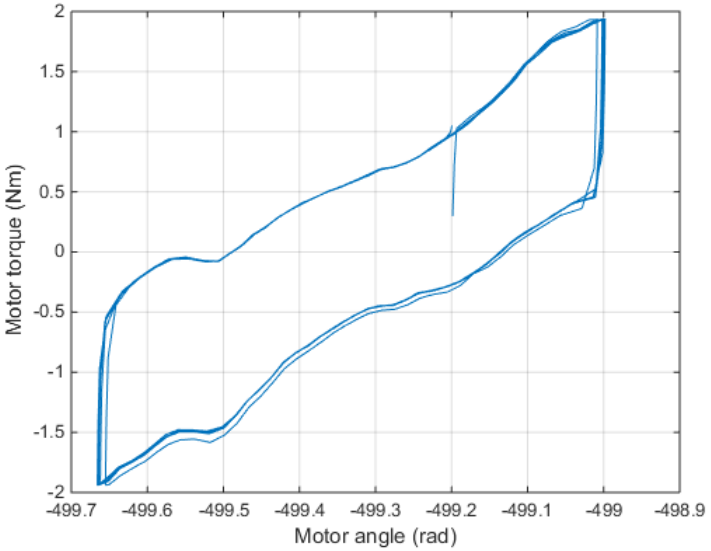

Figure 6.6 Experimental data for joint 6.

6.2 The preprocessed data from *Algorithm I*

The preprocessed data for joint one and three, from *Algorithm I*, are shown in Figures 6.7 and 6.8.

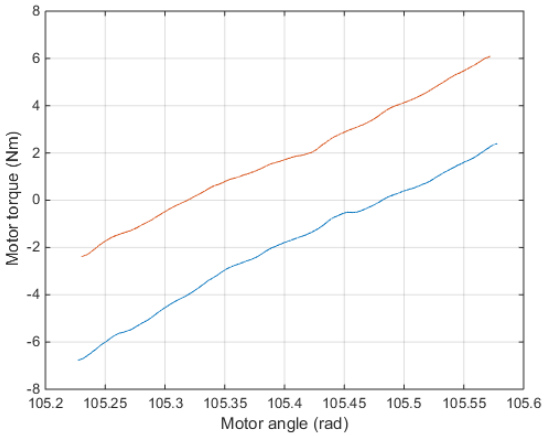

Figure 6.7 The preprocessed data for joint one from *Algorithm I*. The lines have different colors representing that the data has been separated into two sets.

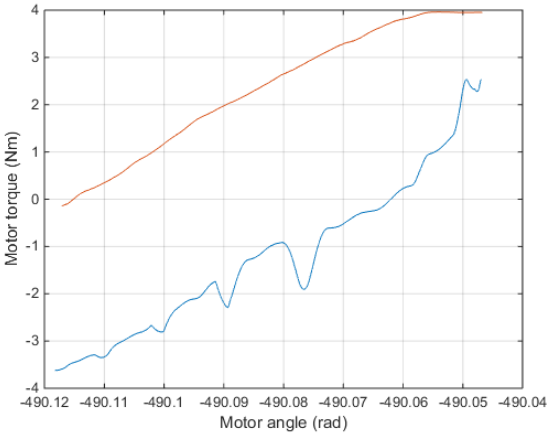

Figure 6.8 The preprocessed data for joint three from *Algorithm I*. The lines have different colors representing that the data has been separated into two sets.

6.3 The preprocessed data from *Algorithm II*

The preprocessed data from *Algorithm II*, for joint one and three, are shown in Figures 6.9 and 6.10. All preprocessed data from *Algorithm II* is plotted on the experimental data in Figures 6.11, 6.12, 6.13, 6.14 and 6.15. For joint two the preprocessing was rejected, because there were not enough data points.

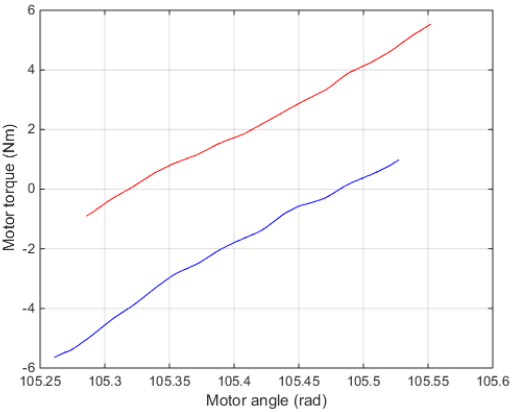

Figure 6.9 The preprocessed data for joint one from *Algorithm II*. The lines have different colors representing that the data has been separated into two sets.

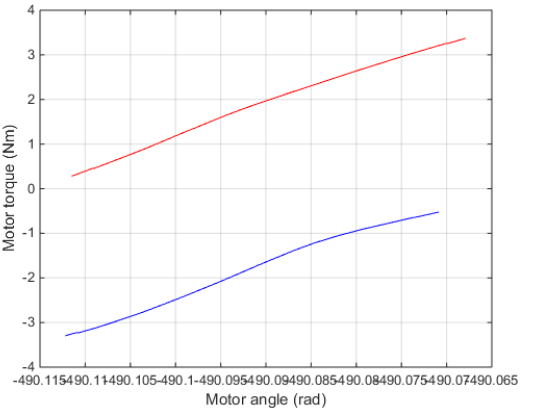

Figure 6.10 The preprocessed data for joint three from *Algorithm II*. The lines have different colors representing that the data has been separated into two sets.

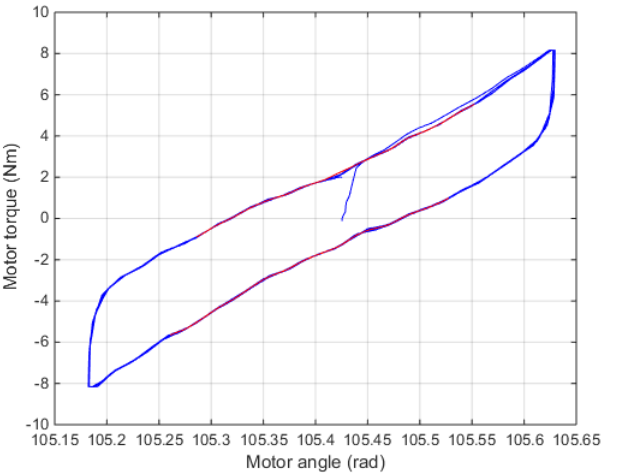

Figure 6.11 The preprocessed data for joint one, from *Algorithm II*, is plotted on the experimental data. The experimental data is blue, and the preprocessed data is red.

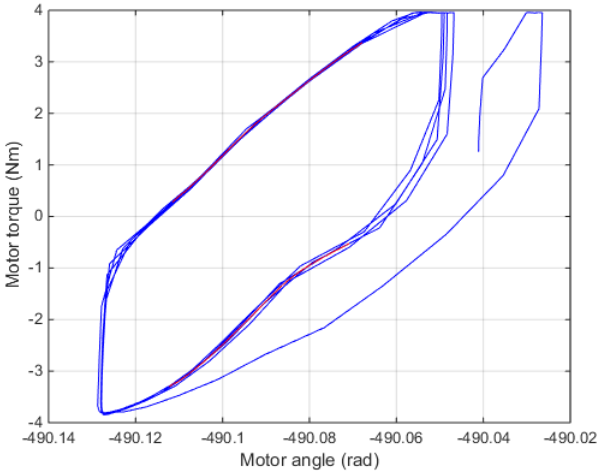

Figure 6.12 The preprocessed data for joint three, from *Algorithm II*, is plotted on the experimental data. The experimental data is blue, and the preprocessed data is red.

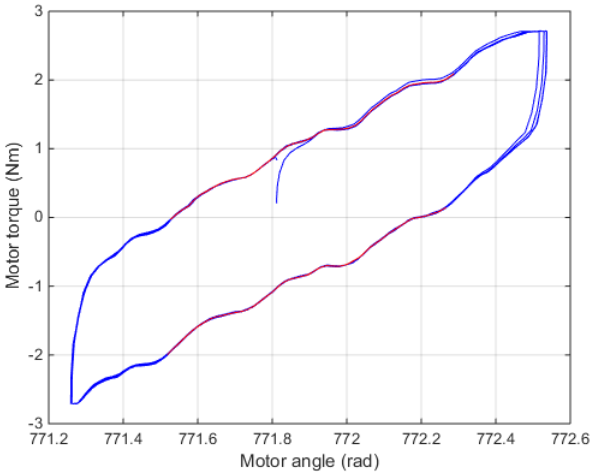

Figure 6.13 The preprocessed data for joint four, from *Algorithm II*, is plotted on the experimental data. The experimental data is blue, and the preprocessed data is red.

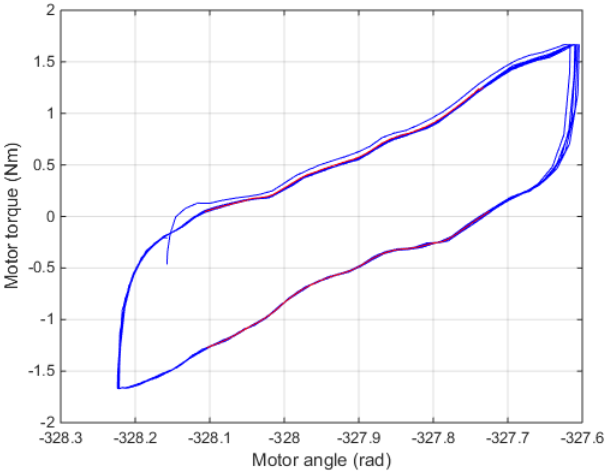

Figure 6.14 The preprocessed data for joint five, from *Algorithm II*, is plotted on the experimental data. The experimental data is blue, and the preprocessed data is red.

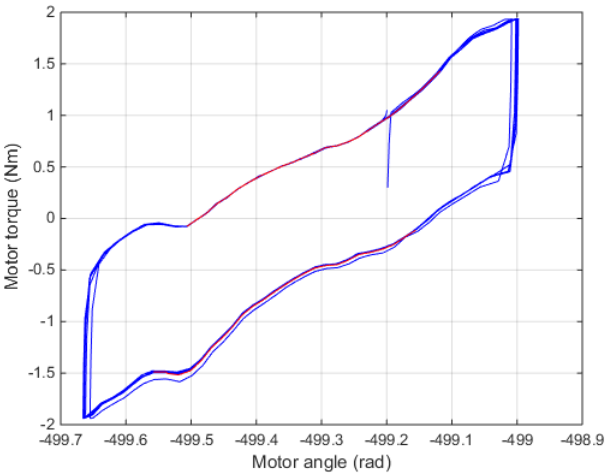

Figure 6.15 The preprocessed data for joint six, from *Algorithm II*, is plotted on the experimental data. The experimental data is blue, and the preprocessed data is red.

6.4 The simulations

The one-dimensional transmission model

Algorithm I only worked for the experimental data for joint 1, as seen in Figure 6.1. *Algorithm I* identified the values for the spring constant, the backlash and the friction. The backlash intervals that were identified are shown in Figure 4.3. The identified parameters are shown in Table 6.1.

spring constant (Nm/rad)	Backlash (rad)	Friction force (Nm)
2.57e+01	1.02e-02	1.91

Table 6.1 The identified parameters for joint 1. The identification was performed using *Algorithm I*.

A simulation with the parameters in Table 6.1 was performed in JModelica.org, and the result is seen in Figure 6.16. The model that was used in the simulation is seen in Figure 3.2. The input to the model was the measured torques from the clamping experiment for joint 1, shown in Figure 6.1, and the logged time when the torques were sampled. The output from the simulation was the motor position, which was the angle of *flange_a* in Figure 3.2. The simulated clamping curve in Figure 6.16 is plotted on the experimental data in Figure 6.17.

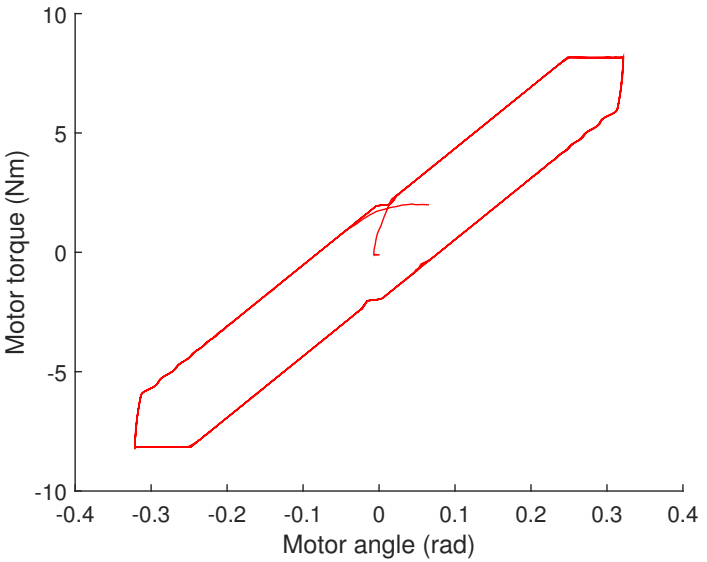

Figure 6.16 The simulated clamping curve for joint 1. The simulation was performed in JModelica.org. In the simulation the same torques that were measured during the clamping experiment were used. The output from the model were the motor positions. What is plotted is the measured torques from the clamping experiment, and the motor positions from the simulation.

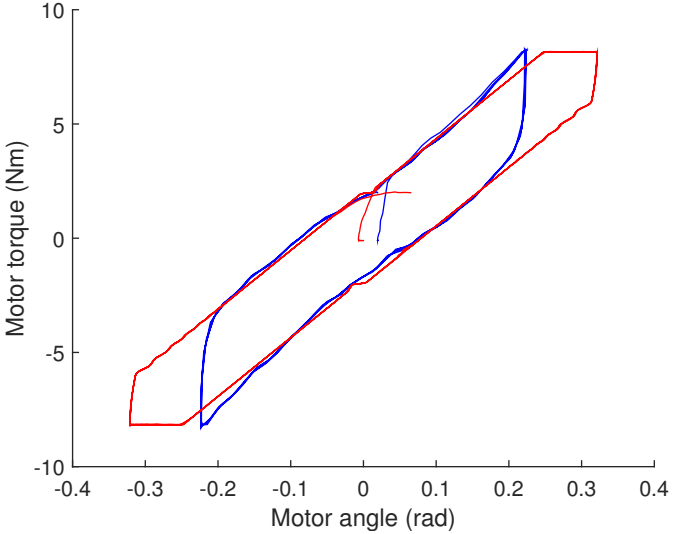

Figure 6.17 The simulated clamping curve is plotted on the experimental data for joint 1. The red curve is from the simulation and the blue curve is the experimental data.

Algorithm II worked for the data from all joints except for joint 2. The data are shown in Figures 6.1, 6.2, 6.3, 6.4, 6.5 and 6.6. The simulations performed with the data and the identified parameters from *Algorithm II* are shown in Figures 6.18, 6.19, 6.20, 6.21 and 6.22

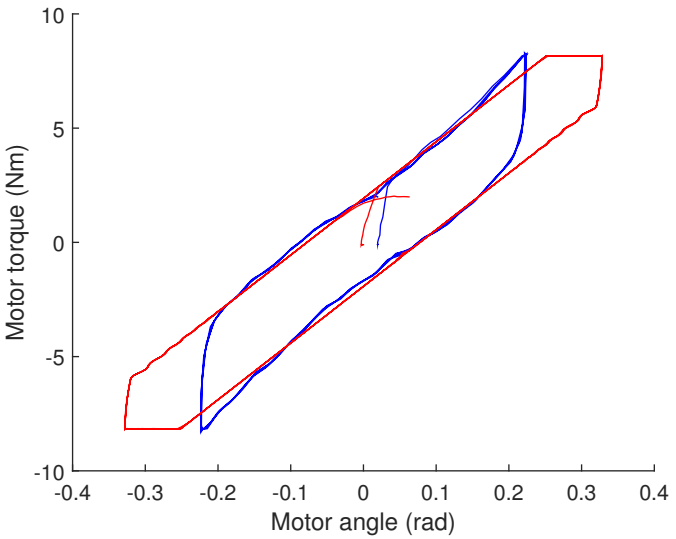

Figure 6.18 The simulated clamping curve is plotted on the experimental data for joint 1. The red curve is from the simulation and the blue curve is the experimental data.

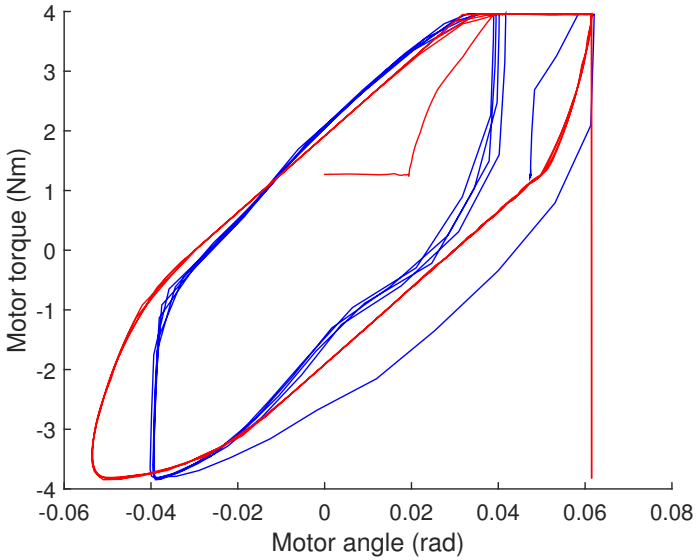

Figure 6.19 The simulated clamping curve is plotted on the experimental data for joint 3. The red curve is from the simulation and the blue curve is the experimental data.

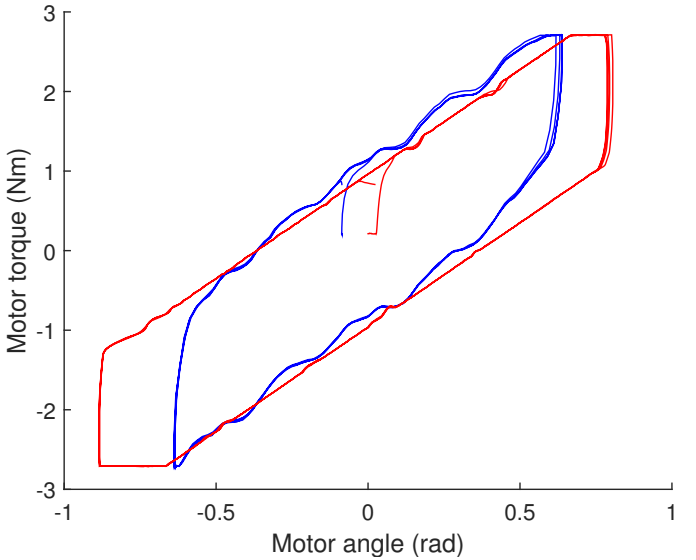

Figure 6.20 The simulated clamping curve is plotted on the experimental data for joint 4. The red curve is from the simulation and the blue curve is the experimental data.

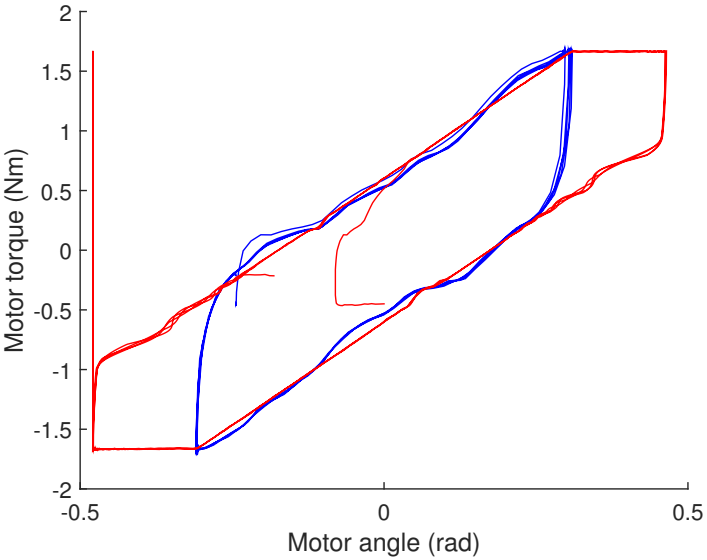

Figure 6.21 The simulated clamping curve is plotted on the experimental data for joint 5. The red curve is from the simulation and the blue curve is the experimental data.

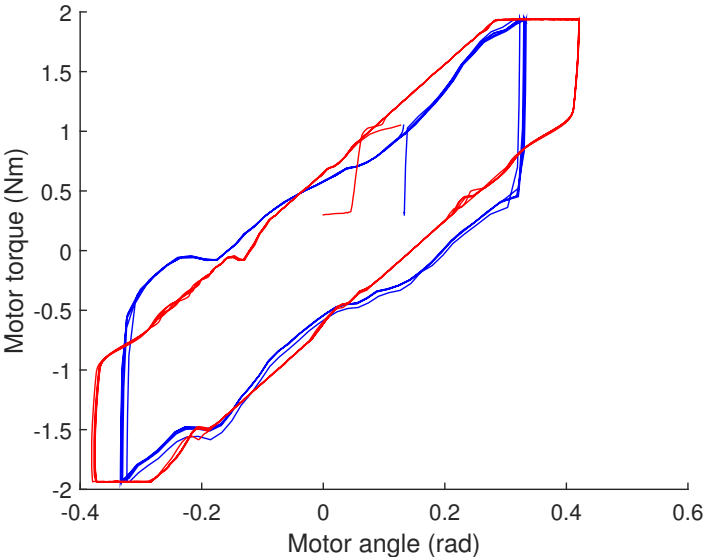

Figure 6.22 The simulated clamping curve is plotted on the experimental data for joint 6. The red curve is from the simulation and the blue curve is the experimental data.

7

Discussion

7.1 The experimental data

The experimental data are shown in Figures 6.1, 6.2, 6.3, 6.4, 6.5 and 6.6. In Figures 6.4, 6.5 and 6.6 it can be seen that the clamping curves are made up by more wavelike lines, compered to the clamping curves showed in Figures 6.1, 6.2 and 6.3. This is probably due to that the regulation of the joint positions for the other joints interfered with the clamping experiment for these joints. This is a source of error that effects the identification of the parameters. But since the simulations of the clamping experiment for joints 4, 5 and 6, shown in Figures 6.20, 6.21 and 6.22, are similar to the experimental data, this source of error did not seem to affect the identification of the parameters too much. In the experimental data for joint 2, presented in Figure 6.2, the data is not centered around zero torque. When that happens it might not be possible to identify the backlash from that data. The backlash is present around zero torque, and if the experimental data does not cover that torque interval where the backlash is present, the backlash can not be identified.

7.2 The logged data and transformations

The torque that was measured at the motor side was scaled by the gear ratio to give the torque at the arm side (this was done by the KUKA system). But in order to run the simulation with the model shown in Figure 3.2, the torque at the motor side was needed. That was taken care of by dividing the torque that the KUKA system gave with the gear ratio for the transmission, and that gave the torque at the motor side.

7.3 The numerical properties of the model

In the model of the one dimensional transmission seen in Figure 3.1, there are two dampers present. No parameters for these dampers were identified. The damping constants for those dampers were set to a small value. The reason for those dampers is the numerical properties for the model. If those dampers were not present the

derived equations for the model became too stiff, and the solver of JModelica.org had problems solving them. This is probably due to oscillations that became present without damping in the simulation that the solver had problem handling. There is a valid assumption that there is damping present in the transmission even if it is not identified in the clamping procedure.

7.4 The identification algorithms

Algorithm II worked better than the first one. That is seen when the data from joint three, shown in Figure 6.3 is preprocessed by *Algorithm I* and *Algorithm II*. The preprocessed data from *Algorithm I* differs greatly from the preprocessed data from *Algorithm II*, shown in Figures 6.8 and 6.10. It is clear that the preprocessed data from *Algorithm II* is better than that from *Algorithm I*, when Figure 6.12 is considered. In Figure 6.12 the preprocessed data from *Algorithm II* is plotted on the clamping data. The main reason that result of the preprocessing of the data from *Algorithm I* shown in Figure 6.8 was not good, is that not enough initial data was removed from the data series in the preprocessing. The initial data shows the transient behavior, and is seen clearly in Figure 6.8. In *Algorithm I* precision was lost due to quantisation from the round of the measured positions. This was not the case for *Algorithm II*, where the torque for a desired position was calculated as the weighted mean of the torques, corresponding to the distances to the two closest measured positions. The preprocessed data from *Algorithm II* is plotted on the clamping data in Figures 6.11, 6.12, 6.13, 6.14, and 6.15. It is seen that the result of the preprocessing of the the data in *Algorithm II* worked satisfactorily. The experimental data shown in Figure 6.2 was rejected by the preprocessing of *Algorithm II*, because the clamping curve did not contain enough data points between the turning regions.

7.5 The simulations

The simulation data is plotted on the experimental data in Figures 6.18, 6.19, 6.20, 6.21 and 6.22. It is seen in Figures 6.19 and 6.21 that there are data points in the turning regions that stand out among the others, and give rise to a vertical line in each Figure. If it could be that it shows a numerical problem that the solver had during the simulation, or that there is something that is not modeled in the turning regions is hard to tell. Anyway, it is still single points, so they could easily be detected by a comparison among the points, and then removed.

7.6 Assumptions made and further work to be done

When the parameters were identified from the clamping data, there was one identified value for the friction force. Since the friction force is velocity dependent, and

the velocity is not constant, the model can not capture the behavior for all parts of the clamping curve. This is seen in Figure 6.17 were the simulation differs the most from the experimental data in the turning regions. The friction force was identified from the data were the turning regions had been excluded. The velocity between the turning regions was also assumed to be constant, which leads to one value for the friction. The assumption that the velocity between the turning regions is constant is a good approximation, based on the way the experiment was performed. The torque was slowly increased or decreased, so the system was always close to an equilibrium between the torque and the force that stemmed from the displacements.There was therefore only a small additional force that could lead to an acceleration and a change in velocity. The assumption seems valid when Figures 6.18, 6.19 6.20, 6.21 and 6.22 are considered, where the simulation agrees well with the experimental data except for the turning regions. The similarity between the experimental data and the data from the simulations shows that the model of the transmission, shown in Figure 3.1, is capable of recreating the behavior of the joints during a clamping experiment.

There was only one value for the friction that was identified. This is not a limitation of the model. A whole table consisting of friction forces for different velocities can be used, including the friction force corresponding to zero velocity, i.e., static friction. To derive this table, a different set of experiments has to be executed, and parameters have to be identified. This is however beyond the scope of this thesis. It is though of interest for further investigation, if the turning regions of the simulated clamping curves shown in Figures 6.18, 6.20, 6.21 and 6.22 could be better recreated using the mentioned additional information.

The multi-axes robot model that was developed was never tested on experimental data. It is of interest to see if it is capable of recreating the clamping curves better than the one-dimensional transmission models did. It should be able to do this, because in the multi-axes robot model the compliance of the links are accounted for, and with the right values inserted in the model, the predictions of the clamping curves should be better. The multi-axes robot model should be validated by a comparison with a real robot, i.e., the multi-axes robot model should be able to recreate the errors that a real robot make, when it has the same reference as input (position references), for example when the reference position input is a circle. To be able to validate the multi-axes robot model through a simulation, models of the servos have to be developed. The compliance for the links are also needed, and need to be measured somehow (perhaps by running a test-program on the robot of interest, and identify the parameters). The mass of different parts of the robot need to be measured and included in the multi-axes robot model to make it capable of recreating the effects of gravitation. If the real robot encounters process forces when it is run, these process forces need to be fed to the multi-axes robot model during the simulation as external torques. How the process forces shall be measured, and how these process forces correspond to the external torques, need to be investigated.

8

Conclusions

The one-dimensional transmission-model was able to recreate the clamping curves with great accuracy, with the derived parameters from the clamping data. This indicates that the method of acquiring data with the clamping procedure, and to simulate models of the transmissions in JModelica.org, is sufficient to predict errors that stems from the transmission. The conformity between the clamping data and the simulations also indicates that the developed algorithm correctly identifies the parameters that describe the transmissions. The robot model that was used in this thesis (a KR 300 R2500 ULTRA), is a representative example for industrial robots. Their structure is similar as well as their characteristics so even though the focus in this thesis was on an individual robot, the developed methods are of a general interest and are applicable for a wide range of industrial robots. The identification algorithm developed in this thesis is going to run on the Cognibotics server, so users can connect to it and get parameters that describe transmissions identified from clamping data.

Bibliography

[1] E. Abele, J. Bauer, T. Hemker, R. Laurischkat, H. Meier, S. Reese, and O. von Stryk. “Comparison and validation of implementations of a flexible joint multibody dynamics system model for an industrial robot”. *CIRP Journal of Manufacturing Science and Technology* **4**:1 (2011), pp. 38–43.

[2] E. Abele, M. Weigold, and S. Rothenbücher. “Modeling and Identification of an Industrial Robot for Machining Applications”. *CIRP Annals* **56**:1 (2007), 387—390.

[3] E. Abele, S. Rothenbücher, and M. Weigold. “Cartesian compliance model for industrial robots using virtual joints”. *Production Engineering* **2**:3 (2008), pp. 339–343.

[4] D. J. Bennett, J. M. Hollerbach, and P. D. Henri. “Kinematic calibration by direct estimation of the jacobian matrix”. In: *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE. Nice, France, 12-14 May 1992, pp. 351–357.

[5] A. C. Bittencourt, E. Wernholt, S. Sander-Tavallaey, and T. Brogårdh. “An extended friction model to capture load and temperature effects in robot joints”. In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE. Taipei, Taiwan, 18-22 October 2010, pp. 6161–6167.

[6] C. Dumas, S. Caro, S. Garnier, and B. Furet. “Joint stiffness identification of six-revolute industrial serial robots”. *Robotics and Computer-Integrated Manufacturing* **27**:4 (2011), pp. 881–888.

[7] fmi-standard.org. *fmi-standard.org*. URL: `https://www.fmi-standard.org/` (visited on 10/16/2015).

[8] G. Hovland, S. Hanssen, E. Gallestey, S. Moberg, T. Brogårdh, S. Gunnarsson, and M. Isaksson. “Nonlinear identification of backlash in robot transmissions”. In: *Proceedings of the 33rd ISR (International Symposium on Robotics)*. Stockholm, Sweden, 7-11 October 2002, pp. 1–6.

[9] JModelica.org. *JModelica.org*. URL: `http://www.jmodelica.org` (visited on 10/16/2015).

[10] C. Lehmann, B. Olofsson, K. Nilsson, M. Halbauer, M. Haage, A. Robertsson, O. Sörnmo, and U. Berger. "Robot Joint Modeling and Parameter Identification Using the Clamping Method". In: *IFAC Conference on Manufacturing Modelling, Management and Control (MIM2013)*. Saint Petersburg, Russia, 19-21 June 2013, pp. 813–818.

[11] K. Nilsson. *Method and system for determination of at least one property of a joint*. WO Patent App. PCT/SE2013/051,224. 2014. URL: `http://www.google.com/patents/WO2014065744A4?cl=en`.

[12] M. Otter, H. Elmqvist, and S. E. Mattsson. "The New Modelica Multibody Library". In: *Proceedings of the 3rd International Modelica Conference (P. Fritzson, ed.), (Linköping), The Modelica Association and Linköping University*. Linköping, Sweden, 3-4 November 2003, pp. 311–330.

[13] Z. Pan and H. Zhang. "Improving robotic machining accuracy by real-time compensation". In: *ICCAS-SICE, 2009*. IEEE. Fukuoka, Japan, 18-21 August 2009, pp. 4289–4294.

[14] C. Reinl, M. Friedmann, J. Bauer, M. Pischan, E. Abele, and O. von Stryk. "Model-based off-line compensation of path deviation for industrial robots in milling applications. ieee". In: *ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. Budapest, Hungary, 3-7 July 2011, pp. 367–372.

[15] M. Ruderman, F. Hoffmann, and T. Bertram. "Modeling and Identification of Elastic Robot Joints With Hysteresis and Backlash". *IEEE Transactions on Industrial Electronics* **56**:10 (2009), pp. 3840–3847.

[16] C. Schweiger and M. Otter. "Modeling 3D Mechanical Effects of 1D Powertrains". In: *Proceedings of the 3rd International Modelica Conference (P. Fritzson, ed.), (Linköping), The Modelica Association and Linköping University*. Linköping, Sweden, 3-4 November 2003, pp. 149–158.

[17] O. Sörnmo, B. Olofsson, U. Schneider, A. Robertsson, and R. Johansson. "Increasing the Milling Accuracy for Industrial Robots Using a Piezo-Actuated High-Dynamic Micro Manipulator". In: *The 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Kaohsiung, Taiwan, 11-14 July 2012, pp. 104–110.

[18] The Modelica Association. *Modelica documentation*. URL: `http://modelica.github.io/Modelica/` (visited on 05/31/2015).

[19] The Modelica Association. *The Modelica Association*. URL: `https://www.modelica.org` (visited on 08/19/2015).

[20] I. Tyapin, G. Hovland, and T. Brogårdh. “Method for estimating combined controller, joint and link stiffnesses of an industrial robot”. In: *Robotic and Sensors Environments (ROSE), 2014 IEEE International Symposium on*. IEEE. Timisoara, Romania, 16-18 October 2014, pp. 1–6.

[21] J. Wang, H. Zhang, and T. Fuhlbrigge. “Improving Machining Accuracy with Robot Deformation Compansation”. In: *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. St. Louis, USA, 10-15 October 2009, pp. 3826 –3831.

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

Document name
MASTER´S THESIS

Date of issue
August 2015

Document Number
ISRN LUTFD2/TFRT--5991--SE

Author(s)
Anders Söder-Hoorn

Supervisor
Klas Nilsson, Cognibotics, Sweden
Mathias Haage, Cognibotics, Sweden
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden
Rolf Johansson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle

Prediction of position errors for an industrial robot, using a model of the robot with parameters acquired from the clamping procedure

Abstract

A robot should follow a given path as accurately as possible. There are however almost always deviations from the desired path, and when the deviations become too large it may be a problem. Some of the deviations stem from transmissions of the robot, and the deviations become more pronounced when there are process forces and gravitational forces present, that affect the transmissions. When position is measured and controlled on the motor side, characteristics of the transmission are not accounted for in the control of the robot, resulting in deviations from the desired path. There are also deviations from the desired path that stem from links of the robot, due to process forces and gravitational forces that affect the links.

To predict the deviations that stem from the transmissions, models of the transmissions were developed and used. Models that should be able to predict the deviations that stem from the links were also developed. To acquire data about the characteristics of the transmissions of a robot, tailored experiments were performed. To acquire the data, the robot's end-effector was locked to a stiff point in space, the robot's motors were run, and the robot's sensors were used to log the data. This procedure is known as a clamping procedure. The collected data were processed to derive parameters that were used in the models of the transmissions. The robot that was used was a KR 300 R2500 ULTRA, which is an industrial robot with six degrees of freedom.

Simulations with the models of the transmissions were able to recreate the data from the clamping procedure with great accuracy. This shows that the models of the transmissions, with the parameters derived from the clamping procedure, capture characteristics of the robot, that are not taken into account in the control of the robot, which leads to deviations from the desired path.

The next step, that is not a part of this thesis, is to use a model of the whole robot, that contains the models of the transmissions and the links, and to validate that the robot model can recreate the deviations from a given path that the robot takes.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-61

Recipient's notes

Security classification