Modeling and Control of a Parallel Robot Using Modelica

Isolde Dressler[1] Johannes Schiffer[1,2] Anders Robertsson[1]
[1]Department of Automatic Control, Lund University
Box 118, 22100 Lund, Sweden
[2] Institute for System Theory and Automatic Control
University of Stuttgart, Germany

Abstract

A new type of high-performance robots has been developed by ABB Robotics, the Robotics Lab at Lund University and Güdel AG, Switzerland. In all parts of the project, ranging from the simulation of the kinematic configuration and reachable workspace, and kinematic and dynamic calibration/grey-box identification, and to code generation of controllers and optimal switching strategies for hybrid control, Modelica and Optimica provide very valuable functionality. We will make a short overview of the different aspects used during the development.

Keywords: Robotics, Multi Body Systems, Dual motor control

Figure 1: Full size Gantry-Tau prototype developed within the SMErobot^{TM}project. The carts (red) are controlled in a coordinated way along the three rails to move the tool/end-plate along a desired trajectory.

1 Introduction

A new type of high-performance manipulator, the Gantry-Tau robot [1], has been developed within the EU FP-6 project SMErobot^{TM}[2] by ABB Robotics, the Robotics Lab at Lund University and Güdel AG, Switzerland. The new concept, which is based on the parallel configuration of the robot's joints (parallel robots), see Fig. 1, is modular, has a large open workspace, is easy to scale and has the inherent benefit of very low inertia of the moving robot parts. This, together with high stiffness of joints and arms, makes it possible to build high-performance robots with respect to accuracy, speed, stiffness and mechanical bandwidth.

Modelica and the MultiBody Library [3] can advantageously be used for modeling and control of robots. In [4] we have reported on how the dynamic model equations of the Gantry-Tau robot were extracted from a MultiBody Modelica model of the robot. The control of a parallel robot using an inverse dynamic model generated by Dymola is presented in [5].

This article presents how different functionalities of Modelica have been used for modeling, simulation, identification and controller generation of the Gantry-Tau manipulator during the different project phases. Two main areas will be discussed: The first is the calibration of the robot's kinematics using Optimica [6], the second the optimization of the actuator control.

In [4], the authors carried out kinematic calibration of the Gantry-Tau robot using a scripting language. In this work, it is shown how a Modelica model for optimization is generated of a subset of the original MultiBody model of the robot. The kinematic parameters are then optimized using Optimica.

To reduce backlash and improve the actuator positioning, the usage of two motors for each cart has been investigated. The actuator system was

DOI: 10.3384/ecp09430118

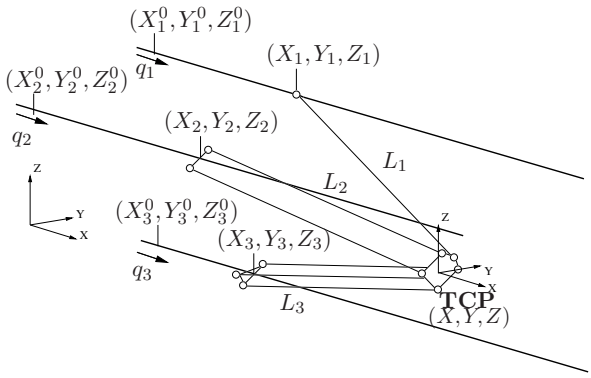

Figure 2: Gantry-Tau schema with variable and parameter notation

modeled in Modelica and a hybrid switching control concept has been tested and optimized using Dymola.

The article is structured as follows: In Sect. 2, the Gantry-Tau manipulator is presented, Sect. 3 describes the kinematic calibration and Sect. 4 the dual motor control. In Sect. 5 the results and methodology are discussed and Sect. 6 concludes the article.

2 The Gantry-Tau Robot

The 3 degree-of-freedom (DOF) parallel Gantry-Tau robot (Figs 1 and 2) consists of three kinematic chains. A prismatic actuator, implemented as a cart moving on a track is connected to an end-effector plate via a link cluster. The altogether 6 links, mounted with passive spherical joints on cart and plate, are distributed in a 3-2-1 configuration to the 3 link clusters. The spherical joint placement on carts and plate is such that links belonging to one cluster form parallelograms, which assures a constant end-effector orientation.

The actuation for the linear motion of the carts along the rails are provided by a so called *rack-and-pinion system*. In all transmissions friction and backlash may severely degrade the performance and accuracy. However, with a rack-and pinion system several motors/carts/robots may be mounted on the same rail and can be controlled independently of each other, which is not the case if the linear motion transmission is made by e.g., ball screws. In Sect. 4 this property will be used for dual motor control and backlash reduction, see also Fig. 5.

As the end-effector orientation is constant for all cart positions, it is sufficient to consider one link per kinematic chain. In addition to a full model with 6 links according to Fig. 2, a simplified Gantry-Tau model has been implemented using the Modelica MultiBody Library (Fig. 3). Here the end-effector orientation is kept constant by a block (blue rectangle in the bottom) which contains 3 passive, serially connected prismatic joints aligned with the 3 coordinate axes. Each of the 3 kinematic chains visible in Fig. 3 consists of a model for track and cart positioned in the base coordinate system by a `FixedTranslation` block and a link connected to the end-effector plate. Input signals of the model are the cart positions.

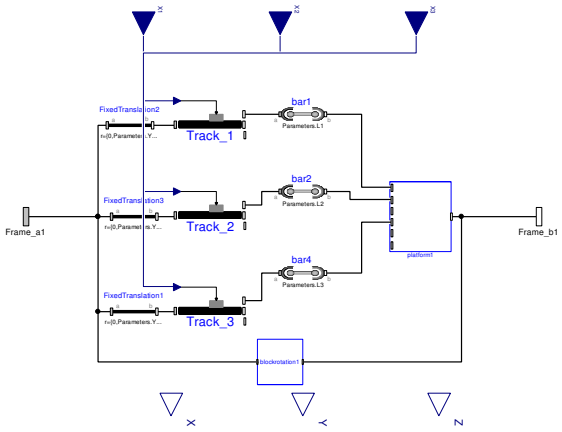

Figure 3: Modelica model of a Gantry-Tau PKM

3 Kinematic Calibration using Optimica

To determine the kinematic parameters, the end-effector position (X, Y, Z) was recorded with a laser tracker for a number of actuator positions (q_1, q_2, q_3). The altogether 21 parameters to optimize are link lengths L_i, the vectors in track direction c_i and the track offsets $(X_i^{\text{offset}}, Y_i^{\text{offset}}, Z_i^{\text{offset}})$, which accumulate the start positions (X_i^0, Y_i^0, Z_i^0) and the offsets between spherical joints and tool center point (TCP) on the end-effector plate, $i = 1, 2, 3$ (see Fig. 2).

The calibration with Optimica is divided in several steps. As the MultiBody Library is not yet compatible with Optimica, a flat Modelica model for optimization has to be generated. For that, the model equations are extracted automatically from the MultiBody model. With a subset of these equations, the kinematic constraint equations, a model for optimization is then generated. After

the optimization, the results are validated.

The extraction of the model equations was first presented in [4]. When translating the MultiBody model in Dymola, the `dsmodel.mof` file with a listing of the translated Modelica code can be generated. This file contains a section with the relevant model equations and assignments relating the large number of variables and parameters that the MultiBody model contains. A script written in python parses this file, extracts the model equations and uses the assignments to successively substitue variables and parameters until the equations are expressed in a desired and previously determined set of parameters and variables. The equations for the kinematic constrains are:

$$\begin{aligned} 0 &= L_i^2 - \Big((X_i - X)^2 + (Y_i - Y)^2 + \\ &\quad + (Z_i - Z)^2\Big),\ i = 1,2,3, \end{aligned} \tag{1}$$

where the cart position (see Fig. 2) $(X_i, Y_i, Z_i)^T = (X_i^{\text{offset}}, Y_i^{\text{offset}}, Z_i^{\text{offset}})^T + q_i \cdot c_i$.

The remaining 9 equations can be found in [4].

Using the measurement data and the kinematic constraint equations among the extracted equation system, a new Modelica model for optimization is then generated:

```
model GTPKinCalib

  parameter Real q1[N] = {data};
  parameter Real q2[N] = {data};
  parameter Real q3[N] = {data};

  parameter Real X[N] = {data};
  parameter Real Y[N] = {data};
  parameter Real Z[N] = {data};

  parameter Real L1;
  parameter Real X1offset;
  parameter Real Y1offset;
  parameter Real Z1offset;
  parameter Real c1[3];
  parameter Real L2;
  parameter Real X2offset;
  ...

  Real f1[N];
  Real f2[N];
  Real f3[N];
  Real cost;

equation
  for i in 1:N loop
    f1[i] = kinematic constraint link 1;
    f2[i] = kinematic constraint link 2;
    f3[i] = kinematic constraint link 3;
  end for;

  cost = f1[1]^2+f2[1]^2+f3[1]^2+ ...;

end GTPKinCalib;
```

The variables `fi[N]` in the model `GTPKinCalib` are the residuals for equation (1) for the given measurement data and parameter values. The variable `cost` is then minimized using Optimica.

3.1 Results

For kinematic calibration, the TCP position (X, Y, Z) was recorded for 176 robot poses with known actuator positions (q_1, q_2, q_3) with a laser tracker. Every second measurement was used for calibration, the remaining ones for the validation of the optimization results.

Figure 4 shows the validation results of the calibration. The calibrated model has a mean absolute positioning error of about 140 μm. Very similar results for parameters and positioning accuracy can be obtained with the Matlab script used in [4].

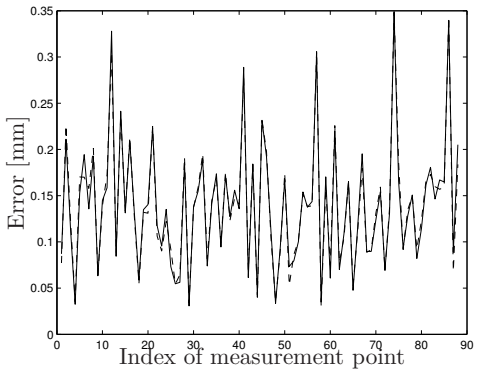

Figure 4: Positioning accuracy of the Gantry-Tau after calibration: absolute positioning error of TCP for the validation measurement points. The model calibrated with Optimica (solid) and the model obtained with the Matlab script used in [4] (dashed) give very similar results.

4 Dual Motor Control

To meet the demands on the system, kinematic sensitivity analysis shows that the robot needs backlashfree gearboxes to achieve the desired positioning accuracy of the robot's TCP. The actuator and drive-line of the robot are based on the rack-and-pinion principle which has been simulated in Modelica.

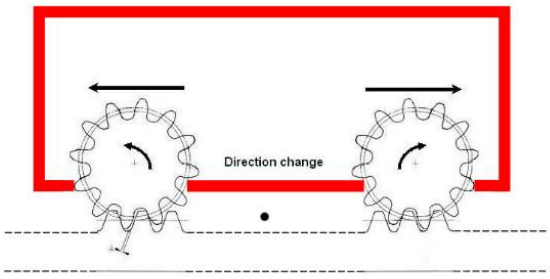

Figure 5: Backlash is present in gearboxes and connection to rails (rack-and-pinion); By using two motors on each cart, which work in opposite directions, improved positioning accuracy and stiffness can be achieved.

In the following, a model representing the robot's actuator drive-line and including a control law for the dual motor control is implemented in Dymola. We will then show, how the Optimization Function of the Dymola Design Library both can be used to optimize the parameters of the motor control law and how to find the the optimal switching instant for when to change direction between the motors..

4.1 Model and control law

A model representing the robot's actuator drive-line consisting of two driving motors and one cart is implemented in Modelica as shown in Fig. 6. Each motor is connected to the load by a free shaft inertia. The backlash is modeled by the 'ElastoBacklash'-block of the Mechanics-package.

This kind of system represents an extension of a two-mass system exhibiting backlash. The latter has been well-studied in literature since the 1940s, due to the fact that in most cases the considered plants which exhibit backlash-effects may be modeled as such a two-mass system. Some older and more recent research on this standard plant can be found in [7], [8]. In the present case, the additional second motor is considered by introducing a third mass, which leads to a three-mass system

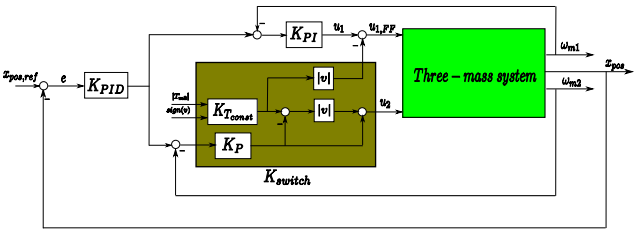

Figure 7: Nonlinear dual motor control structure of a three-mass system. K_{switch} defines the switching of the slave motor and provides the master motor with additional information through a feed-forward structure.

with two nonlinearities representing the backlash between each motor and the cart.

The implemented control structure aims to use both motors for the motion drive while the system is not in backlash and to switch the operating direction of the second motor when the system gets into backlash for fast closing the backlash gap and improving the position accuracy of the carts, and thus of the robot TCP. In the traditional case of a cart driven by just one motor limit cycles may occur in the system due to the backlash. These can be eliminated by the dual motor control. Therefore a nonlinear, smooth switching control law based on a switching variable $v \in [-1, 1]$ ('direction') and a switching function K_{switch} is designed (see Fig. 7). For first simulations nominal parameters for the controllers and the switching function are used. Herewith, the results shown in Fig. 8 are obtained. As desired, no limit cycles occur.

4.2 Parameter Optimization

This section aims to improve the system performance by optimizing the controller parameters and the switching of the second motor. For this purpose, the Optimization Function of the Dymola Design Library is used. This design tool provides several optimization algorithms and allows to optimize parameters of a Dymola model with respect to certain criteria. An introduction to the function is given in [9].

4.2.1 Controller parameter optimization

In a first step, we aim to optimize parameters for the outer-loop PID-controller, controlling the cart position x_{pos}. Therefore the following cost function is defined

$$f = \min\left(\max\left(\frac{1}{W_1}\text{riseTime}(x_{pos}) + \frac{1}{W_2}\text{overshoot}(x_{pos}) + \frac{1}{W_3}\text{settlingTime}(x_{pos})\right)\right)$$

with weights $W_1 = 1.7473$, $W_2 = 10^{-4}$ and $W_3 = 3.957$. Blocks to determine the rise time, the overshoot and the settling time are provided by the Dymola Design Library.

For the optimization an operation with a constant torque on the second motor is chosen, thus $v = -1$. The values of the weighting parameters correspond to the obtained results of the characteristics when simulating with the nominal controller parameters. The available tuning parameters are K_p, T_i and T_d of the PID-controller. The optimization is carried out using the different algorithms implemented in the Optimization Function. As starting values the nominal values of the PID-controller

$$K_P = 200, T_i = 1.5, T_d = 0.05$$

are used. The start value of the cost function is then 3.00354. The Optimization Function allows also to set bounds on parameters. We set the following bounds

$$Kp \in [100, 300],\ Ti \in [0.5, 2],\ Td \in [0.01, 0.1].$$

An overview of the results is provided in Table 1. The best results are obtained by Pattern Search and Genetic Algorithm.

4.2.2 Switching parameter optimization

In this section, we consider the switching strategy, that is when to change direction of the second motor (slave motor), see Fig. 7. The switching is based on the relative position error

$$e_{abs} = \frac{|x_{ref,new} - x_{pos}|}{|x_{ref,new} - x_{ref,old}|}.$$

A schematic view of the switching v is depicted in Fig. 9, where the parameters e_{max} and e_{min} parametrize the curve and thus can be used as tuning parameters for the optimization. For a more detailed description, see [10].

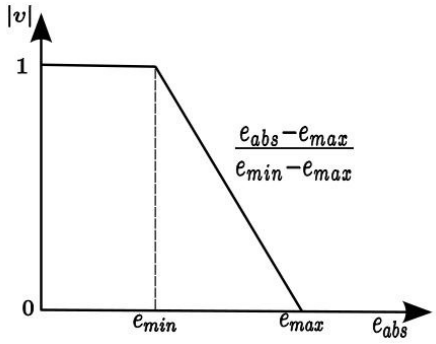

Figure 9: Switching function $v = f(e_{abs})$. $|v|$ takes the value 1, when $e_{abs} < e_{min}$. If $e_{abs} > e_{max}$, v takes the value 0.

To minimize energy and reduce overshoot for a step response in the position reference, the following cost function is defined

$$f_1 = \min(\max(\frac{1}{W_1}\int_0^T (u_1^2 + u_2^2)d\tau + \frac{1}{W_2}\text{overshoot}(x_{pos}))), \quad (2)$$

with $W_1 = \psi \cdot 2165$ and $W_2 = 3 \cdot 10^{-4}$. The weighting parameters correspond to the values obtained for the characteristic parameters of the optimization function, when operating with the previously chosen values $e_{max} = 0.25$ and $e_{min} = 0.01$, as well as an input step reference of $x_{pos,ref} = 0.1$ m and a simulation time of $t = 10$ s. As the optimization goal consists of minimizing the energy input by avoiding any overshoot in the system's step response, the energy input is additionally weighted with a factor $\psi = 10$.

For the optimization different start values and optimization methods are chosen. As initial values for the switching parameters two sets are chosen, $[e_{max} = 0.5,\ e_{min} = 0.25]$ and $[e_{max} = 0.25,\ e_{min} = 0.01]$. The cost function has then a start value of $f_1(start) = 9.54322$ and $f_1(start) = 10.9464$ respectively. The tuning parameters are limited to $e_{max} \in [0.1, 1]$ and $e_{min} \in [0.01, 1]$ in order to avoid switchings in the area where limit cycles occur. An overview of the different setups and the corresponding results is given in Table 2.

All algorithms give similar results for the optimal switching parameters. These lie in a range of $e_{max} \in [0.62, 0.69]$ and $e_{min} \in [0.40, 0.45]$. Only the SQP and the Simplex-method lead to different results when starting with $[e_{max} = 0.25,\ e_{min} = 0.01]$. However, the Genetic Algorithm and the Pattern Search seem to give more reliable results,

as the value of the cost function and the optimal switching parameters are almost identical for both initial sets.

To recheck the optimization results, the cost function f_1 is reformulated to f_2

$$f_2 = \min\left(\max\left(\frac{1}{W_1}\int_0^T (u_1^2+u_2^2)d\tau, \frac{1}{W_2}\text{overshoot}(x_{pos})\right)\right). \quad (3)$$

Then the start values are $f_2(start) = [8.8531, 0.6901]$ and $f_2(start) = [10.02246, 0.9240]$. The results of this optimization are shown in Table 3. For the cases with initial values $[e_{max} = 0.5,\ e_{min} = 0.25]$ the results are similar to the ones obtained with the previous cost function. However, for the initial set $[e_{max} = 0.25,\ e_{min} = 0.01]$ there seem to exist at least two different minima, one in the neighbourhood of $[e_{max} = 1.0,\ e_{min} = 0.1]$ and one around $[e_{max} = 0.6,\ e_{min} = 0.4]$. Again the Genetic Algorithm gives the best results. The optimal values for the switching parameters obtained with this method are almost identical to the ones obtained with the previous cost function f_1. Thus, one can conclude that a pair of parameters $[e_{max} \approx 0.6,\ e_{min} \approx 0.4]$ may satisfy the optimization goal best.

As a consequence, the switching parameters are set to $[e_{max} = 0.6,\ e_{min} = 0.4]$. Then the integrated square sum of the required input signals for a reference step of $x_{pos,ref} = 0.1$ m and a simulation time of $t = 10$ s is reduced from 2165 to 1845, which represents an energy saving of about 15 %.

5 Discussion

The authors showed that the kinematic calibration method presented gives accurate results. In comparison to the Matlab script used in [4], the method is more flexible.

Changes in the MultiBody model of the Gantry-Tau robot, which would make a cumbersome reprogramming of a calibration script necessary, can be handeled with minor changes. Such changes may include kinematic error models (e.g. to consider all 6 links in a slightly non-ideal configuration so that the end-effector orientation varies) or new robot components (e.g. to increase the robot's DOF).

A similar procedure can be used for calibrating the dynamic model of the Gantry-Tau robot or models of a different robot.

6 Conclusion and Future Work

This article shows how different functionalities of Modelica were used for modeling, identification and controller generation for the parallel kinematic Gantry-Tau robot. The work focuses on two aspects, kinematic calibration with Optimica and the evaluation and optimization of the actuator system control.

A method for kinematic calibration of the Gantry-Tau robot using Modelica and Optimica was presented and shown to give accurate results.

A nonlinear three-mass system representing the robots actuator drive-line and a previously designed switching control law has been implemented in Dymola, which Optimization Function of the Dymola Design Library has then been used to optimize the control and switching parameters.

In the future, the flexibility of the calibration method presented can be used for calibrating a kinematic error model. With a similar procedure, the dynamic model of the Gantry-Tau will be calibrated and the inverse dynamic model used for feedforward control. The dual motor control tested successfully in simulations will be implemented and tested in practice. The possibility of code generation for hardware-in-the-loop simulations from the Gantry-Tau Modelica models presented here will be considered.

7 Acknowledgements

This work has partially been funded by the European Commission's Sixth Framework Programme under grant no. 011838 as part of the Integrated Project SMErobot™. The authors would like to thank Johan Åkesson for discussions about kinematic calibration with Optimica and ABB Robotics for supporting the laser tracker measurements.

References

[1] L. Johannesson, V. Berbyuk and T. Brogårdh, "Gantry-Tau – A New Three

Degrees of Freedom Parallel Kinematic Robot", *in Parallel Kinematic Machines in Research and Practice; The 4th Chemnitz Parallel Kinematics Seminar*, 2004, pp. 731-734.

[2] SMErobot™ homepage: `http://www.smerobot.org` (2009).

[3] M. Otter, H. Elmqvist, S.-E. Mattson, "The New Modelica MultiBody Library", *in Proc. of the 3rd International Modelica Conference*, Linköping, Sweden, 2003, pp. 311-330.

[4] I. Dressler, A. Robertsson and R. Johansson, "Accuracy of Kinematic and Dynamic Models of a Gantry-Tau Parallel Kinematic Robot", *in Proc. International Conference on Robotics and Automation (ICRA'07)*, Rome, 2007.

[5] M. Krabbes and C. Meißner, Dynamic modeling and control of a 6 DOF parallel kinematics. In: Proceedings of the 5th Modelica Conference 2006, Vienna, Austria, Modelica Association, 2006.

[6] J. Åkesson, Optimica–An Extension of Modelica Supporting Dynamic Optimization. In: Proceedings of the 6th Modelica Conference 2008, Bielefeld, Germany, Modelica Association, 2008.

[7] M. Nordin and P.-O. Gutman (2002). Controlling mechanical systems with backlash - a survey. In *Automatica* 38 (pp. 1633-1649).

[8] P. Rostalski, T. Besselmann, M. Baric, F. Van Belzen and M. Morari (2007). A hybrid approach to modelling, control and state estimation of mechanical systems with backlash. In *International Journal of Control* Vol.80, No. 11 (pp. 1729-1740).

[9] H. Elmqvist, H. Olsson, S.E. Mattsson, D. Brück, C. Schweiger, D. Joos, M. Otter (2005). Optimization Design and Parameter Estimation. The Modelica Association.

[10] J. Schiffer (2009), Dual motor control for backlash reduction. Master's Thesis report, Department of Automatic Control, Lund University, Sweden, ISRN LUTFD2/TFRT--5841--SE.

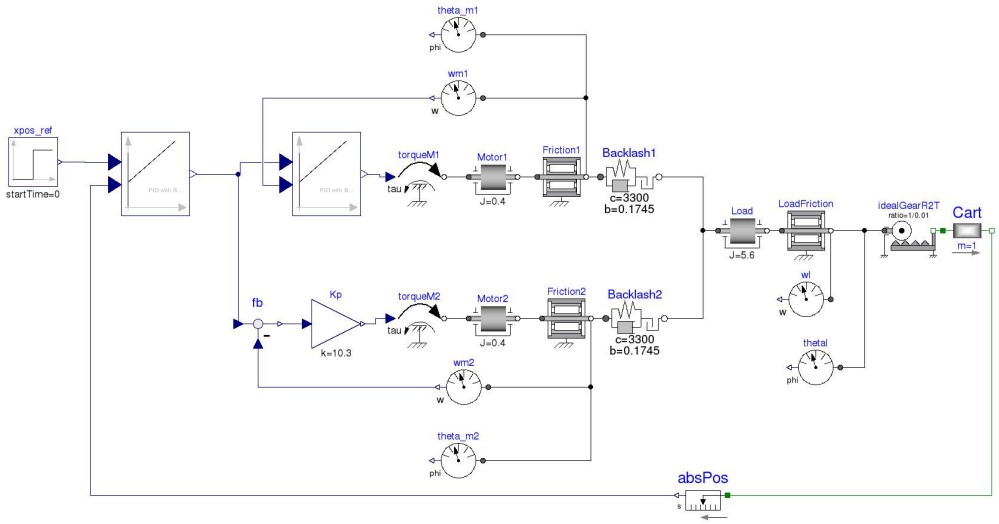

Figure 6: Three-mass system representing the robots actuator drive-line implemented in Dymola. The backlash is represented using 'ElastoBacklash'-block of the Mechanics-package.

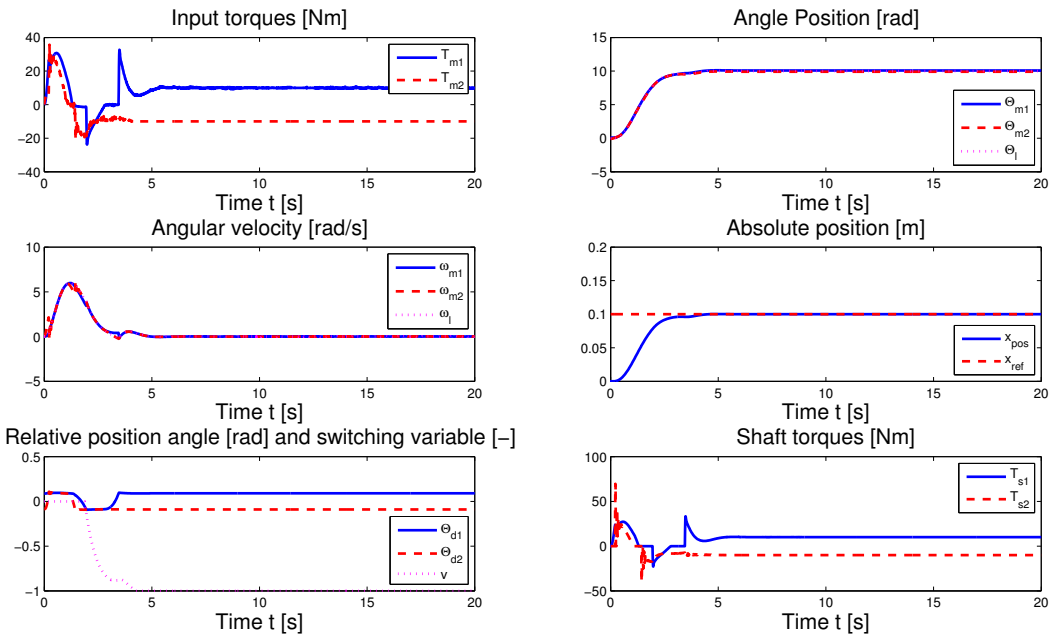

Figure 8: Dual motor control with $v = f(e_{abs})$ simulated with Dymola. The limit cycles are oppressed and the required controller energy is distributed on both motors in the beginning of the motion. Furthermore a smooth response is obtained and the strategy is robust against disturbances.

Table 1: Optimization of controller parameters. The Genetic algorithm gives the best result. The start value of the cost function is 3.00354.

Optimal values $[K_p,T_i,T_d]$	**Optimization method**	**Optimal value f**
$[292.01, 1.24, 0.01]$	Pattern Search	1.67645
$[229.32, 1.38, 0.05]$	SQP	1.82310
$[223.38, 1.4, 0.073]$	Simplex	1.88427
$[281.92, 1.25, 0.087]$	Genetic Algorithm	1.67730

Table 2: Optimization of switching parameters with cost function f_1 of Eq.(2). The Genetic algorithm gives the best results. The start value of the cost function is 10.9464.

Start values	**Optimization method**	**Optimal value** f_1	**Optimal values**
$[0.5, 0.25]$	Pattern Search	9.08898	$[0.69, 0.41]$
$[0.5, 0.25]$	SQP	9.06205	$[0.62, 0.44]$
$[0.5, 0.25]$	Simplex	9.08140	$[0.63, 0.45]$
$[0.5, 0.25]$	Genetic Algorithm	9.06148	$[0.62, 0.45]$
$[0.25, 0.01]$	Pattern Search	9.08444	$[0.68, 0.4]$
$[0.25, 0.01]$	SQP	9.47738	$[1, 0.01]$
$[0.25, 0.01]$	Simplex	9.19005	$[1, 0.24]$
$[0.25, 0.01]$	Genetic Algorithm	9.06148	$[0.62, 0.45]$

Table 3: Optimization of switching parameters with cost function f_1 of Eq.(3). The Genetic algorithm gives again the best results. The start values of the cost function are [10.02246, 0.924029].

Start values	**Optimization method**	**Optimal value** f_1	**Optimal values**
$[0.5, 0.25]$	Pattern Search	$8.50622, 0.51865$	$[0.67, 0.40]$
$[0.5, 0.25]$	SQP	$8.60525, 0.51835$	$[0.55, 0.33]$
$[0.5, 0.25]$	Simplex	$8.49155, 0.57118$	$[0.60, 0.44]$
$[0.5, 0.25]$	Genetic Algorithm	$8.49167, 0.57253$	$[0.61, 0.45]$
$[0.25, 0.01]$	Pattern Search	$8.80359, 0.65697$	$[1, 0.033]$
$[0.25, 0.01]$	SQP	$8.81529, 0.66332$	$[1, 0.01]$
$[0.25, 0.01]$	Simplex	$8.55782, 0.58015$	$[0.98, 0.28]$
$[0.25, 0.01]$	Genetic Algorithm	$8.49013, 0.57150$	$[0.62, 0.44]$