Polishing robot: vibrations due to impact

To simulate and correct unwanted behavior

M.L. Postma

TU Delft

Polishing robot: vibrations due to impact

To simulate and correct unwanted behavior

by

M.L. Postma

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday June 30, 2017 at 15:00 h.

Student number:	4098161	
Project duration:	September 19, 2016 – June 30, 2017	
Thesis committee:	Prof. dr. ir. A.W Heemink,	TU Delft, supervisor
	Ir. T. Gotthardt,	Fraunhofer IPT, supervisor
	Dr. M. Möller,	TU Delft
	Dr. J.W. van der Woude,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Preface

This research is conducted at Fraunhofer IPT in Aachen as a master graduation project for Applied Mathematics at the TU Delft from September 2016 till June 2017. The project was offered by Fraunhofer IPT on their website, where I found it. The topic of this thesis is to simulate and correct unwanted behavior in the polishing robot present at the Fraunhofer IPT. This results in a mathematical model to simulate this behavior and choosing a good correction method. The correction method used improved the simulated model greatly. The members of the thesis committee are Prof. dr. ir. A.W Heemink as TU Delft supervisor, Ir. T. Gotthardt as Fraunhofer supervisor, Dr. M. Möller as external member from the numeric analysis department and Dr. J.W. van der Woude as TU Delft advisor on system theory. This graduation project was not only a mental challenge, but also living in Aachen for these months was a great experience. I have learned a lot about conducting a research and, personally, got to know some great people and what I want to do for a next step after graduation. I would specifically like to thank my supervisor Titus Gotthardt from Fraunhofer IPT in Aachen for his guidance and support during this thesis.

M.L. Postma
Delft, June 2017

Abstract

The specific issue researched in this thesis is correcting some inaccuracies that occur when the robot is polishing certain shapes. The biggest challenge is to mathematically describe this phenomenon in order to find an appropriate correction. Since there are a lot of influences in the process (friction, movement, impact, degrees of freedom) this is very difficult to describe. The main topic of this thesis is therefore the description and simulation of this problem. The research question defined is: what is causing the jumping of the polishing tool in specific situations on non-flat surfaces and how can this be prevented?
The problem has multiple challenges due to combining a rotating movement with degrees of freedom in the lateral movement. The cause that has been found with the simulations is the flexibility in the spindle and the airbearing of the polishing robot. Together with resultant forces, caused by the rotation of the tool and a certain angle of elevation of the material, the flexibility causes the tool to jump in all directions. The goal of these simulations is to acquire data that describes the problem phenomenon. When the realistic data of the simulation is known a correction method is proposed and first simulated.
The method tested to correct this is a static state feedback where the controller is found by minimizing the H_∞-norm. The results show that this type of controller and feedback can decrease the jumping to $0-10$ % of the original displacements. This correction is still theoretical, but actual physical solutions can be implemented to test if it could solve the problem.

List of symbols

The general notation used is presented in this table. Variables or notations missing from this table are always explained. Variations of the presented notation is also possible (i.e. a caption).

$\mathbf{X}, \mathbf{Y}, \mathbf{Z}$	Coordinate frames.
$\mathbf{i}, \mathbf{j}, \mathbf{k}, \mathbf{I}, \mathbf{J}, \mathbf{K}$	Coordinate vectors.
$\mathbf{q}, \dot{\mathbf{q}}$	State vectors, generalized coordinates.
$\mathbf{x}, \dot{\mathbf{x}}$	State vectors, Cartesian displacements.
x, y, z	Displacements of polishing disc in Cartesian coordinate frame [m].
x_m, y_m, z_m	Displacements of airbearing in Cartesian coordinate frame [m].
$\mathbf{v}$	(Relative) velocity [m/s].
$\mathbf{a}$	(Relative) acceleration [m/s^2].
θ, ψ, ϕ, ρ	Angular displacements [rad].
$\underline{\Omega}, \underline{\omega}, \lambda, \dot{\theta}, \dot{\psi}$	Angular velocities [rad/s].
$\underline{\alpha}$	Angular acceleration [rad/s^2].
h	Height [m].
b	Width [m].
L, l	Length [m].
d	Deflection [m].
m	Mass [kg].
r	Radius [m].
t	Time [s].
τ	Natural period [s].
n	Percentage [-].
E	Energy [J].
H	Hamiltonian function.
K	Kinetic energy [J].
V	Potential energy [J].
F	Force [N].
T	Torque [N m].
P	(Generalized) momentum [kg m/s].
M	Moments (also torques) [N m].
G	Generalized mass matrix [kg].
$I, \mathbf{I}$	Moment of inertia [kg m^2] or unit matrix.
k	Spring constants for linear [N/m] or torsional [Nm/rad] spring.
c	Damping coefficient [Ns].
ζ	Damping ratio total energy [-].
N	Young's modulus [N/m^2].
ξ	Material elevation angle [rad].
μ	Coefficient of friction [-].
σ	Angle of friction [rad].
α, β	Angles of resultant forces in C_1 and C_2 [rad].
η	Angle between crooked radius and $\mathbf{j}$-axis [rad].
$\mathbf{u}$	Controller vector.
$\mathbf{y}$	Output vector.
$\mathbf{w}$	Exogenous input.
H_2, H_∞	Minimization norms.

Contents

1 Introduction

The definition of industrial robots according to the Robot Institute of America is [17]:"A robot is a re-programmable, multi-functional manipulator designed to move materials, parts, tools, or specialized devices, through variable programmed motions for the performance of a variety of tasks." This technical definition shows already the advantages of the industrial robot. In this graduation project a polishing robot is studied. This robot is used and improved at the Fraunhofer Institut für Produktionstechnologie (IPT) in Aachen. Fraunhofer is with more than 80 research facilities in over 40 German cities the largest research and development institute of Europe (followed by TNO). Fraunhofer has a lot of different institutes with their own topic.

The specific issue researched in this thesis is correcting some inaccuracies that occur when the robot is polishing certain shapes. The biggest challenge is to mathematically describe this phenomenon in order to find an appropriate correction. Since there are a lot of influences in the process (friction, movement, impact, degrees of freedom) this is very difficult to describe. The main topic of this thesis is therefore the description and simulation of this problem. The problem has multiple challenges due to combining a rotating movement with degrees of freedom in the lateral movement. The goal of these simulations is to acquire data that describes the problem phenomenon. When the realistic data of the simulation is known a correction method is proposed and first simulated. This correction is still theoretical, but actual physical solutions can be implemented to test if it could solve the problem.

To get to the answer of the research question multiple steps are taken and described in this thesis. First an explanation is given about the specific polishing robot that is used at Fraunhofer IPT. Then a study to the different types of problems and solving methods is given for these types of robots and applications. Next the exact research question is defined and explained in chapter 4. To answer this a mathematical background is needed with some basic explanations and definitions, which is presented in chapter 5. Then in chapters 6, 7 and 8 the mathematical model is build and expanded. The simulation results of this model are presented in chapter 9. Then the correction method and possible solution is defined, explained and simulated in chapters 10 and 11. Then finally a final conclusion and recommendations are given.

2

The polishing robot

The robot that is key in this research is a Mitsubishi Industrial polishing robot of the type RV-6SD with a CR2D-711 controller. It is a six-axes standard arm, it has 2 arms of different lengths (upper: 280 mm, fore: 315 mm)[28]. The polishing tool has a size of 20 mm and is attached to a spindle or rod of length 20 mm. The connection between the polishing brush and the spindle is a ball joint around which the polishing brush can move. The robot has 6 degrees of freedom. The controller can control 6 axis simultaneously. It is driven by an AC servo motor with a brake on all axes. The repeatability (accuracy of returning multiple times to the same position) is ±0.02 mm [28].

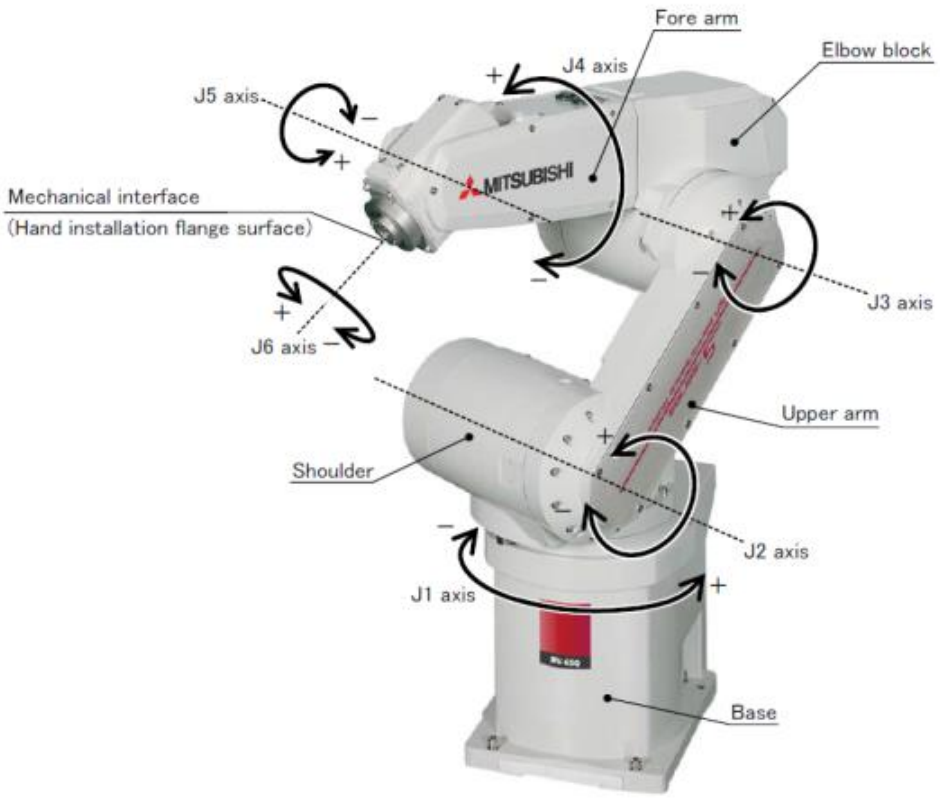

Figure 2.1: *RV-6SD Mitsubishi Industrial Polishing Robot.*[28]

The Mitsubishi Industrial polishing robot has an extra airbearing inserted at the polishing tool, such that forces during the polishing are compensated for. This airbearing is a NewWay S305002 of 50 mm inside diameter [7]. An airbearing works as a line of springs around the tool itself. It is a small cylinder where a layer of air is present (see figure 2.2). Even though having made such adjustments, forces such as friction and drag and the very low stiffness can still cause the polishing tool to drift of the programmed path. The polishing tool is being moved in a circular motion of maximum radius of 3 mm and a transverse movement of approximately 1 mm/s.

Figure 2.2: *NewWay 50 mm inside diameter air bearing [7]*

2.1. Polishing

The goal of polishing is the refinement of a surface structure for aesthetic reasons, correction prevention, remove rust and small imperfections. The polishing robot at the Fraunhofer IPT uses diamond as grind. There are other materials possible, such as silicon carbide, aluminum oxide or alumina and boron carbide [40]. The silicon carbide is used more in applications where a rough result is needed and not for smooth surface finishes. The alumina is used for fine surface finishes, since it breaks down over time. Alumina is also less expensive. The boron carbide results in a moderate surface quality and is used when fast removal is done. The diamond is the hardest grind known and used for polishing for its removal rates and end-result quality. Another advantage of diamond is its thermal conductivity.
The polishing is done in multiple steps, where in each step the polishing material has finer grains:

1. Grinding: the grains are in the tool and grinding against the material. Used to reduce the material to a suitable size or to remove large irregularities from the surface. A course abrasive is used ($> 40\mu$m) [40].;
2. Lapping: lose grains (typically $5-20\mu$m) rolling over material in a slurry. Used to produce a smooth, flat, unpolished surface. Removes subsurface damage caused by sawing or grinding;
3. Lap/polishing: variant of lapping where the tiny particles are embedded in the tool, finer surface finish;
4. Polishing: all particles (size $< 3\mu$m) are embedded in the tool. Removes the material to produce a scratch free surface [40];
5. Buffing: very fine diamond, wiping away work path with a cloth tool. Leads to a smooth, shiny result.

The objects to be polished with the robot are moulds used for optic instruments. This means that the precision of the polishing must be very good, since small errors can cause these type of instruments not to work. When looking at polishing moulds for industrial and automotive applications the relative precision is of less importance.

2.2. The use of robots

The specific difficulty with a robot such as a polishing robot is the fact that during its action the robot is in contact with its environment and due to this contact extra forces and vibrations become important. Even

though a very high precision is requested, this is not possible with some sort of adjustment to compensate for these forces and vibrations [39]. One of the main limitations in the robot is the low mechanical stiffness [39]. The advantage of using a robot for the polishing is the high productivity flexibility and the large workspace [39]. Also the fact that a robot is reprogrammable is a big advantage above machine based polishing.

Specifically using the Mitsubishi robot for the polishing shows the following. The tool path (path over which the rotating tool is moved) is not a straight line. Since the rotating tool holder is not in the middle of the mechanical interface, the tool path exists out of a rotating motion (that is slower than the tool rotation) that is moving in a direction. This means that there are a lot of moments in which the tool is still rotating, but the robot arm itself is approximately in the same position. The material that is being polished has a free geometry. With free geometry is meant that it can be any geometric form. This means in the rotating motion the tool can come at such a location with a slope and not have the tool itself press entirely on the material.

3

Previous work and scientific publications

In this chapter some basic concepts of industrial robots and improving their accuracy are discussed. This is done by describing different problems and ways to solve them. First the basic concepts and definitions are given in section 3.1. Then first methods for improving the accuracy of the robot by taking the entire robot into account are discussed. These are divided in static and dynamic calibration. Then different approaches have been used for industrial robots that are used for machining operations where the tool is in contact with the product. Different approaches are force control, vibration reduction and Eulers' rigid body equations. These methods zoom in more on the tool at the end of the robot arm and look less at the robot as an entire structure.

3.1. Definitions and basic concepts

Production/industrial robots can be classified in roughly two groups: serial and parallel robots. Serial robots are robots whose kinematic structure describes an open loop chain, while parallel robots describe a closed loop chain [43]. This means that a parallel robot has two or more independent kinematic chains connecting the base to the tool of the robot [43]. Another term often used for robots is manipulator or robot manipulator. A robot consists of links connected by joints with one end attached to a base and the other end to the end effector which is equipped with a specialized gripper or tool [43], [29]. There are mainly two types of joints: revolute and prismatic joints. Revolute joints are similar to hinges that allow relative rotation between two links. Prismatic joints allow linear relative motion between two links [43]. The arm (or mainframe) of the robot manipulator can generally move with three degrees of freedom [17]. Many industrial robots can be divided in one of the four basic motion-defining categories [17]:

1. Cartesian coordinates (three linear axes)
2. Cylindrical coordinates (two linear and one rotary axes)
3. Spherical coordinates (one linear and two rotary axes)
4. Revolute or articulated coordinates (three rotary axes)

With robot calibration is meant the process of improving the robot positioning accuracy of a given manipulator through software modification instead of changing the design of the robot or its control system [15], [42]. Calibration is needed because a robot behaves different in real life from expected. This is caused for example by small differences in the mechanism, temperature changes during use of the robot, errors due to vibrations, friction, wear of components, etc. [35].

Two types of calibration can be distinguished: static and dynamic calibration. With static calibration the identification of those parameters which influence primarily the static (time invariant) positioning characteristics of a manipulator are meant [8]. The static calibration improves the accuracy by finding better estimates of the true parameter values of the kinematic model used to control the robot's motion [51]. Optimizing these parameters will result in better positional accuracy and repeatability.

Robot kinematics can be described as the analytical study of the geometry of motion of a robot arm with respect to a fixed reference coordinate system without regard to the forces/moments that cause the motion [17]. The differences in parameters can be classified under geometric (kinematic) or non-geometric

(non-kinematic) errors. Geometric errors are constant for all robot configurations and non-geometric errors can vary for different configurations [19]. Examples of geometric errors are component length errors and assembly/joint-axis identification orientation. Examples of non-geometric errors are friction, backlash, wear, control errors, measurement errors, static loads and thermal errors [19]. To map the position of the end-effector location either a model-based (parametric) method or a modeless (non-parametric) method can be used [48]. In model-based methods an important step is the identification of accurate robot models, while in modeless methods the errors will not be explicitly modeled, but approximated [48].
Another distinction in static calibration can be made in the calculation of parameters and assumption of the reference coordinate system [49]. Absolute calibration calculates the position of the end effector with respect to one world coordinate frame. Relative calibration calculates the position of the end effector in an arbitrary coordinate frame with its own origin. Then forward calibration predicts the location of the tool in world coordinates given the joint parameters. Inverse calibration then calculates the parameters for putting the tool at a desired location in the world coordinate frame [49].

Robot arm dynamics describe the mathematical formulations of the equations of robot arm motion [17]. With dynamic calibration the identification of parameters influencing primarily motion characteristics of the manipulator (forces, actuator torques, accelerations) and dynamic effects that occur on a manipulator such as friction and link stiffness are meant [8]. These dynamic related characteristics are for example distribution of mass in the links, friction in actuators and joints and stiffness. When dynamic calibration is done the trajectory is optimized and/or tracking errors are minimized.

The model-based calibration process has the following steps [15], [42]:

1. Modeling: setting up a mathematical model describing the behavior of the robot.
2. Measurement: gathering data from the actual robot.
3. Identification: calculating the error sources and new parameters or other values from the model and measured data.
4. Compensation or correction: the model implementation step in which the new parameters following from the identification step are implemented.
5. Verification: validation of the improved model. In this step the same characteristics have to be measured again to see if the changes have had any effect on the accuracy. When the accuracy is not improved return to step 3 to adjust modeling techniques.

Different general methods can be used to improve different aspects of robot manipulators. The different aspects are: accuracy and repeatability, trajectory optimization, force control and vibration reduction. Most aspects can be improved using model-based methods.

3.2. Accuracy and repeatability: static calibration

3.2.1. Model-based static calibration

The kinematic model-based calibration is generally seen as the global calibration method that improves the accuracy across the whole robot space [15]. In [31] an overview of previous methods is given for the modeling and identification steps.

Modeling step For the model-based approach the kinematic model is mostly set up using the Denavit-Hartenberg convention [35]. This method is very popular due to amongst other properties the minimal set property [33]. The Denavit-Hartenberg convention is not unique, but gives a procedure to set up the different coordinate frames of each joint of a robot manipulator. Some variations and extensions of the D-H model are the S-model and zero-reference model. The Stone model (or S-model) estimates the six parameters for each link based on circle point analysis [15]. It still has the same problem as the D-H convention when axes are parallel [1]. The zero-reference model is based on Rodrigues equation and only uses two coordinate frames: fixed in work space and the end-effector [29]. The zero-reference position is the wanted position for the robot when all joint displacements are zero. In [10] a 5 parameter extension of the D-H convention is used.

To get good results from the calibration the model parameters must satisfy three properties: continuous, proportional and completeness [35]. These properties lead to a parameter set that can handle small deviations.

Another method to describe the model is using the POE formula. This formula came up in the 90s and represents the forward kinematics as a product of matrix exponentials and is based on the concept of a one-parameter subgroup of the Lie group. This is based on a geometric interpretation of classical screw theory [33]. This came up as a proposition to overcome singularity, but also to avoid calculating a derivative and introduce a kinematic model that can also be used for different applications [33].
In [13] a local POE formula is used. All joint axes are now expressed in their respective local frames, instead of a base frame representation as in [33]. The advantage of this method is that the local coordinate frames can be arbitrarily assigned to the links, such that the kinematic calibration becomes a process of finding new local coordinate frames to reflect the correct geometric characteristics of the robot [13]. The big difference between the two methods is that the Denavit-Hartenberg parameters are with respect to the previous link frame, while the twist coordinates of the POE representation are with respect to a base frame.

Other less popular methods are for example the zero-reference model [33] and continuous and parametrically complete (CPC) modeling approach (a six parameter model which is singularity-free) [13].

To conclude the Denavit-Hartenberg convention is the standard procedure for the modeling step in this type of problems. Some literature such as [30] do however state that the product of exponentials is a superior alternative to the use of the Denavit-Hartenberg parameters.
Identification step The most common method to calculate the optimized parameters is the least squares algorithm. Other methods have been used to calculate the parameters such as nonlinear optimization (Levenberg-Marquardt, Newton, etc.), iterative linearization, (extended) Kalman filter and maximum likelihood.

In [38] a global model for error estimation is build with a maximum likelihood estimate on a robot with 6 degrees of freedom. This maximum likelihood estimator takes into account backlash in transmission units, measuring device errors and manufacturing tolerances. The best results were an improvement of the error to 0.5 mm without considering the non-geometrical errors. This is an improvement from an error of factor 10 to a region of $0.5 - 1.0$.

In [51] the weighted least squares method is used instead of the standard least squares method. They showed that 20 % decrease in average position error is possible compared to using the least squares method. This method is expected to be useful when a large number of similar robots must be calibrated. In this situation the random error behavior becomes worthwhile, since it reduces the number of actual measurements.

In [35] a comparison between a nonlinear optimization procedure, an iterative linearization and an extended Kalman filter is made on a SCARA robot. All three methods generally converged, each with different computation times. The parameter estimations and the efficiency of the extended Kalman filter were better than the nonlinear optimization estimations. The parameter estimations of the linearized iterator were also better than the ones of the nonlinear optimization method, but this method is not efficient and showed some convergence problems for some parameters. The final conclusion of [35] was that the extended Kalman filter was by far the best method by being fast, reliable and being able to give an estimation of the uncertainties.

In [37] a comparison is made between the least squares estimation and the extended Kalman filter on a simulated 7 degrees of freedom MyBot humanoid arm. For this a non-singular Jacobian matrix is derived that shows the influence of each parameter error on the difference of the measured and theoretical positions. The least squares estimation is a non-iterative method that calculates the parameters using the singular value decomposition and optimizes the root-mean square residual error of the model [50]. The extended Kalman filter is again used to approximate the state when there are uncertainties. Both methods are iterated until the desired norm is reached. [37] showed that the least squares estimate converges faster than the extended Kalman filter, but the Kalman filter has better performance than the least squares estimation, especially when only a small number of measurements are used.

3.2.2. Modeless static calibration

Another approach is the modeless calibration. In this type of calibration the calculations are based on approximation of robot kinematic relationships. Relationships such as between the robot joint readings and its position errors or between the robot positions and its position errors. Methods that have been used here

are for example radial basis function networks (RBFN), artificial neural networks (ANN), fuzzy logic, genetic algorithm and utilization of polynomials (Fourier, ordinary, Jacobi, Laguerre, Hermite and Bessel).

The advantage of a modeless method is that the kinematic modelling and identification step can be skipped [48]. In these type of methods the robot workspace is divided in a grid on which all position errors are measured. Then an error compensation can be realized by interpolating errors from its neighboring grid points [48]. The big disadvantage is that a choice has to be made between calibration accuracy and computational load (number of grid points), [48].

The modeless method has two major steps, [48]. The first step is identifying all position errors for the grid points. The second step then is targeting a random location and compensation the error with a linear interpolation on the data from the first step [48].

In [24] both geometric and non-geometric errors are compensated for the D-H joint parameters. For the other parameters only the geometric error is considered. The experiment is conducted on a six axes industrial robot manipulator. The approach in [24] is to divide the workspace of the robot into subregions where the parameter errors can be calculated. Then the continuous parameter error functions can be obtained by interpolation, in this case by a Radial Basis Function Network (RBFN).

3.2.3. Combining modeless and model-based approaches

The down side on almost all model-based approaches is that the non-geometric errors cannot be correctly included in the model itself [31]. In [10] the non-geometric errors are included in the kinematic model and taken into account when using the least squares method. The measured absolute accuracy before the calibration is then 2.82 mm with standard deviation 0.88 mm. Calibrating only the geometric errors gives an absolute accuracy of 0.69 mm with standard deviation 0.34 mm. Calibrating the combined method gives 0.58 mm with standard deviation 0.21 mm.

In [3] a classical least square estimation technique is used to determine the parameters of the system. Then using this model-based technique the polynomial coefficients are determined to approximate the position error as a function of the joint angles. The error in [3] can be reduced by as much as 57 %. With just the least square estimation this reduction is around 35 %.

In [4] the geometric errors are modeled with a nonlinear least squares method and the non-geometric errors are compensated for with neural networks. This method is used on a seven degrees of freedom articulated robot. In the identification step a nonlinear least squares method is first used to calibrate the geometric parameters. Then two methods are used for accounting for the non-geometric parameters.

In [31] itself a combination of model-based and modeless methods is used. For the geometric errors a model-based model is used where a model using the D-H convention is set up. The identification of the geometric errors is then done using the extended Kalman filter. The non-geometric errors are included using an error compensation with an artificial neural network.

3.3. Trajectory optimization: dynamic calibration

From already early on (1985) methods have been developed for robot dynamic parameter identification. The goal of the dynamic calibration is trajectory optimization (path accuracy). This means improving the accuracy of the robot during its motion. In the past improving the path accuracy has not been done much for two reasons [2]: there were no measuring systems available to measure the accuracy of the path during the motion and, compared to the static pose accuracy, the path accuracy was of minor importance. The trajectory planning can be done in the joint-variable space or in the Cartesian space [17]. The joint-variable space has the advantages that the trajectory is planned in the controllable variables, the planning can be done in near real time and the joint trajectories are easier to plan. The disadvantage is that it is difficult to determine the locations of the links and the end-effector during the motion.

The same steps for dynamic calibration can be distinguished as for the static calibration [2]. The modeling step now means setting up a dynamic model describing not only position, but also acceleration and velocity. Some of the properties that are taken into account for dynamic calibration are elasticity and joint backlash [2].

According to [11] two different dynamic calibration procedures are mostly used: Newton-Euler and Lagrangian. The Newton-Euler procedure uses a recursive formulation of the Newton-Euler motion equations. The Lagrangian procedure is based on the Lagrangian energy approach. The actual estimation of the dynamic parameters is often done with the least-square algorithm.

In [18] three different optimization methods are used to find a trajectory that minimizes the condition number. The three methods are a heuristic method, a quasi Newton method and a iterative gradient conjugate type of method by Powell (1964) and Minoux (1983). This last method gave the best results, although no actual results were presented in [18].

In [9] two approaches are applied; the Newton-Euler approach and the energy theorem using the Hamiltonian. The actual identification is done using a least squares method. Based on the obtained model a feedforward model based control is applied in a complete and partial form.

In [2] an offline dynamic calibration is done by modelling in analogy of the static calibration and by applying an on-line correction of the desired values. The results, according to [2], was an improvement of the path accuracy up to 85 %.

In most studies the formulations used are Euler-Lagrange and Newton-Euler. In some various subforms are defined such as: Uicker's Lagrange-Euler equations, Hollerbach's Recursive-Lagrange equations, Luh's Newton-Euler equations, Lee's generalized d'Alembert (G-D) equations. The structure of these formulations may differ as they are obtained for various reasons and purposes, but describe the same dynamic behavior [17].

3.4. Contact forces: force control and vibration reduction

All previous methods address the positional accuracy and repeatability (static calibration) and the trajectory optimization (dynamic calibration). What has not been addressed is the problem of forces and vibrations caused by the contact with the material (environment). As mentioned before this is caused, amongst other things, by the low stiffness of the robot [39]. The basic variables involved in the force control methods are position, velocity, acceleration and force.

3.4.1. Force control

In [39] the adoption of force control strategies is discussed for machining operations, specifically drilling. Machining operations are operations that require a high power and a good precision. This could also be operations which have force contact with the environment, such as polishing, milling and grinding. Here passive and active force control are explained. Passive force control adjustments to the robot itself are made with, for example, springs and dampers in the robot end effector. Active force control uses controller parameters to perform the robot tasks. This can be done in two ways [27]. The first is directly modifying the robots controller parameters, which is a lot of work since the controller architecture is usually closed and the required bandwidth for force control might not be available in the robot[27]. The second method is indirectly using a special purpose force controlled auxiliary device attached to the robot arm.

[21] and [52] give an overview of the types of force control methods. For more details the sources from [52] can be studied. This gives the following classical types of methods [21], [52]:

- Indirect force control:
  - Methods using the relation between position and applied force:
    - ◇ Stiffness control by position only (passive): end-effector has an extra mechanical system composed of springs (or springs and dampers). Successful for specific tasks as handling pegs and orientations [52].
    - ◇ Stiffness control by force feedback correction (active): programmable spring.
  - Methods using the relation between velocity and applied force:

- ◇ Impedance control: the relationship between the velocity and the applied force [52]. These methods vary based on how the measured signals are used. [27] defines it more specifically as imposing a dynamic relation between the end-effector position error and the force of interaction with the environment (desired impedance). According to [27] this desired impedance is usually chosen linear and of second order (as a mass-spring-damper system).
- ◇ Admittance control: inverse relation of the impedance control [52]. Concept: making modifications of the admittance of a position controlled robot to enable the execution of constrained tasks [52].

- Direct force control:
  - Methods using direct position and applied force:
    - ◇ Hybrid position/force control: combines force and torque information with positioning data, based on Mason's concept [52]. Position control and force control can be studied separately and the control laws can be designed independently. A lot of research has been done to this method.
    - ◇ Hybrid impedance control: combines impedance control and hybrid position/force control into one strategy [52].
  - Methods using direct applied force feedback:
    - ◇ Explicit force control: two categories, force-based and position-based. Position-based explicit control has the same structure as admittance control. For force-based explicit methods the measured forced is used directly to form an force-error vector [52].
    - ◇ Implicit force control: control of position based on the pre-definition of position for a desired force [52].

[52] also mentions an extra category of advanced force control methods for accurate force tracking or perfect task accomplishment when there are unknown parameters and/or uncertainties.

The impedance control used in [27] preserves the stability for contour following operation. It is expected that there will always remain some force peak at the moment of impact, i.e. the moment the tool touches the material. However the impedance control shows some natural robustness to the impact phase. There is still a peak force, but the oscillations afterward quickly vanish which shows stability preservation [27].

An indirect active force control method is used in [39] by using a pre-programmed Machining Force Control Pressure package of ABB. The downside of using such pre-programmed functions is that it is basically a black box of which the parameters have to be determined empirically. The force and torque need to be described to use this method in [39]. The results of the displacement during the contact force are an improvement. When starting with 20 N during the process and 10 N in the beginning a difference in displacements is found when having first contact from 0.1 mm to 0.06 mm.

3.4.2. Vibration reduction

Because of the low stiffness in the joints vibration exists in the robot arm [12]. This vibration problem tends to be nonlinear and time-varying. This means that a set of partial differential equations can be used to describe the vibrations. Then the boundary conditions are formulated in such a way that the restrictions of the robot manipulator and the contact forces are included in the model.

In the literature two approaches to solve this can be found: open-loop feedforward methods and closed-loop feedback methods [12]. The closed-loop methods have better performance, but increase the complexity of the system and are computationally more expensive. The open-loop methods do not have these drawbacks, but are less robust against disturbances and parameter variations [12].

in [12] an extended version of the input shaping technique (IST) is applied. The basic idea of this method is to calculate a response function that exactly corrects the displacement vibrations. This technique was first used in 1990 by Singer and Seering. Since then more extensions have been made to handle more complex non-linear systems.

4

Research question

In this chapter a precise description of the research question and solution method is given. This takes into account the work already done in optimizing robot manipulators (see chapter 3).

4.1. The phenomenon

As mentioned before, the robot arm at hand is used for polishing moulds used for optic instruments. This means that the polishing must give a very accurate result. There are however some situations in which the robot cannot keep the polishing brush exactly on the desired position. In these situations the polishing brush can tilt and precess and continues to jump over the material. This phenomenon is not always happening, but specifically when the polishing brush is moved in a material shaped as a tube. This indicates that some external forces on the polishing brush are causing this. The tool keeps vibrating and precessing from side to side when this starts. This can be caused by two things: a continuing precessing due to new impulses or a resonance induced by the right forces. Stated differently: is the reason for the jumping movement an impulse or a certain vibrational frequency that causes resonance? Another explanation could be that the vibration is caused by the controllers of the robot itself. The robot arm positions itself perpendicular to the material in the middle of the outer moving circle. Then the polishing brush makes a circular movement with a radius of maximum 3 mm. This means the spindle or rod does not have to be perpendicular to the material it is touching. This means that when the robot controllers notice an irregularity to correct, it is only of the inner center of the movement circle and not of the correct position. This would have to happen often enough to cause an effect. This situation is less likely, since measurements of the deviations by the robot are only done at the joints. This would mean that the joints would be moving or vibrating the same way in their degrees of freedom. The observed movement of precessing and tilting however does not seem to work through the entire robot, but seems to be localized to the end effector part with the tool.

The general question for this problem is: what is exactly causing this and can this be corrected?

4.2. Possible modeling techniques

The first big question for choosing a modeling technique is: is it necessary to take the entire robot arm into account or only a portion of the system. This depends not only on the problem to be solved, but also on the information available.

Is it expected that the entire structure is necessary for solving this problem? The answer to this question is actually no. Due to the air bearing added in the tool head, an extra flexibility has been added. In order to be able to correct displacements the air bearing must give extra flexibility. A reasonable explanation of the phenomenon is that the forces of the contact and impact feed the tool movement to tilt in the air bearing. This would mean that the flexibility in the joints might not influence this greatly. Combining this with the discussing in the last section, it seems reasonable to start by assuming the largest contribution to this problem is localized in the end-effector with the rotating tool. This is also convenient since the Mitsubishi robot has a closed structure, meaning many parameters are not known. For these reasons the general static and dynamic calibration will not be used.

The first step is then to look at the part of the robot manipulator that exists out of the airbearing, holder and tool head. This eliminates any geometric information needed about the robot arm itself! Then the question remains how to model this. Since the model exists out of a spring system combined with a rotating element, two modeling options seem most promising.

The first option is defining this as a rotating beam system in which the influence of the air bearing and contact forces are included as boundary conditions. This system is then integrated over the length of the tool. The advantage of this method is that any resonance can be described in detail. The disadvantages are that the specific geometry of the tool head and areas of contact are not included and that these models can become very complicated.

The second option would be to start from the tool head. By assuming that there is some force or impulse that is causing the tilting or precessing and using this force and impulse in an extended model. The tool head can first be seen as a rotating disk. Here a rigid body can be defined with Euler's equations of rotational movement. The effect of tilting and precessing is then similar to that of a gyroscope. Then to include the airbearing and torsional stiffness a second equation module can be added. It is expected that the impulses and the forces will cause motion in the extended module of the airbearing and spindle. The advantage of this method is the relative simplicity of the method. The resulting system can still be complicated, but the essential method is more straightforward.

When the model is actually working and able to describe the phenomenon a solution must be found. This can still be done in a passive or active way. First the computer model can indicate what can actually solve this: can some mechanical part be added or is some type of controller necessary? If the cause of the tilting and precessing is indeed local to the tool head and air bearing, adding a controller to the entire robot might not work sufficient. This would have to be a controller defined outside the existing controller and every calculation the found adjustment has to go through the existing controller and still has to have the correct result. This also means that the adjustments that can be made are dependent on the calculation frequency of the existing controller. For this reason some local adjustment (passive or active) would be best.

4.3. Research question

To summarize the research question is:

What is causing the jumping of the polishing tool in specific situations on non-flat surfaces and how can this be prevented?

This is done by first assuming this is mainly caused by flexibility in the end-effector. Both methods discussed in section 4.2 have their merits, but based on the observations and discussions the second method is used. For this model Euler's rotational equation of motion is used for rigid bodies. The model actually seems similar to that of a gyroscope. This information is used to formulate the model. For the airbearing and spindle model a relative standard spring-damper-mass system can be derived.

If the simulations are evaluated and validated with the situation a theoretical solution in some feedback law can be made of which the actual applicability must be studied. This would be the active approach of solving the problem. The passive approach can also be used by researching possible mechanical adjustments on the end-effector.

5

Mathematical background

5.1. Spring-mass-damper systems

There are two types of springs: translational springs and rotational springs [46]. Translational springs are springs with no rotating object on both sides and forces acting on one or both sides of the spring. When a linear spring is assumed, the translational stiffness or spring constant k_L is constant. The general formula for the force is then

$$F_k = k_L x, \tag{5.1}$$

where F_k is the spring force and x the relative displacement. Normally the spring is assumed to have a negligible mass. The rotational or torsional spring has stiffness k_T and an external torque on one or both sides of the spring is applied. It satisfies the following relation [46]

$$T_k = k_T \theta, \tag{5.2}$$

where T_k is the torque applied and θ the relative angular displacement.

There are three types of well known damping [46]:

1. Viscous damping (fluid damping);
2. Coulomb damping (dry friction damping);
3. Structural damping (hysteresis damping).

With viscous damping two surfaces are separated by a liquid film. Due to a force the damping exists in opposite direction of the force and also depends on the fluid properties. The exact friction force is difficult to describe, so most of the time a linear relationship is used for the damper. The magnitude can then be described as

$$F_c = c v, \tag{5.3}$$

where F_c is the damping force, c the damping coefficient and v the relative speed of the mass. When $v > 0$ this is positive damping, which helps to stabilize the system. If $v < 0$ the damping becomes self-excited vibrations and adds energy into the system. This is called negative damping [46]. Again a difference can be made between translational and rotational viscous dampers. The formulas are analogous with those of the springs. For a translational viscous damper the equation assuming linear properties becomes

$$F_c = c v. \tag{5.4}$$

For a rotational damper this becomes

$$T_c = c_T \omega, \tag{5.5}$$

where c_T is the torsion viscous damping coefficient and ω is the relative angular velocity.

Coulomb damping happens between two dry surfaces. If the relative velocity between the two surfaces is zero the friction force is static. This static friction force can attain a maximum until sliding occurs. Then this becomes kinetic friction force. In most situations the assumption is made that the kinetic friction force is nearly constant [46].

Structural damping is the type of energy dissipation due to the stress and strain. Because of this stress and strain an internal friction is present in the material [23]. Other types of damping are for example air damping and displacement-squared damping [23]. In [23] all these 5 types are discussed and explanations of the damping coefficients are given. Another method that is presented in this book is making an equivalent damping coefficient to the linear viscous damping. This is done by introducing the dissipated energy per cycle, ΔE_i for damping mechanism i. Then the equivalent viscous damping constant is

$$c_{eq} = \frac{\sum_{i=1}^{n} \Delta E_i}{\pi \omega A^2}, \tag{5.6}$$

where ω is the frequency of the system and A is the amplitude of the solution of the system. Instead of using the ΔE_i it is sometimes convenient to assume a certain percentage of the systems energy is dissipating. Then using $n \in [0,1]$ as percentage the equivalent damping coefficient would become

$$c_{eq} = \frac{n * E_{max}}{\pi \omega A^2}. \tag{5.7}$$

The big advantage of such a simplified definition is that not all different types of damping need to be identified and less detailed information is needed.

These dampers and springs are part of a system of equations. If a single mass m is studied attached to a wall of some kind with a spring and a damper and no external force, its movement can be described by [23]

$$m\ddot{x} + c\dot{x} + kx = 0. \tag{5.8}$$

The same can be done for a single rotating disc with a certain torsional stiffness, moment of inertia I and damping effect. Then the movement can be described with [23]

$$I\ddot{\theta} + c\dot{\theta} + k\theta = 0. \tag{5.9}$$

5.2. State-Space representation

To work with systems and controls a state space representation of the problem is very useful. For this a definition of state variables is needed. These form the smallest possible set of independent variables that completely describe the state of a system [46]. The idea is that at a certain time these variables can describe the system behavior. The independence means that the variables cannot be expressed as functions of each other and the inputs. The state-variable equations are then formed by the time derivative of the state variables as function of the state variables and inputs [46]. The final basic state-space representation of the system then looks like

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{5.10}$$

$$\mathbf{y} = C\mathbf{x} + D\mathbf{u}, \tag{5.11}$$

where A is the state matrix, B the input matrix, C the output matrix and D the direct transmission matrix [46]. In a lot of situations the $D = 0$, since the input vector is not that interesting for the measurements. The $\mathbf{x}$ is the state vector (another common notation is $\mathbf{q}$) [1], $\mathbf{u}$ the input vector and $\mathbf{y}$ the output vector. The notation $\dot{\mathbf{x}}$ is used for the time derivative of a variable or function. For other derivatives the notation $\frac{d}{d\mathbf{x}}$ or $\frac{\partial}{\partial x_i}$ is used. An extended model system can be defined if there is a disturbance present in the system and if not only there is an output vector $\mathbf{y}$, but also an output that is relevant to the outside world $\mathbf{z}$. This full system looks like [45]

$$\dot{\mathbf{x}} = A\mathbf{x} + B_w\mathbf{w} + B\mathbf{u}, \tag{5.12}$$

$$\mathbf{z} = C_z\mathbf{x} + D_{zw}\mathbf{w} + D_z\mathbf{u}, \tag{5.13}$$

$$\mathbf{y} = C\mathbf{x} + D_w\mathbf{w}. \tag{5.14}$$

$$\tag{5.15}$$

Here $\mathbf{w}$ is an exogenous input. This means a disturbance or noise. With these type of systems the goal can be to find a feedback controller to correct some type of unwanted behavior.

[1] For Cartesian coordinates $\mathbf{x}$ is used and for generalized coordinates $\mathbf{q}$. Generalized coordinates can also be angles, line elements, etc.

5.2.1. Feedback controller

There are different types of feedback that can be used. The most basic state and output feedback methods are the state feedback and output feedback. Two other methods are the dynamic compensator and the disturbance rejection problem [34]. As stated before, a basic system with input $\mathbf{x}$, output $\mathbf{y}$ and controller $\mathbf{u}$ looks like

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{5.16}$$

$$\mathbf{y} = C\mathbf{x}. \tag{5.17}$$

The state feedback controller is defined as $\mathbf{u} = D_c\mathbf{x}$ such that

$$\dot{\mathbf{x}} = (A + BD_c)\mathbf{x}, \tag{5.18}$$

where D_c is chosen such that this closed loop system has the desired behavior. If the state is not available the output can be used, i.e., $\mathbf{u} = H\mathbf{y}$ output feedback, such that

$$\dot{\mathbf{x}} = (A + BHC)\mathbf{x}. \tag{5.19}$$

An extension of the state feedback is adding the possibility of influencing the system after the feedback, i.e.,

$$\mathbf{u} = D_c\mathbf{x} + G\mathbf{x}_{new}, \tag{5.20}$$

where $\mathbf{x}_{new}$ is the new input. When there is an online dynamic system, a dynamic compensator can be used. This uses a control law and an observer of the system. This method is relevant if it is possible to feed online information to the system. The last useful method is the disturbance rejection method. This aims at correcting a system given by

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + B_1\mathbf{w}, \tag{5.21}$$

$$\mathbf{y} = C\mathbf{x}, \tag{5.22}$$

$$\mathbf{z} = C_1\mathbf{x} + D_1\mathbf{w}. \tag{5.23}$$

Here $\mathbf{w}$ can be interpreted as the disturbance. The criteria described in [34] are however only valid in perfect defined conditions. The idea behind this method is used to make an optimization method for these type of problems [45]. First two types of norms are introduced; H_2 and H_∞ norms. Minimizing the H_∞ norm means that the peak values of the system are being minimized. Minimizing the H_2 norm means minimizing the impulse energy present in the system. Depending on the goal or focus of the wanted controller one of the norms can be used.

The output feedback should be used if there is something known about measurements or information that can be taken from the robot arm and end-effector.

5.3. System definition

In the definition of systems an important concept is the number of degrees of freedom. These are defined as the number of independent generalized coordinates that specify the configuration of the system [46]. Another definition is that the number of degrees of freedom equals the number of dependent generalized coordinates n that specify the configuration of the system minus the number of independent equations of constraint m, i.e.

$$\#DOF = n - m. \tag{5.24}$$

There are also different types of inputs that can be used for mechanical systems. The two general types are generalized forces and generalized displacements [46]. The generalized forces are forces or moments and generalized displacements can be displacement, velocity or acceleration in translational or rotational direction.

The methods of deriving the system equations can differ based on the objective. The most common methods are the force based Newton method and the energy based Hamiltonian and Lagrangian approaches. Each of these approaches will be explained in this section.

5.3.1. Newtonian mechanics

To start, the three laws of Newton of mechanics are [5]

1. A body remains at rest or in uniform motion unless acted upon by a force.

2. A body acted upon by a force moves in such a manner that the time rate of change of momentum equals the force. In formula
$$\mathbf{F} = \dot{\mathbf{P}} = m\dot{\mathbf{v}} = m\mathbf{a}, \tag{5.25}$$
with $\mathbf{P}$ the momentum, m the mass, $\mathbf{v}$ velocity, F the force and $\mathbf{a}$ the acceleration.

3. If two bodies exert forces on each other, these forces are equal in magnitude and opposite in direction. Or equivalent: action = − reaction.

Setting up a system with the second law of Newton gives for each body a differential equation in which all forces acting upon the body are identified (gravity, friction, etc.). The coordinates that are used must be Cartesian for using Newton's laws. With more degrees of freedom and complex structures finding solutions to the equations of motion following from Newtons laws gets more difficult [6]. Different modeling techniques could better be used in these situations.

A related law to Newton's second law is Euler's law. This is used for rotational movements where moments about a certain reference point O are summed. This law is generally given by [23]

$$\sum \mathbf{M}_O = I_O \underline{\alpha}, \tag{5.26}$$

where $\mathbf{M}_O$ are the moments around O, I_O is the mass moment of inertia of the mass around O and $\underline{\alpha}$ is the angular acceleration vector.

5.3.2. Lagrangian

To obtain an equivalent system to that following from Newton's laws is often done to get around some practical difficulties. One of the most used formulations is that of Lagrange. This gives for the Lagrangian function $\mathscr{L}$ that this equals the difference between the kinetic energy K and potential energy V, or $\mathscr{L} = K - V$. The potential energy function is such that the force acting on each particle is determined by [5]

$$F_i = -\frac{\partial}{\partial q_i} V(q_1, \cdots, q_n). \tag{5.27}$$

Then the system of equations of motion are given by [14]

$$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial \mathscr{L}}{\partial \mathbf{q}} = \mathbf{F}_g, \tag{5.28}$$

where $\mathbf{F}_g$ is a vector of generalized forces acting on the system. Since the potential energy is only a function of the generalized coordinates and not its derivatives, it is possible to rewrite this equation to

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial}{\partial \mathbf{q}}(K - V) = \mathbf{F}_g. \tag{5.29}$$

The kinetic energy is equal to

$$K = \frac{1}{2} m\mathbf{v}^2 = \frac{1}{2} m\dot{\mathbf{q}}^2 \tag{5.30}$$

or some sort of variation or sums of these terms with more complicated systems. This can be generally described as

$$K(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^T G(\mathbf{q}) \dot{\mathbf{q}}, \tag{5.31}$$

where the $n \times n$ inertia (generalized mass) matrix $G(\mathbf{q})$ is symmetric and positive definite for all $\mathbf{q}$ [14]. The potential energy is for example due to gravitation or the energy stored in a spring. In the gravitational situation the potential energy is given by $U_g = mg\Delta h$ and for a one dimensional spring this becomes $U_s = \frac{1}{2} kx^2$.

5.3.3. Hamiltonian

Hamilton's principle shows that the path that a dynamical system follows is that which minimizes the time integral of the difference between the kinetic and potential energies [5]. In terms of the Lagrangian function this means that the following action integral must be minimized [5]:

$$\int_{t_1}^{t_2} \mathscr{L}(\mathbf{q}, \dot{\mathbf{q}}, t)\, dt. \tag{5.32}$$

In this principle the coordinates do not have to be Cartesian, but are generalized coordinates and velocities.

The link between the Euler-Lagrange equation and the Hamiltonian equations is described with defining the generalized momenta for any Lagrangian as [14]

$$\mathbf{P} = \frac{\partial \mathscr{L}}{\partial \dot{\mathbf{q}}}, \tag{5.33}$$

and defining the state vector to be $(q_1, \cdots, q_n, P_1, \cdots, P_n)^T$. Then the equation for the momentum using (5.31) simply becomes

$$\mathbf{P} = G(\mathbf{q})\dot{\mathbf{q}}. \tag{5.34}$$

So the final Hamiltonian equations become

$$\dot{\mathbf{q}} = \frac{\partial H}{\partial \mathbf{P}}(\mathbf{x}, \mathbf{P}), \tag{5.35}$$

$$\dot{\mathbf{P}} = -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{q}, \mathbf{P}) + \mathbf{F}_g. \tag{5.36}$$

Here H is the Hamiltonian function and is defined as the total energy of the system.

With both the Hamiltonian and Lagrangian modeling it needs to be clear when which description is used. For this two types of energies are introduced: energy and co-energies [25]. The relation between the two depends on the energy and the complementary relation between the energy and co-energy. In the situation of kinetic energy for example there is a linear relation between the velocity v and the generalized moment p. Then using Newton's second law, the kinetic energy of a mass is the integral of the work done by the force, i.e. [25]

$$K(\mathbf{P}) = \int \mathbf{F} d\mathbf{x} = \int \frac{d\mathbf{P}}{dt} d\mathbf{x} = \int d\mathbf{P} \frac{d\mathbf{x}}{dt} = \int \mathbf{v} d\mathbf{P} = \int \frac{\mathbf{P}}{m} d\mathbf{P} = \frac{\mathbf{P}^2}{2m}. \tag{5.37}$$

The kinetic co-energy cannot be calculated from the work like this, but is defined as the integral of the momentum with respect to the velocity which is equivalent to [25]

$$K^*(\mathbf{v}) = \mathbf{P}\mathbf{v} - K(\mathbf{P}) = \frac{m}{2}\mathbf{v}^2. \tag{5.38}$$

Since the kinetic energy and kinetic co-energy are both quadratic in $\mathbf{P}$ and $\mathbf{v}$ respectively the two equal each other. Thus it is not necessary to make a distinction between $K(\mathbf{P})$ and $K^*(\mathbf{v})$. This is all because the velocity and momentum are linearly related. For potential energy the same distinction can be made. Then the co-potential energy is defined as the integral of the displacement with respect to the force, i.e.

$$V^*(\mathbf{F}) = \int \mathbf{x} d\mathbf{F}. \tag{5.39}$$

This leads to not only a Lagrangian and Hamiltonian, but also the possibility to define a co-Lagrangian and co-Hamiltonian.

In this new definition paradigm the Lagrangian is defined as

$$\mathscr{L} = K^*(\mathbf{v}) - V(\mathbf{q}), \tag{5.40}$$

where $K^*(\mathbf{v})$ is the kinetic co-energy and V the potential energy. The co-Lagrangian is then defined as

$$\mathscr{L}^* = V^*(\mathbf{F}) - K(\mathbf{P}). \tag{5.41}$$

Since the Hamiltonian is the total energy it is given by the sum of the potential and kinetic energy, i.e.

$$H(\mathbf{q}, \mathbf{P}) = K(\mathbf{P}) + V(\mathbf{q}) = \frac{1}{2}\mathbf{P}^T G^{-1}(\mathbf{q})\mathbf{P} + V(\mathbf{q}). \tag{5.42}$$

The co-Hamiltonian is then defined as

$$H^*(\mathbf{v}, \mathbf{F}) = K^*(\mathbf{v}) + V^*(\mathbf{F}) = \frac{1}{2}\mathbf{v}^T G(\mathbf{q})\mathbf{v} + V^*(\mathbf{F}). \tag{5.43}$$

5.4. Numerical integration

To solve the second order differential equations that follow from the system definition a numerical method is often used. This since in a lot of situations an analytic solution is not possible. The type of integration that is interesting for these type of problems are numerical time integration methods with an initial condition. The most simple integration methods are the one step Euler forward and backward methods. If we have a one dimensional system of the form

$$\dot{x} = f(t, x), \tag{5.44}$$

then for example the Euler forward method calculates the unknown function value x each time step with step size h by [47]

$$x_{n+1} = x_n + hf(t_n, x_n). \tag{5.45}$$

There are a lot of methods, but using Matlab gives one of the most stable methods called the ode45 solver. This solver uses the Runge-Kutta integration method of the 4th and 5th order and varies the step size (which does not have to be uniform) until the two solutions are close enough to each other.

6

Polish brush model

The rotation of the tool is a very important factor in the movement of the polishing brush. In this chapter a model is set up of the polishing brush. This is simplified to a round disc moving over a non-flat surface. The model can be derived using Euler's rotational equations. For the derivations of equations [22] is used.

6.1. Tool model

First assume that there is only a tool polishing some random surface. The tool is rotating with approximately 50 Hz. This is equal to an angular speed of $50 \cdot 2\pi$ rad/s. When the polishing surface is exactly parallel to the polishing tool there is no resultant momentum in the x- or y- direction to cause tilting and precessing behavior. When the polishing surface has a free geometric form this does not have to be the case. The model is a disc on a non-flat surface where it is assumed that the disc is in contact with the material only at two points, see figure 6.1.

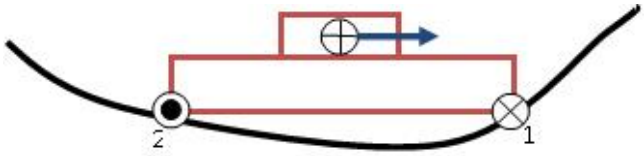

Figure 6.1: *Disc model of the polishing tool.*

This model can be categorized as a rigid body. A rigid body can be seen as a continuous collection of particles moving together as a "rigid" unit [22]. This movement is a rotational movement described by a single angular velocity and/or angular acceleration. The general movement of a rigid body can be a combination of two types of movements: translation and rotation. The translation can be described using linear momentum and the rotation by angular momentum. The angular momentum can be expressed as the product of a tensor with the angular velocity [22].

This tensor is in this case a 3×3 matrix and is called the inertia matrix and is noted with I. Some properties are that this matrix is real and symmetric, i.e.

$$\mathbf{I} = \begin{pmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{pmatrix}. \tag{6.1}$$

The elements on the diagonal I_{xx}, I_{yy}, I_{zz}, are the moments of inertia and are given by

$$I_{ii} = \int_m (j^2 + k^2) dm, i, j, k \in x, y, z. \tag{6.2}$$

The other elements are called products of inertia and defined as

$$I_{ij} = \int_m ij dm, i, j \in x, y, z. \tag{6.3}$$

The value of the elements of this matrix is dependent on the chosen coordinate system. For simplicity it is assumed that the center of rotation of the disc is located at the center of the disc (in height and width). This means that the radius from point 1 and point 2 to the center of rotation is not the radius of the polishing brush r_{disc} but actually $r = \sqrt{r_{disc}^2 + \left(\frac{h}{2}\right)^2}$, where h is the height of the polishing brush.

6.1.1. Coordinate systems

To be able to describe the equations of motion for this problem multiple coordinate systems have to be defined. The choice for coordinate systems is not unique and is often done in such a fashion to simplify equations. This is done by finding this coordinate system such that the inertia matrix is very simple [22]. Then this inertia matrix can be transformed to the "actual" coordinate frame with a transformation matrix. It is important that the coordinate systems are made of principle axes. These form a basis for which the products of inertia vanish [22]. Two other properties of principle axes are [22]

1. A principle axis of a rigid body is perpendicular to a plane of symmetry.
2. Principle axes are dependent on the choice of origin.

For this problem three principle coordinate systems are defined: the local, the reference and the world coordinate systems. The local coordinate system is noted with

$$\mathbf{X}^l = x\mathbf{i}, \qquad \mathbf{Y}^l = y\mathbf{j}, \qquad \mathbf{Z}^l = z\mathbf{k}. \tag{6.4}$$

The reference coordinate system as

$$\mathbf{X} = X\mathbf{I}, \qquad \mathbf{Y} = Y\mathbf{J}, \qquad \mathbf{Z} = Z\mathbf{K}. \tag{6.5}$$

The world coordinate system as

$$\mathbf{X^W} = X^W\mathbf{I^W}, \qquad \mathbf{Y^W} = Y^W\mathbf{J^W}, \qquad \mathbf{Z^W} = Z^W\mathbf{K^W}. \tag{6.6}$$

First of all the world coordinate system is the system in which the material that has to be polished is defined, see figure 6.2. Then the reference coordinate system is a coordinate system that is defined on the contact

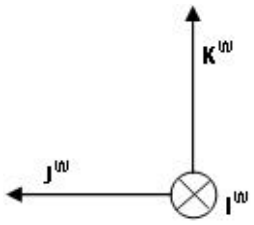

Figure 6.2: *World coordinate system.*

point in the direction of the movement of the tool, see figure 6.3. To describe the reference coordinate system in terms of the world coordinate system an extra variable is introduced: the material elevation angle $\xi(X^W, Y^W)$. This is a 3D function which returns the angle of the material in the direction of the movement. Assuming the material can be located such that the **Y**-axis is parallel to $\mathbf{X}^W$. Then the reference coordinate axes are given by

$$\mathbf{I} = -\cos(\xi)\mathbf{J}^W + \sin(\xi)\mathbf{K}^W \tag{6.7}$$

$$\mathbf{J} = \mathbf{I}^W, \tag{6.8}$$

$$\mathbf{K} = \sin(\xi)\mathbf{J}^W + \cos(\xi)\mathbf{K}^W. \tag{6.9}$$

The local coordinate system is then positioned on the disc. The z-axis is chosen to go vertically up from the disc. Then the x- and y- axes are defined as

$$\mathbf{i} = \mathbf{K} \times \mathbf{k}, \qquad \mathbf{j} = \mathbf{k} \times \mathbf{i}. \tag{6.10}$$

This means that the y-axis is parallel to the longitude movement vector and the x-axis goes into the paper.

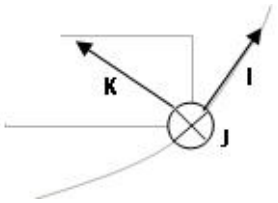

Figure 6.3: *Reference coordinate system.*

6.2. Euler angles

To describe the tilting and precessing of the disc, variables are used different from x, y and z. The variables that fit best in this situation are the Euler angles. These are θ, ψ and ϕ. ϕ describes the rotation of the tool, where $\dot{\phi} = p =$ constant. θ is defined as the angle between $\mathbf{k}$ and $\mathbf{K}$, where $\dot{\theta}$ describes the nutation. ψ is defined as the angle between $\mathbf{i}$ and $\mathbf{I}$, where $\dot{\psi}$ describes the precession. In aircraft dynamics a different terminology is used for the different movements. Then p describes the yaw movement and $\dot{\theta}$ the pitch movement. $\dot{\psi}$ is a bit more difficult, since in this model it is measured around the $\mathbf{K}$-axis and not the $\mathbf{i}$-axis. This rotation around the $\mathbf{i}$-axis is rolling. So $\dot{\psi}$ is a combination of rolling and yawing. The component rolling is then $\dot{\psi}\sin(\theta)$.

6.2.1. Using three Euler angles

With introducing these angles the local coordinate system can be described in terms of the reference coordinate system:

$$\mathbf{i} = \cos(\psi)\mathbf{I} + \sin(\psi)\mathbf{J} \tag{6.11}$$

$$\mathbf{k} = \sin(\theta)\mathbf{I} + \cos(\theta)\mathbf{K}. \tag{6.12}$$

The Euler angles can now be described as vector angular velocities;

1. $\mathbf{p} = \dot{\phi}\mathbf{k} = p\mathbf{k}, p > 0$.
2. $\underline{\dot{\theta}} = \dot{\theta}\mathbf{i}$.
3. $\underline{\dot{\psi}} = \dot{\psi}\mathbf{K} = \dot{\psi}(\sin(\theta)\mathbf{j} + \cos(\theta)\mathbf{k})$.

For the last angle $\mathbf{K}$ is written in terms of the local coordinate system. Then the absolute motion of the spin axis can be denoted as

$$\underline{\Omega} = \underline{\dot{\psi}} + \underline{\dot{\theta}} = \dot{\theta}\mathbf{i} + \dot{\psi}\sin(\theta)\mathbf{j} + \dot{\psi}\cos(\theta)\mathbf{k} \tag{6.13}$$

$$= \Omega_x\mathbf{i} + \Omega_y\mathbf{j} + \Omega_z\mathbf{k}. \tag{6.14}$$

The total angular velocity can be described as

$$\underline{\omega} = \underline{\Omega} + \mathbf{p} \tag{6.15}$$

$$= \Omega_x\mathbf{i} + \Omega_y\mathbf{j} + (\Omega_z + p)\mathbf{k} \tag{6.16}$$

$$= \omega_x\mathbf{i} + \omega_y\mathbf{j} + \omega_z\mathbf{k}. \tag{6.17}$$

6.2.2. Using two Euler angles

A question one may ask is why is a variable component necessary around the $\mathbf{k}$-axis, while it is already known that the rotation of the tool is the constant p. If instead of the above discussed Euler angles ψ is taken as the rotation around the $\mathbf{j}$-axis this would give

1. $\mathbf{p} = \dot{\phi}\mathbf{k} = p\mathbf{k}, p > 0$ constant.
2. $\underline{\dot{\theta}} = \dot{\theta}\mathbf{i}$.
3. $\underline{\dot{\psi}} = \dot{\psi}\mathbf{j}$.

Then $\Omega_y = \dot{\psi}$, $\omega_y = \dot{\psi}$, $\Omega_z = 0$ and $\omega_z = p$ in the above equations.

This would simplify the equations greatly in the next models. The question remains if it has any physical influence on the model.

6.3. Equations and assumptions

The equations of motion for a rigid body are dependent on the angular momentum **H**. This is given by the cross-product of the radius $\mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ and the momentum $\mathbf{P} = m\mathbf{v}$, i.e.

$$\mathbf{H} = \mathbf{r} \times m\mathbf{v}. \tag{6.18}$$

Taking the time derivative of this equation yields

$$\frac{d\mathbf{H}}{dt} = \dot{\mathbf{r}} \times m\mathbf{v} + \mathbf{r} \times m\dot{\mathbf{v}} = \mathbf{r} \times m\dot{\mathbf{v}}, \tag{6.19}$$

since $\dot{\mathbf{r}}$ and $\mathbf{v}$ are parallel. Then remember Newton's second law $\sum_i \mathbf{F}_i = m\mathbf{a} = m\dot{\mathbf{v}}$. This changes the right hand side to

$$\frac{d\mathbf{H}}{dt} = \mathbf{r} \times \sum_i \mathbf{F}_i = \sum_i \mathbf{r} \times \mathbf{F}_i, \tag{6.20}$$

which is exactly the definition of the momentum due to forces, so the equation of motion becomes

$$\frac{d\mathbf{H}}{dt} = \sum_i \mathbf{M}_i. \tag{6.21}$$

Continuing with deriving an exact equation, **H** must be studied. The definition was $\mathbf{H} = \mathbf{r} \times m\mathbf{v}$. The velocity can be written in terms of the angular velocity $\underline{\omega} = \omega_x\mathbf{i} + \omega_y\mathbf{j} + \omega_z\mathbf{k}$ by $\mathbf{v} = \underline{\omega} \times \mathbf{r}$. This gives

$$\mathbf{H} = m\mathbf{r} \times (\underline{\omega} \times \mathbf{r}) = m(||\mathbf{r}||^2\underline{\omega} - (\mathbf{r} \cdot \underline{\omega})\mathbf{r}). \tag{6.22}$$

This last equality follows from the fact that

$$A \times (B \times C) = (A \cdot C)B - (A \cdot B)C. \tag{6.23}$$

Expanding equation (6.22) in all separate terms gives the possibility for the following notation

$$\mathbf{H} = \mathbf{I}\underline{\omega}, \tag{6.24}$$

where **I** is defined as equation (6.1). Then using this form of **H** in the equation of motion gives

$$\sum_i M_i = \frac{d\mathbf{I}\omega}{dt} = \underline{\omega} \times \mathbf{I}\underline{\omega} + \left.\frac{d}{dt}\right|_{rel} (\mathbf{I}\underline{\omega}) = \underline{\omega} \times \mathbf{I}\underline{\omega} + \mathbf{I}\underline{\alpha}, \tag{6.25}$$

where $\underline{\alpha}$ is the angular acceleration. This equation is known as Euler's equation.

The equalities are possible since:

1. Looking at the time derivative of a vector **A**:

$$\frac{d\mathbf{A}}{dt} = (\dot{A}_x\mathbf{i} + \dot{A}_y\mathbf{j} + \dot{A}_z\mathbf{k}) + (A_x\dot{\mathbf{i}} + A_y\dot{\mathbf{j}} + A_z\dot{\mathbf{k}}) \tag{6.26}$$

   The first term is the change observed in the rotating system, so ignoring the actual rotation. This can be defined as the relative time derivative $\dot{\mathbf{A}}_{rel}$. Another notation for this relative time derivative due to some angular velocity is

$$\dot{\mathbf{A}}_{rel} = \left.\frac{d}{dt}\right|_{\underline{\omega}} \mathbf{A}. \tag{6.27}$$

   Then the derivatives of basis vectors describe the rotation of the system. This means that the rotation can be described as the change in the unit vector due to the angular velocity $\underline{\omega}$. So then

$$\dot{\mathbf{i}} = \underline{\omega} \times \mathbf{i}, \qquad \dot{\mathbf{j}} = \underline{\omega} \times \mathbf{j} \qquad \dot{\mathbf{k}} = \underline{\omega} \times \mathbf{k}. \tag{6.28}$$

   This then leads to

$$(A_x\dot{\mathbf{i}} + A_y\dot{\mathbf{j}} + A_z\dot{\mathbf{k}}) = \underline{\omega} \times (A_x\mathbf{i} + A_y\mathbf{j} + A_z\mathbf{k}) = \underline{\omega} \times \mathbf{A}. \tag{6.29}$$

   So

$$\frac{d\mathbf{A}}{dt} = \underline{\omega} \times \mathbf{A} + \dot{\mathbf{A}}_{rel}. \tag{6.30}$$

2. If instead of **A** the angular velocity $\underline{\omega}$ is used, the time derivative becomes

$$\dot{\underline{\omega}} = \underline{\omega} \times \underline{\omega} + \dot{\underline{\omega}}_{rel}. \tag{6.31}$$

   Then since $\underline{\omega} \times \underline{\omega} = 0$ this becomes $\dot{\underline{\omega}} = \dot{\underline{\omega}}_{rel}$ or $\underline{\alpha} = \underline{\alpha}_{rel}$.

3. The time derivative relative to $\underline{\omega}$ is given by

$$\left.\frac{d}{dt}\right|_{rel} (\mathbf{I}\underline{\omega}) = \dot{\mathbf{I}}_{rel}\underline{\omega} + \mathbf{I}\dot{\underline{\omega}}_{rel}. \tag{6.32}$$

   Then $\dot{\mathbf{I}}_{rel} = 0$, since the relative system rotates with the body.

So now the general equation of motion is derived. The question remains on how the angular velocity, inertia matrix and angular acceleration are defined. In the local coordinate system $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ the inertia matrix is generic. This means it is very simple due to the symmetry of the polishing brush. The products of inertia are all equal to zero and the moments of inertia in the x and y direction are the same since the disc is symmetric here. For simplicity it is assumed the polishing brush is a disc with a height h and a radius r. Then the values of the moments of inertia are generally known [16]

$$\mathbf{I} = \begin{pmatrix} \frac{1}{12}m(3r^2+h^2) & 0 & 0 \\ 0 & \frac{1}{12}m(3r^2+h^2) & 0 \\ 0 & 0 & \frac{1}{2}mr^2 \end{pmatrix}. \tag{6.33}$$

Now using in Euler's equation (6.25) the fact that the total angular velocity is given by $\underline{\omega} = \underline{\Omega} + \mathbf{p}$

$$\sum_i \mathbf{M_i} = (\underline{\Omega} \times \mathbf{I}\underline{\omega} + \mathbf{p} \times \mathbf{I}\underline{\omega}) + \mathbf{I}\left(\left.\frac{d}{dt}\right|_{\underline{\Omega}} \underline{\omega} + \underline{\Omega} \times \underline{\omega} \right), \tag{6.34}$$

where is used that (see equation (6.30)

$$\frac{d}{dt}\underline{\omega} = \left.\frac{d}{dt}\right|_{\underline{\Omega}} \underline{\omega} + \underline{\Omega} \times \underline{\omega}. \tag{6.35}$$

Then expanding more

$$\begin{aligned} \mathbf{p} \times \mathbf{I}\underline{\omega} &= \mathbf{p} \times \mathbf{I}(\underline{\Omega} + \mathbf{p}) && (6.36) \\ &= \mathbf{p} \times \mathbf{I}\underline{\Omega} + \mathbf{p} \times \mathbf{I}\mathbf{p}, && (6.37) \end{aligned}$$

here the last term vanishes since $\mathbf{p}$ is defined along the principle axis $\mathbf{k}$ and $\mathbf{I}$ is a diagonal matrix defined such that the two terms are parallel to each other. The grouping of the $\mathbf{I}$ and $\underline{\Omega}$ is also possible because of these reasons. Then

$$\begin{aligned} \mathbf{I}(\underline{\Omega} \times \underline{\omega}) &= \mathbf{I}(\underline{\Omega} \times \underline{\Omega}) + \mathbf{I}(\underline{\Omega} \times \mathbf{p}) && (6.38) \\ &= \mathbf{I}(\underline{\Omega} \times \mathbf{p}) && (6.39) \\ &= (\mathbf{I}\underline{\Omega}) \times \mathbf{p} = -\mathbf{p} \times \mathbf{I}\underline{\Omega}. && (6.40) \end{aligned}$$

Combining equations (6.36) and (6.38) in the equation of motion then gives

$$\sum_i \mathbf{M_i} = \underline{\Omega} \times \mathbf{I}\underline{\omega} + \mathbf{I}\left.\frac{d}{dt}\right|_{\underline{\Omega}} \underline{\omega} = \underline{\Omega} \times \mathbf{I}\underline{\omega} + \mathbf{I}\underline{\alpha}_{rel}. \tag{6.41}$$

6.3.1. Three Euler angles

Now using the definitions (6.13) and (6.15) gives for $\underline{\alpha}_{rel} = \dot{\underline{\Omega}} + \dot{p}\mathbf{k}$;

$$\underline{\alpha}_{rel} = \ddot{\theta}\mathbf{i} + (\ddot{\psi}\sin(\theta) + \dot{\psi}\dot{\theta}\cos(\theta))\mathbf{j} + \frac{d}{dt}(\dot{\psi}\cos(\theta) + p)\mathbf{k}. \tag{6.42}$$

Now for the cross product $\underline{\Omega} \times \mathbf{I}\underline{\omega}$ the same can be done

$$\underline{\Omega} \times \mathbf{I}\underline{\omega} = (I_{zz}\Omega_y(\Omega_z + p) - I_{yy}\Omega_y\Omega_z)\mathbf{i} - (I_{zz}\Omega_x(\Omega_z + p) - I_{xx}\Omega_x\Omega_z)\mathbf{j}, \tag{6.43}$$

where $\Omega_x, \Omega_y, \Omega_z$ are given in (6.13). Note that $I_{xx} = I_{yy} = I_O$ due to symmetry.

Then the final equations become

$$\sum_i \mathbf{M_i} = \left(I_O(\ddot{\theta} - \dot{\psi}^2 \sin(\theta)\cos(\theta)) + I_{zz}\dot{\psi}\sin(\theta)\omega_z\right)\mathbf{i} \tag{6.44}$$

$$+ \left(I_O(\ddot{\psi}\sin(\theta) + 2\dot{\psi}\dot{\theta}\cos(\theta)) - I_{zz}\dot{\theta}\omega_z\right)\mathbf{j} \tag{6.45}$$

$$+ I_{zz}\dot{\omega}_z\mathbf{k}. \tag{6.46}$$

In this equation ω_z is not expanded. This is motivated by the assumption that in this system no moment will arise in the z-direction and $\dot{\omega}_z$ only appears in this direction.

6.3.2. Two Euler angles

Now doing the same for the definition where **p** is the rotation around the **k**-axis, $\underline{\dot{\theta}}$ the rotation around the **i**-axis and $\underline{\dot{\psi}}$ the rotation around the **j**-axis gives for $\underline{\alpha}_{rel} = \underline{\dot{\Omega}} + \dot{p}\mathbf{k}$;

$$\underline{\alpha}_{rel} = \ddot{\theta}\mathbf{i} + \ddot{\psi}\mathbf{j} + \dot{p}\mathbf{k}. \tag{6.47}$$

Since the rotation around the **k**-axis is assumed known and constant, $\dot{p} = 0$. Now for the cross product $\underline{\Omega} \times \underline{I\omega}$ the same can be done

$$\underline{\Omega} \times \mathbf{I}\underline{\omega} = (I_{zz}\Omega_y(\Omega_z + p) - I_{yy}\Omega_y\Omega_z)\mathbf{i} - (I_{zz}\Omega_x(\Omega_z + p) - I_{xx}\Omega_x\Omega_z)\mathbf{j}. \tag{6.48}$$

Note that $I_{xx} = I_{yy} = I_O$ due to symmetry.

Then the final equations become

$$\sum_i \mathbf{M_i} = \left(I_O\ddot{\theta} + I_{zz}\dot{\psi}p\right)\mathbf{i} \tag{6.49}$$

$$+ \left(I_O\ddot{\psi} - I_{zz}\dot{\theta}p\right)\mathbf{j} \tag{6.50}$$

$$+ 0\mathbf{k}. \tag{6.51}$$

These equations are a lot simpler than (6.44).

6.4. State-space representation

To be able to simulate the model a state-space representation is needed of the form

$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}), \tag{6.52}$$

where $\mathbf{f}(\mathbf{q})$ can be a nonlinear function. First of all: there are three axes on which equations can be formed. Assume the following notation for the moments in each direction:

$$M^x = \sum_i M_i^x, \qquad M^y = \sum_i M_i^y, \qquad M^z = \sum_i M_i^z. \tag{6.53}$$

Then for the three angles model the state variables used are $\mathbf{q} = (\theta, \dot{\theta}, \psi, \dot{\psi}, \omega_z)^T$. Then rewriting (6.44) gives

$$\dot{q}_1 = q_2 \tag{6.54}$$

$$\dot{q}_2 = \frac{M^x}{I_O} + q_4^2 \sin(q_1)\cos(q_1) - \frac{I_{zz}}{I_O} q_4 \sin(q_1) q_5 \tag{6.55}$$

$$\dot{q}_3 = q_4 \tag{6.56}$$

$$\dot{q}_4 = \frac{M^y}{I_O \sin(q_1)} - 2q_4 q_2 \frac{\cos(q_1)}{\sin(q_1)} + \frac{I_{zz} q_2 q_5}{I_O \sin(q_1)} \tag{6.57}$$

$$\dot{q}_5 = \frac{M^z}{I_{zz}} \tag{6.58}$$

For the two angles model only four equations are needed:

$$\dot{q}_1 = q_2 \tag{6.59}$$

$$\dot{q}_2 = \frac{M^x}{I_O} - \frac{I_{zz}}{I_O} q_4 p \tag{6.60}$$

$$\dot{q}_3 = q_4 \tag{6.61}$$

$$\dot{q}_4 = \frac{M^y}{I_O} + \frac{I_{zz}}{I_O} q_2 p. \tag{6.62}$$

7

Forces and influences

Due to the free geometric forms of the material contact forces can arise on the polishing tool which cause non-zero moments on the system from the last section. The longitudinal movement of the tool is assumed to have a speed of v. There is also a polishing force applied on the material in the $-z$-direction from the robot arm. The value of this polishing force is between 10 and 15 Newton. There can be different influences on the polishing disc that create moments in the system. In this chapter different possible causes are discussed to describe this behavior.

Because there is no full contact with the material in the assumed situation, resultant forces can arise in the two outer contact points. The question is what this longitude movement means for the resultant force. The only reason an object moves is that the force with which the tool is moving exceeds a certain limit. First an introduction on friction force and the angle of friction must be given.

7.1. Angle of kinetic friction

There are two types of friction; static and kinetic friction [20]. They are both caused by the roughness of surfaces. Static friction can be defined as the friction that causes motion of solid ground. Kinetic friction is related to the movement of bodies in contact and results in loss of energy. If the direction of the resultant force due to the normal force and the friction force exceeds a certain angle σ movement will be possible [20]. This is showed by Coulomb. This angle is called the angle of friction. He showed that there is a good approximation between the friction force F_r and the movement:

1. The friction force F_r is proportional to the normal force with proportionality factor μ.

2. The friction force F_r is oriented in the opposite direction of the velocity vector.

The resulting law of friction then is [20]

$$F_r = \mu F_N, \tag{7.1}$$

where F_N is the normal force. The proportionality factor μ is the coefficient of kinetic friction. Mathematically the vector force can be noted as

$$\mathbf{F}_r = -\mu F_n \frac{\mathbf{v}}{\|\mathbf{v}\|}. \tag{7.2}$$

The angle of friction and the coefficient of kinetic friction are related as

$$\tan(\sigma) = \mu. \tag{7.3}$$

The σ can be graphically interpreted as a cone around the axis of the normal force with angle of σ. The actual value of the coefficient of kinetic friction is still difficult to determine. This value is dependent on the material properties of both the polishing brush and the material. This is a value that has to be approximated in some way.

7.2. Calculating the resultant forces

Even without having the exact magnitude of the friction forces, with help of the last section, they can be approximated with the angle of friction. Since there is a translational movement due to the robot arm, the angle of friction must be reached to allow movement. This means the resultant force that reacts on the movement and the polishing pressure follows the angle σ. This resultant force is a combination of the normal force and the friction force and its component in the **k**-direction can cause a moment in the **i**-direction (right-hand rule).

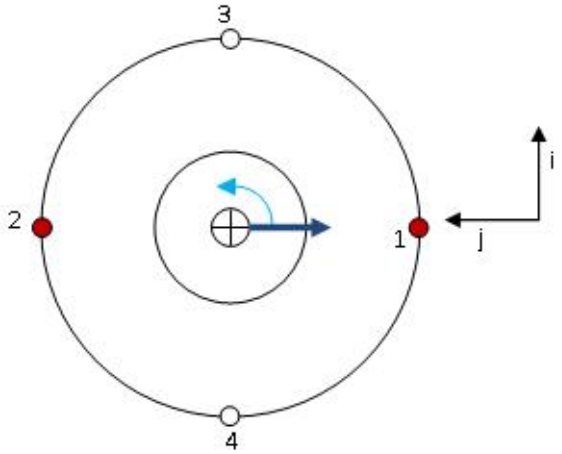

Figure 7.1: *Polishing tool head from above with* **k***-axis coming out of the paper.*

7.2.1. Resultant forces in contact points

First start with the situation where the only contact points are in the direction of the movement. These two points are C_1 and C_2, see figure 7.1. The two other points C_3 and C_4 float above the material. To simplify maters the disc in figure 6.1 is now assumed as one unit with one width. As discussed in the introduction of this section there is in both points C_1 and C_2 a resultant force that has an angle of σ with the normal direction to the material, see figure 7.2.

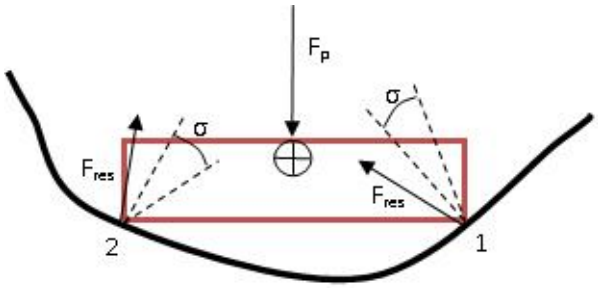

Figure 7.2: *Polishing tool head sideways with all forces on it (direction of movement to the right).*

Since the movement causes friction force opposite to the movement, the resultant force is located to the far left of the friction cone for both points. There must be a balance of forces so with the known angles and known polishing force F_p the problem in figure 7.3 can be solved. First the angles α and β must be determined. It was assumed that the angle of the material elevation, ξ, is known at each point. Assume that the direction of the translational movement corresponds to the minus and plus signs of ξ.

Then first α, the angle between F_p and F_{res}^{C2}. The angle between the normal axis of the material and the horizontal axis is $\frac{1}{2}\pi - \xi(C_2)$. This makes the angle between the horizontal and F_{res}^{C2} equal to $\frac{1}{2}\pi - \xi(C_2) + \sigma$. Then the horizontal, part of F_p and F_{res}^{C2} form a triangle with a corner of $\frac{1}{2}\pi$. So then $\alpha = \frac{1}{2}\pi - \left(\frac{1}{2}\pi - \xi(C_2) + \sigma\right) = \xi(C_2) - \sigma$.

For β the same type of argument can be made. The angle between the horizontal and the material is $\xi(C_1)$. Then with some basic geometry it can be proven that the angle between F_p and the normal axis from

the material is $\xi(C_1)$. This means that $\beta = \xi(C_1) + \sigma$.

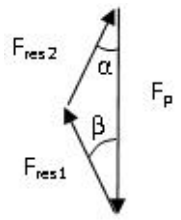

Figure 7.3: *Forces working on the polishing tool.*

Then calculating the actual forces can be done by solving the following two equations:

$$F_{res}^{C1}\cos(\beta) + F_{res}^{C2}\cos(\alpha) = F_p \tag{7.4}$$
$$-F_{res}^{C1}\sin(\beta) + F_{res}^{C2}\sin(\alpha) = 0. \tag{7.5}$$

Solving this in part gives the next formulas

$$F_{res}^{C2} = \frac{F_p}{\cos(\beta)}\left(\frac{\sin(\alpha)}{\sin(\beta)} + \frac{\cos(\alpha)}{\cos(\beta)}\right)^{-1} \tag{7.6}$$
$$F_{res}^{C1} = \frac{F_p}{\cos(\beta)} - \frac{\cos(\alpha)}{\cos(\beta)}F_{res}^{C2}. \tag{7.7}$$

Especially F_{res}^{C2} can be simplified using standard trigonometric identities from Calculus. The resulting formulas are then

$$F_{res}^{C2} = F_p\frac{\sin(\beta)}{\sin(\alpha+\beta)} \tag{7.8}$$
$$F_{res}^{C1} = \frac{F_p}{\cos(\beta)}\left(1 - \frac{\cos(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}\right). \tag{7.9}$$

For the actual moments, only the components that are perpendicular to the direction of $\mathbf{r}$ can contribute to them. This means the angle between $r = \sqrt{r_{disc}^2 + \left(\frac{h}{2}\right)^2}$ and the $\mathbf{j}$-axis must be calculated. This angle is defined as $\eta = \tan^{-1}\left(\frac{h}{2r_{disc}}\right)$. The component of the forces will cause a moment in the $\mathbf{i}$-direction (following the right-hand-rule), i.e.

$$M^x = \left(F_{res}^{C1}\cos(\beta+\eta) - F_{res}^{C2}\cos(\alpha+\eta)\right)\cdot r. \tag{7.10}$$

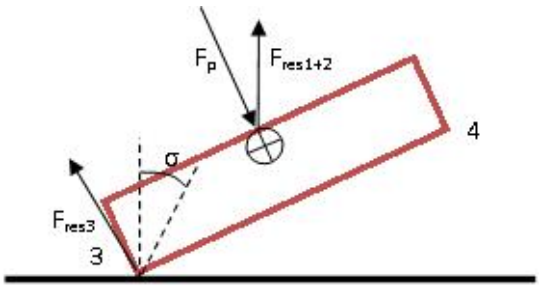

Figure 7.4: *Polishing tool head sideways with all forces working on it (**i**-axis following disc to left).*

7.2.2. Resultant forces in tilting points

When the polishing tool comes in contact with the material (see figure 7.4) a reaction exists at the extra point of contact. For this derivation the extra point of contact is assumed to be C_3 (for C_4 it is all similar, but mirrored). The forces (7.8) and (7.9) are still present, but are now balanced with F_{res}^{C3}, see figure 7.5. The

direction of F_{res}^{C3} is to the far left of the friction cone. As the disc hits the material the relative movement is in the direction of the material, so the counter friction will have the opposite direction. The same way as before the angles must be determined first. First focus on γ. γ is actually the component of ψ that rolls around the **j**-axis and not the **K**-axis. The angle between the **j**-axis and the **K**-axis is $\frac{1}{2}\pi - \theta$. To get an equation for γ consider $\sin(\gamma)$. In terms of ψ this becomes $\sin(\gamma) = \sin(\psi)\cos(\frac{1}{2}\pi - \theta) = \sin(\psi - \psi_0)\sin(\theta)$. So an equation for γ is

$$\gamma = \sin^{-1}\left(\sin(\psi - \psi_0)\sin(\theta)\right), \tag{7.11}$$

here ψ_0 is the initial value of ψ. For δ the same as before can be done. In the picture no elevation is assumed, but if there was, a simple addition of $\xi(C_3)$ is needed. In the picture the normal axis to the material is parallel to the resultant forces of C_1 and C_2, so the general angle is $\delta = \xi(C_3) + \sigma$.

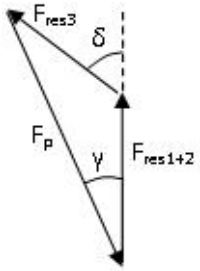

Figure 7.5: *Force balance of the polishing tool* .

Now calculating the actual forces is similar. F_{res}^{C1} and F_{res}^{C2} are the same except for the F_p term which is now replaced with

$$F_p^* = F_p\cos(\gamma) - F_{res}^{C3}\cos(\delta). \tag{7.12}$$

So the resulting forces for C_1 and C_2 are

$$F_{res}^{C2} = F_p^*\frac{\sin(\beta)}{\sin(\alpha+\beta)} \tag{7.13}$$

$$F_{res}^{C1} = \frac{F_p^*}{\cos(\beta)}\left(1 - \frac{\cos(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}\right). \tag{7.14}$$

Then the resultant force for C_3 and the force F_p^* satisfy

$$F_p^* + F_{res}^{C3}\cos(\delta) = F_p\cos(\gamma) \tag{7.15}$$

$$F_{res}^{C3}\sin(\delta) = F_p\sin(\gamma). \tag{7.16}$$

This leads to the following values

$$F_{res}^{C3} = F_p\frac{\sin(\gamma)}{\sin(\delta)} \tag{7.17}$$

$$F_p^* = F_p\cos(\gamma) - F_{res}^{C3}\cos(\delta). \tag{7.18}$$

Then this force causes only a moment in the **j** -direction due to the component in the **k**-direction. The moment it causes is in the negative direction. Since $\gamma > 0$, $F_p > 0$ and $\delta > 0$ in this situation this means

$$M^y = -F_{res}^{C3}\cos(\delta + \eta)\cdot r. \tag{7.19}$$

In the situation C_4 touches the material, $\gamma < 0$ and the angles are the same. If the assumption is made that the angle of the material elevation is the same the forces have the same values. Now again the component in the **k**-direction only contributes to the moment in the **j**-direction. In this situation the force component leads to a moment in the positive direction, i.e.

$$M^y = F_{res}^{C4}\cos(\delta + \eta)\cdot r = -F_{res}^{C3}\cos(\delta + \eta)\cdot r. \tag{7.20}$$

Here it is important to notice that for the calculation of F_p^* the absolute value is taken of F_{res}^{C3}, otherwise wrong forces are calculated.

Finally it is important to know when one or the other situation occurs. For this the initial height of the center of rotation must be calculated. This can be approximated multiple ways by using the two angles. If more information is known about the material better approximations can be made. Here the following calculation is done:

$$\Delta h = n r_{disc} \sin(\xi(C_3)) + (1-n) r_{disc} \sin(\xi(C_1)), \tag{7.21}$$

where n is a percentage that can be chosen realistically. The The actual height of C_3 or C_4 with respect to the center of rotation can be calculated with $h' = r\sin(\gamma)$. When $\gamma < 0$ and $h' < 0$ this means the disc is approaching the material at C_4. Since it is a symmetric disc the situation that there is contact with the material is when $\Delta h - |h'| \leq 0$.

7.2.3. Summary

To summarize the last section, the following resultant forces arise:

If $\Delta h - |h'| > 0$ (no contact with C_3 and C_4):

$$F_{res}^{C1} = \frac{F_p}{\cos(\beta)} \left(1 - \frac{\cos(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}\right) \tag{7.22}$$

$$F_{res}^{C2} = F_p \frac{\sin(\beta)}{\sin(\alpha+\beta)} \tag{7.23}$$

$$F_{res}^{C3} = 0. \tag{7.24}$$

If $\Delta h - |h'| \leq 0$:

$$F_{res}^{C1} = \frac{F_p^*}{\cos(\beta)} \left(1 - \frac{\cos(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}\right) \tag{7.25}$$

$$F_{res}^{C2} = F_p^* \frac{\sin(\beta)}{\sin(\alpha+\beta)} \tag{7.26}$$

$$F_{res}^{C3} = F_p \frac{\sin(\gamma)}{\sin(\delta)} \tag{7.27}$$

$$F_p^* = F_p \cos(\gamma) - |F_{res}^{C3}| \cos(\delta). \tag{7.28}$$

The moments resulting from these forces are given by (using F_p or F_p^* when appropriate)

$$M^x = \left(F_{res}^{C1} \cos(\beta+\eta) - F_{res}^{C2} \cos(\alpha+\eta)\right) \cdot r \tag{7.29}$$

$$M^y = -F_{res}^{C3} \cos(\delta+\eta) \cdot r \tag{7.30}$$

$$M^z = 0. \tag{7.31}$$

The forces in the three directions are slightly different and given by (only the F_p^* can be replaced by F_p when appropriate)

$$F_x = F_p \sin(\gamma) \tag{7.32}$$

$$F_y = 2F_p^* \frac{\sin(\alpha)\sin(\beta)}{\sin(\alpha+\beta)} \tag{7.33}$$

$$F_z = F_p^* \left(\left| \frac{\sin(\alpha)\cos(\beta)}{\sin(\alpha+\beta)} \right| + \left| \frac{\cos(\alpha)\sin(\beta)}{\sin(\alpha+\beta)} \right| + \left| \frac{\sin(\gamma)\cos(\delta)}{\sin(\delta)} \right| \right). \tag{7.34}$$

Here for $F_x = F_{res}^{C3} \cdot \sin(\delta)$, $F_y = F_{res}^{C1} \sin(\beta) + F_{res}^{C2} \sin(\alpha)$ and $F_z = |F_{res}^{C1}| \cos(\beta) + |F_{res}^{C2}| \cos(\alpha) + |F_{res}^{C3}| \cos(\delta)$.

7.3. Contact impulse

Another option of influence can be an impact impulse or force. This means looking at the exact moment one of the points comes in contact with the material. In a small time interval there will be an initial acceleration and a crash with the material and a final acceleration after the crash. The force difference in the angle

variables can be given by

$$\Delta \mathbf{F}^{impact} = m\Delta\alpha = m(\alpha_f - \alpha_i). \tag{7.35}$$

The initial acceleration is calculated in the model, but about the final acceleration an assumption must be made. Since the polishing brush is stopped by the material in its motion, there will be a moment in which the acceleration is zero. If this is used, the moment added is only for the two angles in both the two and three angle model

$$\mathbf{M} = -\begin{pmatrix} m\ddot{\theta}r \\ m\ddot{\psi}r \end{pmatrix}. \tag{7.36}$$

In the θ angle this moment of crashing happens any time the polishing brush is not in equilibrium position $\theta = \theta_0$. This means that any time this happens a moment is existing in C_1 or C_2. In C_1 this moment will be negative and in C_2 positive. The moments will happen when $\theta - \theta_0 \neq 0$. Since in the contact points there is not a lot of moving room the assumption that $\ddot{\theta}_f = 0$ seems quite realistic.

For ψ this moment might be adjusted for the three angle moment, since its movement is then not perpendicular to the material. The impact will happen when C_3 or C_4 will come in contact with the material (see section 7.2.2). Then in C_3 a negative moment will be added and in C_4 a positive moment. This will happen when $|r\sin(\psi - \psi_0)| \geq \Delta h$ (see equation (7.21)). Only the assumption that $\ddot{\phi}_f = 0$ may be too strict, since there is enough room for the polishing brush to bounce back.

7.4. Damping effects

Stabilizing effects in simulation models can be made by adding damping. Damping is an influence that is present in all real world physical systems[23]. Eventually vibrations decay and die out. Mathematical modeling and physical observations give rise to a simple addition of a $c\dot{q}$ term, where c is the viscous damping coefficient. This type of damping is most often used. The problem remains where to identify the damping in the model. The following options can be found:

1. Damping due to the friction between the polishing brush and the material with the liquid layer with diamond particles.

2. Damping in the center of rotation. In reality a ball joint is present here which could cause friction and damping in the movement of ψ and θ.

The damping effects can also be added to the airbearing model presented in the next section. This would mean that the choice has to be made if the damping is present in the airbearing or in the rod stiffness or if there is a separate damping at the end of the polishing tool. Furthermore the damping coefficient must be determined. This is for a large part dependent on the results of the system. When deviations are physically not possible because they explode the damping coefficient should be increased.

7.5. Impact and impact duration

Following the derivations made in [44] a description of the impact and approximation of the impact duration can be made. Impact is defined here as a sudden contact of a moving body with a motionless barrier or with a body with a much larger size. An impact exists out of 3 phases: the loading phase, the rest moment and the unloading phase. The loading phase begins at the moment of the first impact and in this phase a deformation takes place as result of inertia forces. At the end of this phase the entire object has zero velocity. Then the unloading phase starts in which the body returns to its own shape and regains speed in the opposite direction.

The situation of C_3 and C_4 touching the material with a certain velocity is a simplified situation described as a body impact against a no-slip surface [44]. Using the preservation principle the following impulse and final angular velocity can be defined:

$$\alpha_f = \frac{m v_0 e_x}{I_0}, \tag{7.37}$$

$$P_{impulse} = m v_0, \tag{7.38}$$

where v_0 is the initial normal velocity component, e_x the distance between the center of rotation and the location of impact. The initial normal velocity of the movement is the angular velocity times the radius r. The impulse has as unit $[Ns]$. To really calculate the magnitude of the force, the duration of impact must be approximated. [44] also has approximations for this value. First of all it is assumed that when the polishing disc comes in contact with the material its behavior can be described as a linear spring. Then the duration of impact is one fourth of the entire period, since it is only a quarter of a total movement. The natural period is then defined as

$$\tau = 2\pi\sqrt{\frac{m}{k_p}}, \tag{7.39}$$

where k_p is some assumed spring stiffness. Then the duration of impact is

$$t_{im} = \frac{\tau}{4} = \frac{\pi}{2}\sqrt{\frac{m}{k_p}}. \tag{7.40}$$

For C_1 and C_2 it is only an impact if the initial velocity $\dot{\theta}$ is not zero and the angle θ is not equal to θ_0. For C_3 and C_4 a different approximation must be made. Since the contact is modeled as a linear spring it is possible to derive its maximum deflection, d_{max}. This can be done by writing a simple energy balance of a mass-spring system. This would give

$$E_{kin} = E_{pot} \tag{7.41}$$

$$\frac{1}{2}mv^2 = \frac{1}{2}d_{max}^2 k_p. \tag{7.42}$$

This then leads to the maximum deflection

$$d_{max} = \sqrt{\frac{m}{k_p}}v, \tag{7.43}$$

where v is the linear velocity. When C_3 and C_4 are a distance d_{max} from the material the theoretical linear spring is activated.

7.6. Discussion on realistic influences

The resultant forces in C_3 and C_4 are not valid for this specific robot arm, since the spindle or rod is actually positioned orthogonal to the reference point for the robot and not moving with the polishing disc. The damping is a difficult behavior to describe, since this is very specific to the material properties. The best description found is the impact theory last described. The damping can then be used to decrease growing frequencies.

8

Airbearing model

The next step to get a realistic model is to include a big influence on the amount of movement possible of the tool: the airbearing. As explained before, the airbearing is a tube which contains air between the tool rod inside and the outside tube. This air behaves as springs on the tool rod. A way to model this is to divide the airbearing in a horizontal component and a vertical component. This gives different spring constants and adjusted masses.

8.1. Adding translational vibrations

For small horizontal vibrations it can be assumed that the movement is translational. So it can be modeled as a simple spring-masses system, see figure 8.1. The bending stiffness of the rod, k_x, is also taken into account. This can have a different value dependent on the material of the rod. A general formula used for bending stiffness is [23]

$$k_x = \frac{3NI}{l^3}, \tag{8.1}$$

where l is the length of the rod, N the elastic or Young's modulus in N/m^2, I the cross-sectional moment of inertia in m^4. For a circular rod this cross-sectional moment of inertia or, short, area moment is given by [32]

$$I^A = \frac{\pi}{4} r_{rod}^4. \tag{8.2}$$

Then the translational stiffness of the airbearing, k_t, in N/m, is again a specification dependent on the type of airbearing. The only question now is what the horizontal component of the mass of the airbearing is, m_x.

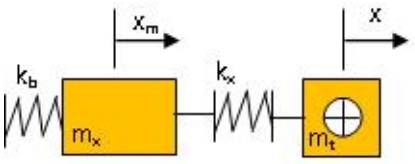

Figure 8.1: *Horizontal model of the airbearing and tool head as masses and springs.*

Then to model figure 8.1 some notes must be made. Since the resultant forces in x- and y-directions are different, both directions get two equations. So using basic modeling techniques for these systems gives [23]

$$m_x \ddot{x}_m = -k_t x_m + k_x(x - x_m), \tag{8.3}$$
$$m_t \ddot{x} = -k_x(x - x_m) - F_x, \tag{8.4}$$
$$m_x \ddot{y}_m = -k_t y_m + k_x(y - y_m), \tag{8.5}$$
$$m_t \ddot{y} = -k_x(y - y_m) - F_y. \tag{8.6}$$

The minus signs for the resultant forces in x- and y-direction are based on the assumption that $C2$ and $C3$ are located left of the model.

8.1.1. Reduced mass in horizontal direction

In the x-directional movement there is an arm between the center of rotation (in the middle of the airbearing) and the movement of the polishing brush. For this reason the actual mass of the airbearing cannot be used for this reduced model. The mass relative to the inertia of the movement must be used. This can be calculated the following way: assume that the movement x_t is an approximation of the movement from the center of rotation in the airbearing, φ_b. This means that $x_t = r_b\varphi_b$ and $\dot{x} = r_b\dot{\varphi}_b$. Then in the model $x_t = x + x_m$, so the total mass of the movement is $m = m_x + m_t$. The total radius of the movement is given by $r_b = \frac{h_{bearing}}{2} + l_{rod} + \frac{h_{disc}}{2}$. Now since this rotational movement from the airbearing is assumed to be so small that it can be approximated by a translational movement, the kinetic energy of the rotational movement should equal the kinetic energy of the translational movement to still have conservation of energy. This means that

$$\frac{1}{2}m\dot{x}_t^2 = \frac{1}{2}I_{bearing}\dot{\varphi}_b^2. \tag{8.7}$$

Then since $\dot{x} = r_b\dot{\varphi}_b$,

$$\frac{1}{2}mr_b^2\dot{\varphi}_b^2 = \frac{1}{2}I_{bearing}\dot{\varphi}_b^2, \tag{8.8}$$

and thus

$$m = \frac{I_{bearing}}{r_b^2}. \tag{8.9}$$

So the reduced mass in the x and y-direction is $m_x = \frac{I_{bearing}}{r_b^2} - m_t$. Note: most of the time the mass of the rod is neglected in the model (since its mass is relatively small), but if it was included in the model its mass should also be subtracted from m_x. The next problem is how to describe the moment of inertia. Since the moment of inertia is a summation or integration about mass it is possible to divide the moment of inertia in three different standard components of which the moments are known:

1. A rectangle with rotational axis in the center for the airbearing.

$$I_{b1} = \frac{m_{bearing}}{12}\left(h_{bearing}^2 + b_{bearing}^2\right), \tag{8.10}$$

   where b is the symbol used for width.

2. A rod with rotation from a distance $\frac{h_{bearing}}{2}$ from the edge of the rod.

$$I_{rod} = m_{rod}\left(\frac{l_{rod}^2}{12} + \frac{h_{bearing}^2}{4}\right). \tag{8.11}$$

3. A rectangle rotation with a distance $\frac{h_{bearing}}{2} + l_{rod}$ from the center of rotation from the middle of the edge.

$$I_{disc} = m_t\left(\frac{4}{12}\frac{h_{bearing}^2}{4} + 4r_{disc}^2 + \left(\frac{h_{bearing}}{2} + l_{rod}\right)^2\right). \tag{8.12}$$

Then the total moment of inertia from the airbearing is given by

$$I_{bearing} = I_{b1} + I_{disc} + I_{rod}. \tag{8.13}$$

8.2. Using total movement for airbearing spring constant

The spring constant for the airbearing can be seen as a combination of the transverse and torsional movement in the airbearing. The spring constants for both movements are given in the design list of the airbearing and are presented in table 9.1. If a certain equilibrium situation from the center of the airbearing is considered (acceleration is zero) given by figure 8.2 the following two equations of motion are valid

$$F + k_{tr}x_t = 0 \tag{8.14}$$

$$FL + k_r\rho = 0. \tag{8.15}$$

Here $x_{tot} = x + \rho L = x_m + x$. If the beam stiffness in the rod is neglected for the moment, the equation of motion from the center of rotation is

$$F + k_t x_{tot} = 0. \tag{8.16}$$

This means that to calculate the total spring constant $-\frac{F}{x_{tot}}$ is needed. Combining the two equations of motion by multiplying by k_r and $k_{tr}L$ and adding them gives

$$F(k_r + k_{tr}L^2) + k_r k_{tr}(x + \rho L) = 0. \tag{8.17}$$

This leads to

$$k_t = -\frac{F}{x_{tot}} = \frac{k_{tr}k_r}{k_r + k_{tr}L^2}. \tag{8.18}$$

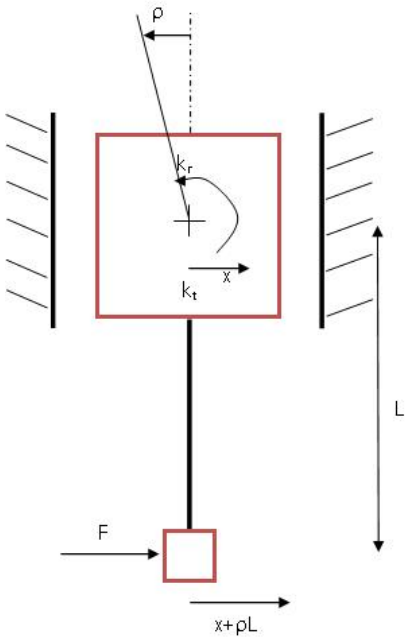

Figure 8.2: *Total movement within the airbearing due to force on the polishing tool.*

8.3. Adding vertical vibrations

Figure 8.3 shows the mass and spring model for the vertical model of the airbearing and tool. Here only one spring is present for which the following formula exists for the spring constant k_z [23]

$$k_z = \frac{NA}{l}, \tag{8.19}$$

where l is again the length of the rod, N the same Young's modulus and A the area of an intersection of the rod given by $A = \pi r_{rod}^2$.

The mass in the vertical direction m_z is assumed to equal the actual mass of the airbearing. The forces here are the polishing force F_p and all resultant forces with components in the z-direction (impact and normal forces). As it will turn out however with the simulations, something is needed in the model to hold the polishing tool in contact with the material. Physically speaking this is the holder in the robot which makes sure the polishing tool with spindle stays in the robot. The behavior of this holder can also be modeled as a very stiff spring, which describes the very tiny space the tool has in the holder in the z-direction which also takes the impact forces due to the polishing disc. The spring constant k_A should be very large. This leads to the following two equations

$$m_z \ddot{z}_m = -k_z(z_m - z) - k_A z_m - F_p, \tag{8.20}$$

$$m_t \ddot{z} = k_z(z_m - z) + F_z. \tag{8.21}$$

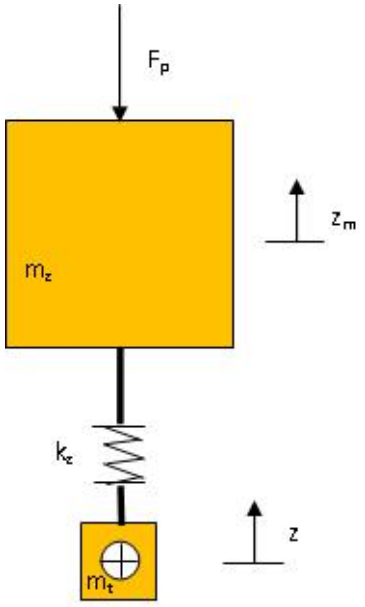

Figure 8.3: *Vertical model of the airbearing and tool head as masses and springs.*

8.4. Adding damping

Using the different types of damping can be very difficult. In this situation at least three types of damping can be easily identified. The airbearing is subject to air damping or velocity-squared damping. The spindle or rod that is bending is subject to structural damping and the friction over the material causes Coulomb damping. Following [23] a detailed approximation can be made of the damping coefficients, but another way is to assume a certain percentage n of the total energy as dissipation (see section 5.1). The formula from this section was

$$c_{eq} = \frac{n \cdot E_{max}}{\pi \omega A^2}. \tag{8.22}$$

Adding two dampers next to each spring in the x and y model means that two damping coefficients need to be calculated. For a spring system the total energy is defined as a summation of the kinetic and potential energy. In this situation this means that if the spring is fully stretched the potential energy is maximum and the kinetic energy is zero. This means that

$$c_{eq} = \frac{n \cdot \left(\frac{1}{2} k A^2\right)}{\pi \omega A^2} = \frac{1}{2} \frac{n \cdot k}{\pi \omega}. \tag{8.23}$$

This means that c_A is calculated using k_t and ω_A and c_D using k_x and ω_D. Adding the dampers to the systems equations gives for x (same for y)

$$m_x \ddot{x}_m = -k_t x_m + k_x (x - x_m) - c_A \dot{x}_m + c_D (\dot{x} - \dot{x}_m), \tag{8.24}$$

$$m_t \ddot{x} = -k_x (x - x_m) - c_D (\dot{x} - \dot{x}_m) - F_x. \tag{8.25}$$

For the z-direction the same can be done by adding damping, c_R, due to internal damping. The equations for z become

$$m_z \ddot{z}_m = -k_z (z_m - z) - k_A z_m - c_R (\dot{z}_m - \dot{z}) - F_p, \tag{8.26}$$

$$m_t \ddot{z} = k_z (z_m - z) + c_R (\dot{z}_m - \dot{z}) + F_z. \tag{8.27}$$

8.5. Natural frequencies and damping ratios

The horizontal and vertical systems vibrate with some natural frequency that can be calculated. If this natural frequency is very high it means that it might be wise to exclude this effect in the model. A very high natural frequency gives small, but a lot of, displacements. Since they might be too small to be of real importance for

the model it is wise to calculate the frequencies beforehand. There are three different systems the x-, y- and z- system. To calculate the natural frequency a harmonic solution is assumed to the system of the form

$$x_m = A^1 \sin(\omega t + \upsilon) + B^1 \cos(\omega t + \upsilon), \tag{8.28}$$

$$x = A^2 \sin(\omega t + \upsilon) + B^2 \cos(\omega t + \upsilon). \tag{8.29}$$

For y and z the same form is assumed. The following homogeneous system is used

$$m_x \ddot{x}_m = -k_t x_m + k_x (x - x_m), \tag{8.30}$$

$$m_t \ddot{x} = -k_x (x - x_m). \tag{8.31}$$

Using the assumed harmonic solution gives

$$\begin{pmatrix} -\omega^2 m_x + k_t + k_x & -k_x \\ -k_x & -m_t\omega^2 + k_x \end{pmatrix} \begin{pmatrix} x_m \\ x \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{8.32}$$

Note: for y the exact same system applies due to symmetry and thus also the same natural frequency is present. For the calculation of m_x the mass of the rod is assumed to be zero.

Solving for the natural frequencies is equivalent to solving this system for eigenvalues. This means the determinant of the matrix above must equal zero, i.e.,

$$(-\omega^2 m_x + k_t + k_x)(-m_t\omega^2 + k_x) - k_x^2 = 0 \tag{8.33}$$

$$m_t m_x \omega^4 - m_t\omega^2(k_t + k_x) - m_x\omega^2 k_x + k_x(k_t + k_x) - k_x^2 = 0 \tag{8.34}$$

$$m_t m_x \omega^4 - \omega^2(m_t k_t + m_t k_x + m_x k_x) + k_x k_t = 0. \tag{8.35}$$

Solving this gives two positive solutions $f_1^x = 2.0696 \cdot 10^3$ Hz = 2.0696 kHz and $f_2^x = 194.0691$ Hz. So for y the frequencies are also $f_1^y = 2.0696 \cdot 10^3$ Hz = 2.0696 kHz and $f_2^y = 194.0691$ Hz.

For the z-direction the same procedure can be followed:

$$\begin{pmatrix} -\omega^2 m_z + k_z + k_A & -k_z \\ -k_z & -m_t\omega^2 + k_z \end{pmatrix} \begin{pmatrix} z_m \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{8.36}$$

This means solving the equation

$$m_z m_t \omega^4 - ((m_z + m_t)k_z + m_t k_A)\omega^2 + k_A k_z = 0. \tag{8.37}$$

This leads to $f_1^z = 4.6107 \cdot 10^3$ Hz = 4.6107 kHz and $f_2^z = 2.8246 \cdot 10^3$ Hz= 2.8246 kHz . This frequency is very high and in situations where the computation time gets to long simplification might be a solution.

In Matlab this can be calculated by the function $eig(A)$, with A the system matrix from the next section. This can then also be done in the situation with damping. The eigenvalues are then in symbolic notation [36]

$$\lambda_{1,2} = \sigma \pm i\omega, \tag{8.38}$$

where σ relates to the damping and ω to the frequencies of the oscillation. Then the damping ratio and the frequency is calculated using [36]

$$\zeta = -\frac{\sigma}{\sqrt{\sigma^2 + \omega^2}} \tag{8.39}$$

$$f = \frac{\omega}{2\pi}. \tag{8.40}$$

Using a damping percentage of $n = 0.1$ % the frequencies are the same as calculated before and the corresponding damping ratios are

$$\zeta_{x_m} = \zeta_{y_m} = 8.1742 \cdot 10^{-5}, \tag{8.41}$$

$$\zeta_x = \zeta_y = 7.9374 \cdot 10^{-5}, \tag{8.42}$$

$$\zeta_{z_m} = 1.9923 \cdot 10^{-5}, \tag{8.43}$$

$$\zeta_z = 5.7860 \cdot 10^{-13}. \tag{8.44}$$

What can be concluded is that the damping ratio for z is a lot lower than the others. This might be a problem for the simulations. With some computations is also found that there is no damping value in which ζ_z comes near the other values. This would imply that some other sort of damping is necessary. Adding an extra damping $-c_Z\dot{z}$ to equation (8.27) with $c_Z = 10^{-8}$ Ns results in a damping ratio of $\zeta_z = 5.0028 \cdot 10^{-5}$. This is closer to the other values. Physically this would mean applying c_Z means z is damping due to the polishing disc on the material. In the z-direction this is not expected to be large since the actual Coulomb friction is expected in the x and y-directions and just a small part could go up. So 10^{-8} Ns seems reasonable.

8.6. Final model equations

The final equations of the airbearing in different axial directions now are

$$\begin{aligned}
m_x\ddot{x}_m &= -k_t x_m + k_x(x - x_m),\\
m_t\ddot{x} &= -k_x(x - x_m) - F_x,\\
m_x\ddot{y}_m &= -k_t y_m + k_x(y - y_m),\\
m_t\ddot{y} &= -k_x(y - y_m) - F_y,\\
m_z\ddot{z}_m &= -k_z(z_m - z) - k_A z_m - F_p,\\
m_t\ddot{z} &= k_z(z_m - z) + F_z.
\end{aligned} \tag{8.45}$$

And the system with damping becomes

$$\begin{aligned}
m_x\ddot{x}_m &= -k_t x_m + k_x(x - x_m) - c_A\dot{x}_m + c_D\left(\dot{x} - \dot{x}_m\right),\\
m_t\ddot{x} &= -k_x(x - x_m) - c_D\left(\dot{x} - \dot{x}_m\right) - F_x,\\
m_x\ddot{y}_m &= -k_t y_m + k_x(y - y_m) - c_A\dot{y}_m + c_D\left(\dot{y} - \dot{y}_m\right),\\
m_t\ddot{y} &= -k_x(y - y_m) - c_D\left(\dot{y} - \dot{y}_m\right) - F_y,\\
m_z\ddot{z}_m &= -k_z(z_m - z) - k_A z_m - c_R\left(\dot{z}_m - \dot{z}\right) - F_p,\\
m_t\ddot{z} &= k_z(z_m - z) + c_R\left(\dot{z}_m - \dot{z}\right) - c_Z\dot{z} + F_z.
\end{aligned} \tag{8.46}$$

For the state space representation the **x** notation can be used for the state, since it contains only Cartesian

coordinates. The state space representations then become

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{(k_t+k_x)}{m_x} & 0 & \frac{k_x}{m_x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{k_x}{m_t} & 0 & -\frac{k_x}{m_t} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{(k_t+k_x)}{m_x} & 0 & \frac{k_x}{m_x} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{k_x}{m_t} & 0 & -\frac{k_x}{m_t} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{(k_z+k_A)}{m_z} & 0 & \frac{k_z}{m_z} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k_z}{m_t} & 0 & -\frac{k_z}{m_t} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\frac{F_x}{m_t} \\ 0 \\ 0 \\ 0 \\ -\frac{F_y}{m_t} \\ 0 \\ -\frac{F_p}{m_z} \\ 0 \\ \frac{F_z}{m_t} \end{bmatrix}. \tag{8.47}$$

And the system with damping becomes

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{(k_t+k_x)}{m_x} & -\frac{(c_A+c_D)}{m_x} & \frac{k_x}{m_x} & \frac{c_D}{m_x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{k_x}{m_t} & \frac{c_D}{m_t} & -\frac{k_x}{m_t} & -\frac{c_D}{m_t} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{(k_t+k_x)}{m_x} & -\frac{(c_A+c_D)}{m_x} & \frac{k_x}{m_x} & \frac{c_D}{m_x} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{k_x}{m_t} & \frac{c_D}{m_t} & -\frac{k_x}{m_t} & -\frac{c_D}{m_t} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{(k_z+k_A)}{m_z} & -\frac{c_R}{m_z} & \frac{k_z}{m_z} & \frac{(c_R+c_Z)}{m_z} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k_z}{m_t} & \frac{c_R}{m_t} & -\frac{k_z}{m_t} & -\frac{c_R}{m_t} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\frac{F_x}{m_t} \\ 0 \\ 0 \\ 0 \\ -\frac{F_y}{m_t} \\ 0 \\ -\frac{F_p}{m_z} \\ 0 \\ \frac{F_z}{m_t} \end{bmatrix}. \tag{8.48}$$

9

Simulations of polishing brush and airbearing

In this chapter results from simulations of all options presented before are discussed. The goal is to find the best description of what is happening during this process and what influences are important. The polishing brush will be modeled with the two-angle model for simplicity.

9.1. Values and assumptions

For the simulations some assumptions must be made and values must be set. A lot of values are already known from the design of the robot and the end effector (see table 9.1). CR means center of rotation of the polishing disc.

The situation considered is a tube in which the tool is moved in a rotating movement and lateral movement, see figure 9.1. The function for this material is given by

$$f(x,y) = \frac{1}{10}\left(\frac{5x}{4}\right)^2 + 5(4y)^2, \tag{9.1}$$

where x and y are given in meters. With the location and directional velocity of the center of rotation the angle of the material can be calculated. The location of the center of the polishing tool is given by

$$Loc(CR) = \begin{pmatrix} R_{rot}\cos(\omega t) + vt + r_d + R_{rot} \\ R_{rot}\sin(\omega t) \end{pmatrix}, \tag{9.2}$$

where t is taken between 0 and 5 seconds for all simulations unless stated otherwise. The velocity vector of

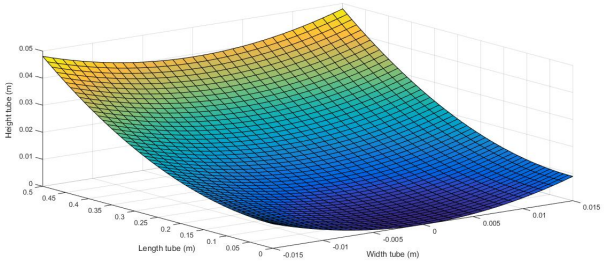

Figure 9.1: *Geometry of the material to be polished.*

the center of rotation is important for calculating the angular elevation of the material. The velocity vector can be used to determine the direction of the tool and thus the direction of the angle that is needed. The velocity vector is given by

$$Vel(CR) = \begin{pmatrix} -\omega R_{rot}\sin(\omega t) + v \\ \omega R_{rot}\cos(\omega t) \end{pmatrix}. \tag{9.3}$$

Table 9.1: Known values

Symbol	Explanation	Value
r_d	radius of brush	0.010 [m]
r_b	radius of rod between tool and air bearing	0.0015 [m]
h	height brush	0.010 [m]
h_b	height of air bearing	0.089 [m]
r	radius from CR to edge point	0.01118 [m]
r_{rot}	radius of robot rotation	0.003 [m]
l	length of rod between tool and air bearing	0.020 [m]
v	longitude speed of robot	0.001 [m/s]
m	mass polishing brush	0.2 [kg]
p	speed of tool rotation	$50 \cdot 2\pi$ [rad/s]
λ	rotation speed of robot	$2 \cdot 2\pi$ [rad/s]
$m_{bearing}$	mass of air bearing	0.337 [kg]
F_p	polishing force	10 [N]
μ	coefficient of friction	$0.5 \cdot \sqrt{(2)} \cdot \frac{1}{2}$ [-]
σ	angle of friction	$\tan^{-1}(\mu)$ [rad]
E	Youngs modulus for steel	$2 \cdot 10^{11}$ [N/m^2]
k_p	spring constant of imaginary contact spring	$1 \cdot 10^8$ [N/m]
k_t	spring constant linear air bearing	$110 \cdot 10^6$ [N/m]
k_r	torsional stiffness air bearing	$23 \cdot 10^6$ [Nm/rad]
k_A	holder stiffness from robot	10^{10} [N/m]
c_A	damper coefficient of the airbearing	1.3159 [Ns]
c_D	damper coefficient of the polishing disc in x and y direction	0.0389 [Ns]
c_R	damper coefficient of the spindle in z-direction	0.5581 [Ns]
c_Z	damper coefficient of the polishing action on the material	$1 \cdot 10^{-8}$ [Ns]
n	dissipation percentage	0.001 [-]
E_{max}	maximum energy present in excitation of spring system	$T_{max} = U_{max}$ [J]
ω_A	natural frequency of airbearing part	$1.3004 \cdot 10^4$ [rad/s]
ω_D	natural frequency of polishing disc part	$1.2194 \cdot 10^3$ [rad/s]
ω_R	natural frequency spindle in z-direction	$2.8970 \cdot 10^4$ [rad/s]

The robot arm positions itself using the center of the outer rotation ω as reference frame. This means that the rod or spindle is positioned in an angle, rod_0, depending on the angle in the center of outer rotation. This value also changes with the direction of the tool movement. When the polishing tool can be pressed down in a straight position the initial values are $\begin{bmatrix}\theta_0\\ \psi_0\end{bmatrix} = \begin{bmatrix}rod_0\\ 0.5*\pi\end{bmatrix}$. The final elevation angle in the direction of the velocity vector $\mathbf{n}$ is then calculated in each point (x, y) using

$$\frac{\partial f}{\partial \mathbf{n}}(x,y) = \frac{1}{\|\mathbf{n}\|}\left(n_x\frac{\partial f}{\partial x} + n_y\frac{\partial f}{\partial y}\right), \tag{9.4}$$

$$\xi(x,y,\mathbf{n}) = \tan^{-1}\left(\frac{\partial f}{\partial \mathbf{n}}(x,y)\right). \tag{9.5}$$

The solver used for integrating the system is the ode45 solver of Matlab. This solver uses the integration method Runge-Kutta of order 4 and 5 to compare the final result with an error stopping criteria. Matlab will adjust the time step size until the stopping criteria has been met.

9.2. Variables and local coordinate systems

To show all variables used for the calculation three schematic pictures are made of the three different views in a 3D situation, see figures 9.2, 9.3 and 9.4. The points C_1, C_2, C_3 and C_4 are located and defined as before. The blue straight arrow in figures 9.2 and 9.4 indicates that the general movement goes from C_1. λ in figure 9.4 is the robot rotation assumed at 2 Hz. θ, ψ and p are positive in the indicated direction. The yellow circle means the ball joint in the middle of the polishing brush.

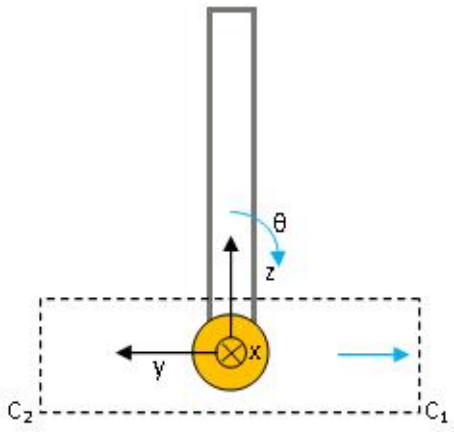

Figure 9.2: *Polishing brush seen from the side, yz-frame.*

9.3. No moment present in polishing disc model

In the situation that there is no moment present in the model, it means that there is no cause for the polishing brush to move out of its equilibrium stable position. Simple simulations show in the two angle model that this is correct, see figure 9.5. The first two graphs are the angles θ and ψ as functions of time in seconds. The two lowest graphs are the elevation in meters and in radians. The elevation in meters is the function value of the height of the material from the assumed zero plane. The elevation in radians is the angle the elevation makes in the direction of the movement.

The material function defined before will be used with all computations.

9.4. Full model

The full polishing disc and airbearing model is formed by two separate parts. In the first part the angular equations of θ and ψ are calculated. In the second part the solution of this system is used to calculate the impact forces in the x, y and z directions to see the effect on the entire airbearing and spindle model. The reason that these two models are separated is because the polishing disc model requires a much higher ac-

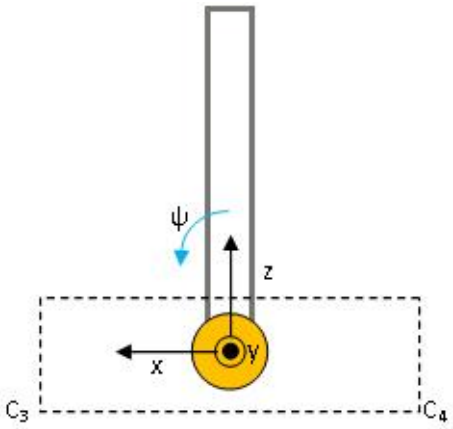

Figure 9.3: *Polishing brush seen from the side, xz-frame.*

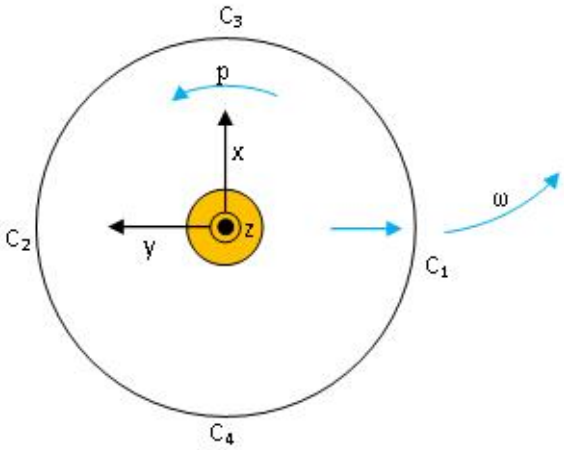

Figure 9.4: *Polishing brush seen from above, xy-frame.*

curacy and therefore smaller time step than the airbearing model. But since the ode45 solver of Matlab looks at an average of the error the solution to the polishing disc model can be very different from the actual one.

There are 6 basic values that are tested. Different contact spring rates k_p are used to study the effect. The different basic simulations can be found in table 9.2. The figures of the results that are not specifically discussed can be found in appendix A.

9.4.1. Results of run 3

The results for $k_p = 10^8$ are presented in this section. In figure 9.6 the angles and angular velocities are presented using a spring height of 1 mm for C_1 and C_2. For C_3 and C_4 the spring height from equation (7.43) is used. In figure 9.7 the forces in the different principal axes are presented. What can be seen is that the magnitude of the impact forces are a lot higher than, for instance, the polishing force of 10 [N].

Then figures 9.8, 9.9 and 9.10 show the deviations from the equilibrium position and the total deviation

Table 9.2: Known values

Number	Spring rate
1	10^4 N/m
2	10^6 N/m
3	10^8 N/m
4	10^{10} N/m

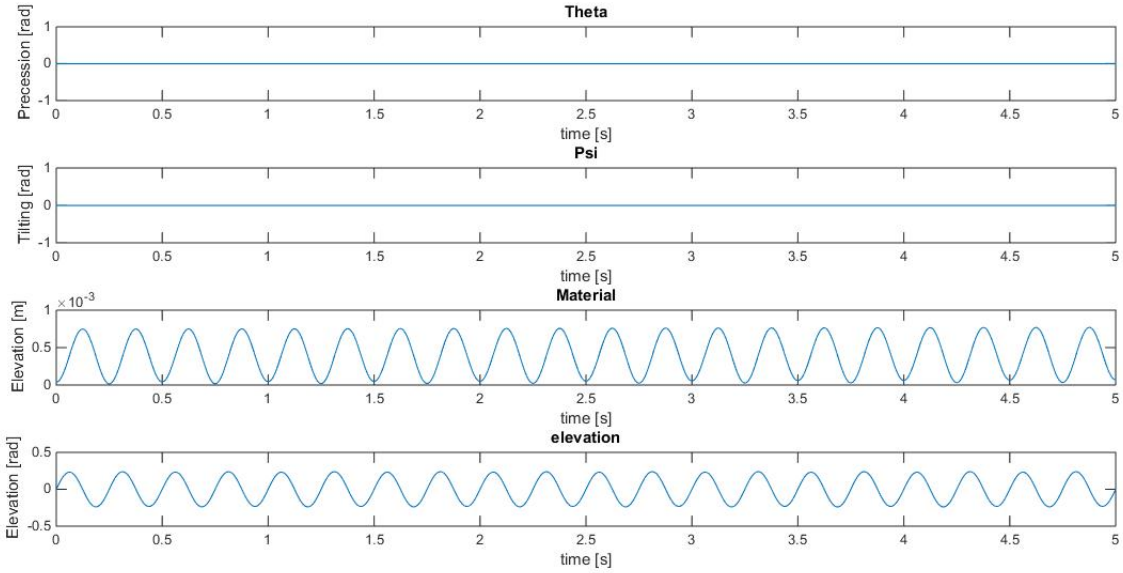

Figure 9.5: *Graph of θ and ψ over an elevated material without impact.*

of the polishing tool without damping. They are all growing without any damping. This is obviously not a realistic physical result.

Adding a simple damping of 0.1% already solves the growing in x-direction and y-direction (see figures 9.11 and 9.12). For the z-direction there does not seem to be a lot of difference. For this reason the extra spring k_A was introduced in section 8.3. The value of the spring constant depends on the information about the difference in flexibility of the spindle and the holder. For $k_A = 10^9$ the results show that the deviations for the holder are 10 times smaller than for the spindle. Using $k_A = 10^{10}$ this difference is 100 times. Since the holder and spindle are both very stiff materials, but the spindle will have a bit more expanding room, choosing $k_A = 10^9$ seems appropriate.

If the damping is increased to 1 % the deviations in x-direction and y-direction are half as small as when the damping is 0.1 % (see figures 9.15 and 9.16). In the z-direction this is not the case. The vibrations are then only thinned, see figure 9.17. So concluding, damping helps a lot, but mostly in the x and y direction. Increasing the damping also decreases the deviations, but still not enough to get a realistic result. Looking at the forces, this might be caused by the magnitude. These forces seem unrealistic large coming from a polishing brush. This is a problem that is caused by the contact force description and is not solved yet.

To be able to say something about the entire model, a workaround is needed. For this the impact force derived in the polishing disc model is divided by a large factor to get a realistic result and to say something about the feedback design. The frequency of the impact force is used from the polishing disc model, but the magnitude is simply adjusted.

9.4.2. Results of run 1

When using $k_p = 10^4$ the behavior of the polishing disc is very different (see figure 9.18). First of all the magnitude of the forces are smaller, see figure 9.19. Comparing the forces of figure 9.7 with figure 9.19 in the x-direction it can be seen that the forces are ten times smaller. The forces are now also a more continuous function. The more varying force results in a smaller deviation in the x-direction, see figure 9.20. For the y-direction the magnitude is now also four times smaller and the forces in y-direction can also be negative. This results in total in smaller deviations, see figure 9.21. With the z-direction there is still the same problem, see figure 9.22.

The effect of 0.1 % damping is a four times smaller deviation in the total x-direction, see figure 9.23. In the y-direction this is already ten times smaller, see figure 9.24. In the z-direction adding the damping and holder spring gives a lot better results (figure 9.25).

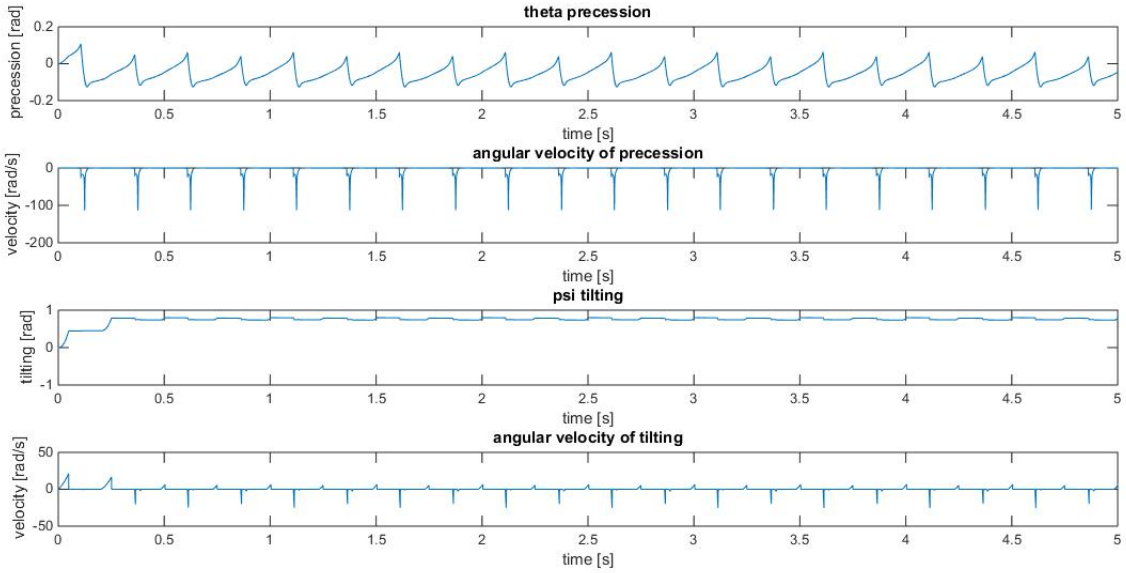

Figure 9.6: Run 3 $\theta, \dot{\theta}, \psi, \dot{\psi}$ *presented over 5 seconds.*

9.4.3. Results of run 4

When $k_p = 10^{10}$ N/m θ remains at one angle and bounces a bit. The reason the result from this run is interesting is because the forces are now defined in such a manner that the deviations in the x-direction are bounded and not increasing continuously, see figure 9.28. This does indeed indicate that the magnitude and frequency of the impact force is very important for damping and stability of the system.

9.4.4. Decreasing contact spring height

The assumed layer of 1 mm in the y-direction or C_1 and C_2 contact is a guess to approximate when the polishing brush is impacting the material. When this is decreased to 1 μm $k_p = 10^{10}$ gives a realistic result, see figure 9.29. The forces in y-direction are still equal in magnitude compared to 1 mm layer.

9.4.5. Workaround with force factor

One of the reasons the force might become too large is that the impact impulse is converted to a force, by division by a very small duration time. The impulse is defined as the total force applied in that impact duration time. Outside of this time there is no applied force. This may raise the question if this duration time is well defined or if the force description is not too high.

Another possibility is that the forces are too high due to the material definition chosen and the friction coefficient choice that causes the velocities and accelerations in the polishing disc model. This can be tested if an actual product is used for the simulation to see if the assumed geometry is realistic.

Even though impact forces should be very large, for some reason the model of the airbearing cannot handle this. A real solution to this problem is not yet found, but in order to simulate the actual situation the forces are reduced to a realistic magnitude. The force frequencies and behavior are still used this way, but simply scaled to a realistic magnitude. Two values are used: dividing the forces by 10000 and multiplying the forces by the impact duration t_m such that the impulse value is used. This might not be physically correct with the units, but it is a way to adjust each run in a proportional way.

The best results for the 10000 workaround come from run 1. Figures 9.31, 9.32 and 9.33 show the results for run 1 for 0.1 % damping. The scale is now a lot better. In the y-direction the maximum deviation lies around 1.5 cm due to the stiffness of the spindle. The deviations of the airbearing go up to 40 μm in the y-direction.

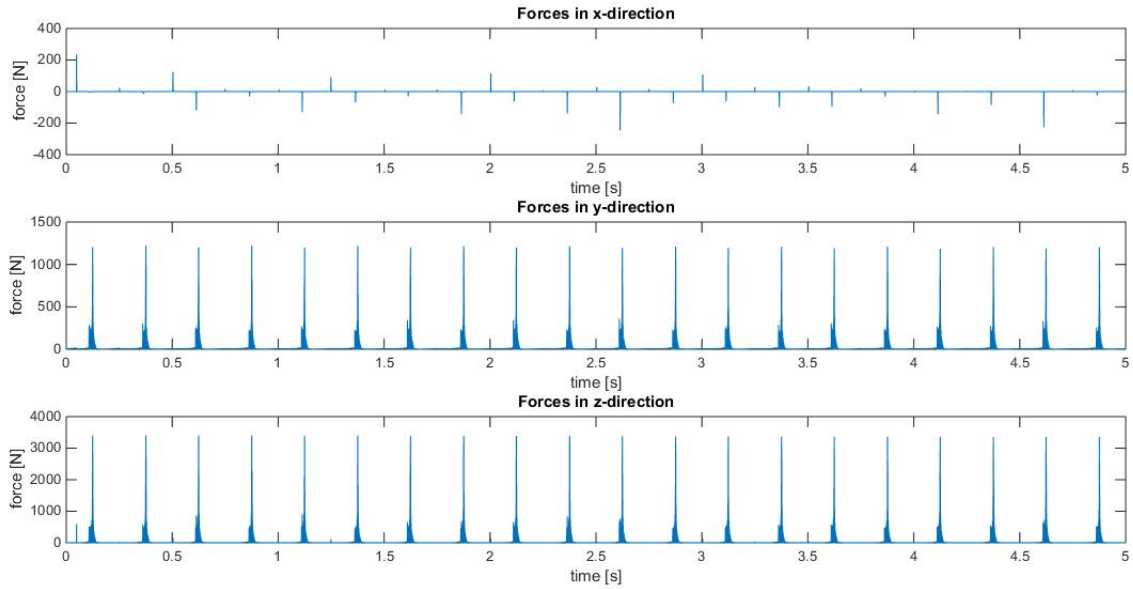

Figure 9.7: *Run 3 forces in the three principal directions in [N].*

Figures 9.34, 9.35 and 9.36 show the results of 1 % in the same situation. These results only show less intense vibrations, less thick. Depending on what seems most realistic the correct damping can be chosen.

The best results for the t_m workaround are from run 4. Figures 9.37, 9.38 and 9.39 show the results for run 1 for 0.1 % damping. In the original results run 3 seemed very promising, but with these factors the deviations in y-direction are still up to 5 cm. This is too high. In figure 9.38 the maximum deviation is up to 5 mm. These deviations seem promising for the problem description.

Figures 9.40, 9.41 and 9.42 show the results of 1 % in the same situation. These results show less intense vibrations, less thick, but also a bit smaller. Again depending on what seems most realistic the correct damping can be chosen.

These results show that smaller forces indeed give realistic vibrations. But this still needs some tuning to find the right magnitude. This is still a problem originating from the polishing disc model. Since the goal in this thesis is not only to model but also to discuss possible correction methods it is important to use the workaround with realistic results to investigate the use of the airbearing model. The four different runs discussed in this section will be used as data sets for testing the correction methods. These correction methods will be discussed in the next chapter.

9.5. Concluding remarks

Simulating situations where rotations, friction and impact forces are present is very difficult. The model found in this thesis still has some problems, but in the end the model simulates the observed problem with a simple workaround. The description of the model consists of all parts present in the end-effector. The biggest problem found in the end is the magnitude of the resultant forces in C_1 and C_2. As it turns out α and β have a magnitude that ensures that the forces are very big and cause a high angular velocity and therefore a large impact force. The obvious reason for this could be that the geometry of the material to be polished is not realistic, i.e., the angles are too steep. Another factor could be that the friction coefficient is too large.

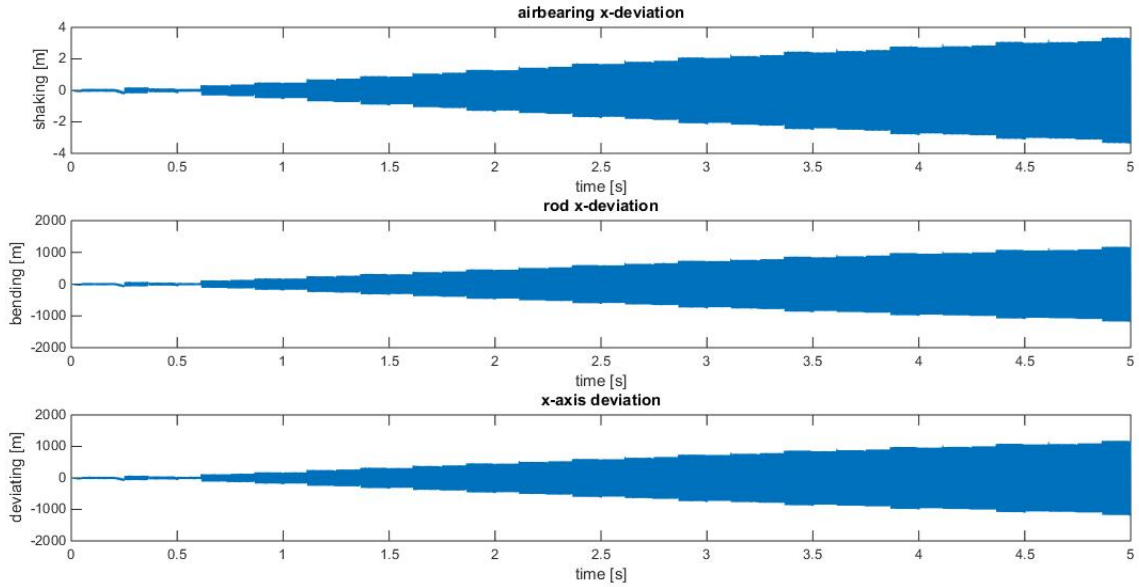

Figure 9.8: *Run 3 deviations in x-direction without damping in [m].*

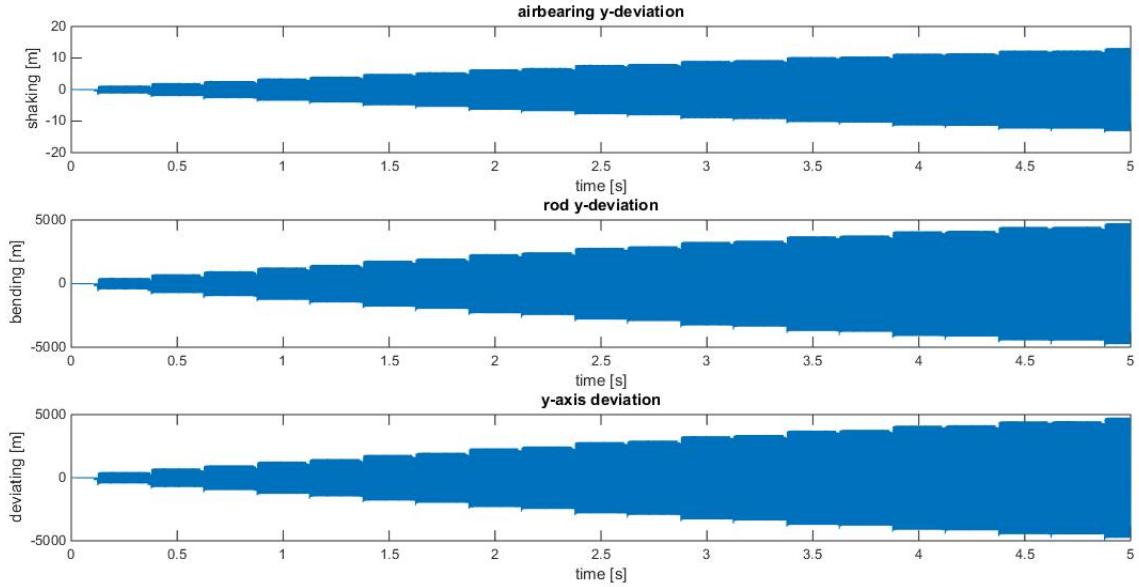

Figure 9.9: *Run 3 deviations in y-direction without damping in [m].*

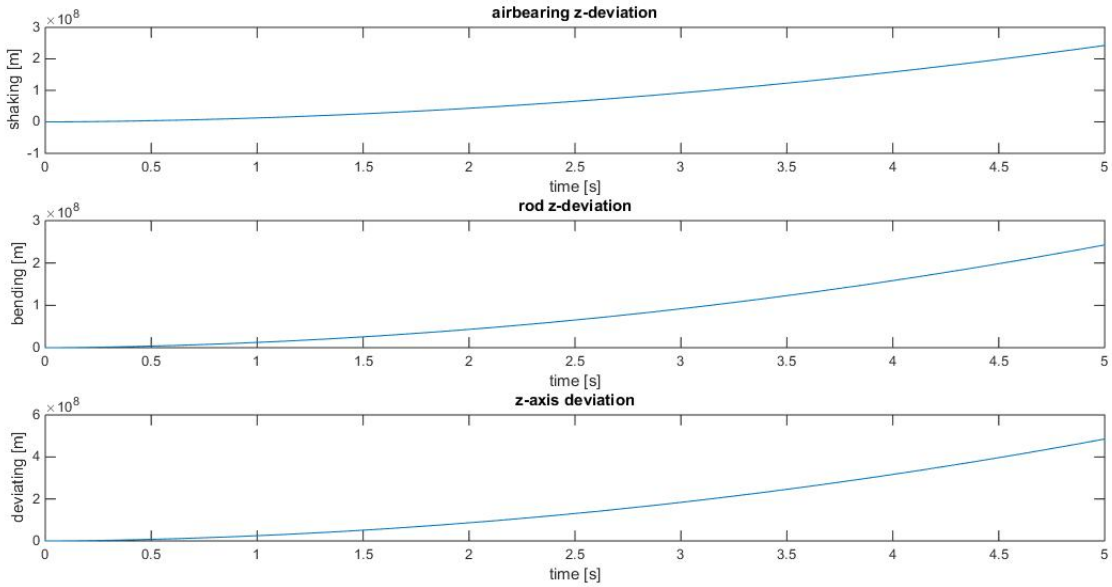

Figure 9.10: *Run 3 deviations in z-direction without damping in [m].*

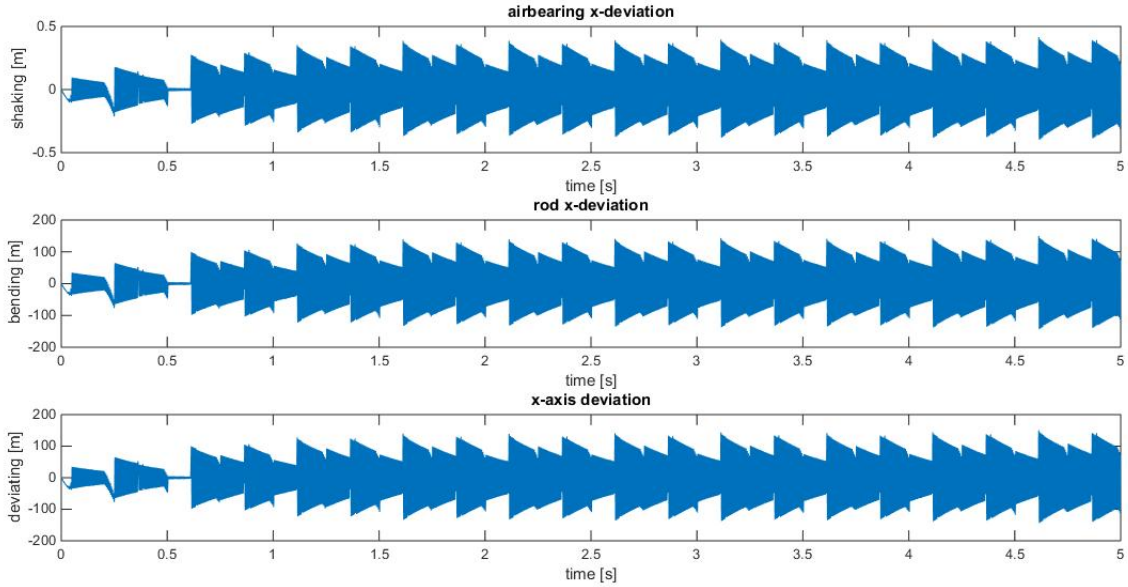

Figure 9.11: *Run 3 deviations in x-direction with* 0.1 *% damping in [m].*

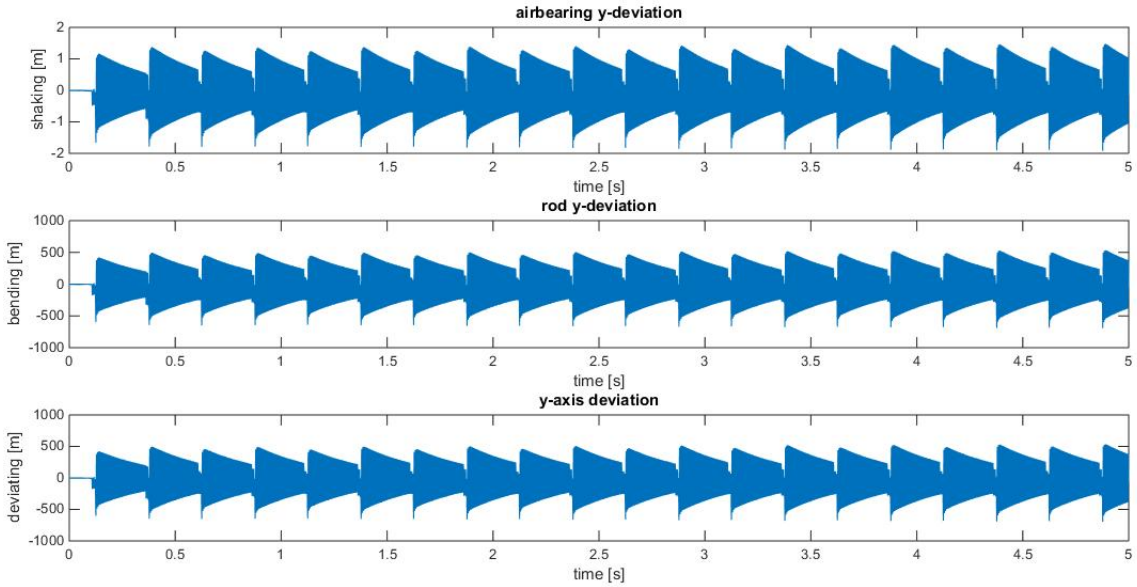

Figure 9.12: *Run 3 deviations in y-direction with* 0.1 *% damping in [m].*

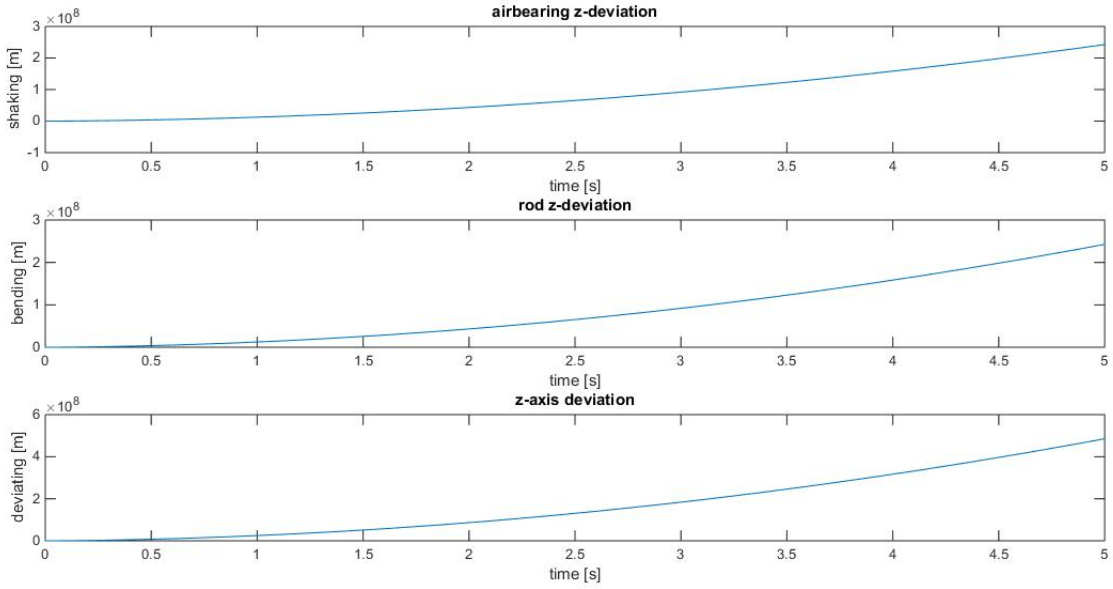

Figure 9.13: *Run 3 deviations in z-direction with* 0.1 *% damping in [m].*

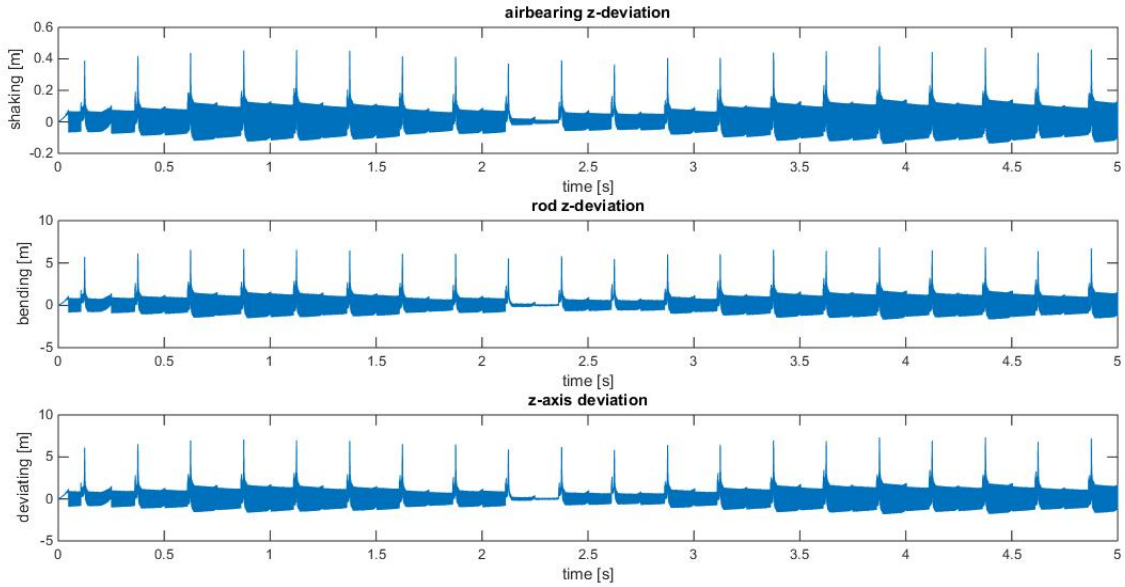

Figure 9.14: *Run 3 deviations in z-direction with* 0.1 *% damping and holder spring in [m].*

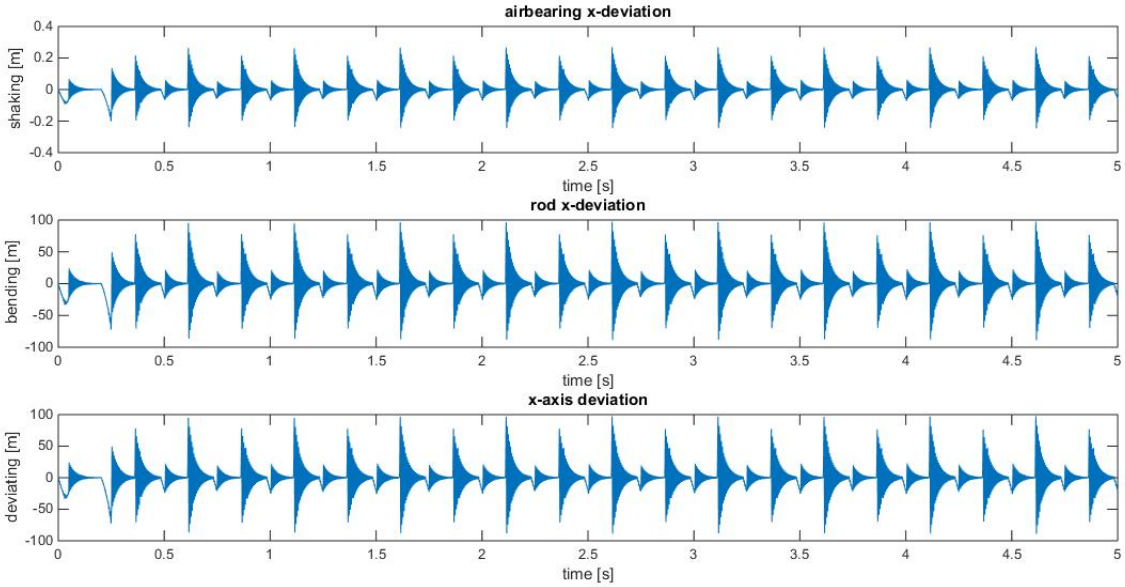

Figure 9.15: *Run 3 deviations in x-direction with* 1 *% damping in [m].*

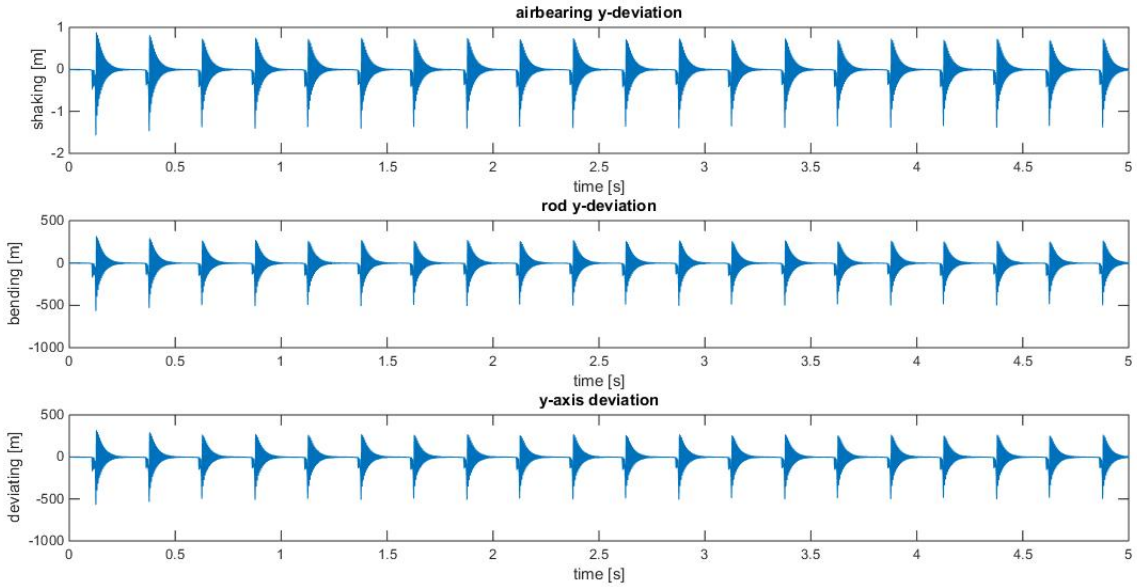

Figure 9.16: *Run 3 deviations in y-direction with* 1 *% damping in [m].*

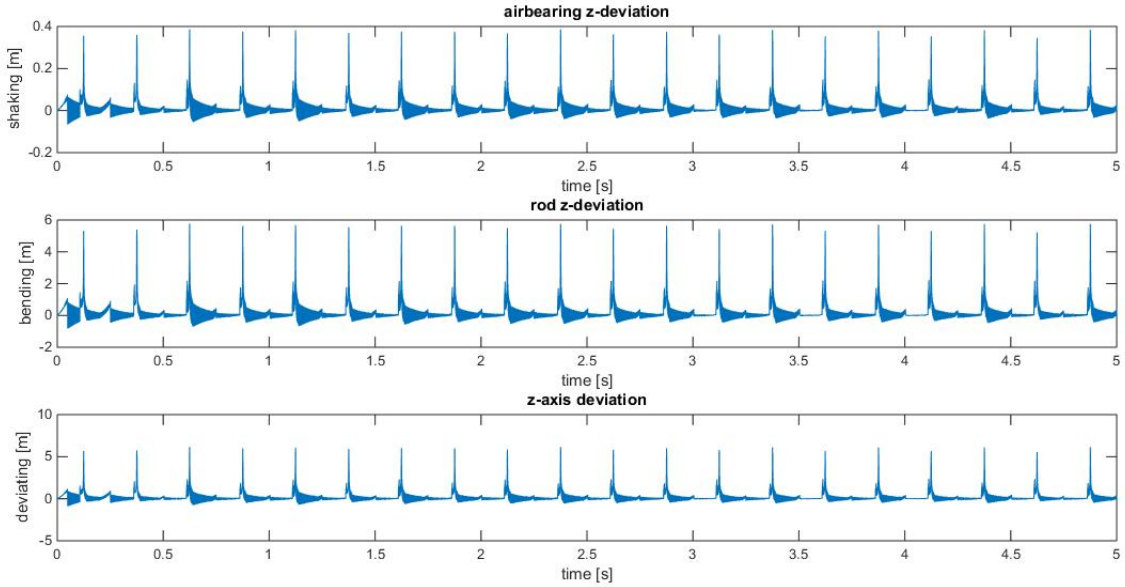

Figure 9.17: *Run 3 deviations in z-direction with* 1 *% damping in [m].*

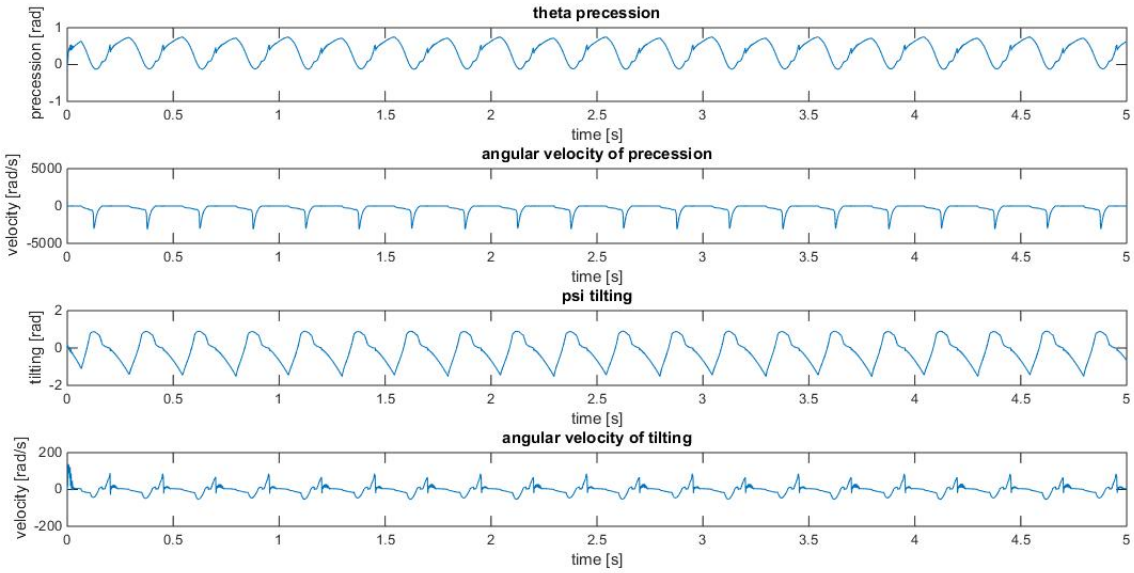

Figure 9.18: *Run 1 $\theta, \dot{\theta}, \psi, \dot{\psi}$ presented over 5 seconds.*

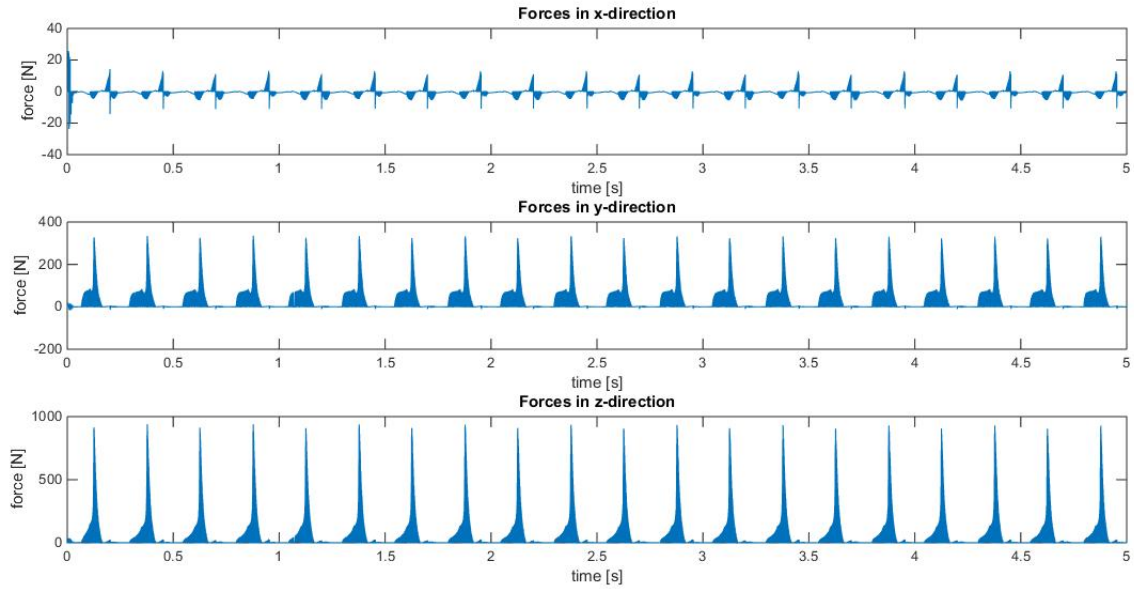

Figure 9.19: *Run 1 forces in the three principal directions in [N].*

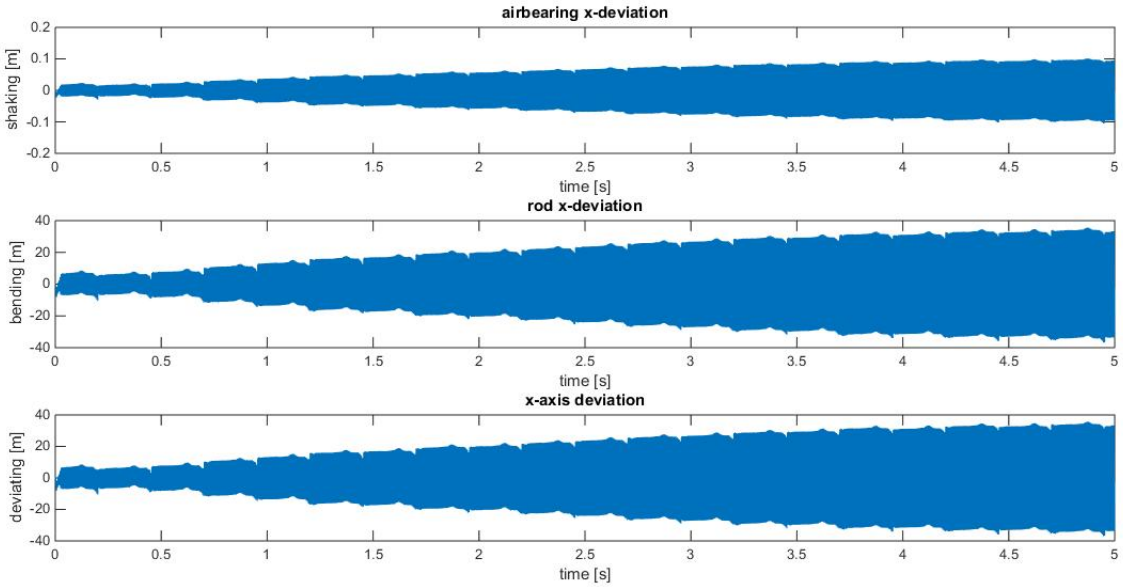

Figure 9.20: *Run 1 deviations in x-direction without damping in [m].*

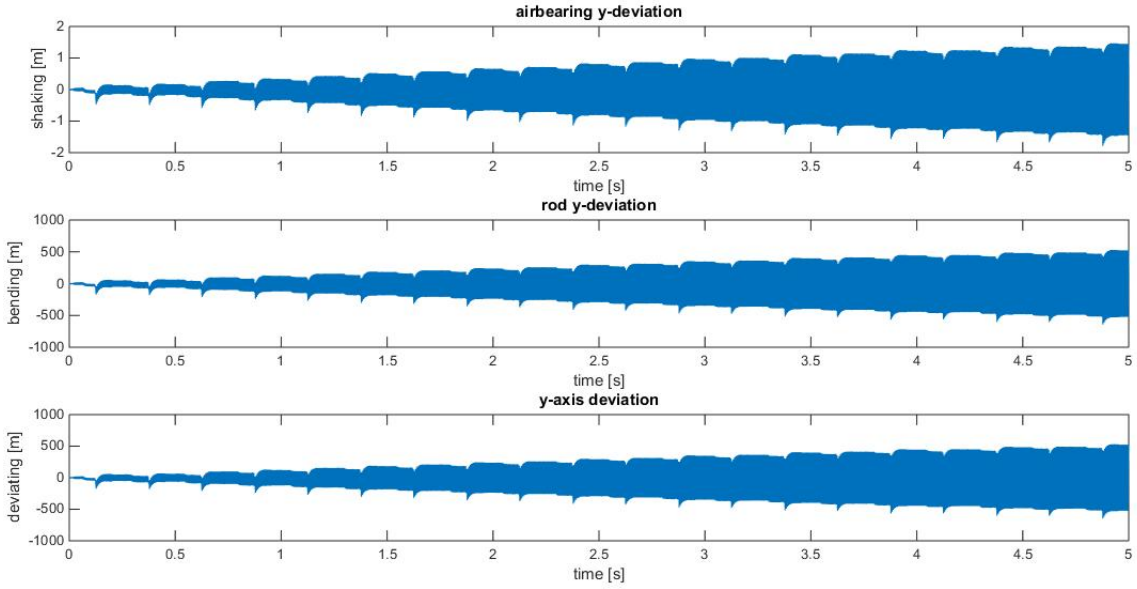

Figure 9.21: *Run 1 deviations in y-direction without damping in [m].*

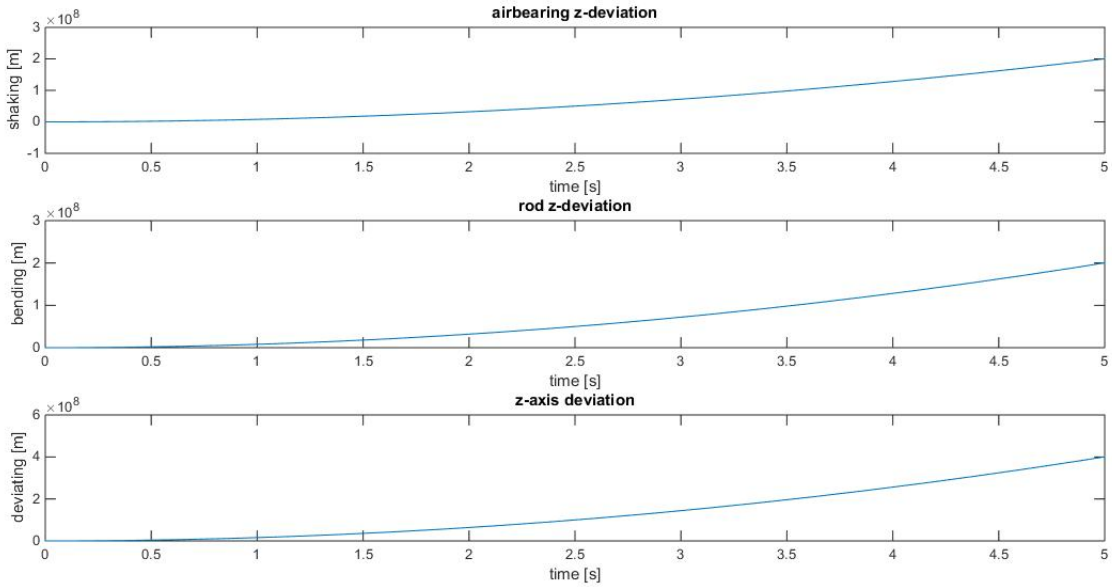

Figure 9.22: *Run 1 deviations in z-direction without damping and holder spring in [m].*

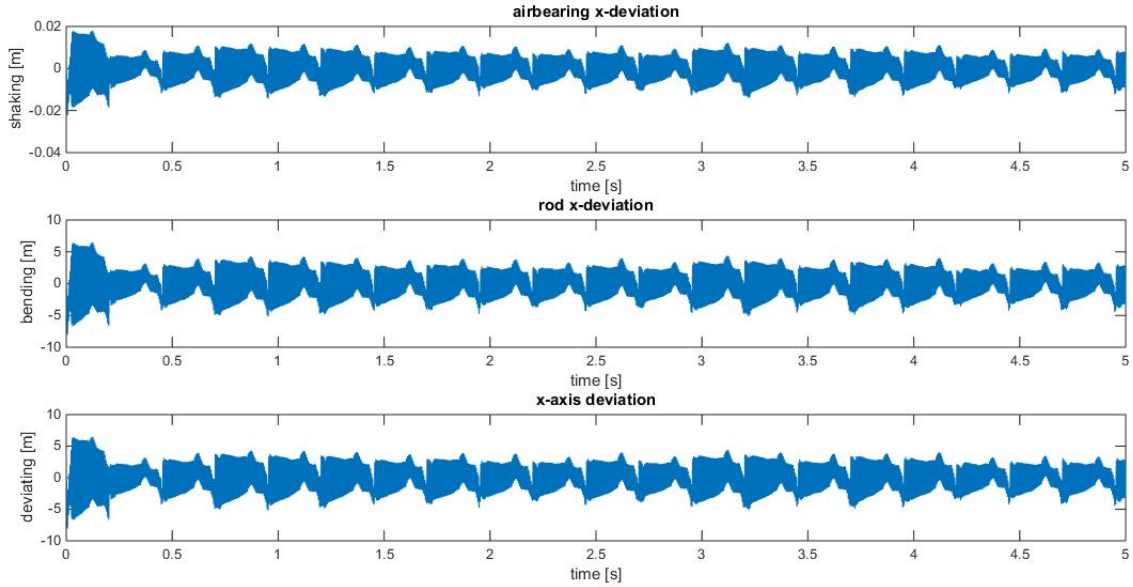

Figure 9.23: *Run 1 deviations in x-direction with* 0.1 *% damping in [m].*

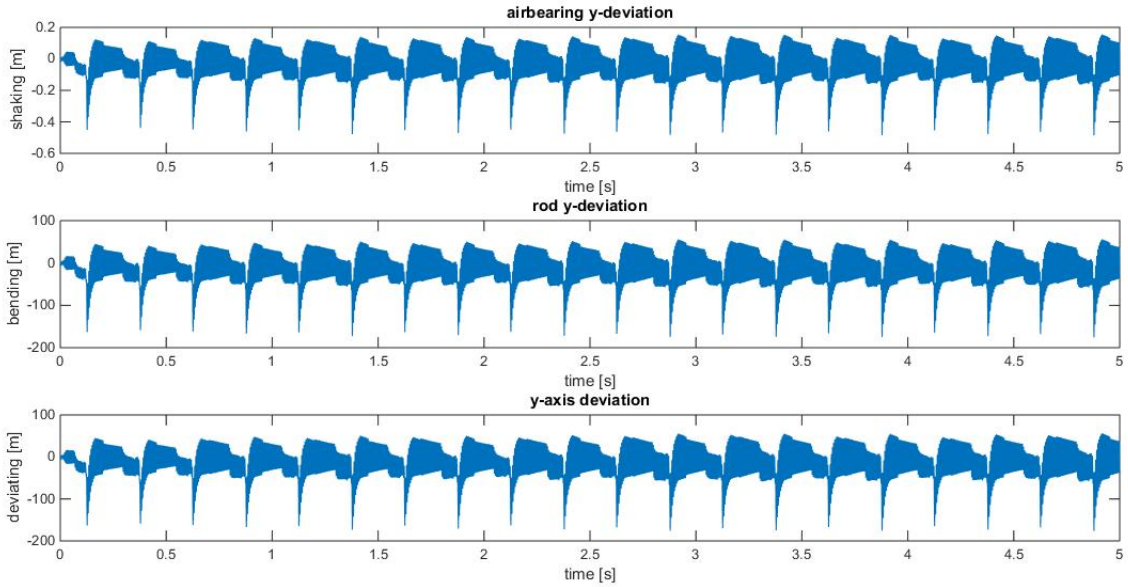

Figure 9.24: *Run 1 deviations in y-direction with* 0.1 *% damping in [m].*

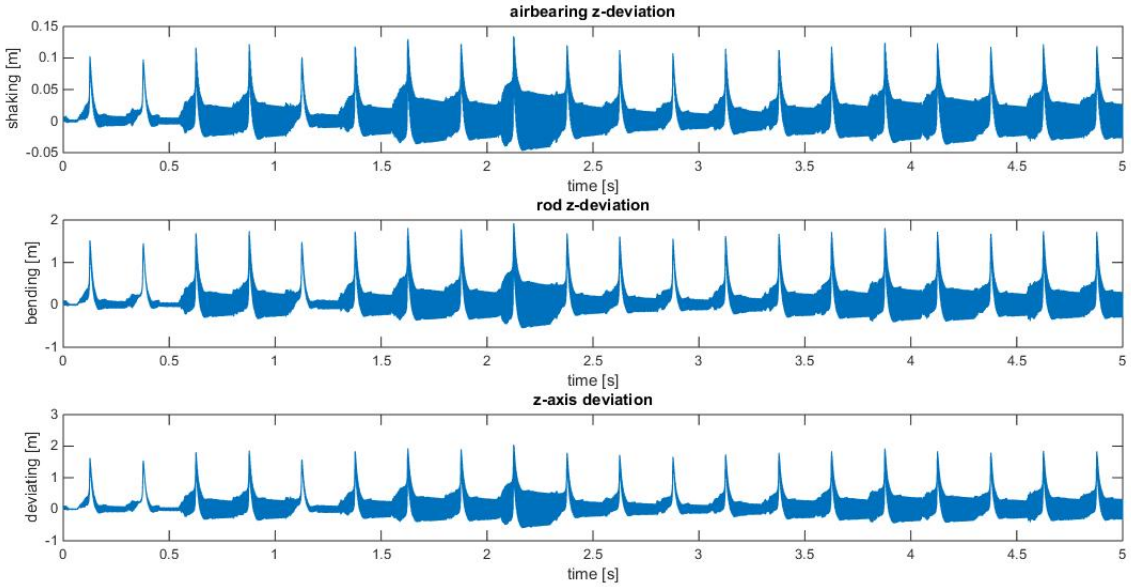

Figure 9.25: *Run 1 deviations in z-direction with* 0.1 *% damping in [m].*

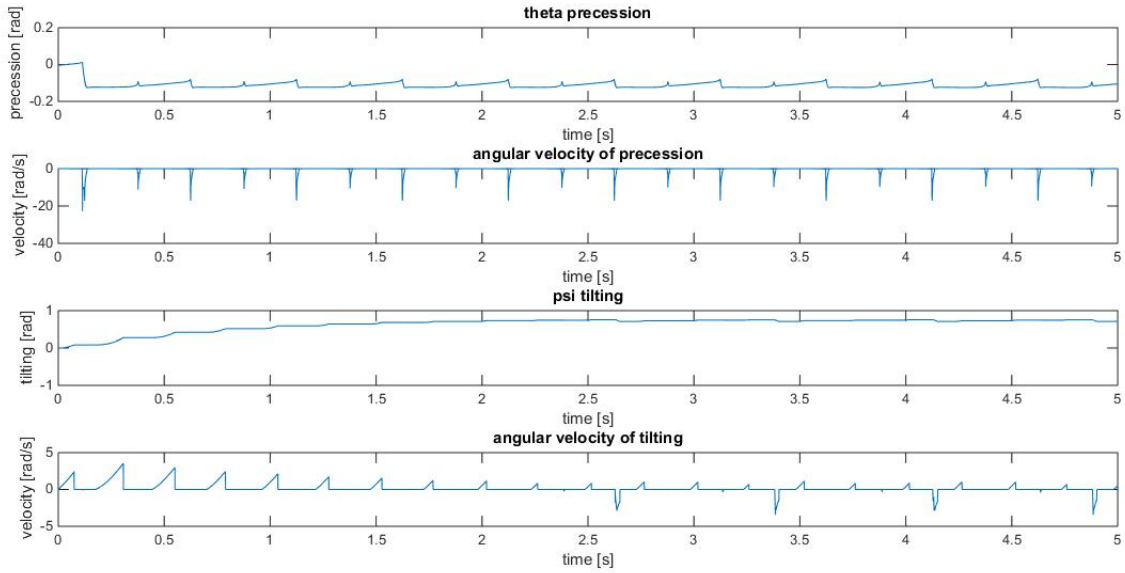

Figure 9.26: *Run 4 $\theta, \dot{\theta}, \psi, \dot{\psi}$ presented over 5 seconds.*

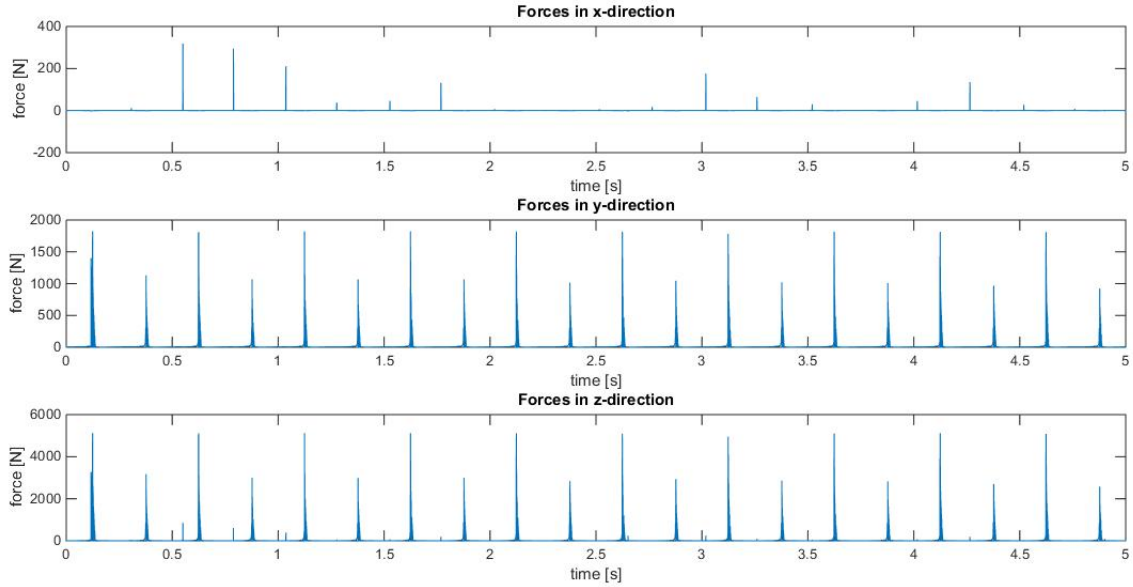

Figure 9.27: *Run 4 forces in the three principal directions in [N].*

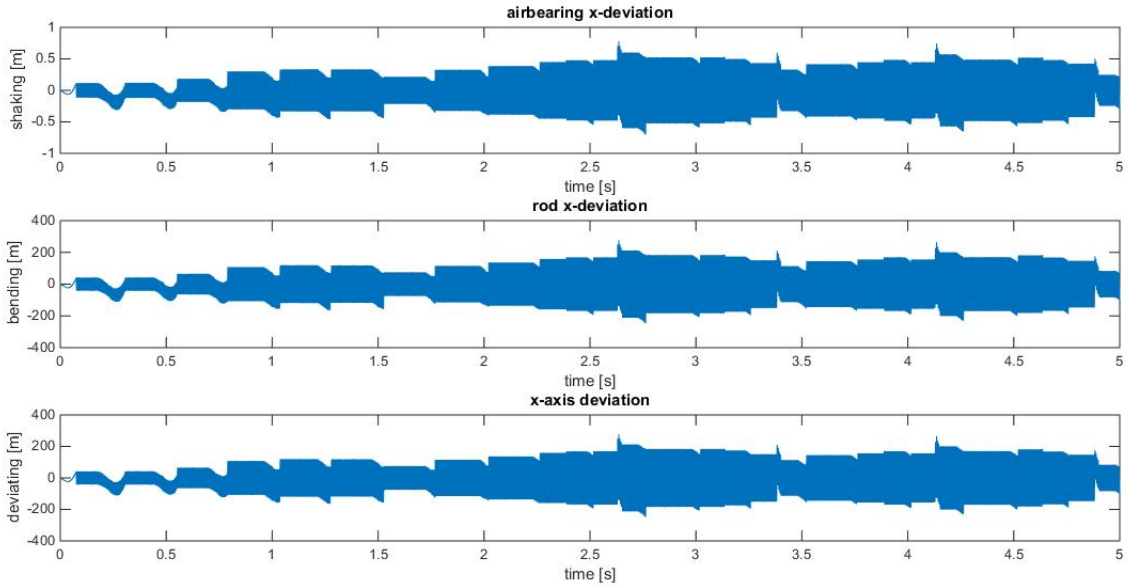

Figure 9.28: *Run 4 deviations in x-direction without damping in [m].*

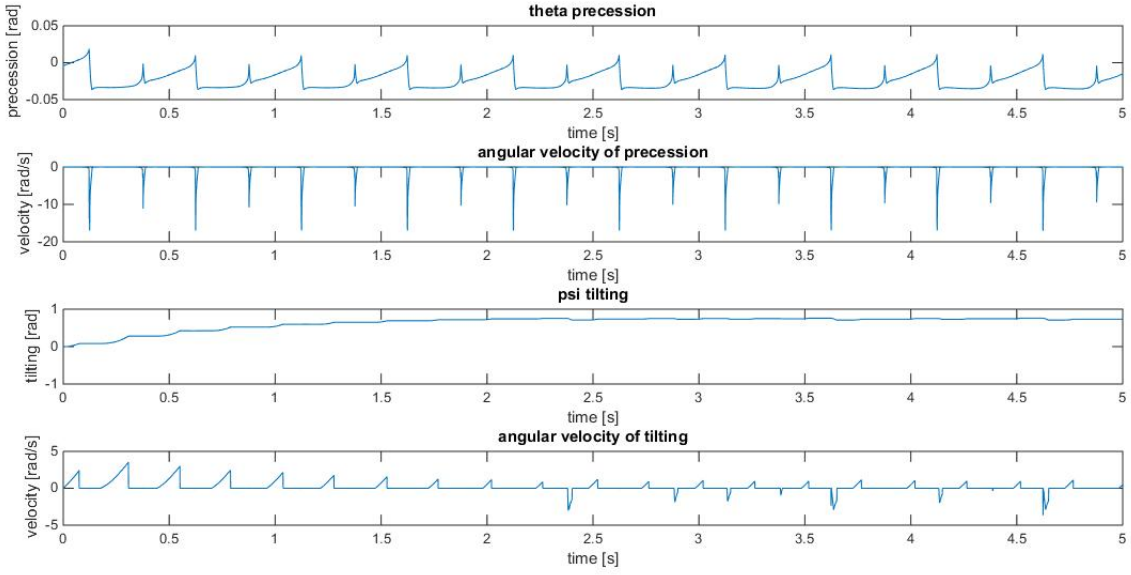

Figure 9.29: $\theta, \dot{\theta}, \psi, \dot{\psi}$ *presented over 5 seconds with decreased contact spring height.*

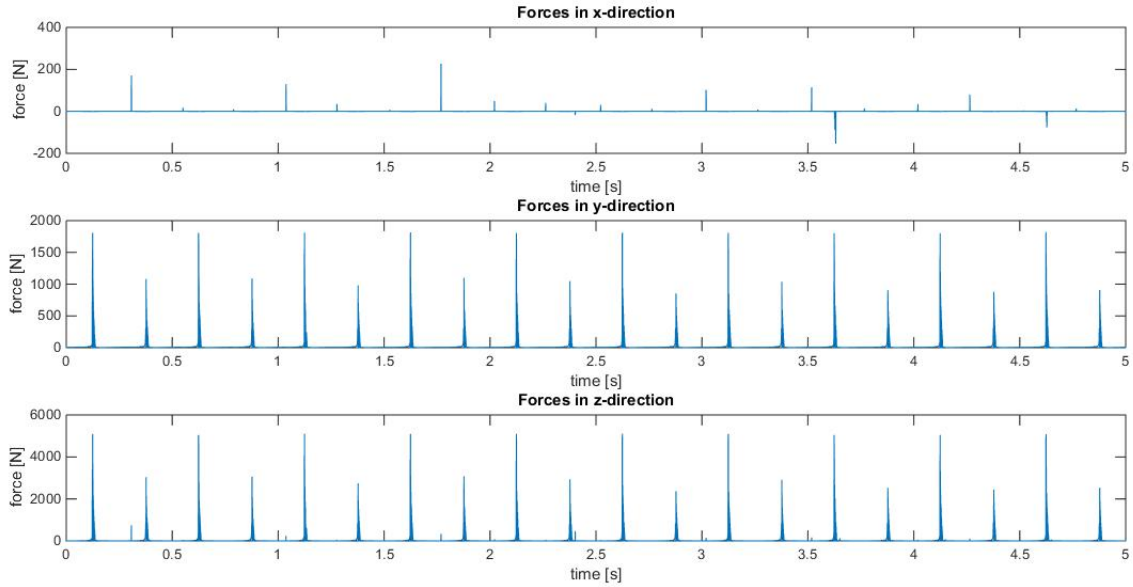

Figure 9.30: *Forces in the three principal directions in [N] with decreased contact spring height.*

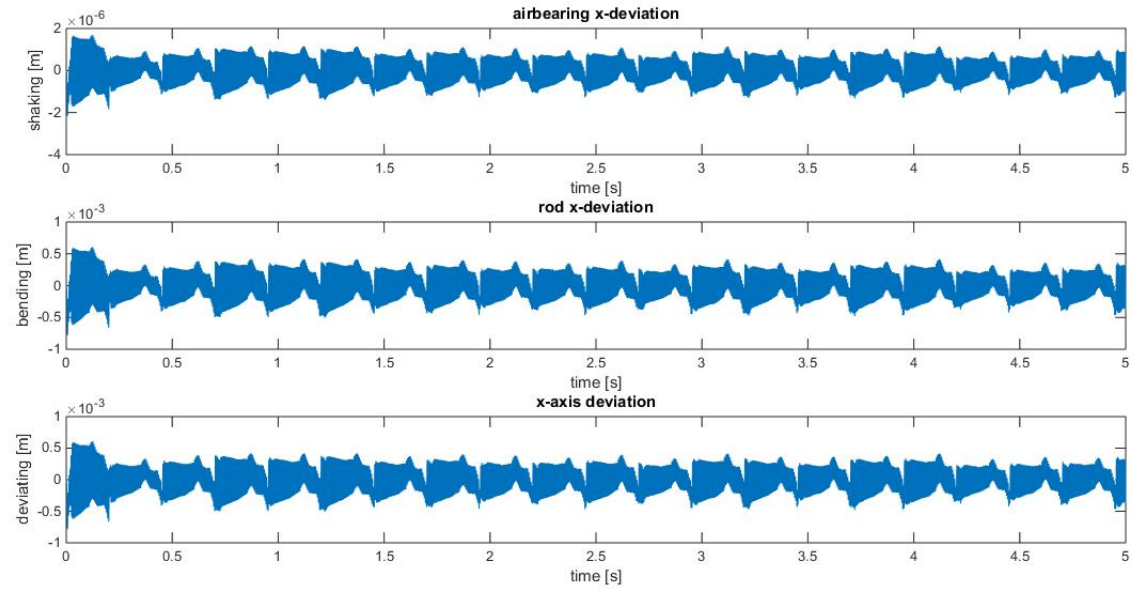

Figure 9.31: *Run 1 workaround deviations in x-direction with* 0.1 *% damping and dividing factor of* 10000 *in [m].*

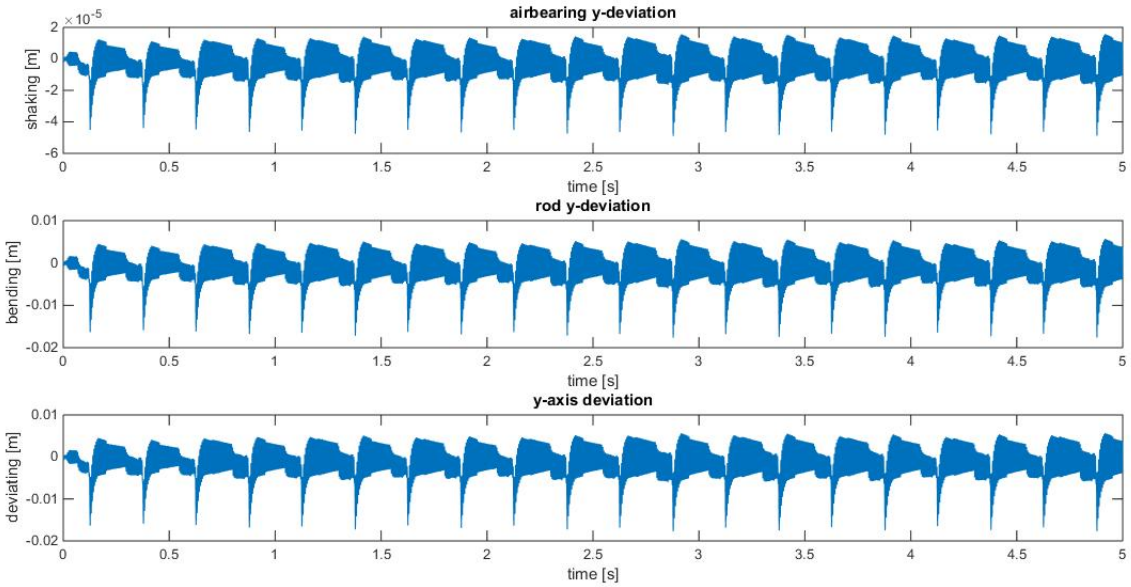

Figure 9.32: *Run 1 workaround deviations in y-direction with* 0.1 *% damping and dividing factor of* 10000 *in [m].*

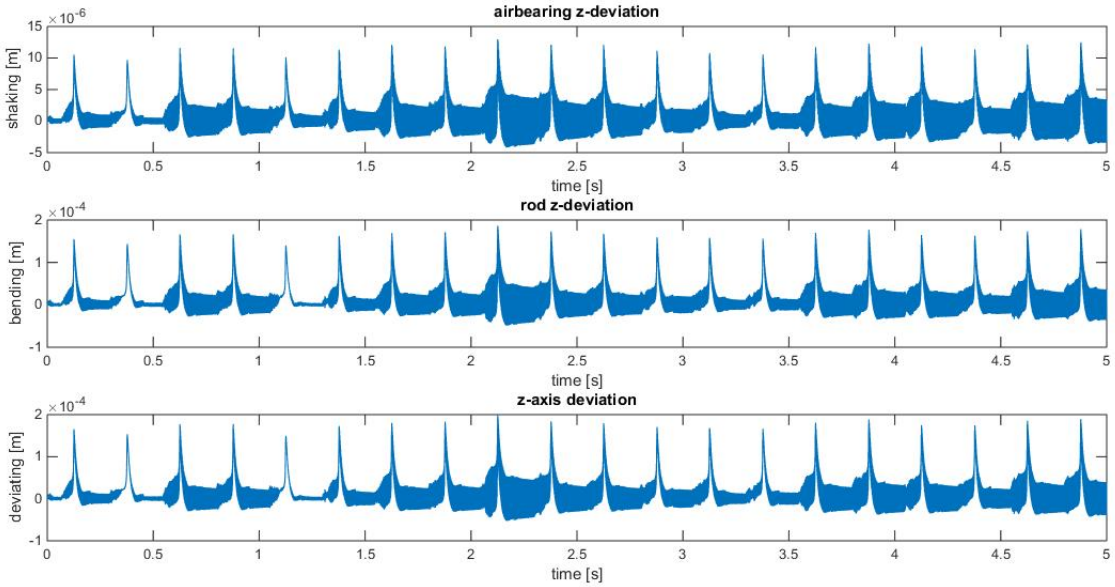

Figure 9.33: *Run 1 workaround deviations in z-direction with* 0.1 *% damping and dividing factor of* 10000 *in [m].*

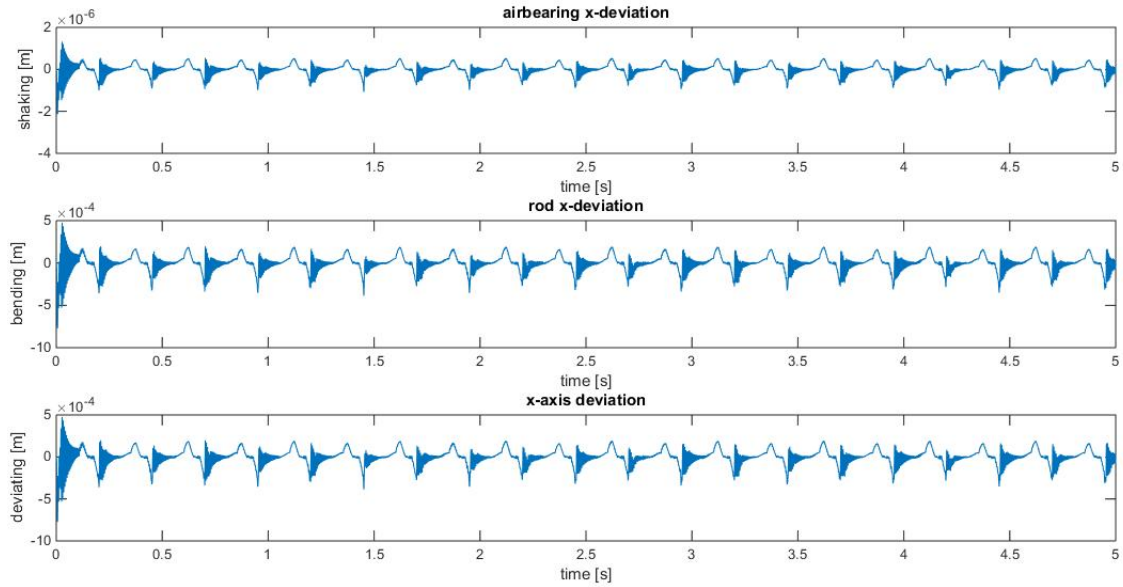

Figure 9.34: *Run 1 workaround deviations in x-direction with* 1 *% damping and dividing factor of* 10000 *in [m].*

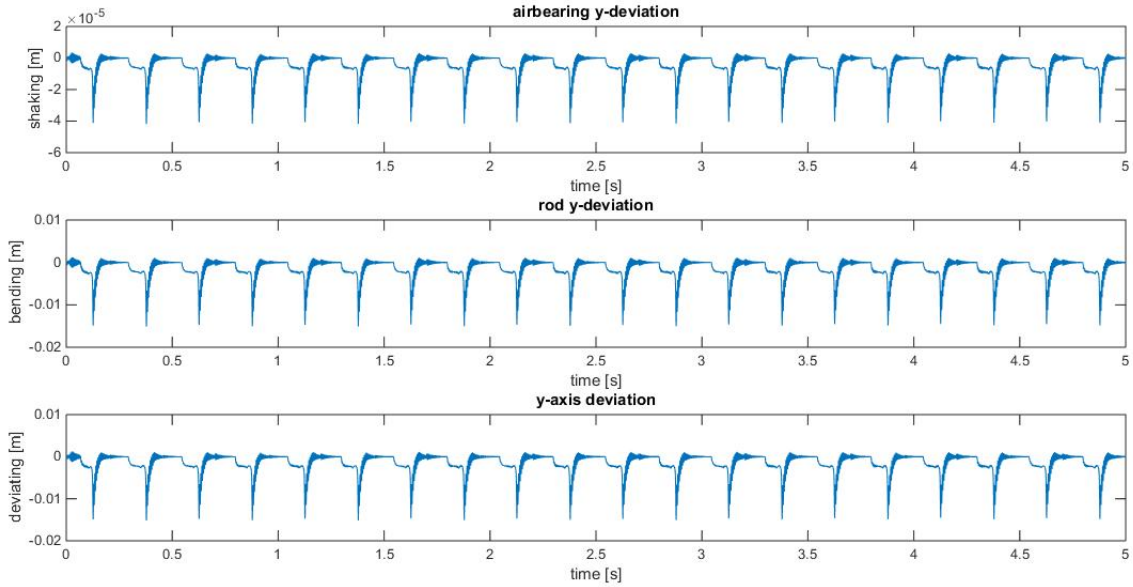

Figure 9.35: *Run 1 workaround deviations in y-direction with* 1 *% damping and dividing factor of* 10000 *in [m].*

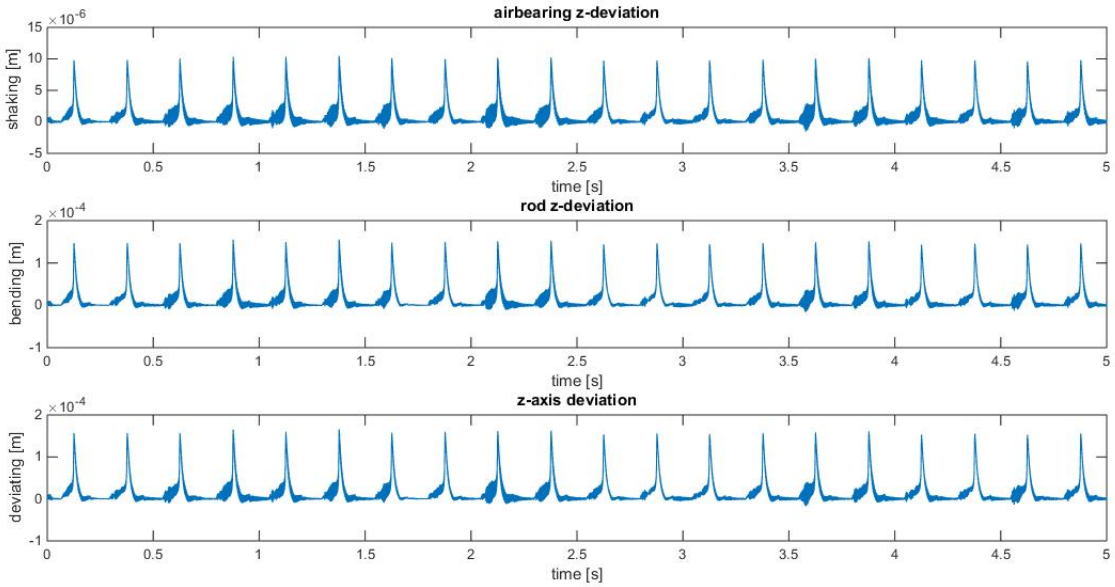

Figure 9.36: *Run 1 workaround deviations in z-direction with* 1 *% damping and dividing factor of* 10000 *in [m].*

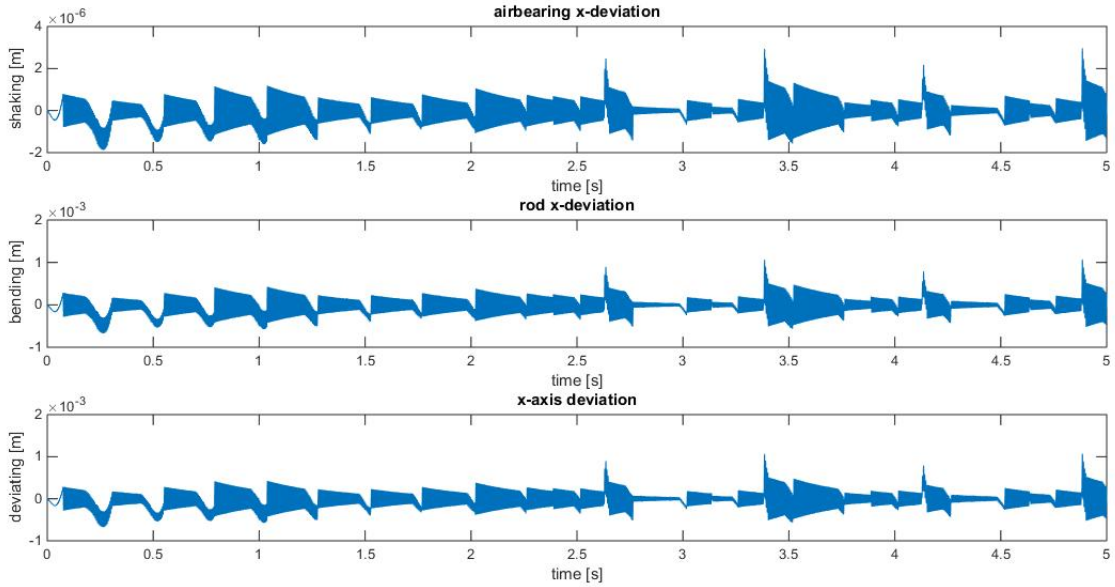

Figure 9.37: *Run 4 workaround deviations in x-direction with* 0.1 *% damping and multiplication factor of* t_m *in [m].*

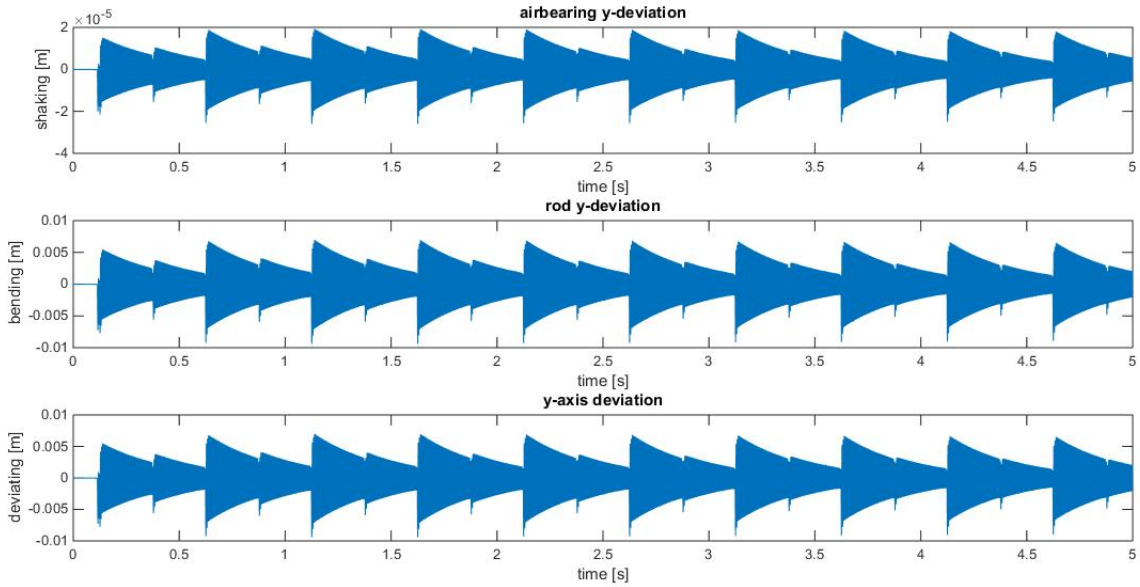

Figure 9.38: *Run 4 workaround deviations in y-direction with* 0.1 *% damping and multiplication factor of* t_m *in [m].*

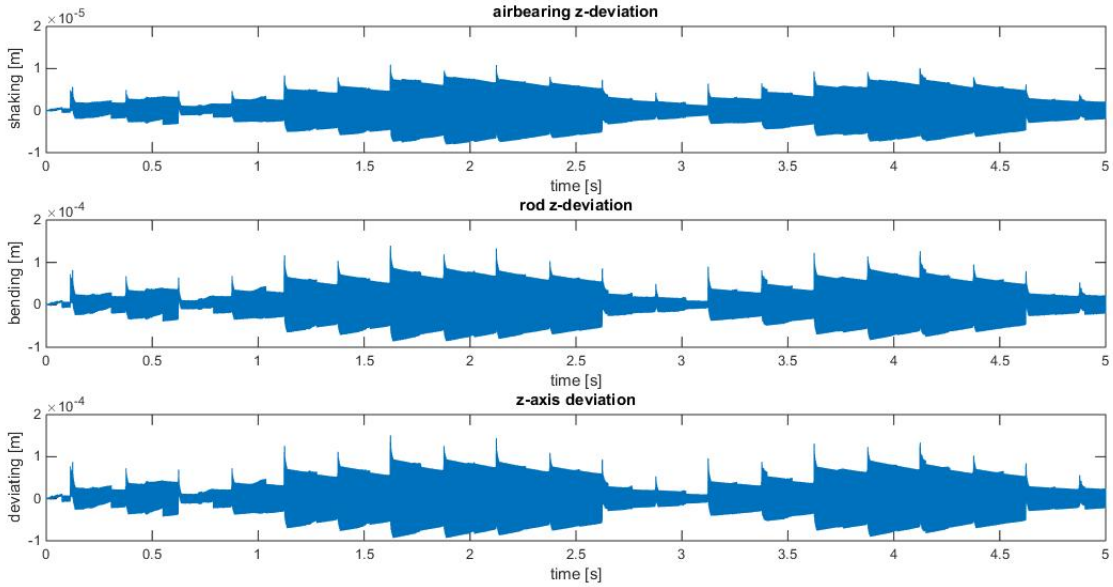

Figure 9.39: *Run 4 workaround deviations in z-direction with* 0.1 *% damping and multiplication factor of* t_m *in [m].*

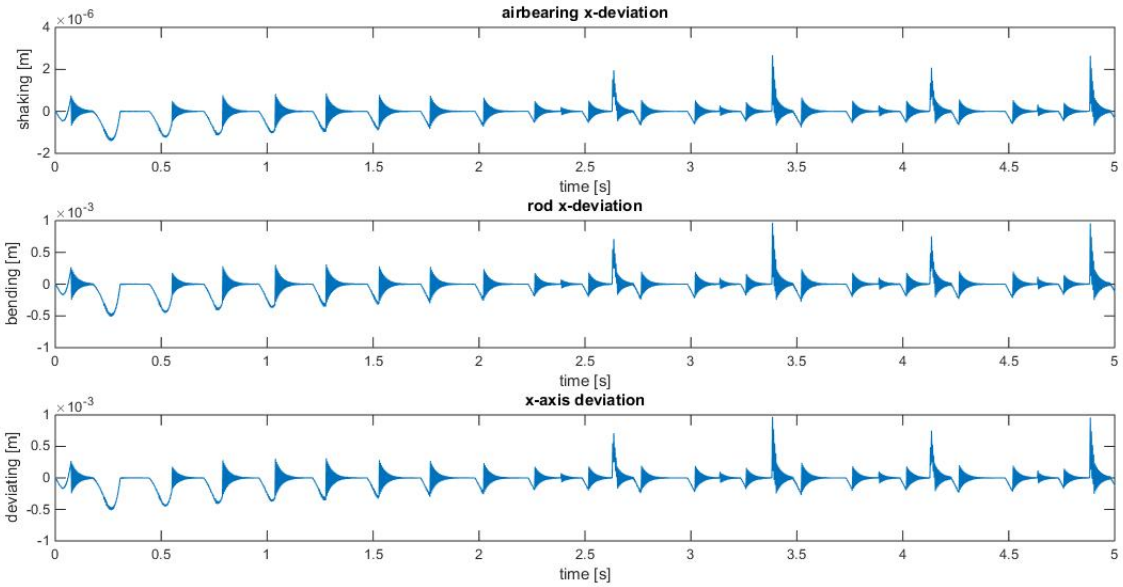

Figure 9.40: *Run 4 workaround deviations in x-direction with* 1 *% damping and multiplication factor of* t_m *in [m].*

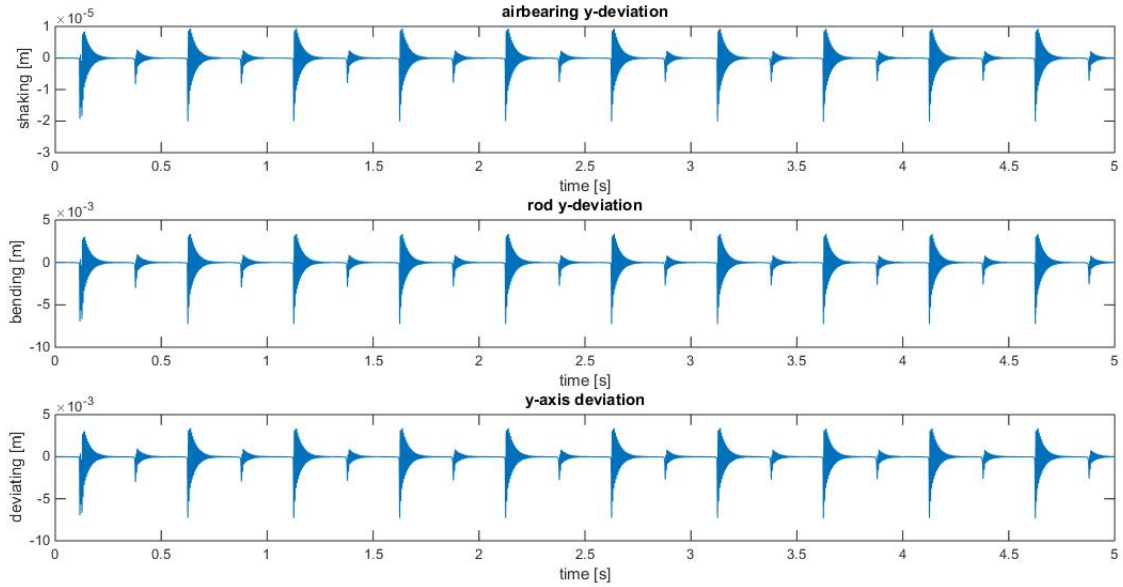

Figure 9.41: *Run 4 workaround deviations in y-direction with* 1 *% damping and multiplication factor of* t_m *in [m].*

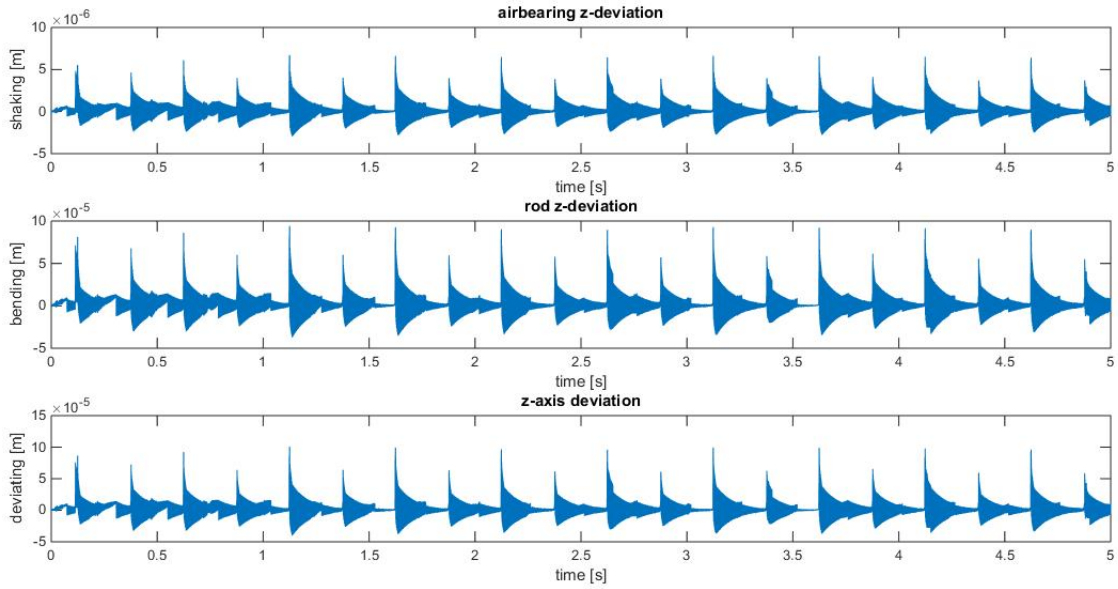

Figure 9.42: *Run 4 workaround deviations in z-direction with* 1 *% damping and multiplication factor of* t_m *in [m].*

10

Feedback design

Now that the equations of motion are established the question of how to improve the behavior can be answered. Some sort of correction needs to be designed to improve the unwanted behavior found. To accomplish this it might be needed to linearise the full system around some stable point. To avoid this linearization it is also possible to divide the model into different parts with their own functionality. This then leads to a linear problem to solve. For this problem a controller can be designed to make the correction. When the controller is known it can be used with the original system to test the influence of this controller. In this chapter the theory and methods used for calculating the controller will be described.

10.1. Linearization around an equilibrium

The system now derived can be generally stated as

$$\dot{\mathbf{q}} = f(\mathbf{q}, \mathbf{u}), \tag{10.1}$$

where $\mathbf{q}$ is the state vector, $\mathbf{u}$ the so called input vector. This is called a time-invariant system [34]. The goal of the linearization is defining a simpler system to analyze. The linearised system can be an accurate model for the problem. The first step is identifying the equilibrium points, or the constant vectors $\mathbf{q}^*$ and $\mathbf{u}^*$ satisfying [34]

$$f(\mathbf{q}^*, \mathbf{u}^*) = 0. \tag{10.2}$$

By expanding in Taylor series and defining the system in a neighborhood of the equilibrium point by $\mathbf{q} = \mathbf{q}^* + \mathbf{z}$ and $\mathbf{u} = \mathbf{u}^* + \mathbf{v}$ the linearised system of (10.1) is defined as [34]

$$\dot{\mathbf{z}} = A\mathbf{z} + B\mathbf{v}, \tag{10.3}$$

where

$$A = \frac{\partial f}{\partial \mathbf{q}}(\mathbf{q}^*, \mathbf{u}^*) \tag{10.4}$$

$$B = \frac{\partial f}{\partial \mathbf{u}}(\mathbf{q}^*, \mathbf{u}^*). \tag{10.5}$$

10.2. Controllers using LMI's

For the beginning it is best to assume that the full state is known. Then looking at the full system of equations of the polishing disc and the airbearing it can be seen that it is a linear system except for the moments and the response angular velocities $\dot{\theta}$ and $\dot{\psi}$ after the impact. Since it is physically not possible to adjust θ and ψ it might be a solution to use linear matrix inequalities (LMI's), where the state is given by $\mathbf{x} = (x_m, \dot{x}_m, x, \dot{x}, y_m, \dot{y}_m, y, \dot{y}, z_m, \dot{z}_m, z, \dot{z})^T$ and the disturbance is given by $\mathbf{w} = (F_x(\theta, \dot{\theta}, \psi, \dot{\psi}), F_y(\theta, \dot{\theta}, \psi, \dot{\psi}), F_z(\theta, \dot{\theta}, \psi, \dot{\psi}), F_p)^T$. The linear matrix inequalities form a set of which an optimal solution can be calculated by minimizing the H_∞ or H_2 norm. The choice of norm is dependent on what is most important to adjust in the original system. The advantage of using LMI's is that multiple objectives can be combined into one control problem [36].

These problems can be solved using the interior point technique in convex programming.

From the simulations it was already concluded that the two systems of the polishing disc and of the air-bearing must be calculated separately. This means that the norms and LMI's can perfectly be used, since **w** is then calculated beforehand and B_1 is the matrix to calculate different components. There exists a Matlab package Yalmip to optimize the H_2 and H_∞ norms.

In [45], [41] and [36] the use of linear matrix inequalities (LMI's) for minimization of norms is explained. In [45] explicit matrix inequalities are defined and solved with the Yalmip Matlab package [26]. In [36] the LMI toolbox is used. In this research the Yalmip optimization tools are used. As stated before, the system of interest is of the form

$$\begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} + B_1\mathbf{w}, \\ \mathbf{y} &= C\mathbf{x}, \\ \mathbf{z} &= C_1\mathbf{x} + D_1\mathbf{w}. \end{aligned} \tag{10.6}$$

The transfer matrix $T(s)$ is the input-output description of the system in the Laplace domain and is given by [45]

$$T(s) = C_1(sI - A)^{-1}B_1 + D_1, \tag{10.7}$$

where I is the identity matrix. Two types of feedback controllers are studied in [45]:

- Static state feedback: $\mathbf{u} = D_c\mathbf{x}$, which gives the closed loop system

$$\dot{\mathbf{x}} = (A + BD_c)\mathbf{x} + B_1\mathbf{w}, \tag{10.8}$$
$$\mathbf{z} = C_1\mathbf{x} + D_1\mathbf{w}. \tag{10.9}$$

- Dynamic feedback:

$$\dot{\mathbf{x}}_c = A_c\mathbf{x}_c + B_c\mathbf{y}, \tag{10.10}$$
$$\mathbf{u} = C_c\mathbf{x}_c + D_c\mathbf{y}. \tag{10.11}$$

  This gives the closed loop system

$$\dot{\mathbf{x}} = (A + BD_cC)\mathbf{x} + BC_c\mathbf{x}_c + B_1\mathbf{w}, \tag{10.12}$$
$$\dot{\mathbf{x}}_c = B_cC\mathbf{x} + A_c\mathbf{x}_c, \tag{10.13}$$
$$\mathbf{z} = C_1\mathbf{x} + D_1\mathbf{w}. \tag{10.14}$$

For the static state feedback it is assumed that the full state is known by measurements, i.e., $C = I$. The difference in behavior of a static and dynamic feedback is that in general the dynamics are needed if the state is not available and needs to be estimated to be used for control. Also a dynamic control calculated at time t does not only depend on the output measured at time t but also on previous measurements [34]. In the next subsections asymptotic stability, H_∞ norm and the H_2 norm are discussed with their LMI characterizations.

10.2.1. Asymptotic stability

To be able to use the LMI's and the norms to calculate a feedback controller the system without the controller must be asymptotically stable. There is a LMI definition for proving that a system is asymptotically stable. This definition is that a system (10.6) without the controller **u** and measurements **y** is asymptotically stable, or equivalent, A is an asymptotically stable matrix if and only if

$$\exists K = K^T \succ 0 : A^TK + KA \prec 0. \tag{10.15}$$

10.2.2. H-infinity norm

The H_∞ norm of a system gives a factor that indicates the height of the peak values in the worst-case situation of the system [45]. When taken from the transfer matrix T it is denoted as $\|T\|_{H_\infty}$. It describes the maximum gain in the principal direction [36]. Then using LMI's the matrices can be found to minimize this height. This

means that minimizing the H_∞ norm gives a more uniform system. The LMI definition for this norm is then [45]: Let $\gamma > 0$ be a given real number and A an asymptotically stable matrix. Then

$$\|T\|_{H_\infty} < \gamma \iff \exists K = K^T : \begin{pmatrix} A^T K + KA + C_1^T C_1 & KB_1 + C_1^T D_1 \\ B_1^T K + D_1^T C_1 & \gamma^2 I + D_1^T D_1 \end{pmatrix} \prec 0. \tag{10.16}$$

It can be shown that if A has to be asymptotically stable this is equivalent to the definition that [45]

$$\|T\|_{H_\infty} < \gamma \iff \exists K = K^T \succ 0 : \begin{pmatrix} A^T K + KA & KB_1 & C_1^T \\ B_1^T K & -\gamma^2 I & D_1^T \\ C_1 & D_1 & -I \end{pmatrix} \prec 0. \tag{10.17}$$

When the H_∞ norm is being minimized by finding a static state feedback $\mathbf{u} = D_c\mathbf{x}$ some adjustments must be made in order to get a proper LMI to solve. In [45] an exact description and derivation are given of how to define the LMI in this situation. Here it is stated that for a $\gamma > 0$ the following must hold to get a asymptotic stable system

$$\exists M, \exists Y = Y^T \succ 0 : \tag{10.18}$$

$$\begin{pmatrix} (AY + BM) + (AY + BM)^T & B_1 & YC_1^T \\ B_1^T & -\gamma^2 I & D_1^T \\ C_1 Y & D_1 & -I \end{pmatrix} \prec 0. \tag{10.19}$$

Here $Y = K^{-1}$ and $M = D_c K^{-1} = D_c Y$. The controller can then be found by calculating $D_c = MY^{-1}$.

For the dynamic controller it becomes a lot more complicated to get to a LMI. The following definition is then given to find a dynamic controller that produces an asymptotically stable system: for $\gamma > 0$

$$\exists \mathbf{v} : X(\mathbf{v}) \succ 0 \tag{10.20}$$

$$\begin{pmatrix} A(\mathbf{v})^T + A(\mathbf{v}) & B_1(\mathbf{v}) & C_1(\mathbf{v})^T \\ B_1(\mathbf{v})^T - \gamma^2 I & D_1(\mathbf{v})^T & \\ C_1(\mathbf{v}) & D_1(\mathbf{v}) & -I \end{pmatrix} \prec 0. \tag{10.21}$$

Here $\mathbf{v} = \{X, Y, K, L, M, N\}$ is a vector of the unknowns and the matrices are given by

$$\begin{aligned} A(\mathbf{v}) &= \begin{pmatrix} AY + BM & A + BNC \\ K & XA + LC \end{pmatrix}, \\ B_1(\mathbf{v}) &= \begin{pmatrix} B_1 \\ XB_1 \end{pmatrix}, \\ C_1(\mathbf{v}) &= \begin{pmatrix} C_1 Y & C_1 \end{pmatrix}, \\ X(\mathbf{v}) &= \begin{pmatrix} Y & I \\ I & X \end{pmatrix}. \end{aligned} \tag{10.22}$$

Here X and Y should be symmetric. Note: I as matrix here means the identity matrix. The final controller can then be calculated by first solving

$$VU^T = I - YX, \tag{10.23}$$

for V and U. This can be done with a LU decomposition where V is a lower triangular matrix and U^T an upper triangular matrix. Then the final controller is given by

$$\begin{pmatrix} A_c & B_c \\ C_c & D_c \end{pmatrix} = \begin{pmatrix} U & XB \\ 0 & I \end{pmatrix}^{-1} \begin{pmatrix} K - XAY & L \\ M & N \end{pmatrix} \begin{pmatrix} V^T & 0 \\ CY & I \end{pmatrix}^{-1}. \tag{10.24}$$

10.2.3. H-2 norm

The H_2-norm gives the overall energy of a system where it is specifically relating the input disturbance to the output response [36]. When taken from the transfer matrix T it is denoted as $\|T\|_{H_2}$. The H_2 norm can be seen as the output when applying a series of unit impulses [36]. Then using LMI's, the controller matrices

can be found to minimize the energy. This means that minimizing the H_2 norm gives a system in which the total energy in the system is minimized. The LMI definition for this norm is then [45]: Let $\gamma > 0$ be a given real number and A an asymptotically stable matrix. Then

$$\|T\|_{H_2} < \gamma \iff \exists K = K^T, \exists Z = Z^T : \begin{cases} \begin{pmatrix} A^T K + KA & KB_1 \\ B_1^T & -\gamma I \end{pmatrix} \prec 0, \\ \begin{pmatrix} K & C_1^T \\ C_1 & Z \end{pmatrix} \succ 0, \text{trace}(Z) < \gamma. \end{cases} \tag{10.25}$$

Then in order to make sure the system is asymptotically stable it can be shown that only one small adjustment has to be made to get [45]

$$\|T\|_{H_2} < \gamma \iff \exists K = K^T \succ 0, \exists Z = Z^T : \begin{cases} \begin{pmatrix} A^T K + KA & KB_1 \\ B_1^T & -\gamma I \end{pmatrix} \prec 0, \\ \begin{pmatrix} K & C_1^T \\ C_1 & Z \end{pmatrix} \succ 0, \text{trace}(Z) < \gamma. \end{cases} \tag{10.26}$$

In computations with the H_2 norm it is assumed that $D_1 = 0$. This means that the disturbance itself is not an output, but just the state which it affects. The same way as for the H_∞ norm an extended definition of LMI's and matrices can be given when searching for a controller. For the static state feedback controller this means again with $\gamma > 0$ that

$$\exists M, \exists Y = Y^T \succ 0 \exists Z = Z^T : \tag{10.27}$$

$$\begin{cases} \begin{pmatrix} (AY + BM) + (AY + BM)^T & B_1 \\ B_1^T & -\gamma I \end{pmatrix} \prec 0, \\ \begin{pmatrix} Y & YC_1^T \\ C_1 Y & Z \end{pmatrix} \succ 0, \quad \text{trace}(Z) < \gamma \end{cases} \tag{10.28}$$

Here $Y = K^{-1}$ and $M = D_c K^{-1} = D_c Y$. The controller can then be found by calculating $D_c = MY^{-1}$.

The dynamic controller is a more complicated description. The following definition is then given to find a dynamic controller that produces an asymptotically stable system: for $\gamma > 0$

$$\exists \mathbf{v} \exists Z = Z^T : X(\mathbf{v}) \succ 0 \tag{10.29}$$

$$\begin{cases} \begin{pmatrix} A(\mathbf{v})^T + A(\mathbf{v}) & B_1(\mathbf{v}) \\ B_1(\mathbf{v})^T - \gamma I & \end{pmatrix} \prec 0. \\ \begin{pmatrix} X(\mathbf{v}) & C_1(\mathbf{v})^T \\ C_1(\mathbf{v}) & Z \end{pmatrix} \succ 0, \quad \text{trace}(Z) < \gamma. \end{cases} \tag{10.30}$$

Here $\mathbf{v} = \{X, Y, K, L, M, N\}$ is a vector of the unknowns and the matrices are given by (10.22). The final controller can be calculated by first solving

$$VU^T = I - YX, \tag{10.31}$$

for V and U. This can be done with a LU decomposition where V is a lower triangular matrix and U^T an upper triangular matrix. This gives the final controller as

$$\begin{pmatrix} A_c & B_c \\ C_c & D_c \end{pmatrix} = \begin{pmatrix} U & XB \\ 0 & I \end{pmatrix}^{-1} \begin{pmatrix} K - XAY & L \\ M & N \end{pmatrix} \begin{pmatrix} V^T & 0 \\ CY & I \end{pmatrix}^{-1}. \tag{10.32}$$

For more details on the derivation the lecture notes of [45] can be used.

10.2.4. Yalmip package

Yalmip is a toolbox for modeling and optimization in Matlab [26]. It has a lot of optimization functions to save a lot of programming time . The three LMI descriptions from before can be solved using the Optimize function for which an objective to minimize is given and some additional options can be set.

10.3. System description

Now the final systems (8.45) and (8.46) can be written in state-space notation. Here the forces that can be calculated from the polishing disc model must first be multiplied with the correct angles to get the solution in the (x, y, z) coordinate frame. For notation use F_θ and F_ψ for the impact forces from the polishing disc model. Then

$$F_x = F_\psi \sin(\alpha), \tag{10.33}$$
$$F_y = F_\theta \sin(\beta), \tag{10.34}$$
$$F_z = F_p + |F_\theta| \cos(\beta) + |F_\psi| \cos(\alpha). \tag{10.35}$$

Here F_p acts as the normal force as reaction to the polishing force (now with positive sign instead of negative sign). The absolute value is taken since both components of z-forces of θ and ψ will be directed to the positive direction, but when they are in a negative x or y direction they can be negative.

10.3.1. System without damping

To get the state description of the system first rewrite (8.45) into

$$\begin{aligned}
\ddot{x}_m &= -\frac{k_t}{m_x} x_m + \frac{k_x}{m_x}(x - x_m), \\
\ddot{x} &= -\frac{k_x}{m_t}(x - x_m) - \frac{\sin(\alpha)}{m_t} F_\psi, \\
\ddot{y}_m &= -\frac{k_t}{m_x} y_m + \frac{k_x}{m_x}(y - y_m), \\
\ddot{y} &= -\frac{k_x}{m_t}(y - y_m) - \frac{\sin(\beta)}{m_t} F_\theta, \\
\ddot{z}_m &= -\frac{k_z}{m_z}(z_m - z) - \frac{k_A z_m}{m_z} - \frac{F_p}{m_z}, \\
\ddot{z} &= \frac{k_z}{m_t}(z_m - z) + \frac{F_p + |F_\theta| \cos(\beta) + |F_\psi| \cos(\alpha)}{m_t}.
\end{aligned} \tag{10.36}$$

Then using $\mathbf{x} = \left(x_m, \dot{x}_m, x, \dot{x}, y_m, \dot{y}_m, y, \dot{y}, z_m, \dot{z}_m, z, \dot{z}\right)^T$ again as before and $\mathbf{w} = (F_x, F_y, F_z, F_p)^T$ the final system becomes

$$\dot{\mathbf{x}} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{(k_t+k_x)}{m_x} & 0 & \frac{k_x}{m_x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{k_x}{m_t} & 0 & -\frac{k_x}{m_t} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{(k_t+k_x)}{m_x} & 0 & \frac{k_x}{m_x} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{k_x}{m_t} & 0 & -\frac{k_x}{m_t} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{(k_z+k_A)}{m_z} & 0 & \frac{k_z}{m_z} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k_z}{m_t} & 0 & -\frac{k_z}{m_t} & 0
\end{bmatrix} \mathbf{x}$$
$$+ \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & -\frac{1}{m_t} & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
-\frac{1}{m_t} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\frac{1}{m_z} \\
0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{m_t} & 0
\end{bmatrix} \mathbf{w}. \tag{10.37}$$

Here the matrices correspond with the system $\dot{\mathbf{x}} = A\mathbf{x} + B_1\mathbf{w}$. What can be seen in these matrices is that for the controller design none of the difficult constants are needed. With difficult constants are the contact spring rates and damping coefficients of the springs meant. The only downside of using this system would be that it is not realistic, since there is always damping in real life.

10.3.2. System with damping

The system with damping then is defined as

$$\begin{aligned}
\ddot{x}_m &= -\frac{k_t}{m_x}x_m + \frac{k_x}{m_x}(x - x_m) - \frac{c_A}{m_x}\dot{x}_m + \frac{c_D}{m_x}(\dot{x} - \dot{x}_m), \\
\ddot{x} &= -\frac{k_x}{m_t}(x - x_m) - \frac{c_D}{m_t}(\dot{x} - \dot{x}_m) - \frac{\sin(\alpha)}{m_t}F_\psi, \\
\ddot{y}_m &= -\frac{k_t}{m_x}y_m + \frac{k_x}{m_x}(y - y_m) - \frac{c_A}{m_x}\dot{y}_m + \frac{c_D}{m_x}\left(\dot{y} - \dot{y}_m\right), \\
\ddot{y} &= -\frac{k_x}{m_t}(y - y_m) - \frac{c_D}{m_t}\left(\dot{y} - \dot{y}_m\right) - \frac{\sin(\beta)}{m_t}F_\theta, \\
\ddot{z}_m &= -\frac{k_z}{m_z}(z_m - z) - \frac{k_A z_m}{m_z} - \frac{c_R}{m_z}(\dot{z}_m - \dot{z}) - \frac{F_p}{m_z}, \\
\ddot{z} &= \frac{k_z}{m_t}(z_m - z) + \frac{c_R}{m_t}(\dot{z}_m - \dot{z}) + \frac{F_p + |F_\theta|\cos(\beta) + |F_\psi|\cos(\alpha)}{m_t}.
\end{aligned} \tag{10.38}$$

With the same state and disturbance vector the state-space representation then becomes

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{(k_t+k_x)}{m_x} & -\frac{(c_A+c_D)}{m_x} & \frac{k_x}{m_x} & \frac{c_D}{m_x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{k_x}{m_t} & \frac{c_D}{m_t} & -\frac{k_x}{m_t} & -\frac{c_D}{m_t} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{(k_t+k_x)}{m_x} & -\frac{(c_A+c_D)}{m_x} & \frac{k_x}{m_x} & \frac{c_D}{m_x} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{k_x}{m_t} & \frac{c_D}{m_t} & -\frac{k_x}{m_t} & -\frac{c_D}{m_t} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{(k_z+k_A)}{m_z} & -\frac{c_R}{m_z} & \frac{k_z}{m_z} & \frac{c_R}{m_z} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k_z}{m_t} & \frac{c_R}{m_t} & -\frac{k_z}{m_t} & -\frac{c_R}{m_t} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{m_t} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m_t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{m_z} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m_t} & 0 \end{bmatrix} \mathbf{w}. \tag{10.39}$$

In this system it is obvious that the damping constants have an influence. So the value of the damping has an effect on the controller that will be designed.

10.4. Further system matrices

The other matrices involved, i.e. B, C, C_1, D_1, are dependent on design and practical considerations. If $\mathbf{u}$ is defined as forces influencing or the airbearing or the spindle in $x-$, y- or z- direction, then $\mathbf{u} = D_c\mathbf{x}$ for static state feedback means that the force is given as a linear function of displacements and/or velocities. A function of all displacements in the uneven equations leads to a B given by

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{10.40}$$

This will be used as first solution for correction. The second solution used is when the correction is given as a function of the derivations when the correction is given by, for example, springs to correct in each direction

and location. The B matrix is then given by

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{m_x} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m_t} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m_x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{m_t} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{m_z} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{m_t} \end{pmatrix}. \tag{10.41}$$

Since the disturbance **w** influences the system internally it is logically to assume $D_1 = 0$. Then C and C_1 correspond to what is known about the output. This means that C describes what part of the state can be used to formulate the controller. For the beginning it is best to assume that $C = I$, the identity matrix. Then C_1 describes what part of the system shows the relevant behavior. Since what is observed are the deviations in the $x-$, y- and z- directions and not the velocities it is assumed that C_1 is given by

$$C_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{10.42}$$

10.4.1. Norms of system without controller

To calculate the starting norms of the system the transfer function can be used, without any controller. This is given by

$$T(s) = C_1(sI - A)^{-1}B_1 + D_1. \tag{10.43}$$

D_1 is assumed zero and all other matrices given as before in this chapter, where A is used as the total system with damping. As explained in [45] computing the H_∞ norm of this transfer function, i.e. $\|T\|_{H_\infty} < \gamma$, is equivalent to calculating

$$\exists K = K^T : \begin{pmatrix} A^T K + KA + C_1^T C_1 & KB_1 + C_1^T D_1 \\ B_1^T K + D_1^T C_1 & -\gamma^2 I + D_1^T D_1 \end{pmatrix} \prec 0. \tag{10.44}$$

Computing this in Matlab gives for γ the result $\gamma = 0.0021182$. This value should be decreased by the use of a controller. The lower this gets the better the performance with the controller.

The H_2 norm has the same type of definition for $\|T\|_{H_2} < \gamma$ given by [45]

$$\exists K = K^T \succ 0 \exists Z = Z^T : \begin{pmatrix} KA + A^T K & KB_1 \\ B_1^T K & -\gamma I \end{pmatrix} \prec 0, \qquad \begin{pmatrix} K & C_1^T \\ C_1 & Z \end{pmatrix} \succ 0, \qquad \text{trace}(Z) < \gamma. \tag{10.45}$$

Computing this in Matlab gives for γ the result $\gamma = 0.0029477$.

10.5. Scaled system

Looking at the matrix entries it can be concluded that the entries in A can vary a lot. This can cause some difficulties in simulating when the goal is to minimize a norm. A workaround for this is to scale the system with diagonal matrices to scale the rows and columns to decrease the difference in magnitude . This can improve the performance of the minimization and after a transformation the result is applicable for the original system.

In Matlab some functions are present to define a scaled system description. This feature could ensure that the calculation of the controller is better and less scattered due to large variations in magnitudes. It is therefore important to take this possibility into account. For this system the Matlab function prescale is used.

A scaled system can be defined by two transformation matrices: T_L and $T_R = T_L^{-1}$. These matrices are both diagonal matrices and define the scaled system:

$$A^s = T_L A T_R, \tag{10.46}$$
$$B^s = T_L B, \tag{10.47}$$
$$C^s = C T_R, \tag{10.48}$$
$$B_1^s = T_L B_1, \tag{10.49}$$
$$C_1^s = C_1 T_R, \tag{10.50}$$
$$D_1 = 0. \tag{10.51}$$

The transformation matrices T_L and T_R for the first solution of B are given by

$$T_L = \text{diag}(256, 7.8125 \cdot 10^{-3}, 64, 6.2500 \cdot 10^{-2}, 256, 7.8125 \cdot 10^{-3}, 64, 6.2500 \cdot 10^{-2}, 256, 3.9063 \cdot 10^{-3}, 128, 7.8125 \cdot 10^{-3}), \tag{10.52}$$

$$T_R = \text{diag}(3.9063 \cdot 10^{-3}, 128, 1.5625 \cdot 10^{-2}, 16, 3.9063 \cdot 10^{-3}, 128, 1.5625 \cdot 10^{-2}, 16, 3.9063 \cdot 10^{-3}, 256, 7.8125 \cdot 10^{-3}, 128). \tag{10.53}$$

For these transformation matrices the norms calculated are comparable to the non-scaled system, i.e.,

$$\|T\|_{H_\infty} < \gamma = 0.0021182, \tag{10.54}$$
$$\|T\|_{H_2} < \gamma = 0.0029474. \tag{10.55}$$

The transformation matrices T_L and T_R for the second solution of B is given by

$$T_L = \text{diag}(4096, 0.125, 1024, 1, 4096, 0.125, 1024, 0.5, 8192, 0.125, 4096, 0.25), \tag{10.56}$$

$$T_R = \text{diag}(2.4414 \cdot 10^{-4}, 8, 9.7656 \cdot 10^{-4}, 1, 2.4414 \cdot 10^{-4}, 8, 9.76562 \cdot 10^{-4}, 2, 1.2207 \cdot 10^{-4}, 8, 2.4414 \cdot 10^{-4}, 4). \tag{10.57}$$

For these transformation matrices the norms calculated are comparable to the non-scaled system, i.e.,

$$\|T\|_{H_\infty} < \gamma = 0.0021182, \tag{10.58}$$
$$\|T\|_{H_2} < \gamma = 0.0029476. \tag{10.59}$$

The only difference with the first solution is the last decimal of the H_2-norm.

These transformation matrices are dependent on the values of the damping constants, the spring constants and system matrices of the airbearing model.

All the calculated controller matrices (for static and dynamic feedback) can be used as calculated for the corresponding system. When the scaled controller matrix is used for the original system it is necessary to transform it to the correct scale again. The use of the scaled system is then a different optimization solution. For the static state feedback D_c is then given by

$$D_c = D_c^s T_R^{-1}. \tag{10.60}$$

10.6. Adding an observer

Right now it is assumed that the full state is known, i.e. $C = I$. This might not be the case, but this depends on the measuring device applied. In this situation another method similar to a dynamic controller can be used with the static state feedback to correct the system. This combines a controller with a so-called observer. For this the definition of an observable system is needed. An observable system means that if $\mathbf{u}$ and $\mathbf{y}$ are known on some interval starting from zero, its initial state $\mathbf{x}_0$ can be determined [45]. Assume n is the dimension of A. To check if a system is observable the following statements are equivalent [45]:

- The pair (C, A) is observable;

- $\text{rank}\begin{pmatrix} sI-A \\ C \end{pmatrix} = n$ for all $s \in \mathbb{C}$;
- $\text{rank}\begin{pmatrix} \lambda I-A \\ C \end{pmatrix} = n$ for all λ eigenvalues of A;

When the system without disturbances $\mathbf{w}$, given by

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{10.61}$$
$$\mathbf{y} = C\mathbf{x}, \tag{10.62}$$

is observable, an observer can be defined of the form

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + B\mathbf{u} + C_c(\mathbf{y} - C\hat{\mathbf{x}}). \tag{10.63}$$

Using this estimate state for defining a state control means that $\mathbf{u} = D_c\hat{\mathbf{x}}$. The resulting system is then given by [45]

$$\dot{\mathbf{x}} = A\mathbf{x} + BD_c\hat{\mathbf{x}}, \tag{10.64}$$
$$\dot{\hat{\mathbf{x}}} = C_cC\mathbf{x} + (A + BD_C - C_cC)\hat{\mathbf{x}}, \tag{10.65}$$
$$\mathbf{y} = C\mathbf{x}. \tag{10.66}$$

The full system with disturbance $\mathbf{w}$ now becomes

$$\dot{\mathbf{x}} = A\mathbf{x} + BD_c\hat{\mathbf{x}} + B_1\mathbf{w}, \tag{10.67}$$
$$\dot{\hat{\mathbf{x}}} = C_cC\mathbf{x} + (A + BD_C - C_cC)\hat{\mathbf{x}}, \tag{10.68}$$
$$\mathbf{y} = C\mathbf{x} + D_1\mathbf{w}. \tag{10.69}$$

Then following the notation of [45] gives

$$\mathscr{A} = \begin{pmatrix} A & BD_c \\ C_cC & A + BD_c - C_cC \end{pmatrix}, \tag{10.70}$$
$$\mathscr{B}_1 = \begin{pmatrix} B_1 \\ 0 \end{pmatrix}, \tag{10.71}$$
$$\mathscr{C}_1 = \begin{pmatrix} C_1 & 0 \end{pmatrix}, \tag{10.72}$$
$$\mathscr{D}_1 = D_1. \tag{10.73}$$

Since the H_∞-norm has a more compatible objective for this problem this is the only norm discussed in this section. The H_∞-norm can be calculated with the following system of inequalities and simultaneously ensure asymptotic stability of $\mathbf{A}$: the H_∞-norm of the system is less than $\gamma > 0$ if and only if

$$\exists \mathscr{K} = \mathscr{K}^T > 0 : \begin{pmatrix} \mathscr{A}^T\mathscr{K} + \mathscr{K}\mathscr{A} & \mathscr{K}\mathscr{B}_1 & \mathscr{C}_1^T \\ \mathscr{B}_1^T\mathscr{K} & -\gamma^2 I & \mathscr{D}_1^T \\ \mathscr{C}_1 & \mathscr{D}_1 & -I \end{pmatrix} < 0. \tag{10.74}$$

Unfortunately these inequalities do not form a LMI (not linear). For this a transformation needs to be defined to be able to calculate the controller. This is a little tricky.

For future work an example for a more realistic situation is when only the displacements are measurable, i.e.,

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{10.75}$$

This means that $C = C_1$.

10.7. Simulation

To actually apply the controller calculated in this chapter, the adjusted system with its controller (static or dynamic) is used. First the angular model is used to derive the impulse forces, then the new system can be integrated using the controllers. For different choices of runs all types of controllers and norms are used. The resulting matrices are presented in appendix C. The calculation of these controller matrices is only dependent on the choice of damping. The results of these runs will be given and discussed in the next chapter.

It is expected that the H_∞-norm will be better suited for this problem than the H_2-norm simply because the objective of the H_∞-norm is to decrease peak values. The H_2-norm improves the overall time behavior and will not just focuss on these extremes that are in this problem so important.

11

Simulating with controllers

The four runs from section 9.4.5 are used to test the controllers. For the static state feedback both the H_∞-norm minimization and the H_2-norm minimization are applied with the scaled system and the original system. The calculated controller matrices can be found in appendix C.

11.1. Static state feedback first solution

For the first B matrix the H_∞-norm shows some interesting results for both the original controller and the scaled controller. As it turns out, the behavior of the controller does not change with a different damping. For this reason only the results of a damping of 0.1 % are presented here.

11.1.1. Controller comparison H infinity norm

The scaled controller must first be transformed back to the right scale before being applied. This is done following equation (10.60). Figures 11.1 and 11.2 show the control $\mathbf{u} = D_c\mathbf{x}$ for the non-scaled problem for run 1. An interesting result is that in the x-direction the total controller equals zero. The only explanation for this to happen is that the controller design is optimal for the x-direction. In the y-direction and the z-direction this is different. In the y-direction some delay can be recognized at the start which damps out very fast. In the z-direction for the airbearing a lot of oscillations can be seen and for the spindle a delay again.

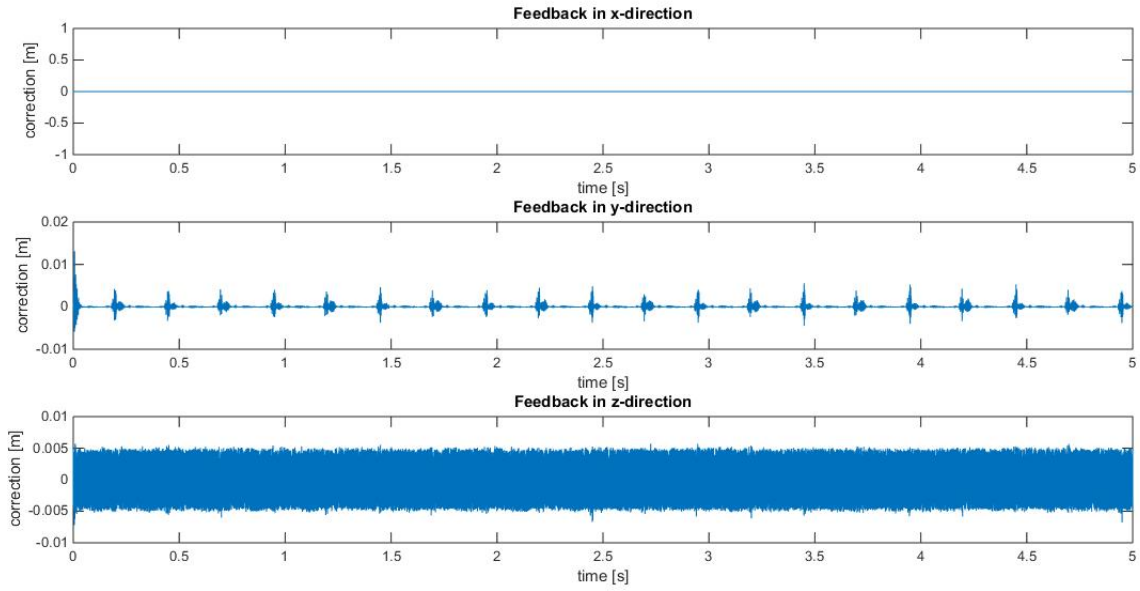

Figure 11.1: *Feedback part of the airbearing $D_cC\mathbf{x}$ of the system in [m/s].*

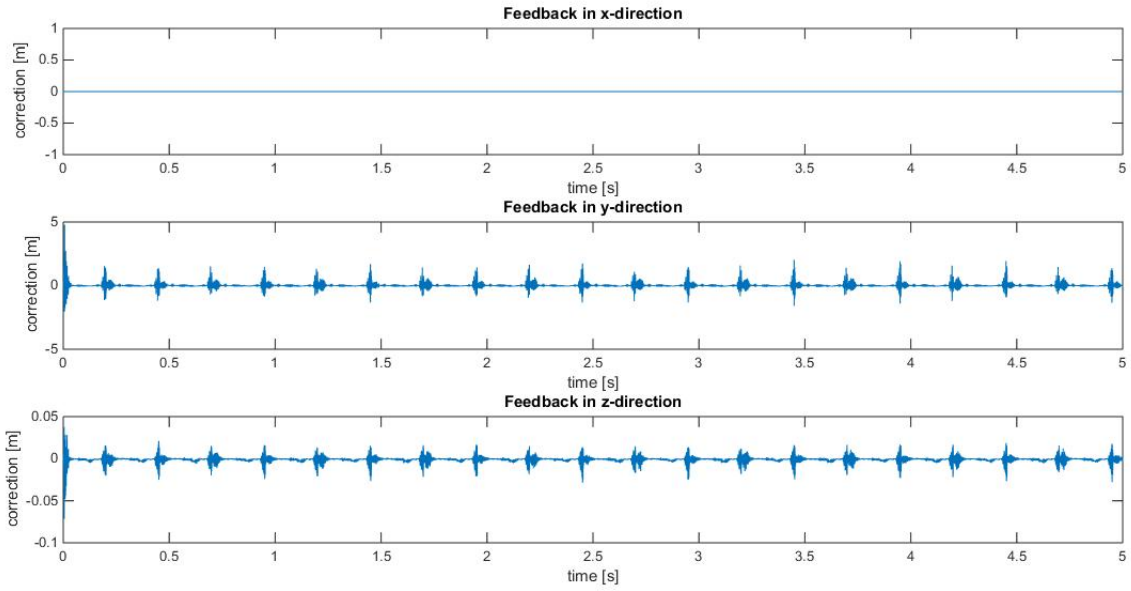

Figure 11.2: *Feedback part of the spindle $D_cC\mathbf{x}$ of the system in [m/s].*

Comparing these controllers $\mathbf{u}$ with the scaled solution $\mathbf{u} = D_c^s T_R^{-1}\mathbf{x}$ shows for the y- and z-direction something entirely different, see figures 11.3 and 11.4. The controller values are smaller and less wild. This can be caused by the fact that with scaling some of the high frequencies are not taken into account. This controller does look more applicable in real life. The x-direction remains at the same optimum value.

The H_∞-norm of these systems with controller is for the original system $\gamma = 3.4459 \cdot 10^{-6}$ and for the scaled system $5.0011 \cdot 10^{-6}$. In comparison with the value for the system without control (0.0021182 for scaled and non-scaled) this is already a lot better.

11.1.2. Original system H infinity norm

The final resulting deviations of the use of the H_∞ controllers show the quality of this method on these types of problems. In this section the results of the original static state controller are presented for run 1 and run 4. These show for two very different impact behavior values how this method works out.

Figures 11.5, 11.6 and 11.7 show the comparison of the total deviation without and with the static state controller. It is obvious to see that in the x-direction the controller works optimal. Note: even though the simulations show absolutely no deviation with a certain controller, this doesn't mean that in real life no deviation takes place. In real life other factors also influence the system and this only shows if the found controller corrects the deviations due to certain identified factors. In the y-direction the deviations are not only decreased, but also the frequency of oscillation is a lot lower. The same can be said about the z-direction.

The final comparison of run 4 is shown in figures 11.8, 11.9 and 11.10. These figures show the same extremely good result for the x-direction. In the y-direction the deviations are approximately 10 times smaller than without the controller and again with less oscillations. In the z-direction the entire movement into the material is gone and only a small deviation in the upward direction is possible. This shows that the general behavior of the controller decreases the deviations enormously.

11.1.3. Scaled system H infinity norm

In this section the results of the scaled static state controller are presented for run 1 and run 4. These show again for two very different impact behavior values how this method works out.

Figures 11.11, 11.12 and 11.13 show the total deviations of scaled run 1. There is no difference for the x-direction, where again the entire deviation is corrected. For the y-direction the same general behavior is observed, only with even less oscillations. In the z-direction not much difference is observed with the original controller results.

Comparing now for run 4 again the original controller performance in the last section with the scaled controller performance in figures 11.14, 11.15 and 11.16 show the same type of results as run 1 does. The x-

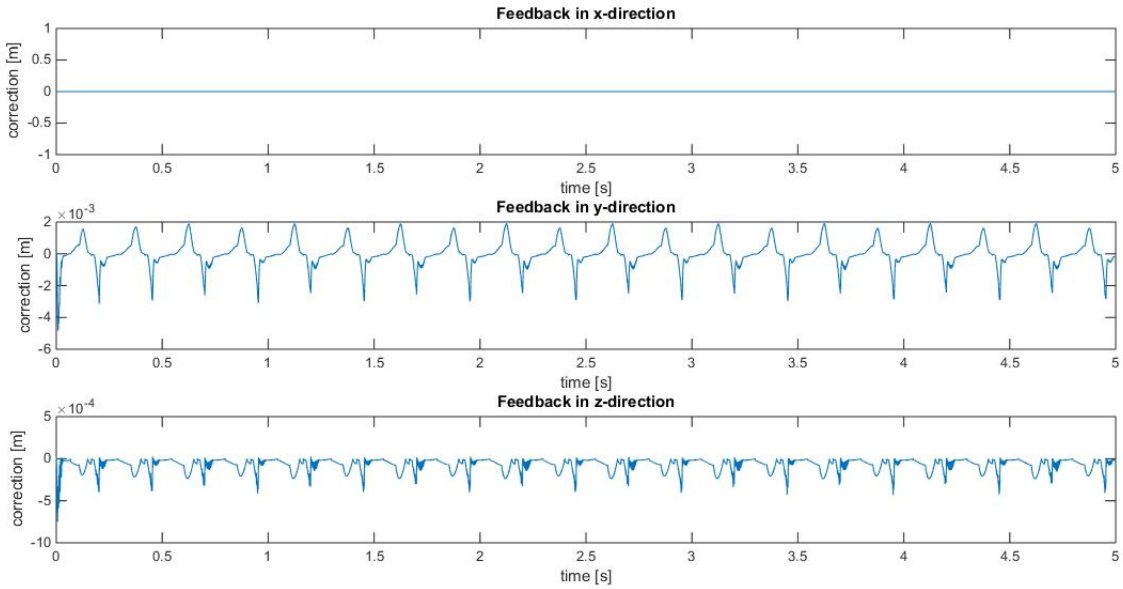

Figure 11.3: *Scaled feedback part of the airbearing $D_c C\mathbf{x}$ of the system in [m/s].*

Table 11.1: Maximum absolute deviation of different solutions of run 1

	x_{max} [m]	y_{max} [m]	z_{max} [m]
Original	$7.8095 \cdot 10^{-4}$	0.0177	$1.9855 \cdot 10^{-4}$
H_∞-norm	0	$7.5486 \cdot 10^{-4}$	$7.1232 \cdot 10^{-6}$
H_∞-norm of scaled system	0	$4.5467 \cdot 10^{-4}$	$8.0705 \cdot 10^{-6}$

direction remains the same. The y- and z-direction are greatly improved. In the y-direction the oscillations are even more reduced than with the original controller.

To see in numbers the improvements made with the controllers calculated from the H_∞-norms, table 11.1 and table 11.3 show the maximum absolute amplitude of the different solutions for run 1 and 4 respectively. Tables 11.2 and 11.4 show the percentage of improvement compared to the situation without a controller. These numbers do not show any trend or obvious relation between the scaled and non-scaled situation or the different runs. It can be said that the behavior of these controllers is very good.

11.2. Static state feedback second solution

For the second solution or the second type of adjustment B-matrix the calculated D_c-matrix shows that the resulting equation BD_c means the physical addition of springs and dampers in the system. This type of controller is already more realistic for testing the H_∞ norm. The same simulations have been done for run 1 and run 4 and the results are presented the same way as in the last section.

11.2.1. Controller comparison H infinity norm

Again a comparison is made between the different controller calculations of the second B-matrix solution. Figures 11.17 and 11.18 show the feedback behavior of the airbearing and the spindle. With this entirely different type of solution a very different scale of values is calculated. The values are now a lot higher and in

Table 11.2: Percentage of improvement of maximum absolute deviation of different solutions of run 1

	x_{max} improvement [%]	y_{max} improvement [%]	z_{max} improvement [%]
H_∞-norm	100	95.73	96.41
H_∞-norm of scaled system	100	97.43	95.93

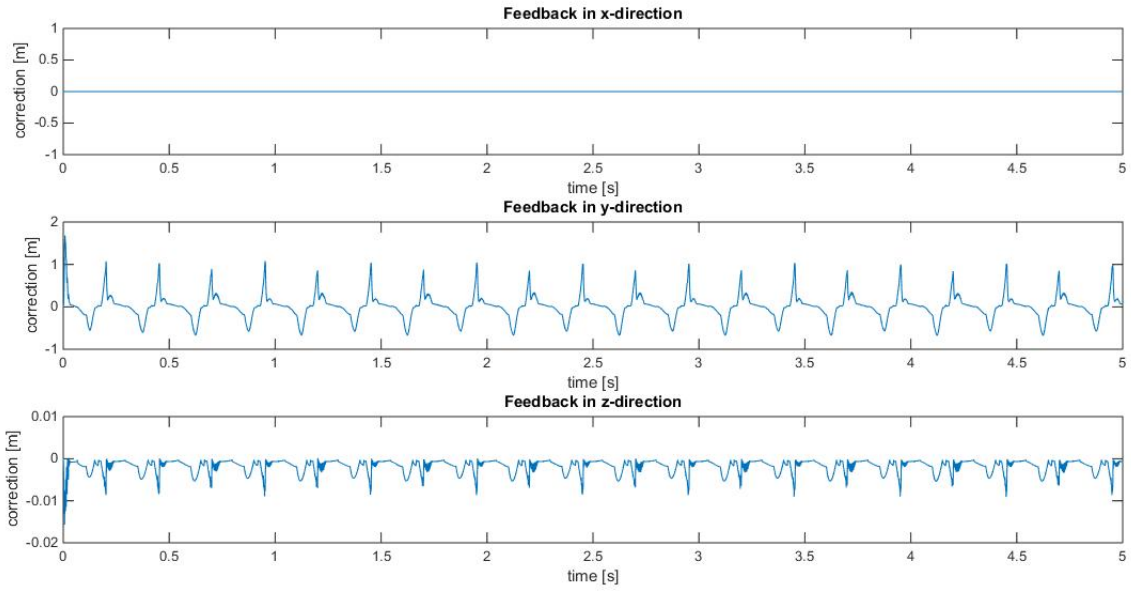

Figure 11.4: *Scaled feedback part of the spindle $D_c C\mathbf{x}$ of the system in [m/s].*

Table 11.3: Maximum absolute deviation of different solutions of run 4

	x_{max} [m]	y_{max} [m]	z_{max} [m]
Original	0.0011	0.0094	$1.5049 \cdot 10^{-4}$
H_∞-norm	0	$7.9841 \cdot 10^{-4}$	$1.2877 \cdot 10^{-5}$
H_∞-norm of scaled system	0	$4.9490 \cdot 10^{-4}$	$1.2861 \cdot 10^{-5}$

Newton. Only in the x-direction the structure of the controller corrects all deviations.

Comparing these forces again with the scaled situation shows a very different result. Figures 11.19 and 11.20 show that now the correcting forces have become in fact a lot smaller. As a realistic solution, this would be easier to implement in real life. This shows that the use of different calculation methods for controllers can also benefit the actual use of the solution. The feedback values in the y- and in the z-direction now resemble more to the behavior of a spring and show more oscillations in the controller.

The values of the H_∞ norm are again lower than of the system without control, but higher than with the first solution. For this solution the normal result is $\gamma = 6.5060 \cdot 10^{-5}$. The result for the scaled system is $\gamma = 5.3719 \cdot 10^{-5}$.

11.2.2. Original system

The comparison without and with controller is again made in this section for the original controller (without scaling). For run 1 this gives figures 11.21, 11.22 and 11.23. This shows that in the x-direction the same happens as in the other B-matrix situation. For the y- and z-direction the behavior of the deviations is the same with or without controller, but the magnitude is a lot smaller and the oscillations are not that wild anymore. For the y-direction the magnitude is even decreased with two digits, i.e., from 10^{-3} scale to 10^{-5} scale. This seems one of the biggest improvements until now. The behavior of the results is more or less the same as in the other B-matrix situation.

For run 4 the same figures have been made for this solution, see figures 11.24, 11.25 and 11.26. For the x-

Table 11.4: Percentage of improvement of maximum absolute deviation of different solutions of run 4

	x_{max} improvement [%]	y_{max} improvement [%]	z_{max} improvement [%]
H_∞-norm	100	91.50	91.44
H_∞-norm of scaled system	100	94.73	91.45

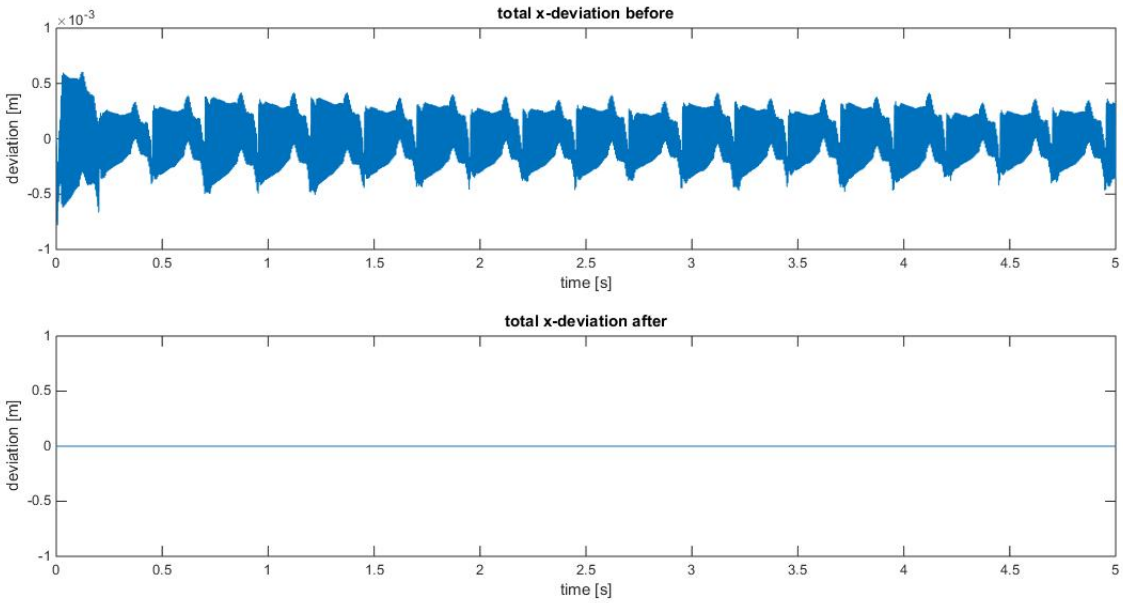

Figure 11.5: *Run 1 comparison of total deviation in x-direction before and after controller in [m].*

direction the same full correction is made. For the y-direction the deviations are a lot reduced in magnitude, from 10^{-3} to 10^{-5} scale. This also happened for this controller for run 1. If improvements in the x- and y- directions are most important than the non-scaled controller with the B-matrix of the second solution would be a very good solution. For the z-direction not that much improvement is made in magnitude. The oscillating behavior is a lot less than in the situation without controller.

11.2.3. Scaled system

In this section the scaled system results of the H_∞-norm are compared to the situation without a controller. For run 1 the results are presented in figures 11.27, 11.28 and 11.29. The x-direction shows again that the controller is optimal for this direction. The y-direction shows again a similar shape but with a smaller magnitude and less oscillations. Compared to the non-scaled situation it actually shows that the scaled matrix has a worse result. Even though the oscillations are less in this situation, the magnitude of the deviations is almost 10 times bigger than for the non-scaled controller. The z-direction is very different from the results before for this direction. There are still a lot of oscillations, but smaller in magnitude than in the non-scaled situation.

For the scaled controller in run 4 the comparison can be found in figures 11.30, 11.31 and 11.32. It is not surprising to see that in the x-direction the controller corrects all deviations again. For the y-direction a great improvement is made in the magnitude and oscillating behavior. The only problem is the few very high peaks in the controller solution. Because of this value the performance based on the maximum absolute amplitude will be a lot higher than in the non-scaled situation. The amount of oscillations is less than in the non-scaled situation, but the magnitude of the deviations is bigger. The z-direction shows the same type of result as in run 1 for this controller: a lot of oscillations, but a smaller magnitude than the non-scaled situation.

To put some of these results into numbers again, tables 11.5, 11.6, 11.7 and 11.8 show the absolute amplitude improvement by the different type of controllers. Looking at tables 11.6 and 11.8 shows again no real relation between different models used or controllers. It does confirm that even with an entirely different solution the performance of the controller is still very good. It just depends on the choice of correction how well a certain adjustment behaves. For run 4 for example it is obvious that the z-direction correction is less effective than for the other solutions.

11.3. Final remarks

The H_∞ norm shows generally very good results. Tables 11.2, 11.4, 11.6 and 11.8 show very good performances for both types of solutions for both runs. It shows that for different types of corrections or possible solutions the H_∞-norm can be used to find the correct parameters corresponding to this solution. A remark-

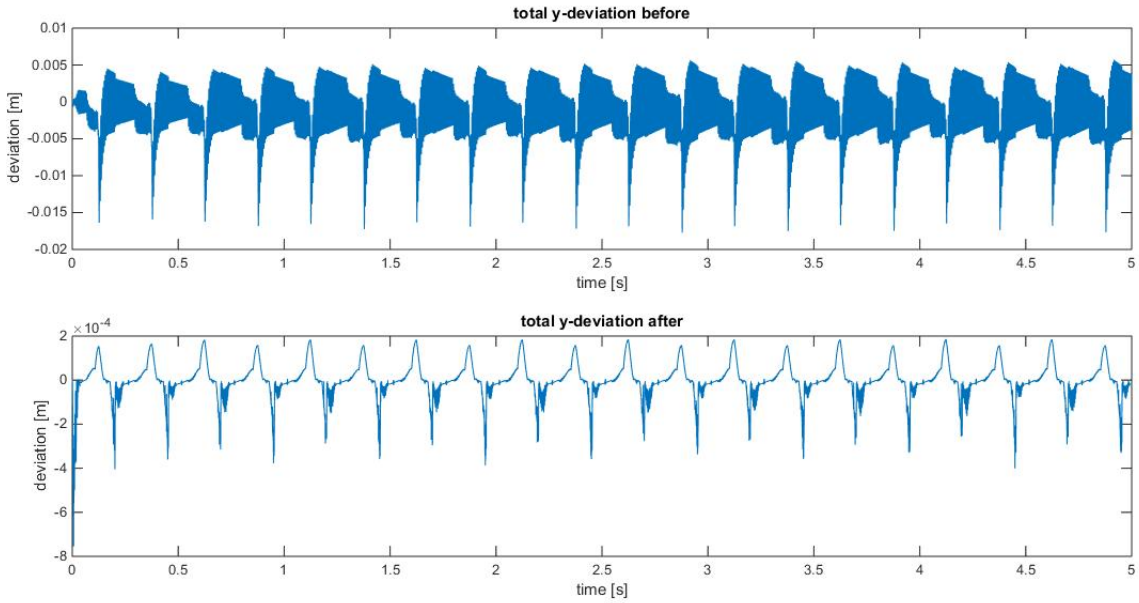

Figure 11.6: *Run 1 comparison of total deviation in y-direction before and after controller in [m].*

Table 11.5: Maximum absolute deviation of different solutions of run 1

	x_{max} [m]	y_{max} [m]	z_{max} [m]
Original	$7.8095 \cdot 10^{-4}$	0.0177	$1.9855 \cdot 10^{-4}$
H_∞-norm	0	$7.4643 \cdot 10^{-5}$	$2.4510 \cdot 10^{-5}$
H_∞-norm of scaled system	0	$6.8915 \cdot 10^{-4}$	$9.7659 \cdot 10^{-6}$

able result for the x-direction is found, which implies that the optimization of the H_∞ norm works from the first variable to the last. Since the other directions also show very good results this is not a problem.

The performance of the designed H_2-norm could not be tested for the two test runs, since these runs take too long computation time. It would be interesting to see if a norm with another objective gives a different performance for the controllers. The H_2-norm would optimize the general performance of the system, which would probably have led to a worse result, since the specific problem here are the peak values. The peak values are the objective of the H_∞ norm.

However, using the workaround, this model still gives useful information. For example the methods used for finding a controller proved to be effective. When the exact data of the model is used as input, this will also lead to a situation for which these methods can be used. The original choice for these type of controllers is based on using an optimization method rather than an analytic solution. The computational load for one run with the controller is unfortunately very large. This means that testing a possible adjustment or solution might take a while. The H_2 norm takes specifically more time than the H_∞ norm to get to a result. The final combination of models and the solution methods for controllers gives a good testing model for possible physical solutions. The assumption for the static state controller is that the controller is in some way a function of the state. As discussed before this could mean for instance adding a spring of a certain spring constant and dampers with a certain damping coefficient as solution to the end-effector on appropriate places.

The H_∞-norm has been tested on two different situations. The results show a very good performance

Table 11.6: Percentage of improvement of maximum absolute deviation of different solutions of run 1

	x_{max} improvement [%]	y_{max} improvement [%]	z_{max} improvement [%]
H_∞-norm	100	99.58	87.66
H_∞-norm of scaled system	100	96.11	95.07

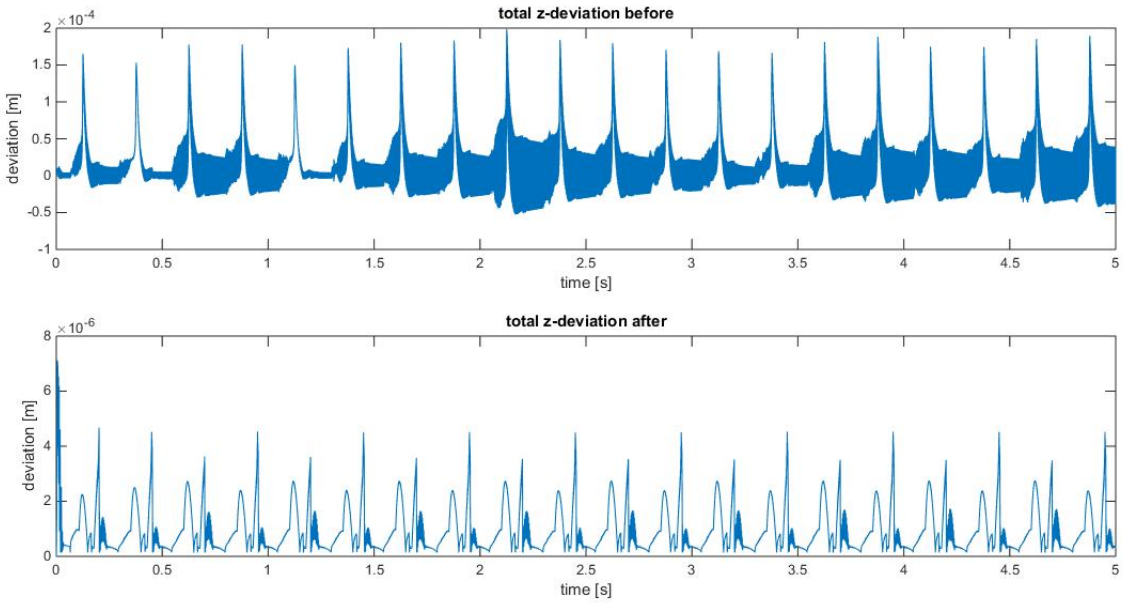

Figure 11.7: *Run 1 comparison of total deviation in z-direction before and after controller in [m].*

Table 11.7: Maximum absolute deviation of different solutions of run 4

	x_{max} [m]	y_{max} [m]	z_{max} [m]
Original	0.0011	0.0094	$1.5049 \cdot 10^{-4}$
H_∞-norm	0	$8.2640 \cdot 10^{-5}$	$4.6829 \cdot 10^{-5}$
H_∞-norm of scaled system	0	$8.0809 \cdot 10^{-4}$	$1.9925 \cdot 10^{-5}$

for these situations. Generally speaking it does not make that much of a difference if the scaled or non-scaled model is used. In most situations the scaled model gives a smoother result with less oscillations, but in magnitude it cannot be said beforehand if the results will be better or worse. The H_2-norm is very difficult to test. As it turns out it is very heavy to compute the results. The behavior of the controller calculated from the H_∞-norm is luckily that good that it can be concluded that it is possible to find a solution to the problem this way.

The choice for the exact type of controller is also depended on what you can adjust. For example the matrix B in the system describes in which equations it is possible to adjust something. The choice for $\mathbf{u}$ is also determined by the physical possibilities. If some solution is suggested its influence on the original system of equations must be described and added to the system this way. The only difficulty is that if $\mathbf{u}$ is not a linear function of $\mathbf{x}$ or $\mathbf{w}$ variables, the method for calculating the controller matrix D_c becomes more difficult. The dynamic controller discussed in this thesis never gave a solid result for controller matrices. Due to some calculations needed (near) singularity of the matrices exists. This does not lead to good behavior. The static state controller is therefore the only controller tested in this thesis. This is a much simpler controller, but gives already good results.

Table 11.8: Percentage of improvement of maximum absolute deviation of different solutions of run 4

	x_{max} improvement [%]	y_{max} improvement [%]	z_{max} improvement [%]
H_∞-norm	100	99.12	68.88
H_∞-norm of scaled system	100	91.40	86.76

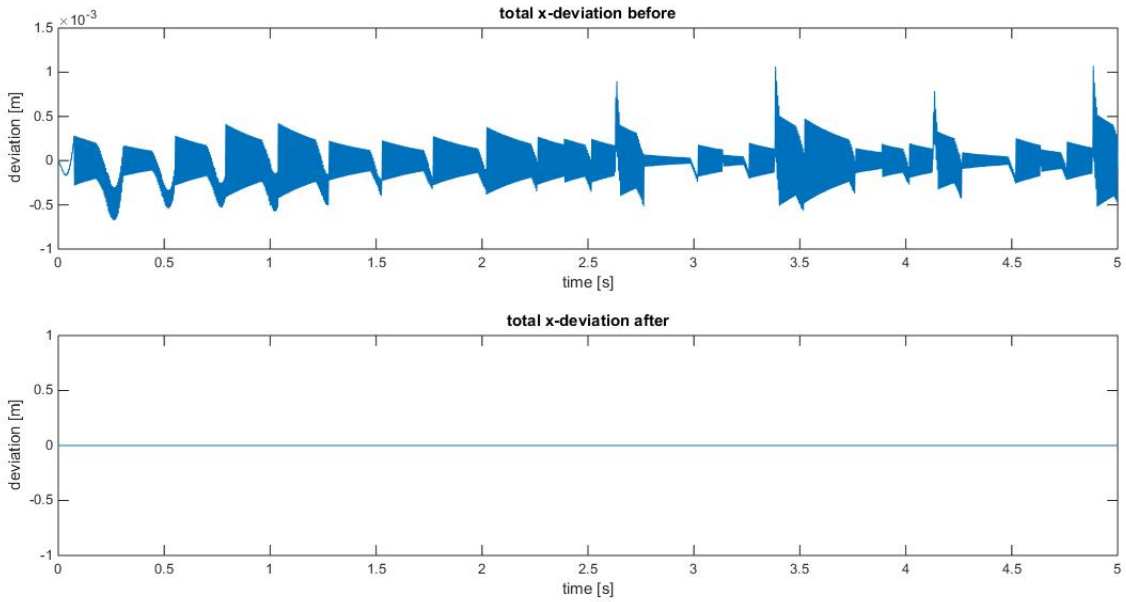

Figure 11.8: *Run 4 comparison of total deviation in x-direction before and after controller in [m].*

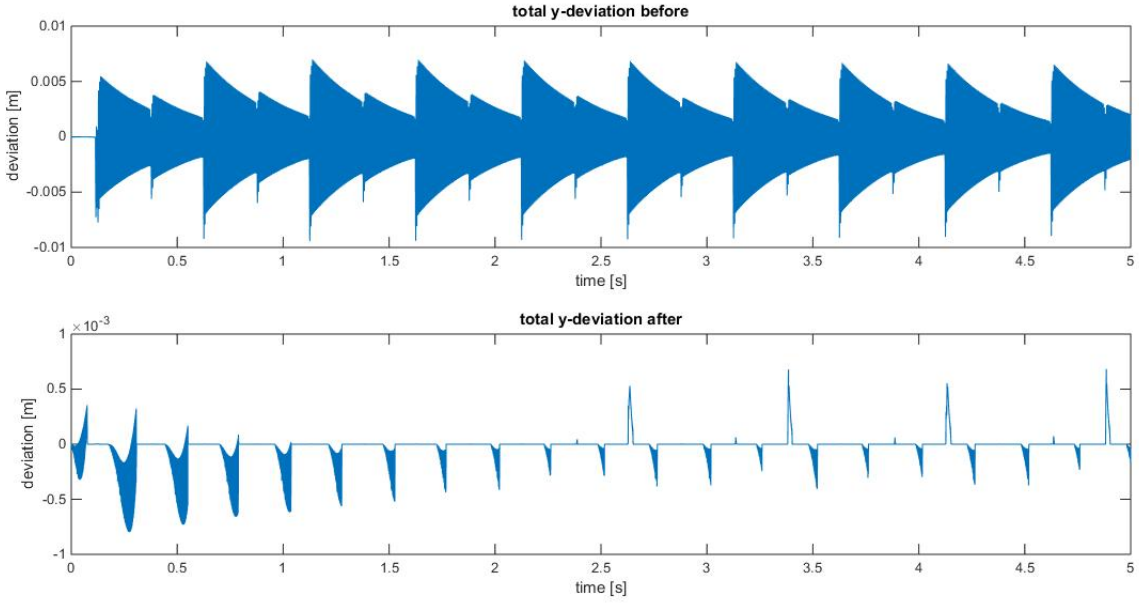

Figure 11.9: *Run 4 comparison of total deviation in y-direction before and after controller in [m].*

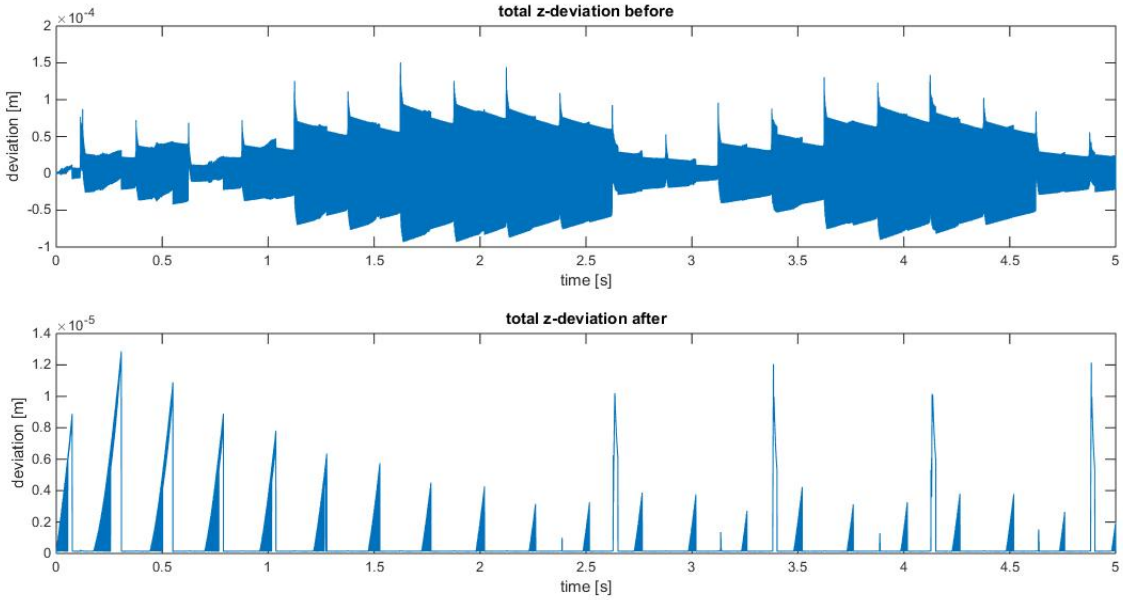

Figure 11.10: *Run 4 comparison of total deviation in z-direction before and after controller in [m].*

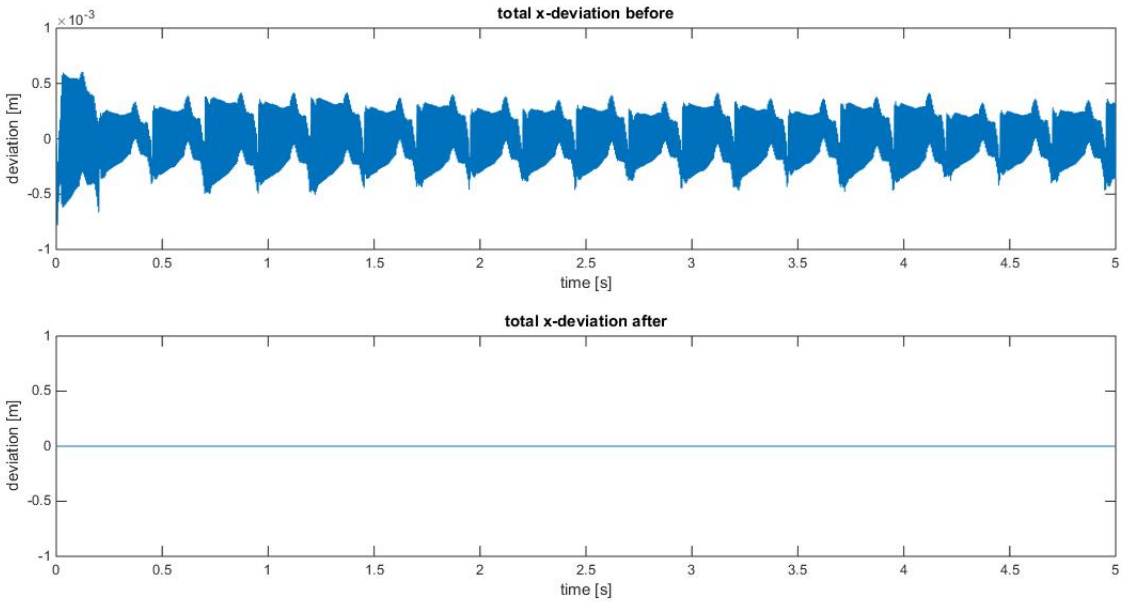

Figure 11.11: *Run 1 comparison of total deviation in x-direction before and after scaled controller in [m].*

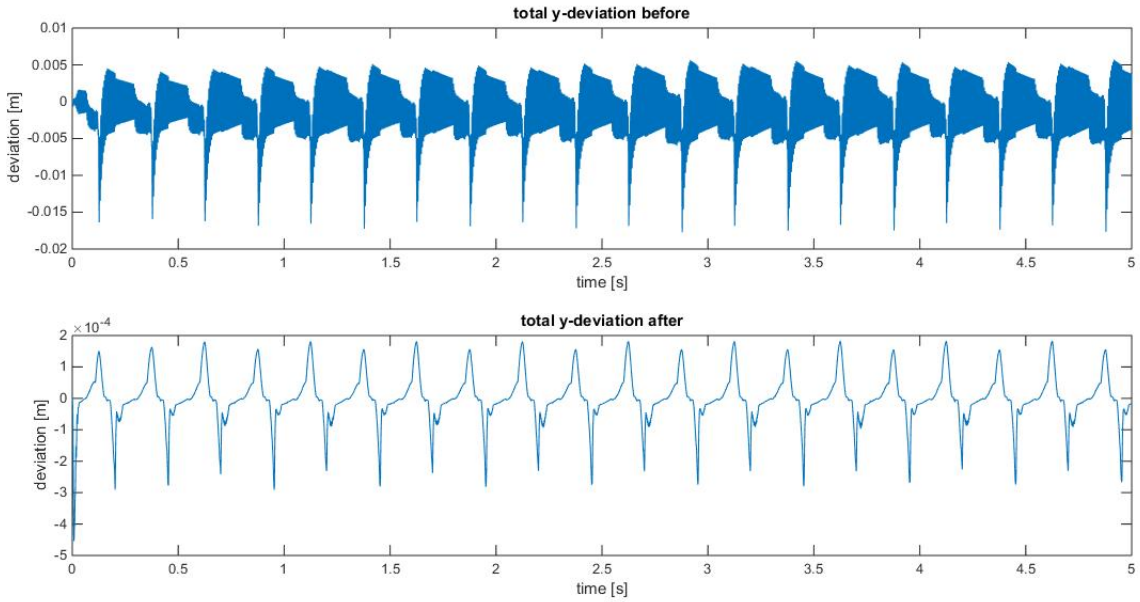

Figure 11.12: *Run 1 comparison of total deviation in y-direction before and after scaled controller in [m].*

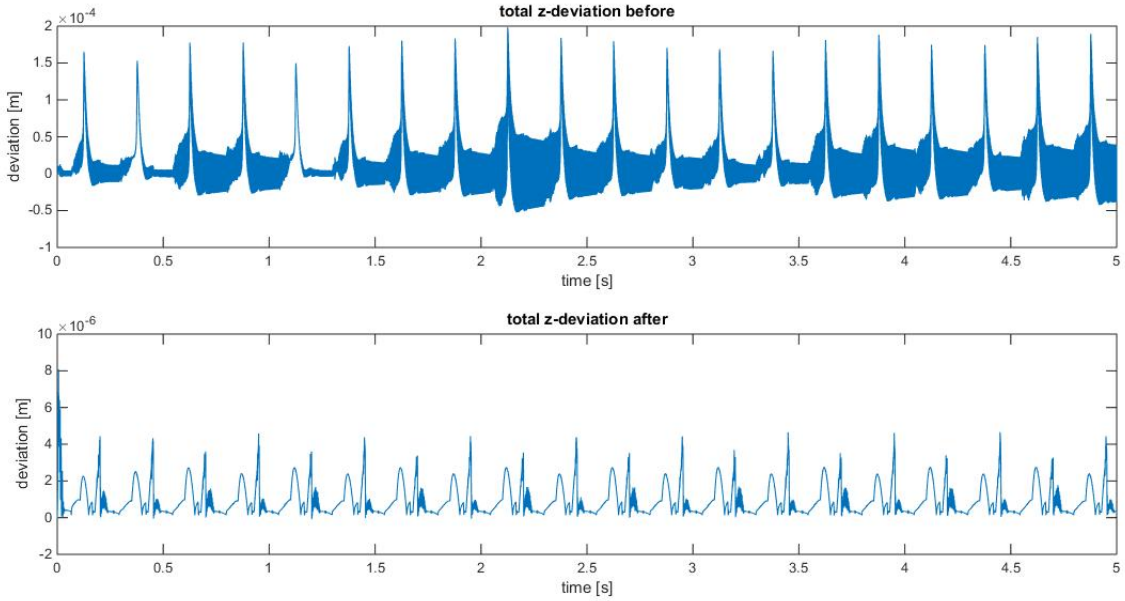

Figure 11.13: *Run 1 comparison of total deviation in z-direction before and after scaled controller in [m].*

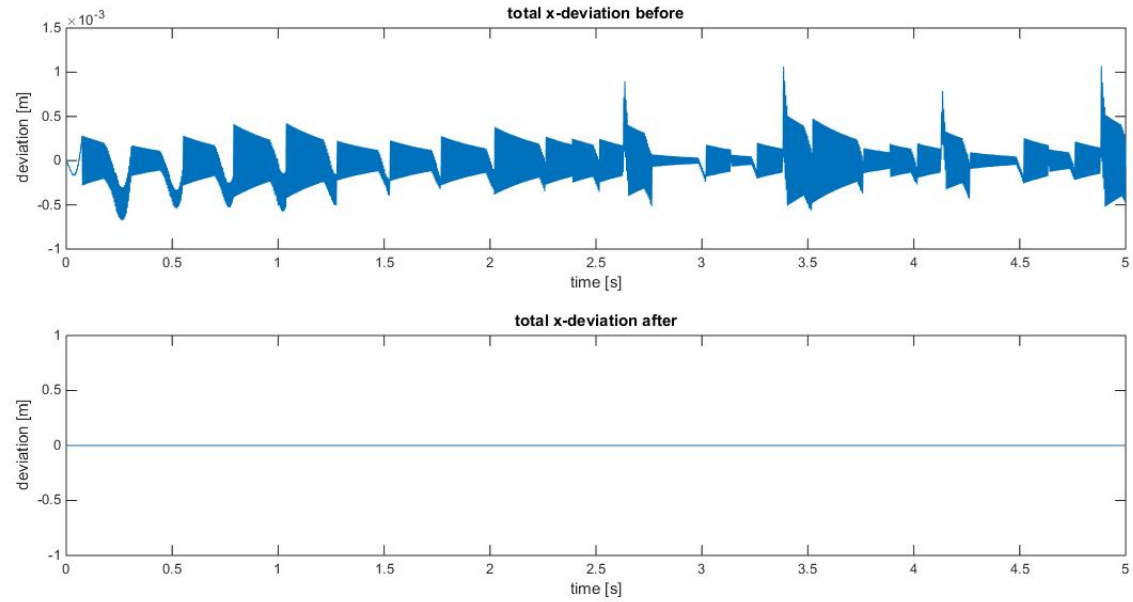

Figure 11.14: *Run 4 comparison of total deviation in x-direction before and after scaled controller in [m].*

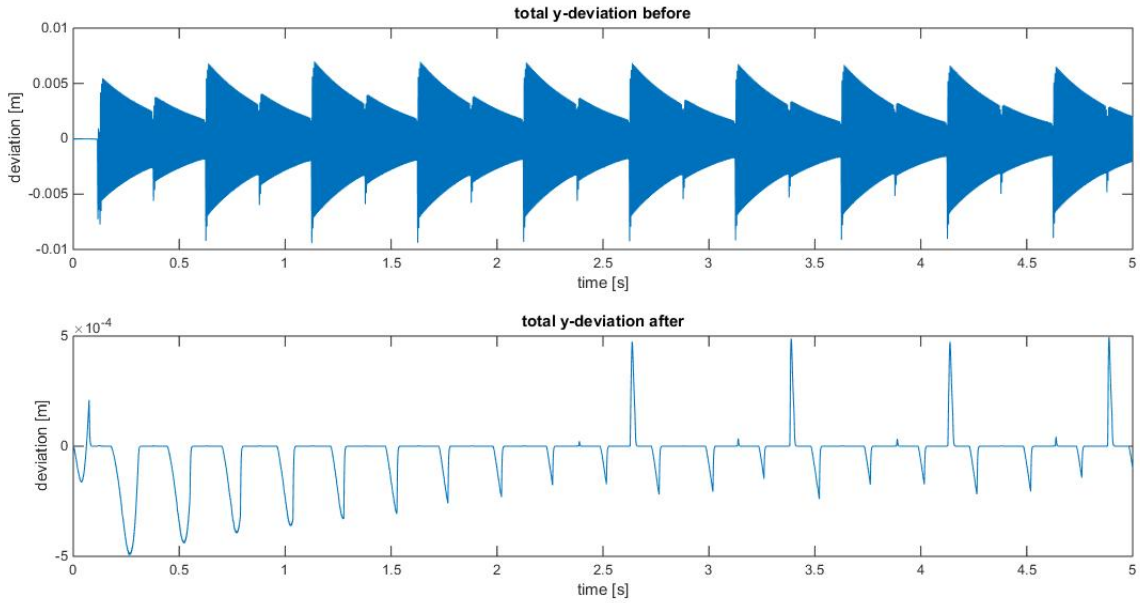

Figure 11.15: *Run 4 comparison of total deviation in y-direction before and after scaled controller in [m].*

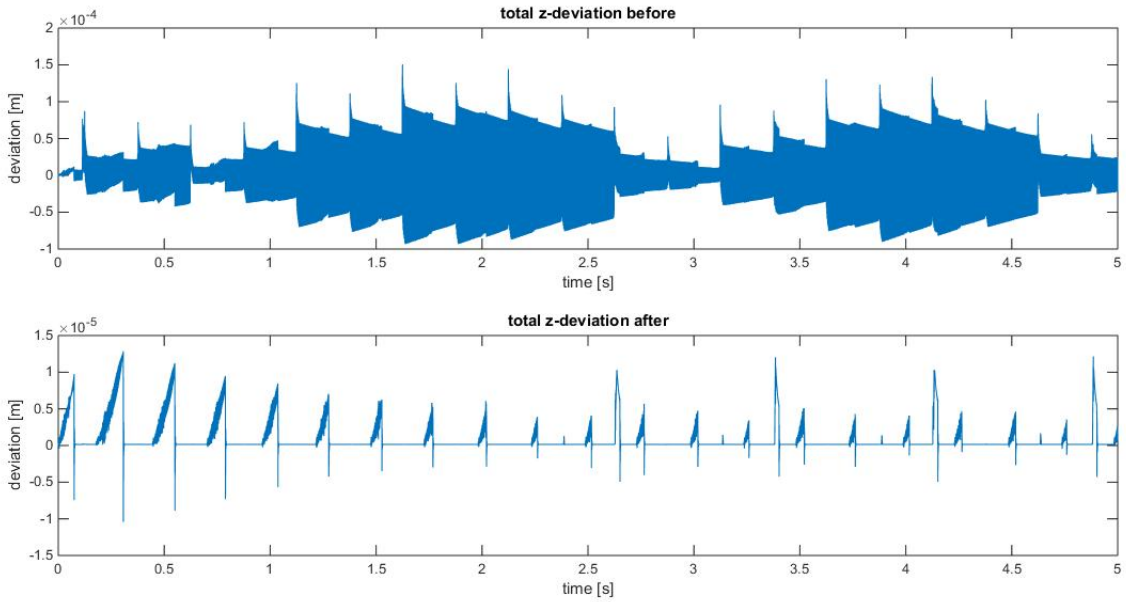

Figure 11.16: *Run 4 comparison of total deviation in z-direction before and after scaled controller in [m].*

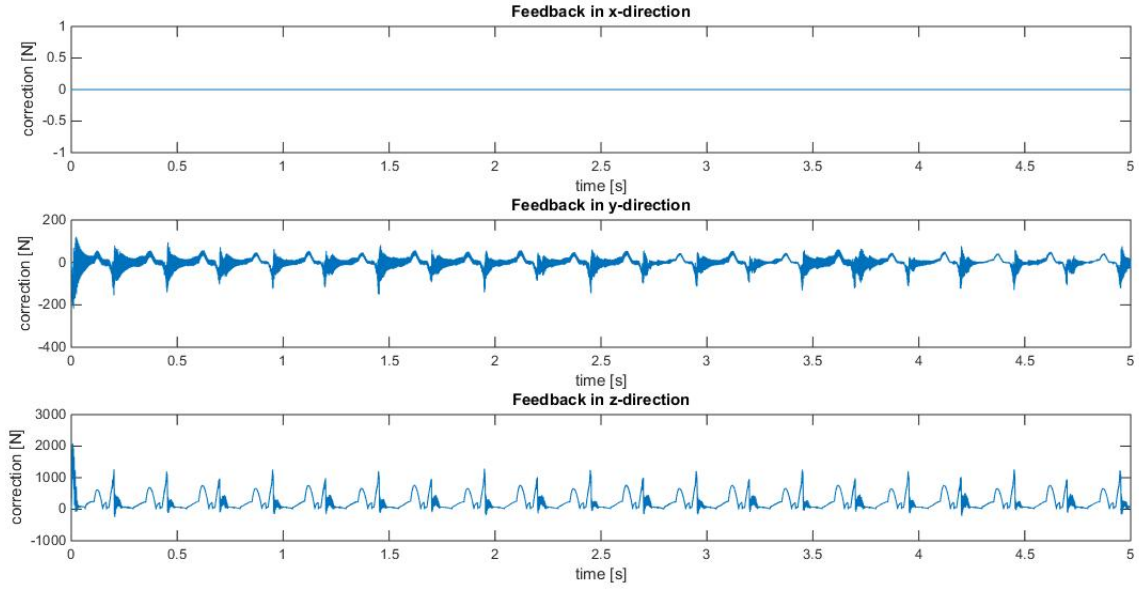

Figure 11.17: *Feedback part of the airbearing $D_c C\mathbf{x}$ of the system in [N].*

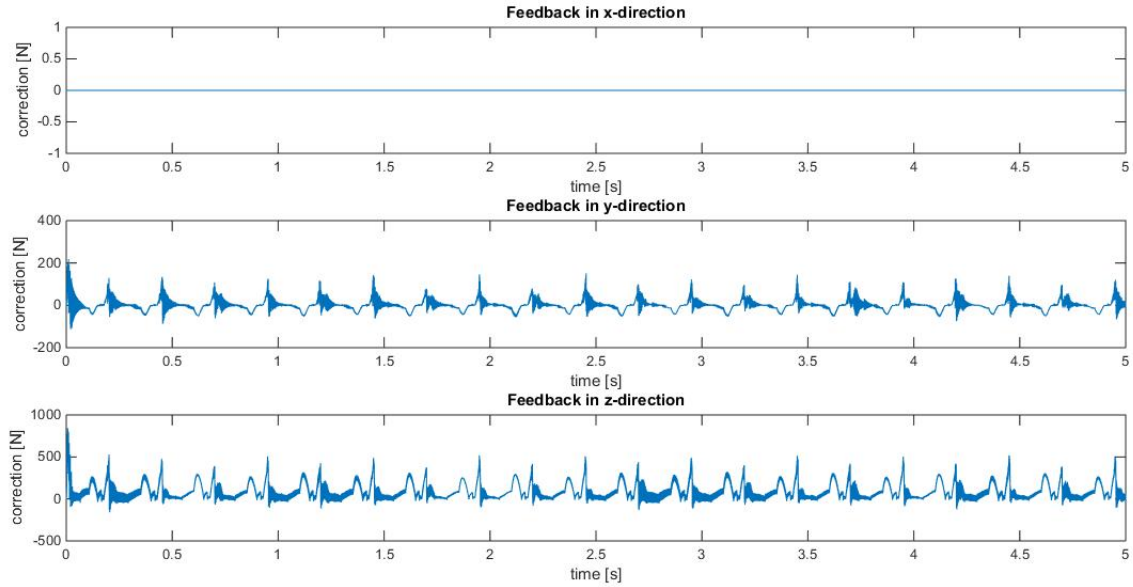

Figure 11.18: *Feedback part of the spindle $D_c C\mathbf{x}$ of the system in [N].*

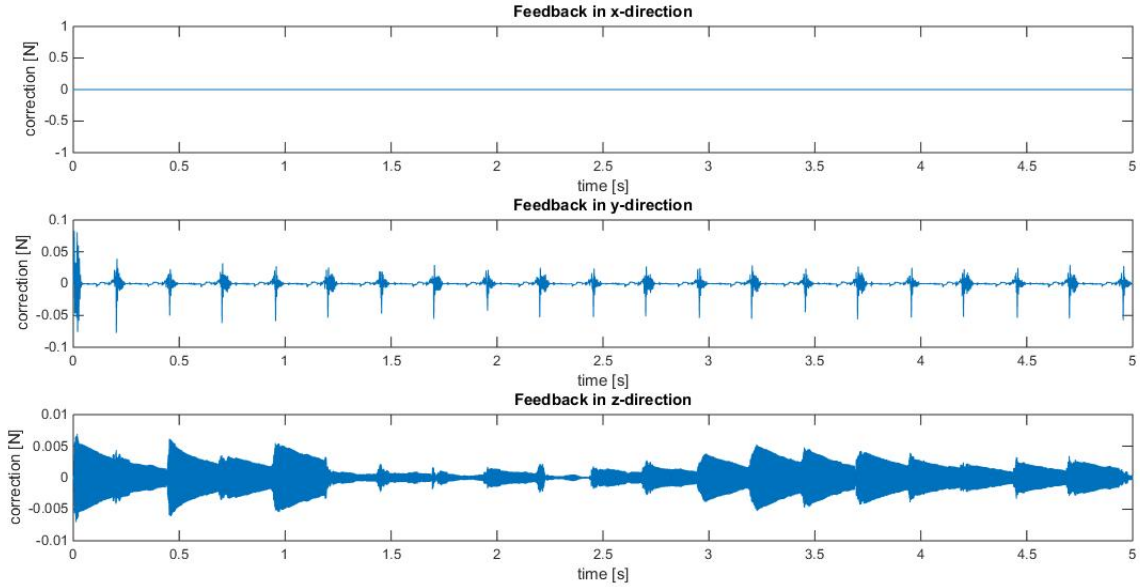

Figure 11.19: *Scaled feedback part of the airbearing $D_c C\mathbf{x}$ of the system in [N].*

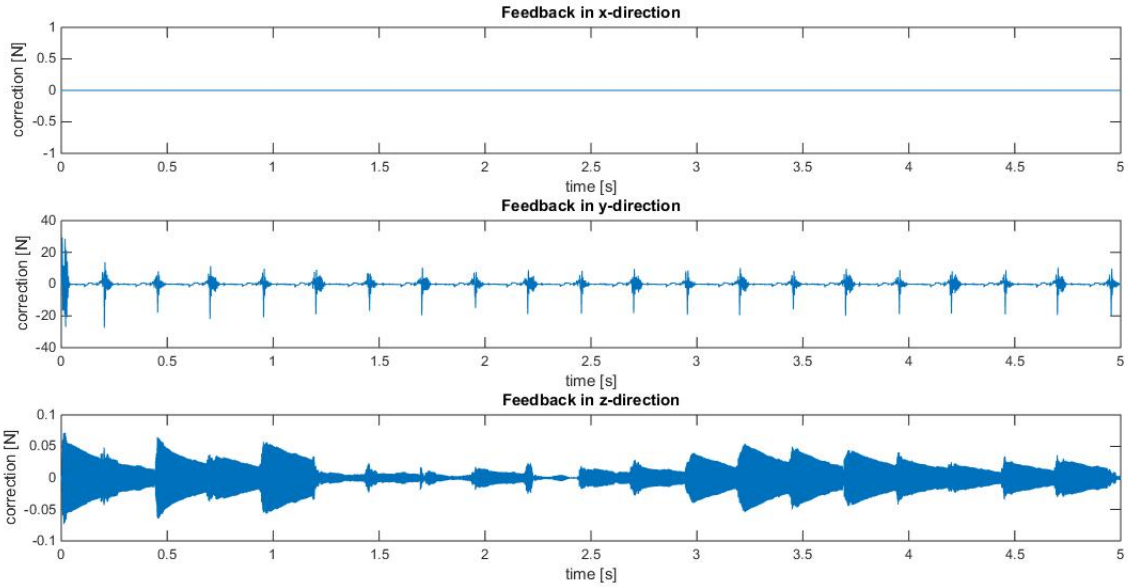

Figure 11.20: *Scaled feedback part of the spindle $D_c C\mathbf{x}$ of the system in [N].*

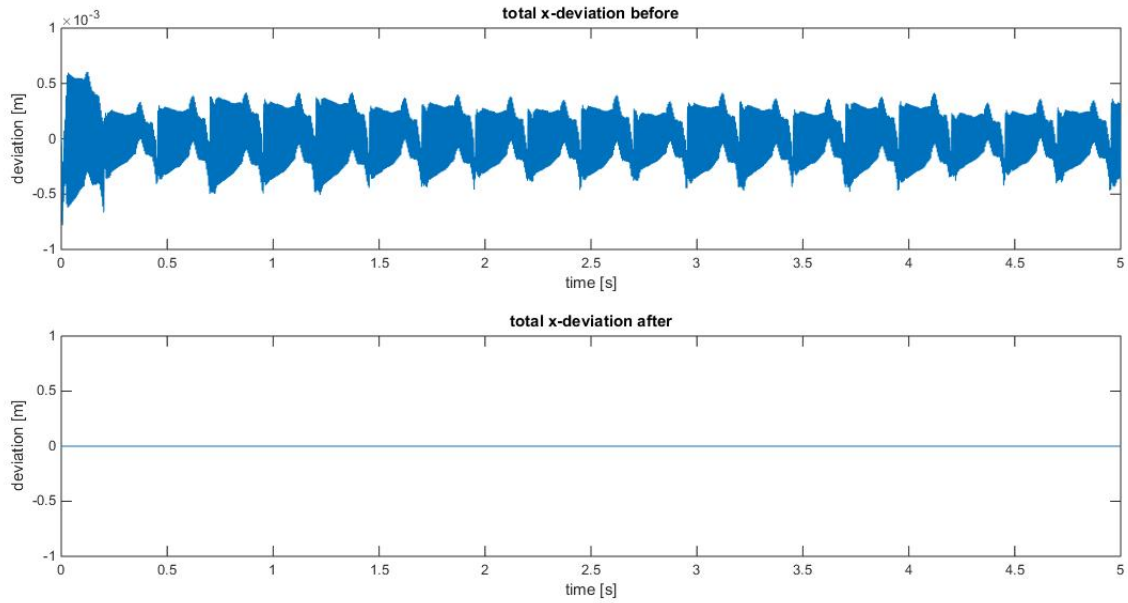

Figure 11.21: *Run 1 comparison of total deviation in x-direction before and after controller in [m].*

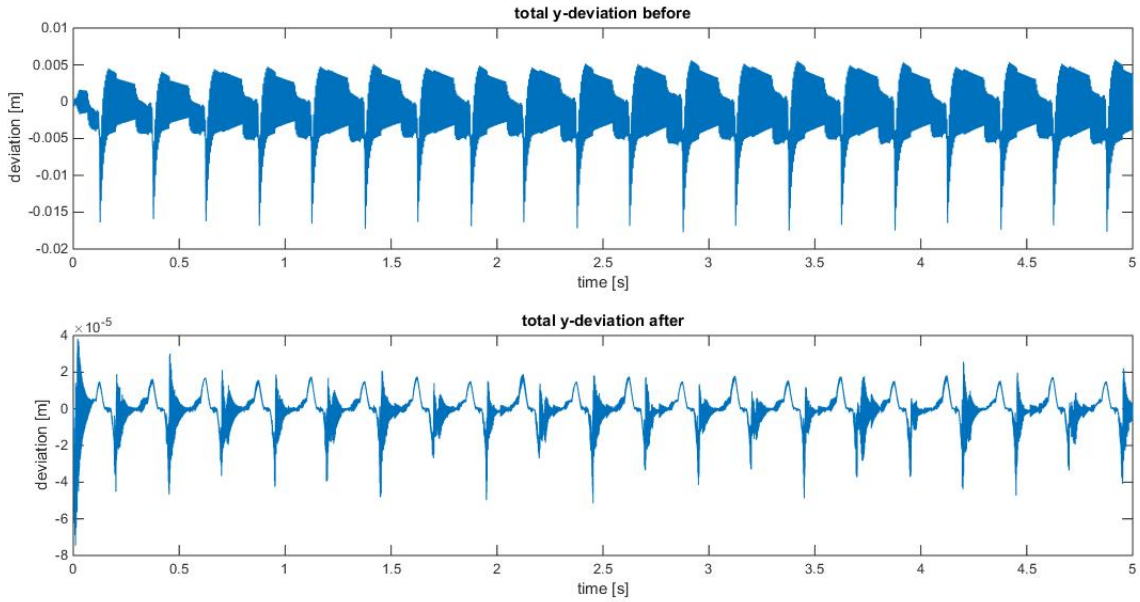

Figure 11.22: *Run 1 comparison of total deviation in y-direction before and after controller in [m].*

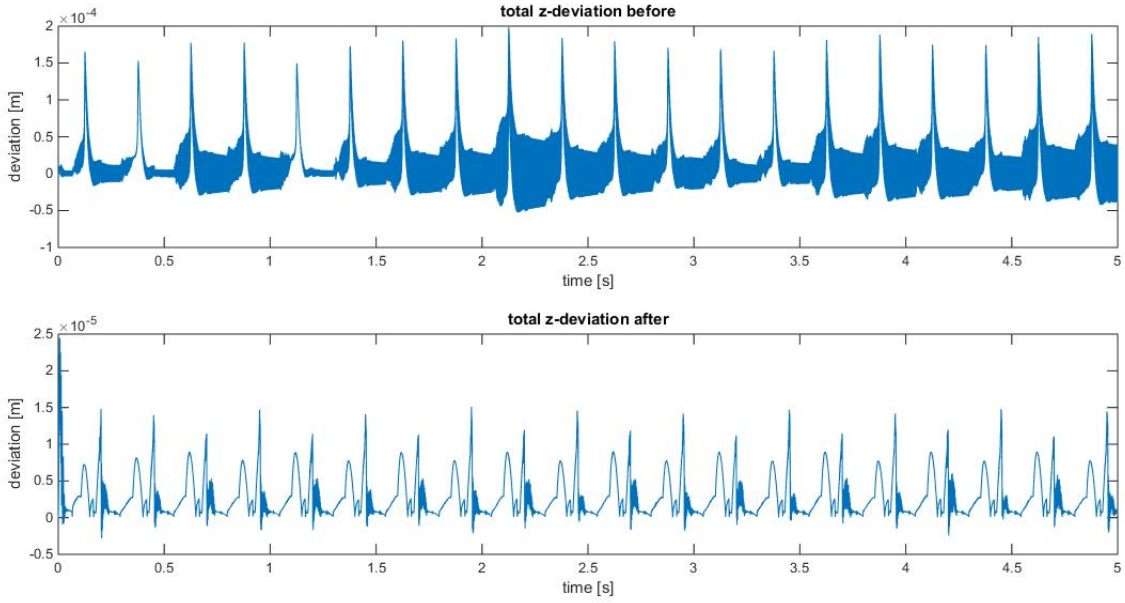

Figure 11.23: *Run 1 comparison of total deviation in z-direction before and after controller in [m].*

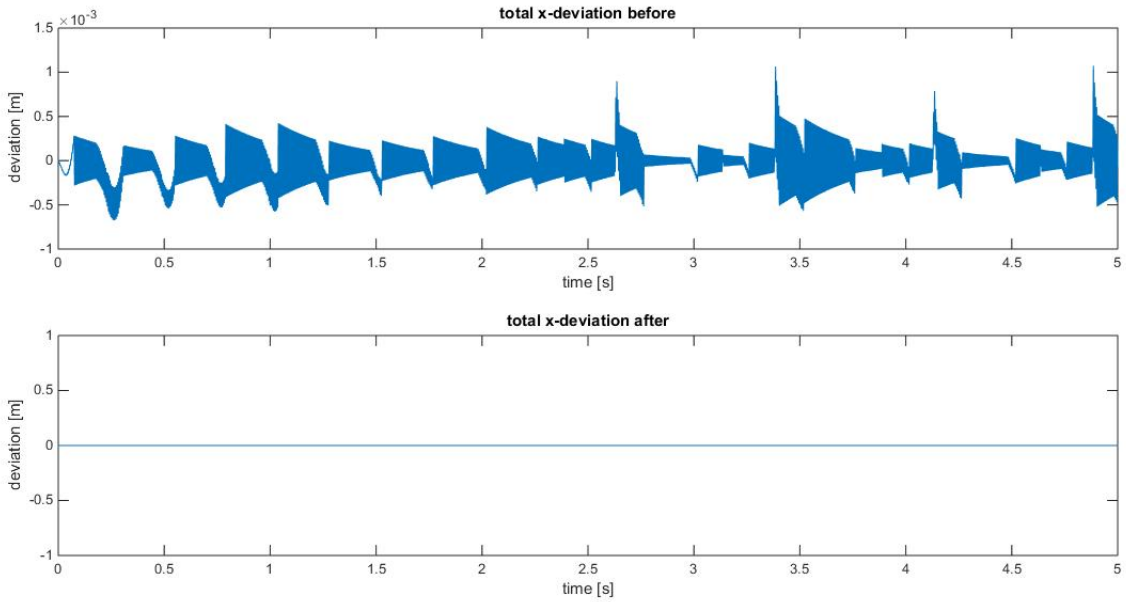

Figure 11.24: *Run 4 comparison of total deviation in x-direction before and after controller in [m].*

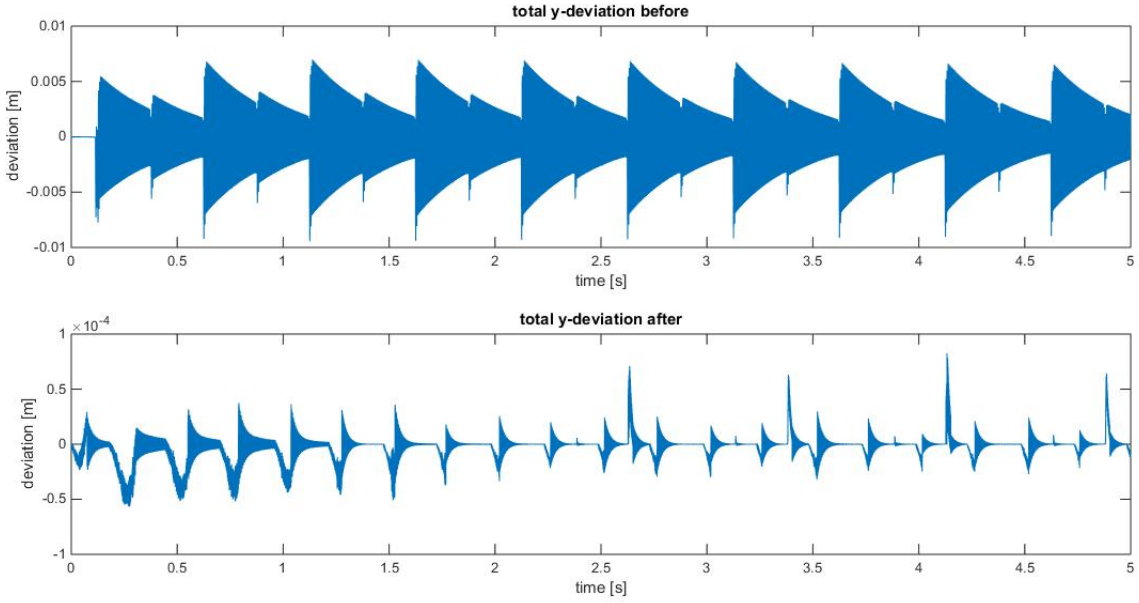

Figure 11.25: *Run 4 comparison of total deviation in y-direction before and after controller in [m].*

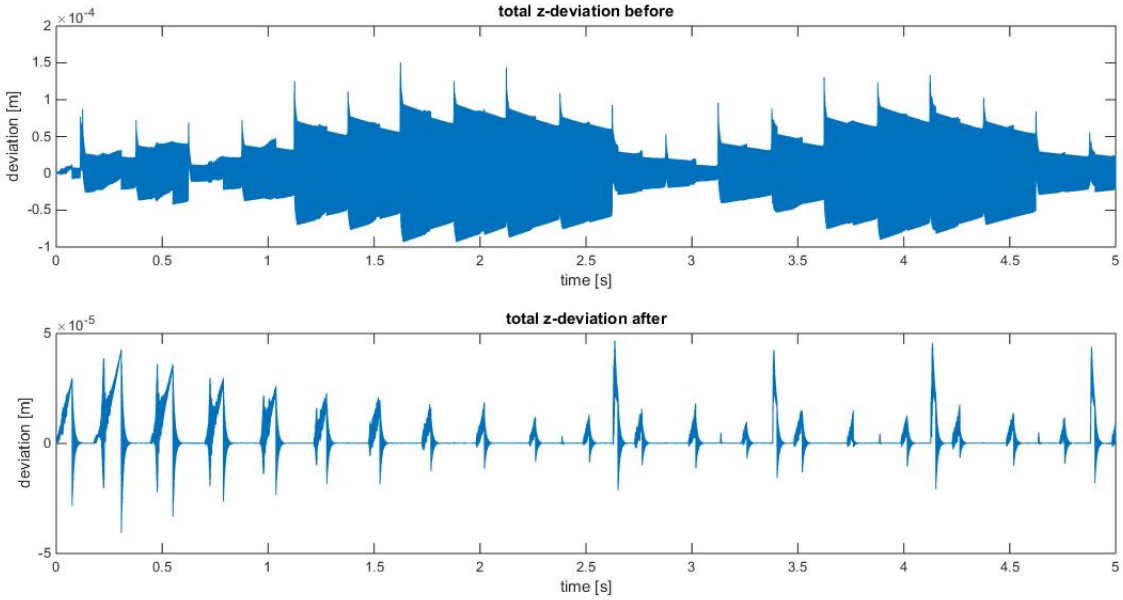

Figure 11.26: *Run 4 comparison of total deviation in z-direction before and after controller in [m].*

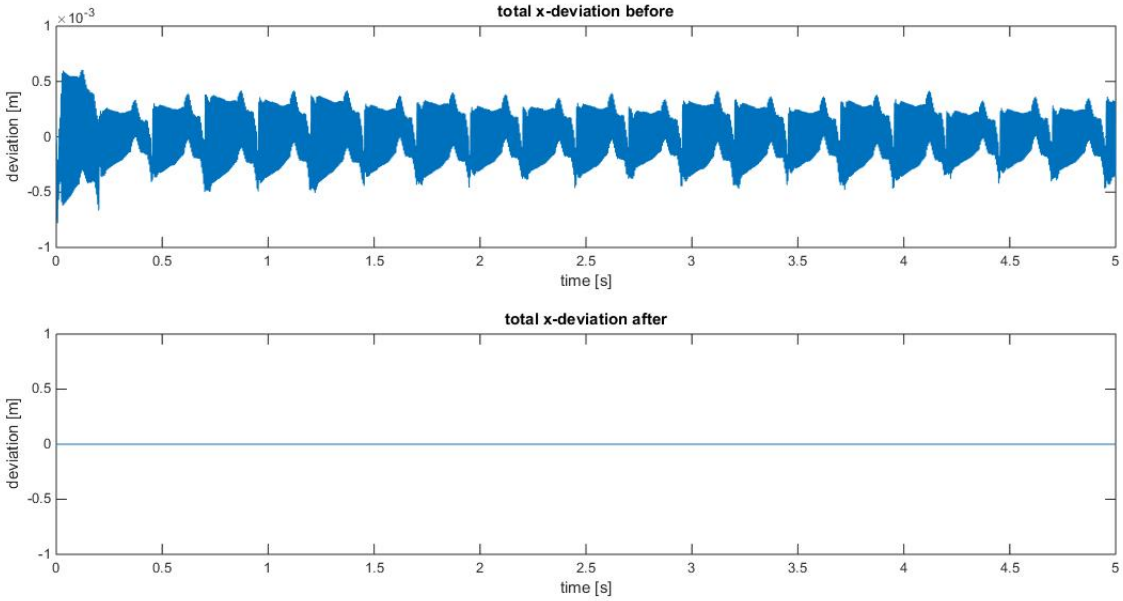

Figure 11.27: *Run 1 comparison of total deviation in x-direction before and after scaled controller in [m].*

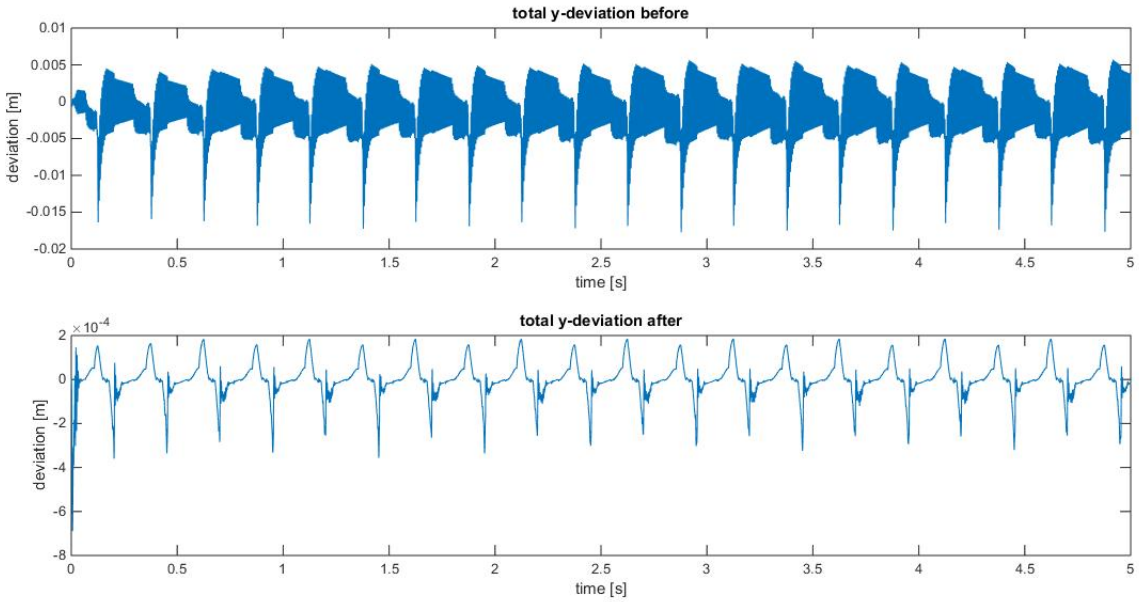

Figure 11.28: *Run 1 comparison of total deviation in y-direction before and after scaled controller in [m].*

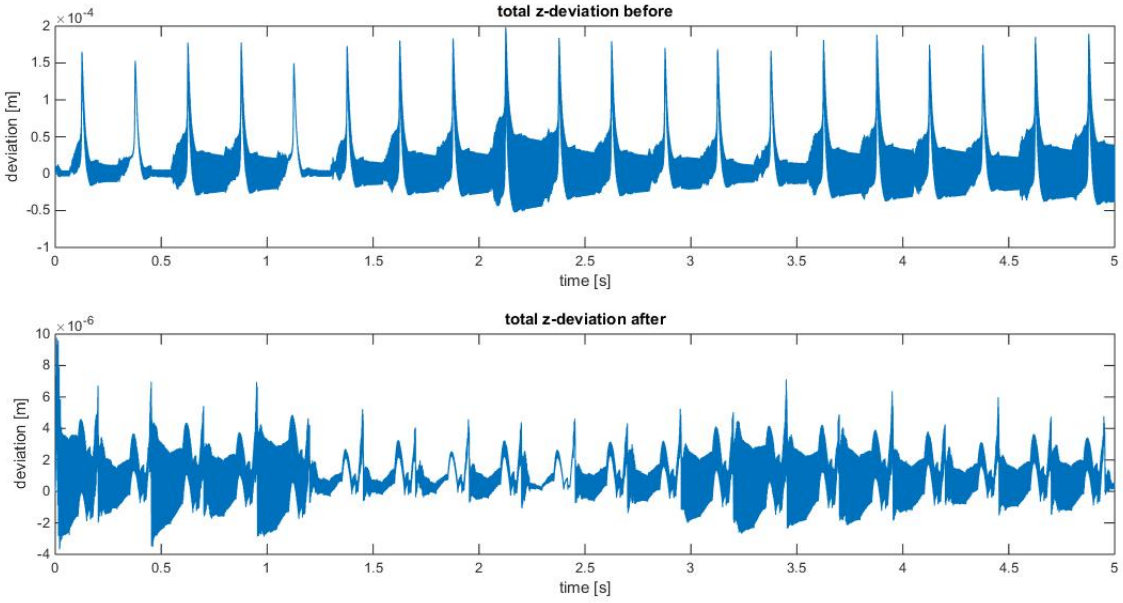

Figure 11.29: *Run 1 comparison of total deviation in z-direction before and after scaled controller in [m].*

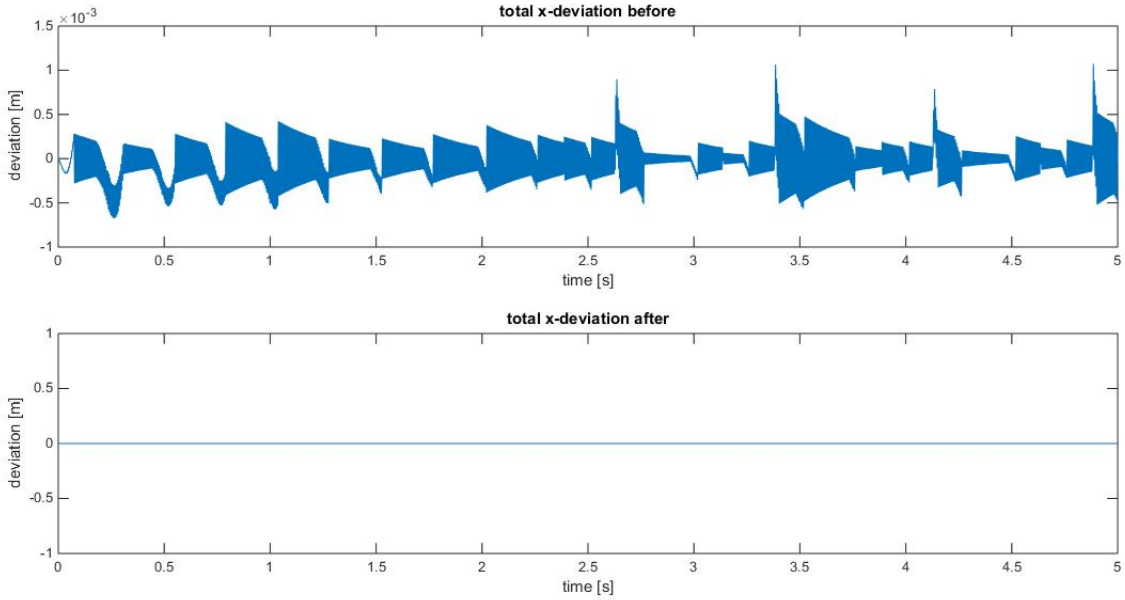

Figure 11.30: *Run 4 comparison of total deviation in x-direction before and after scaled controller in [m].*

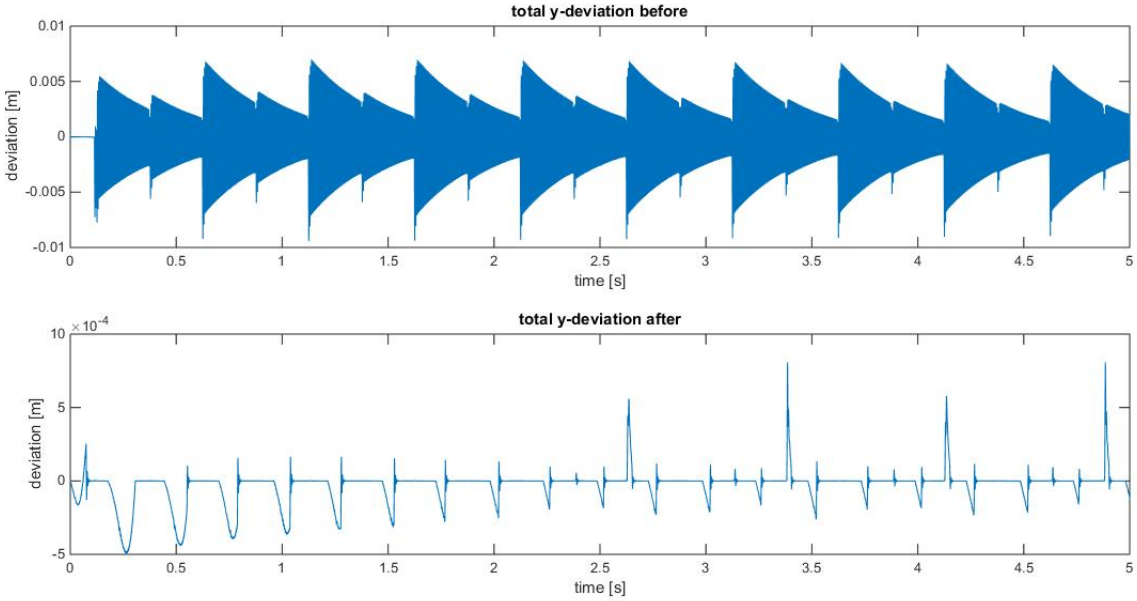

Figure 11.31: *Run 4 comparison of total deviation in y-direction before and after scaled controller in [m].*

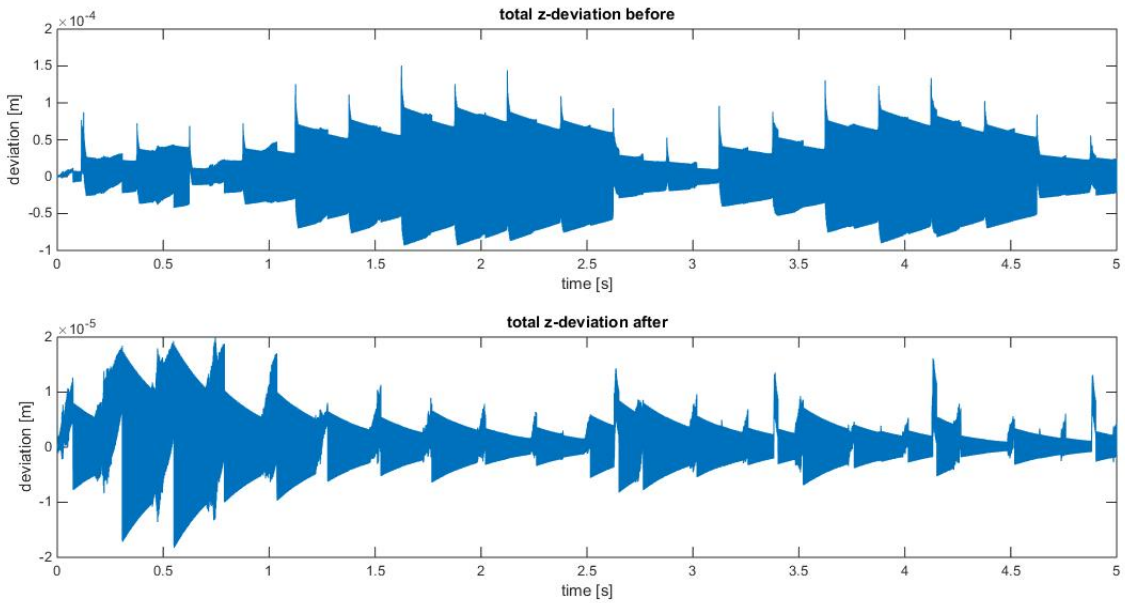

Figure 11.32: *Run 4 comparison of total deviation in z-direction before and after scaled controller in [m].*

12

Conclusion

In this thesis a mathematical model has been derived for the end-effector of a one-armed polishing robot. This mathematical model exists of two parts: the polishing brush and a model extended to the airbearing. Such a model has to our knowledge not been developed before for this exact process and robot. With this model the actual problem has been simulated for different values of impact behavior. The results of these simulations are then used to design a correction for the unwanted behavior. This has been done with static state controllers. The controllers are found by optimization methods. The two different optimization methods treated in this thesis are minimizing the H_∞-norm and minimizing the H_2-norm.

The research question for this thesis is: what is causing the jumping of the polishing tool in specific situations on non-flat surfaces and how can this be prevented? The cause that has been found with the simulations is the flexibility in the spindle and the airbearing of the polishing robot. Together with resultant forces, caused by the rotation of the tool and a certain angle of elevation of the material, the flexibility causes the tool to jump in all directions. The method tested to correct this is a static state feedback where the controller is found by minimizing the H_∞-norm. The results show that this type of controller and feedback can decrease the jumping to $0-10$ % of the original displacements.

The limitations of the simulation of the displacements have been that the forces, caused in the polishing disc by impact behavior on the material, are unrealistically big for these types of models. To have a realistic result of final displacements in the airbearing and spindle model, the forces from the polishing disc are scaled to describe the problem. This method does take into account the frequency and relative magnitude between axes of the forces. The limitation of the correction or feedback method is that only the very basic static state feedback has been tested. Other types of feedback methods such as the dynamic feedback could improve the performance of the polishing tool end-effector even more.

When an actual solution for the polishing robot needs to be tested, the model and controller methods presented in this thesis can be used to approximate the needed parameters and predict an approximation of the performance for that solution.

13

Future work and recommendations

In this thesis an entire model is defined for a specific robot and polishing process. There is still room for improvement, specifically for some properties of the process. For example the material geometry of the example mould is based on a guess. This geometry determines the resultant forces and therefore the angular velocities and with that the impact forces. Adding an actual mould geometry might even solve the problem of too high impact forces. Another element that can add to the quality of the model is a study on the material properties such as the friction coefficient and impact constants such as the impact duration approximation and the spring constant.

Another entirely different method would be the use of measurements. If forces in the x- and y-direction could be measured, the impact forces would be actually known and could be corrected a lot easier. This would avoid the entire polishing disc model and therefore the magnitude of impact forces. Taking measurements during such a process is difficult, since the use of liquids complicate the use of extra electrical devices.

In this thesis only the static state feedback has successfully been applied without singularity problems. There are other ways in Matlab to use the same methods but it might give a result different from that of the Yalmip package. For example as used in [36] the LMI toolbox is used instead of the Yalmip toolbox. There also exist build-in Matlab functions for minimizing the H_2 and H_∞ norm. But solving the singularity problem for the dynamic controller would also give an extra controller that can vary over time. Another method discussed in this thesis but not applied is using an observer in combination with a state controller. If a proper LMI definition is found for calculating the H_∞-norm it could also extend the controller to a situation where only part of the state is known.

Another entirely different direction which has not been studied during this thesis is solving the problem as a rotating beam partial differential equation. The impact forces are then part of the boundary conditions and the final solution can be calculated by using techniques such as finite element methods. The same difficulty remains however with the description of the impact forces.

To conclude there are a lot of ways to extend this model or to adjust this model, but the final product of this thesis can be used for further research by add an actual mould geometry and, if this indeed solves the magnitude of impact forces, use the entire model. Otherwise the workaround still shows the behavior of the impact forces and the results are still helpful. Then actual solutions for the robot itself can be tested using this product. By adjusting the D_c matrix of the state representation in such a manner that it describes the possible solution. The simulation model can then show if the solution does what it supposed to do and can even give a starting approximation of design parameters.

A

Simulation results without damping

In this appendix the results are presented that are not in detail discussed in the corresponding chapter.

A.1. Results of run 2

Figures A.1, A.2, A.3, A.4 and A.5.

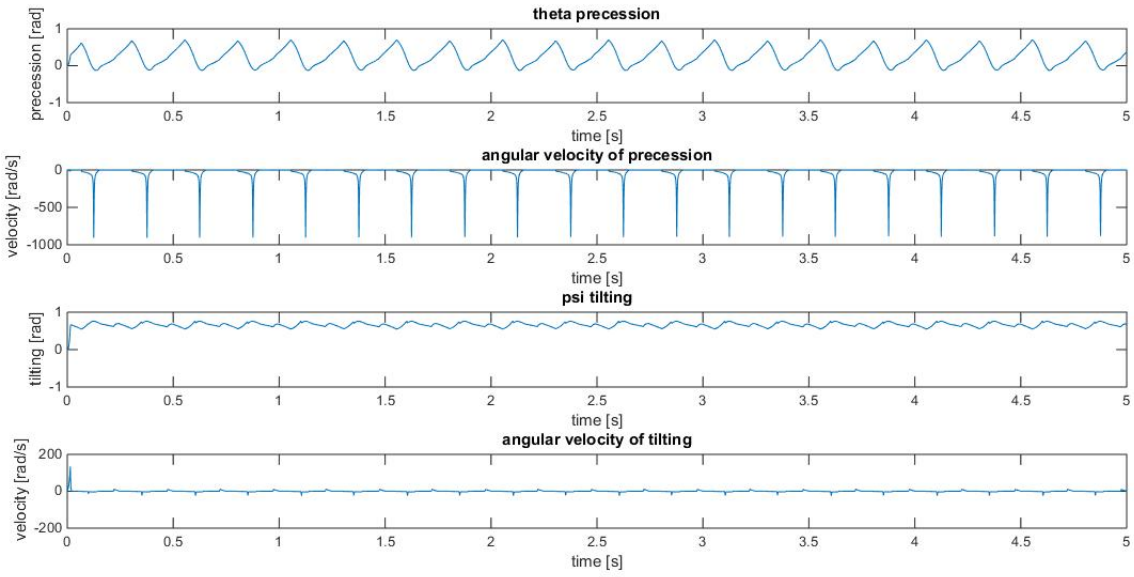

Figure A.1: $\theta, \dot{\theta}, \psi, \dot{\psi}$ *presented over 5 seconds.*

A.2. Extra results of run 4

Figures A.6 and A.7.

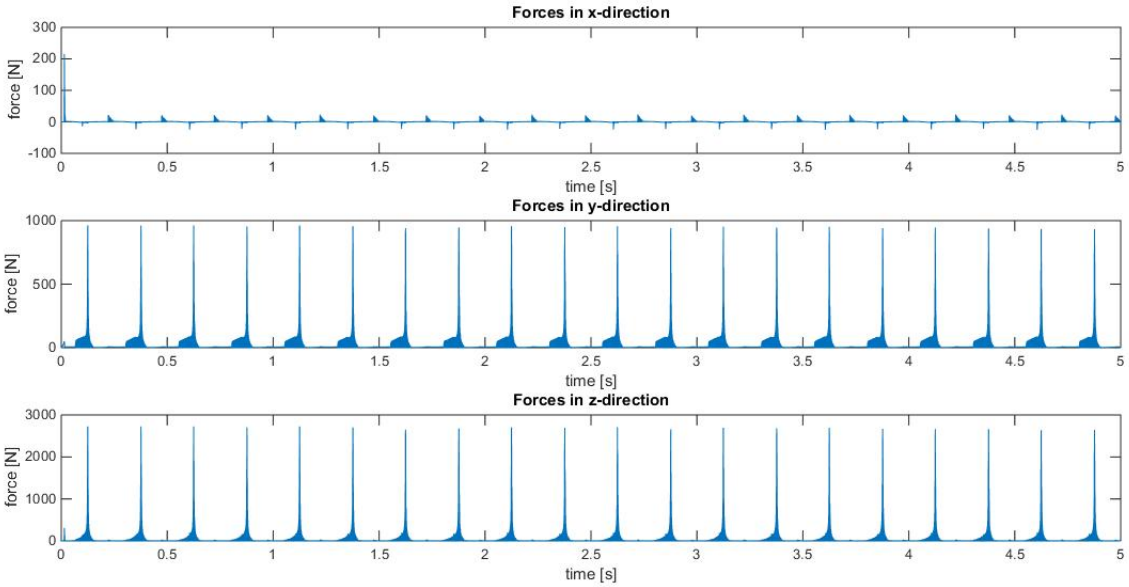

Figure A.2: *Forces in the three principal directions in [N].*

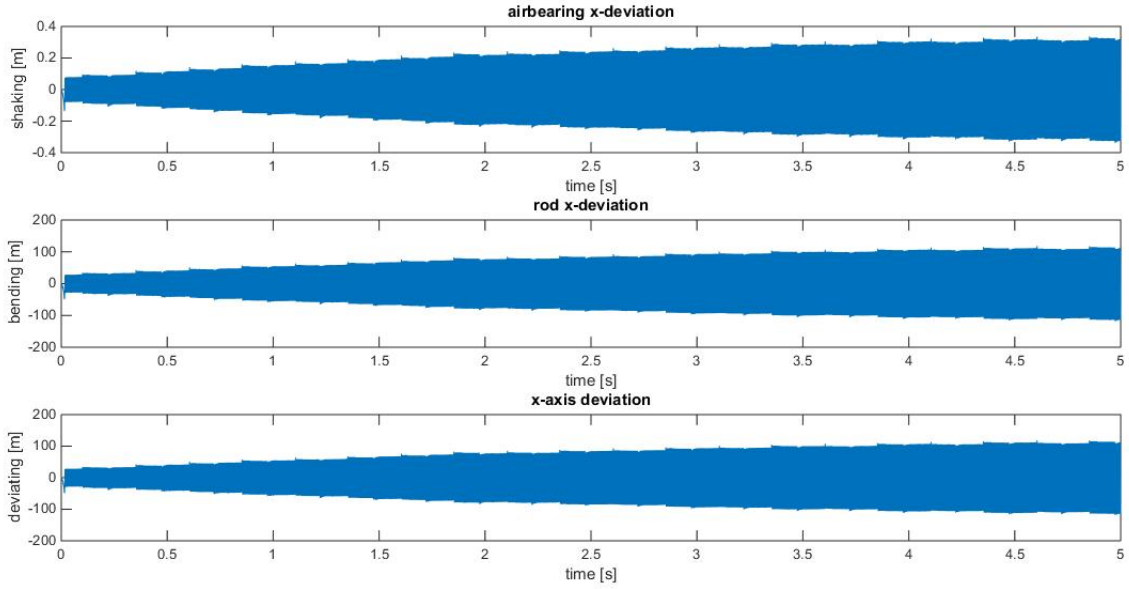

Figure A.3: *Deviations in x-direction without damping in [m].*

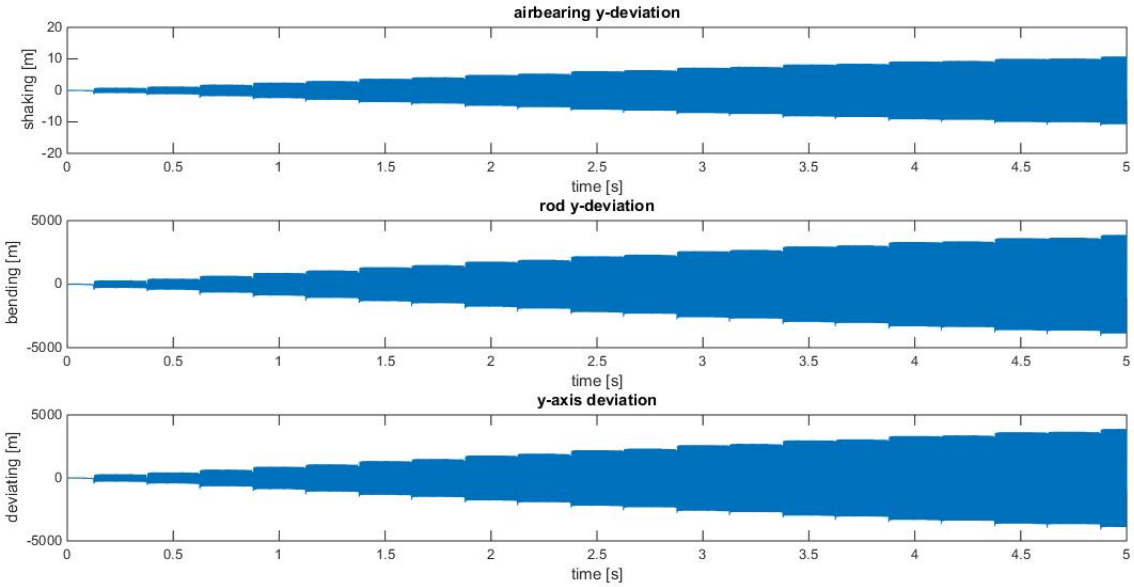

Figure A.4: *Deviations in y-direction without damping in [m].*

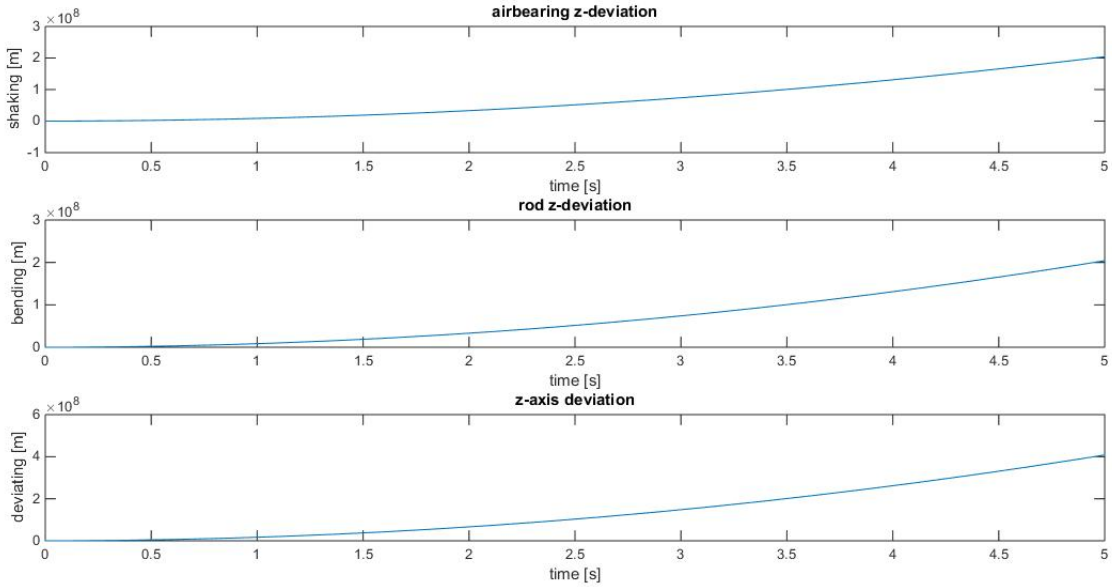

Figure A.5: *Deviations in z-direction without damping in [m].*

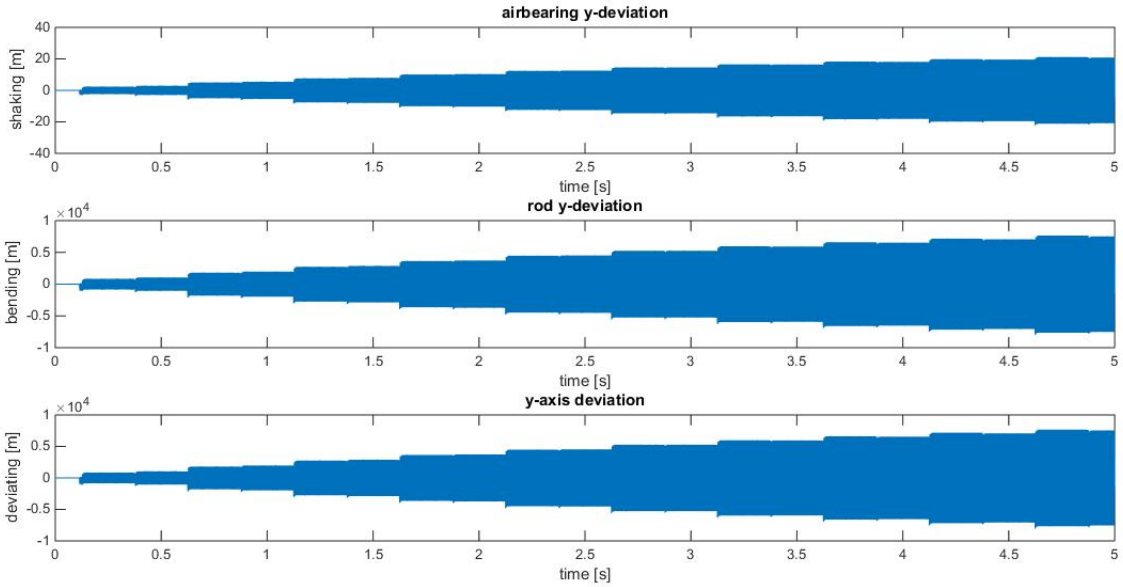

Figure A.6: *Deviations in y-direction without damping in [m].*

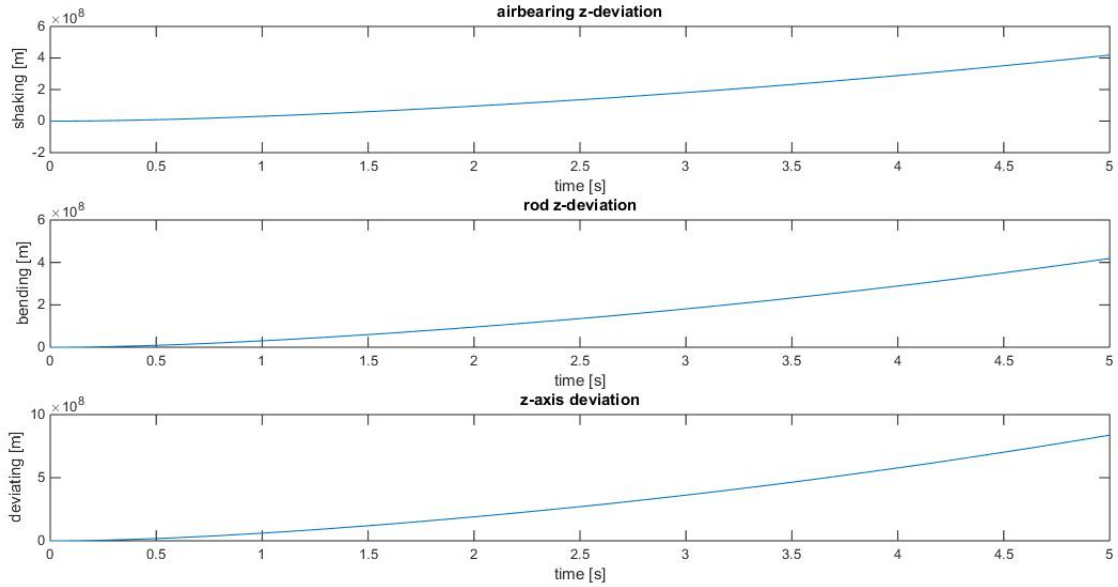

Figure A.7: *Deviations in z-direction without damping in [m].*

B

Simulation results with damping

In this appendix the results are presented that are not in detail discussed in the corresponding chapter.

B.1. Extra results run 1

Damping of 1%. Figures B.1, B.2 and B.3.

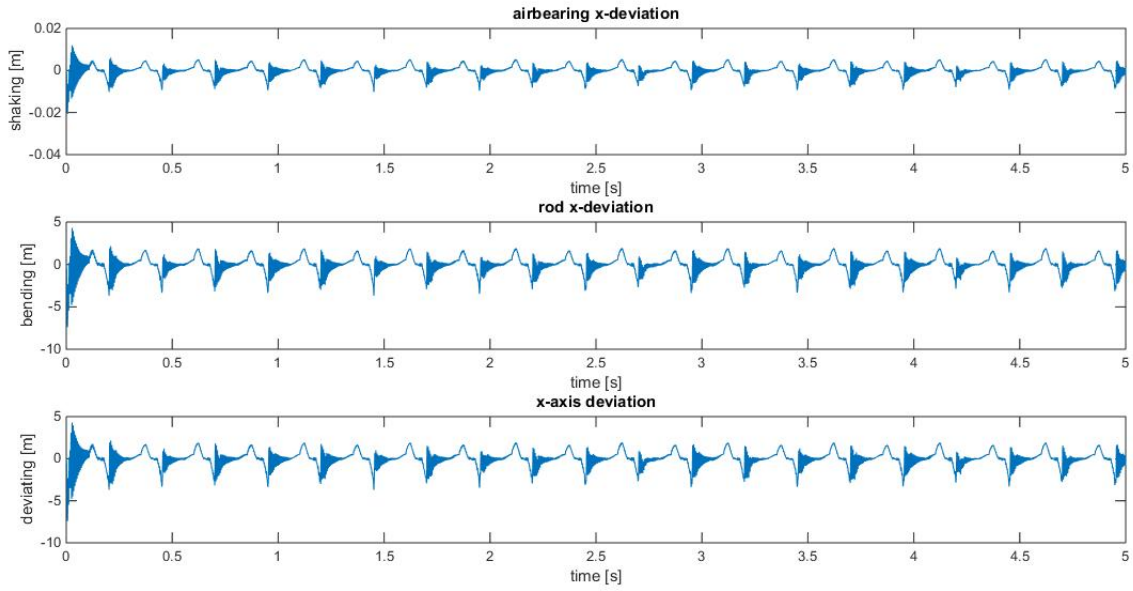

Figure B.1: *Deviations in x-direction with* 1 *% damping in [m].*

B.2. Results run 2

B.2.1. First damping

Damping of 0.1%. Figures B.4, B.5 and B.6.

B.2.2. Second damping

Damping of 1 %. Figures B.7, B.8 and B.9.

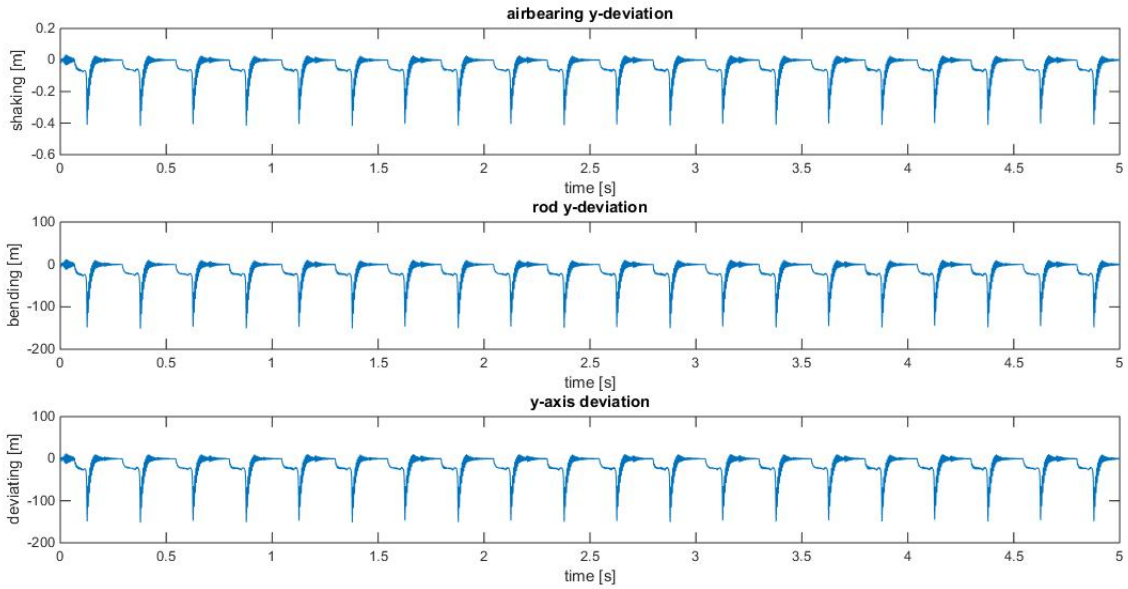

Figure B.2: *Deviations in y-direction with* 1 *% damping in [m].*

B.3. Extra results run 4

B.3.1. First damping

Damping of 0.1%. Figures B.10, B.11 and B.12.

B.3.2. Second damping

Damping of 1 %. Figures B.13, B.14 and B.15.

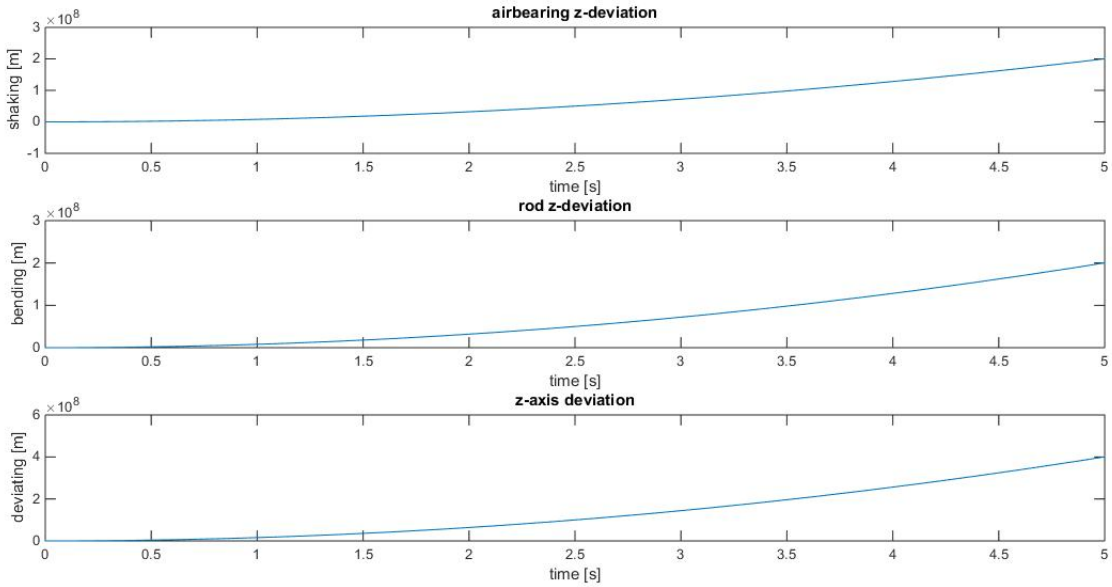

Figure B.3: *Deviations in z-direction with* 1 *% damping in [m].*

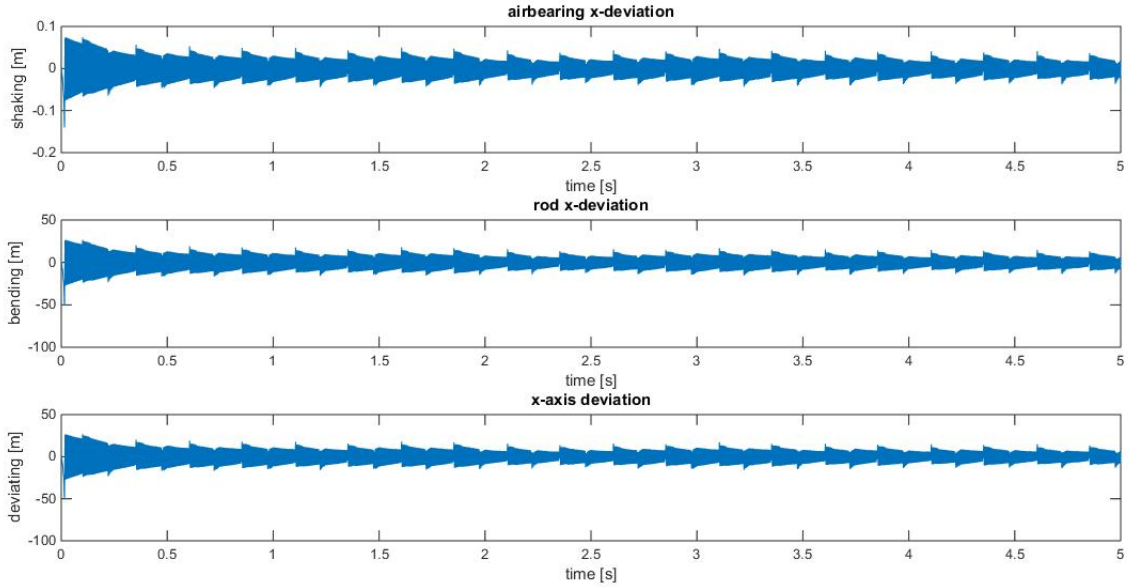

Figure B.4: *Deviations in x-direction with* 0.1 *% damping in [m].*

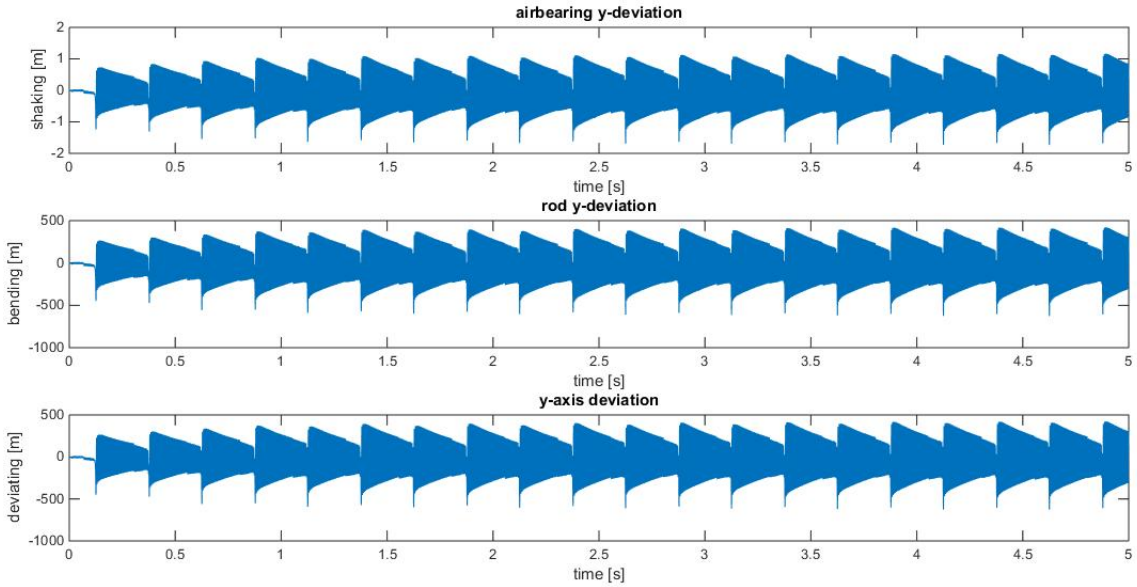

Figure B.5: *Deviations in y-direction with* 0.1 *% damping in [m].*

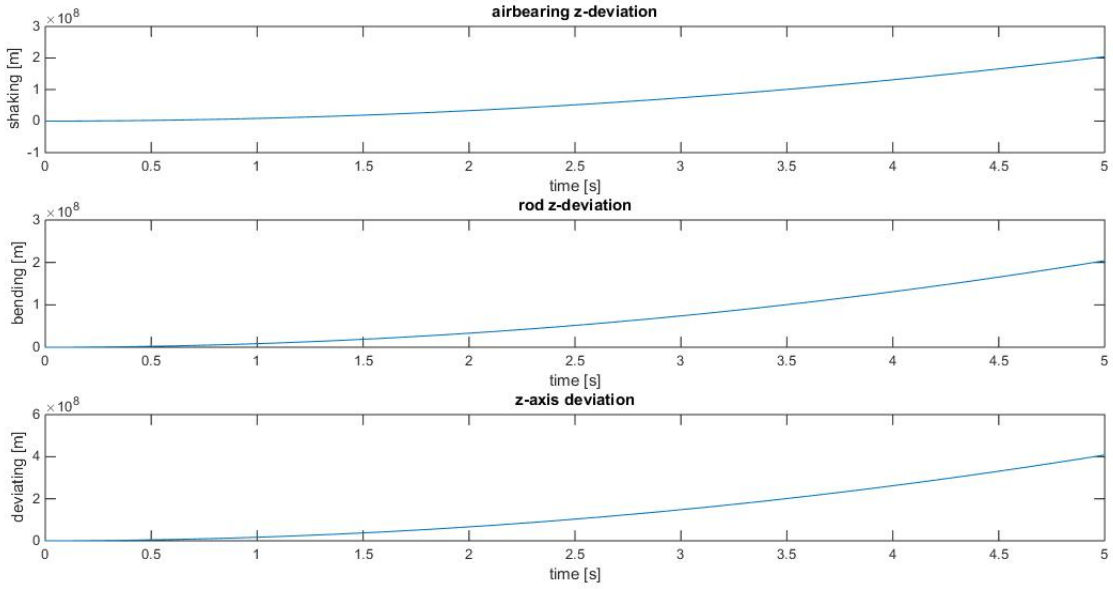

Figure B.6: *Deviations in z-direction with* 0.1 *% damping in [m].*

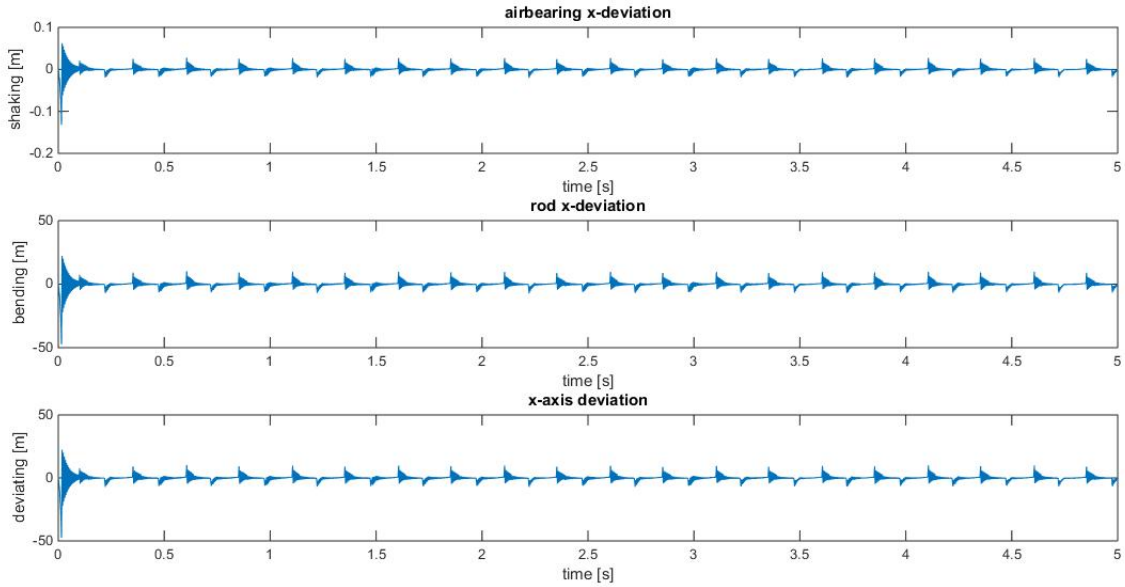

Figure B.7: *Deviations in x-direction with* 1 *% damping in [m].*

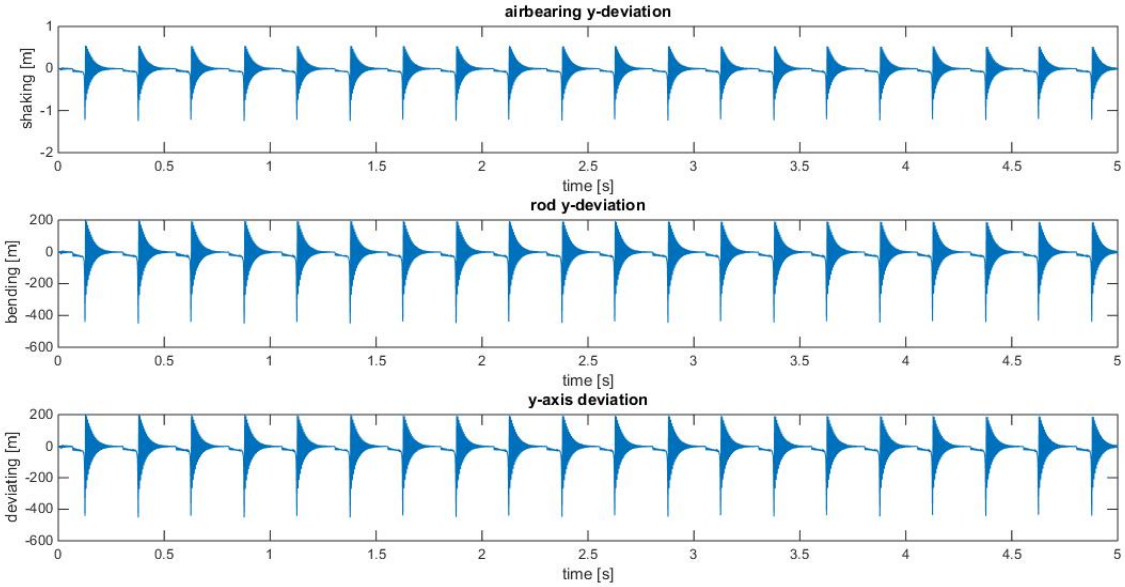

Figure B.8: *Deviations in y-direction with* 1 *% damping in [m].*

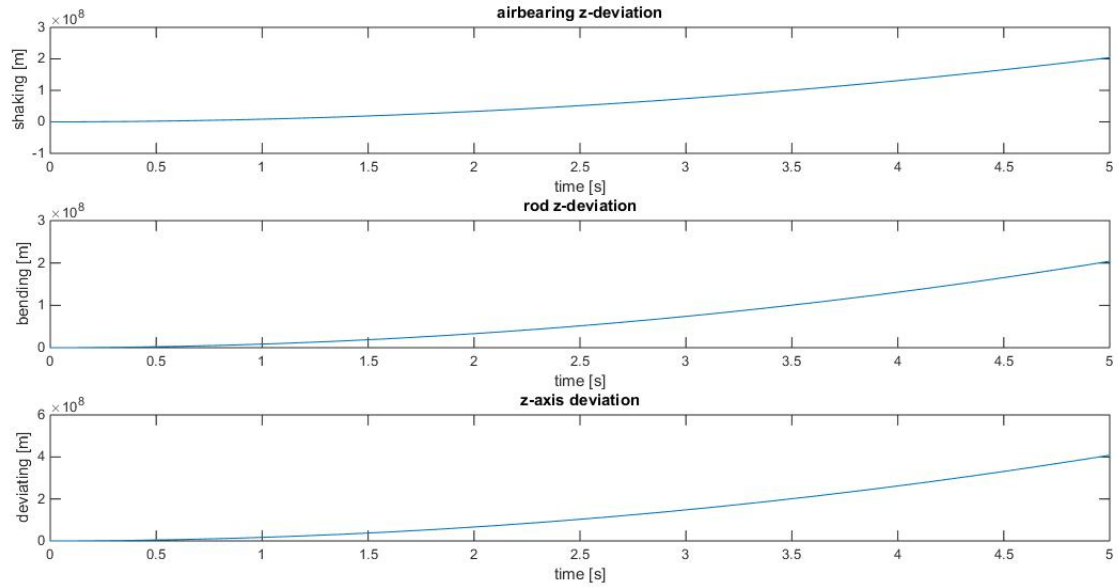

Figure B.9: *Deviations in z-direction with* 1 *% damping in [m].*

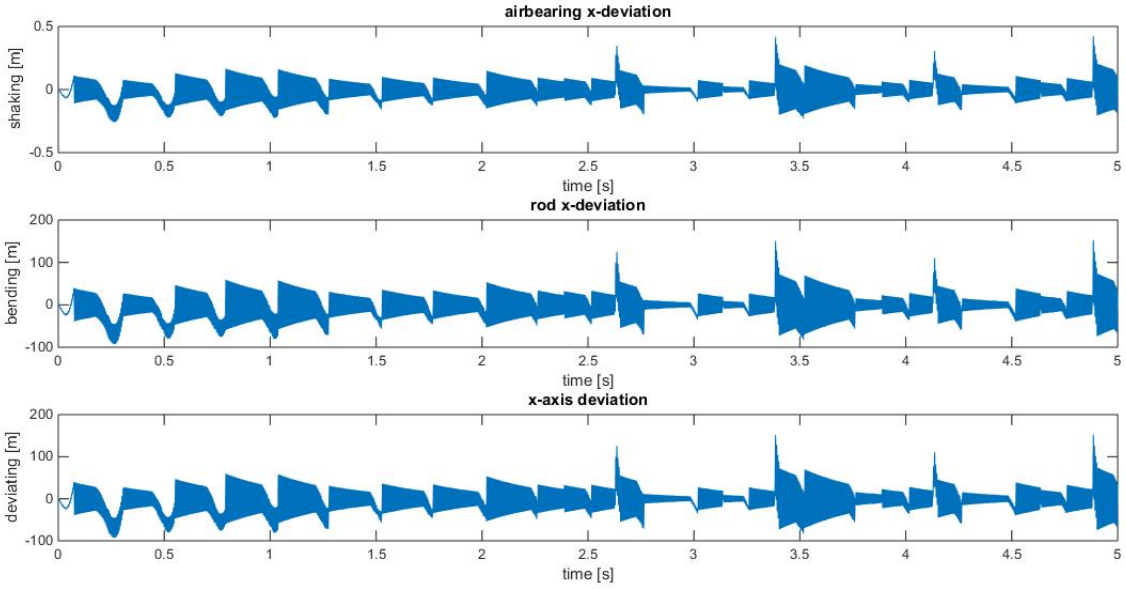

Figure B.10: *Deviations in x-direction with* 0.1 *% damping in [m].*

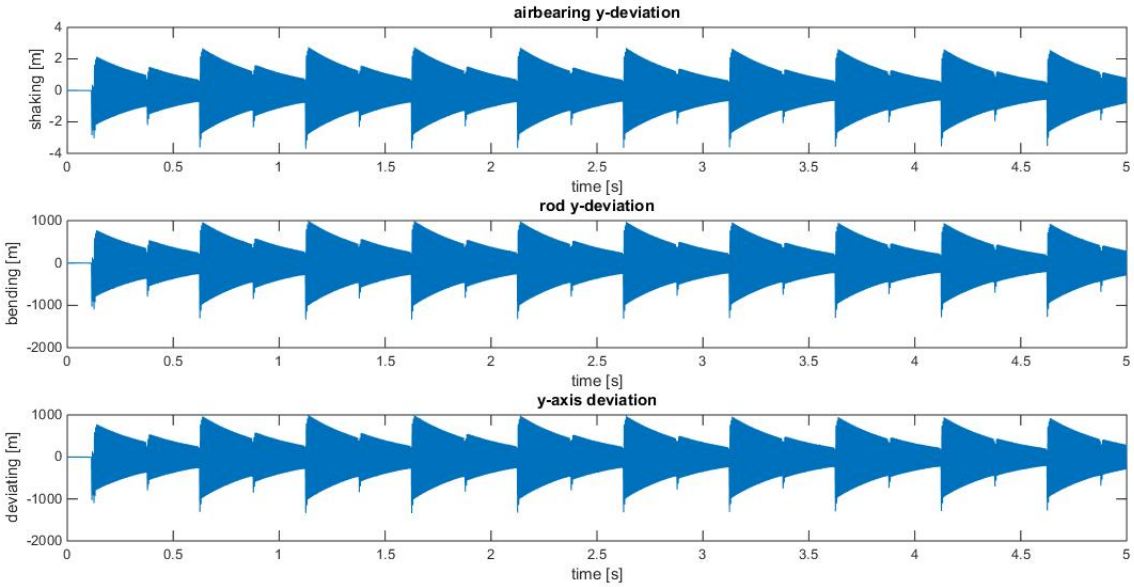

Figure B.11: *Deviations in y-direction with* 0.1 *% damping in [m].*

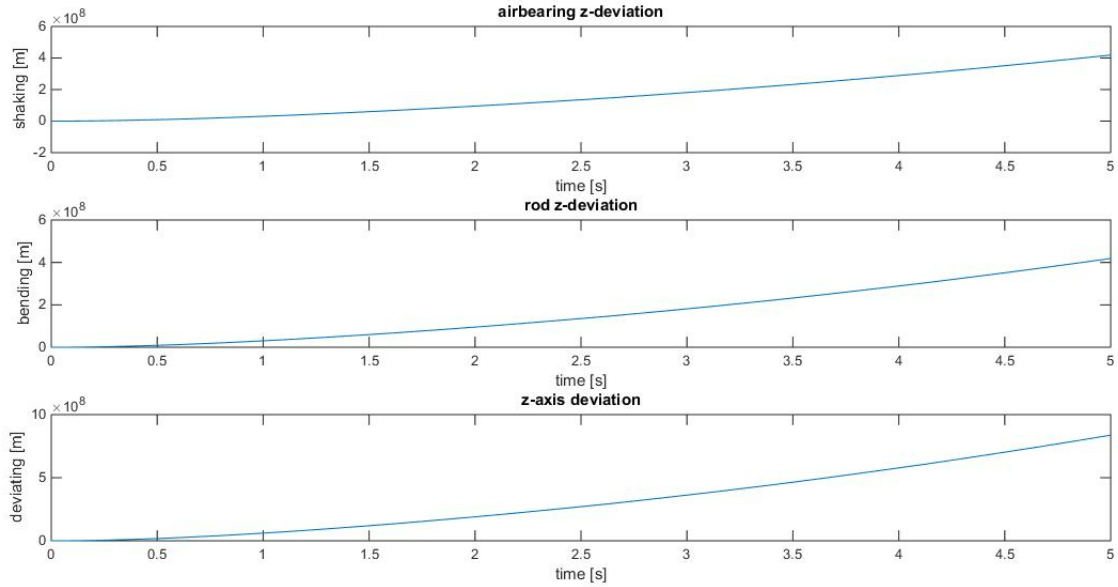

Figure B.12: *Deviations in z-direction with* 0.1 *% damping in [m].*

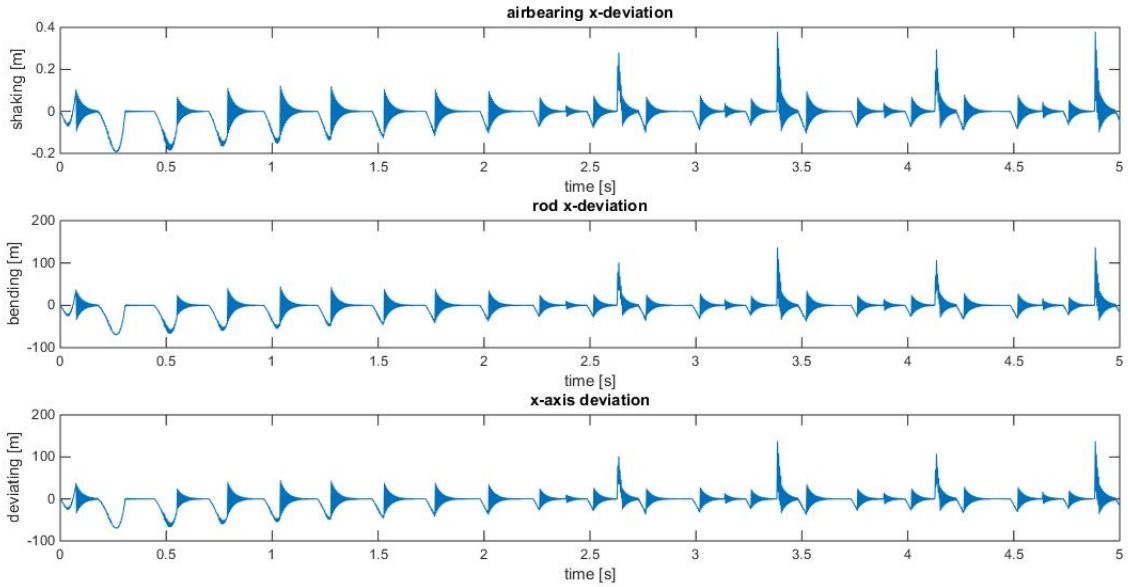

Figure B.13: *Deviations in x-direction with* 1 *% damping in [m].*

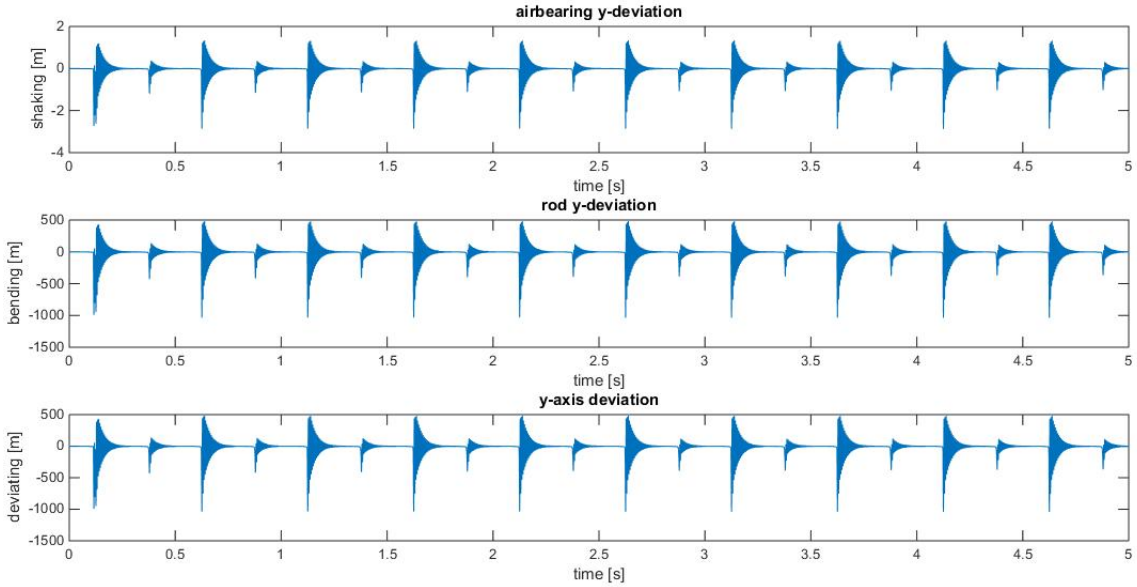

Figure B.14: *Deviations in y-direction with* 1 *% damping in [m].*

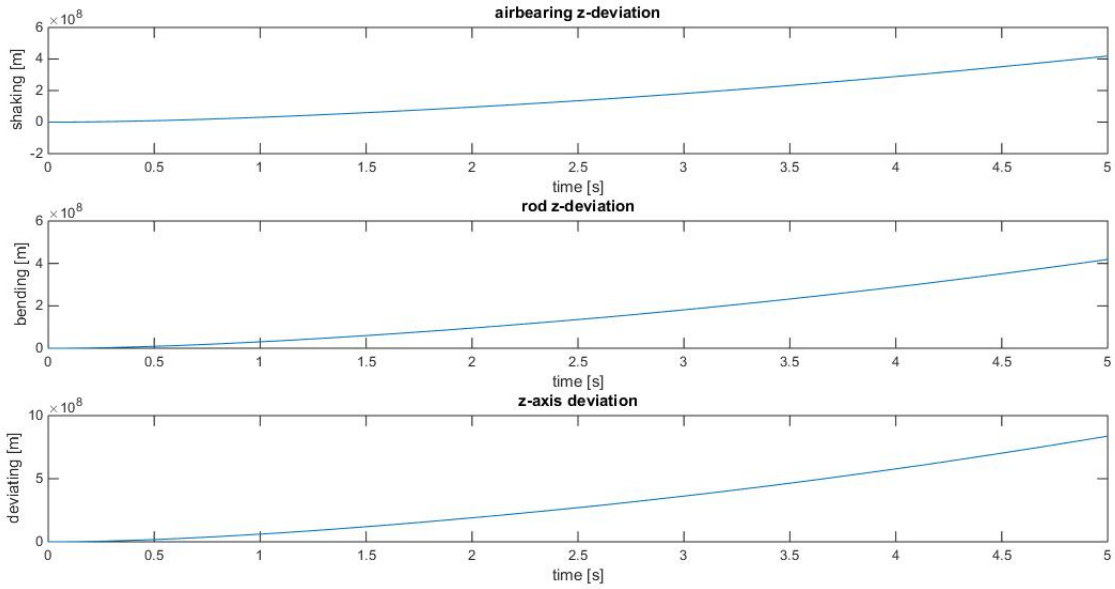

Figure B.15: *Deviations in z-direction with* 1 *% damping in [m].*

C

Simulation results of controllers

In this appendix the results are presented that are not in detail discussed in the corresponding chapter.

C.1. Controller matrices for first solution

The exact matrices resulting from optimizing the LMI's are presented in this section.

C.1.1. Static state controller and the H2-norm

The controller matrix found for 0.1% damping is given by

$$D_c = \begin{pmatrix} -2.1030\cdot10^{17} & 2.0970\cdot10^{9} & 3.0127\cdot10^{17} & 4.3969\cdot10^{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2.9994\cdot10^{17} & -2.9909\cdot10^{9} & -5.6768\cdot10^{17} & 3.8381\cdot10^{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2.1030\cdot10^{17} & 2.0970\cdot10^{9} & 3.0127\cdot10^{17} & 4.3969\cdot10^{8} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2.9994\cdot10^{17} & -2.9909\cdot10^{9} & -5.6768\cdot10^{17} & 3.8381\cdot10^{8} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4.5023\cdot10^{16} & 1.7683\cdot10^{8} & 1.6719\cdot10^{16} & 4.5613\cdot10^{7} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.6744\cdot10^{16} & -1.7638\cdot10^{7} & -9.6861\cdot10^{16} & 3.9255\cdot10^{8} \end{pmatrix} \tag{C.1}$$

The scaled controller matrix with 0.1 % damping is

$$D_c = \begin{pmatrix} -1.3559\cdot10^{12} & 2.9271\cdot10^{5} & 1.1746\cdot10^{11} & 2.1971\cdot10^{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -6.8568\cdot10^{9} & 1.0562\cdot10^{5} & -2.1865\cdot10^{13} & 9.4967\cdot10^{5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1.3559\cdot10^{12} & 2.9271\cdot10^{5} & 1.1746\cdot10^{11} & 2.1971\cdot10^{6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -6.8568\cdot10^{9} & 1.0562\cdot10^{5} & -2.1865\cdot10^{13} & 9.4967\cdot10^{5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.3346\cdot10^{12} & 1.1758\cdot10^{7} & 3.8282\cdot10^{8} & 4.3916\cdot10^{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5.5072\cdot10^{7} & 4.7306\cdot10^{5} & -5.3423\cdot10^{12} & 3.1032\cdot10^{5} \end{pmatrix} \tag{C.2}$$

C.1.2. Static state controller and the H infinity-norm

The controller matrix found for 0.1% damping is given by

$$D_c = \begin{pmatrix} -1.7760\cdot10^{6} & 7.1658\cdot10^{2} & 4.8323\cdot10^{3} & -0.6122 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4.8322\cdot10^{3} & -1.7826 & -3.0898\cdot10^{4} & 1.4777\cdot10^{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1.7760\cdot10^{6} & 7.1658\cdot10^{2} & 4.8323\cdot10^{3} & -0.6122 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4.8322\cdot10^{3} & -1.7826 & -3.0898\cdot10^{4} & 1.4777\cdot10^{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4.3132\cdot10^{6} & 7.4990\cdot10^{2} & 2.8127\cdot10^{5} & -48.0846 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2.8169\cdot10^{5} & 15.5677 & -7.6372\cdot10^{5} & 6.3364\cdot10^{2} \end{pmatrix} \tag{C.3}$$

The scaled controller matrix with 0.1 % damping is

$$D_c = \begin{pmatrix} -5.3415 & -0.0758 & 0.4234 & -0.0556 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0564 & -0.1047 & -44.1852 & -3.7897 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -5.3415 & -0.0758 & 0.4234 & -0.0556 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0564 & -0.1047 & -44.1852 & -3.7897 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5.6772 & -0.1203 & -0.0041 & -0.0811 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0204 & -0.1220 & -16.1900 & -0.1105 \end{pmatrix} \tag{C.4}$$

C.2. Controller matrices for second solution

When the B-matrix is changed to a different type of solution, the controller matrices change. The scaled and original controller matrix from the H_∞-norm for 0.1 % damping are then given by

$$D_c = \begin{pmatrix} 9.0544 \cdot 10^{7} & 0.3096 & 1.8918 \cdot 10^{5} & -0.2062 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 8.9506 \cdot 10^{6} & -4.4892 & -3.2651 \cdot 10^{6} & -17.2076 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 9.0544 \cdot 10^{7} & 0.3096 & 1.8918 \cdot 10^{5} & -0.2062 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8.9506 \cdot 10^{6} & -4.4892 & -3.2651 \cdot 10^{6} & -17.2076 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2.1167 \cdot 10^{8} & -18.0526 & 6.5185 \cdot 10^{7} & -13.4988 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5.6515 \cdot 10^{8} & -46.9699 & -5.1039 \cdot 10^{7} & -36.6347 \end{pmatrix} \tag{C.5}$$

The scaled controller matrix with 0.1 % damping is

$$D_c = \begin{pmatrix} -4.3912 \cdot 10^{-5} & -0.1127 & -0.0029 & -0.5432 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0027 & -2.2877 & -0.9858 & -191.9876 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -4.1680 \cdot 10^{-5} & -0.1095 & -0.0019 & -0.5654 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0035 & -1.1354 & -0.6822 & -200.3145 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3.9457 \cdot 10^{-5} & -0.5892 & -1.5791 \cdot 10^{-4} & -0.3606 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.6238 \cdot 10^{-4} & -1.8623 & -0.0011 & -4.0172 \end{pmatrix} \tag{C.6}$$

D

Matlab code

D.1. Polishing brush ode function

```
function dwdt = odeBrush21(t,w) %two angle model with impuls, variable material
    %Variables:
    vars = getGlobalVars;
    p = vars(1);
    omega = vars(2);
    kp = vars(3);
    %measurements:
    rd = 0.020*0.5; % radius of brush in [m]
    rb = 0.003*0.5; %radius of rod between tool and air bearing [m]
    h = 0.010; %height brush in [m]
    hb = 0.089; %height of air bearing [m]
    wb = 0.05; %width of airbearing [m]
    r = sqrt(rd^2 + (h/2)^2); % = sqrt(0.000125)= 0.01118 [m]
    Rrot = 0.003; %radius of robot rotation [m]
    l = 0.020; %length of rod between tool and air bearing [m]
    A = rb^2*pi; %Area of cross section rod between tool and airbearing [m^2]
    v = 0.001; %longitude speed vector [m/s]

    %masses:
    m = 0.2; %mass brush in kg
    mrod = 0; %mass of rod in kg
    mbearing = 0.337; %mass air bearing in kg
    Ibearing = mrod *(l^2/12 + (hb/2)^2)+ m*((4/3)*((h/2)^2) + 4*(rd^2)+((hb/2)^2+l
        ^2))+mbearing*((hb/2)^2+(wb/2)^2); %Moment of inertia assumed from center of
        rotation in air bearing [kg m^2]
    %masses reduced:
    mz = mbearing;
    mx = Ibearing/((hb/2 + l + h/2)^2) -m - mrod; %Not correct?

    %Springconstants:
    E = 2*10^11 ; % Youngs modulus for steel [N/m^2]
    Ib = (rb^4)*pi/4; %Cross sectional moment of inertia [m^4]
    kt = 110*10^6; % translational springconstant air bearing [N/m] (multiplied by
        6?)
    kx = 3*E*Ib/(l^3); %translational vibration from bending stiffness rod [N/m]
    kz = E*A/l; %Stiffness of the toolholder and rod [N/m]
    kr = 23*10^6; %Torsional stiffness air bearing [Nm/rad]
```

```
35      cp = 4.76*10^(-2); %damper coefficient of slurry with diamond grains, based on
            critically damped assumption [Ns/m]
36
37      %Moments of inertia:
38      I = 0.5*m*r^2; %+ m*Rrot^2? Moment of inertia around principal axis z [kgm^2]
39      I0 = (1/12)*m*(3*r^2+h^2) ; %+ m*Rrot^2? Moment of inertia around principal axes
            x and y [kgm^2]
40
41      %Angles:
42      mu = 0.5*sqrt(2)*0.5; %Coefficient of friction, not correct probably
43      sigma = atan(mu); %angle of friction
44
45      %variable geometry of material
46      %xi= [elevation([delta_p/sqrt(2)+rd delta_p/sqrt(2)]);
47      %     elevation([delta_p/sqrt(2)-rd delta_p/sqrt(2)]);
48      %     elevation([delta_p/sqrt(2) delta_p/sqrt(2)+rd]);
49      %     elevation([delta_p/sqrt(2) delta_p/sqrt(2)-rd])];
50
51      Center1 = Rrot*cos(omega*t) +v*t+ rd+Rrot ;
52      CenterToolLoc = [Center1 ; Rrot*sin(omega*t)];
53      Vel1 = -omega*Rrot*sin(omega*t) +v;
54      CenterToolVel = [Vel1; omega*Rrot*cos(omega*t)];
55      psi0 = pi/2;
56      rod0 = tube_elevation(CenterToolVel, r+Rrot,0);
57      theta0 = rod0;
58      C1 = rd*abs(cos(w(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
59      C2 = -rd*abs(cos(w(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
60      Center3 = (Rrot-rd*abs(cos(w(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
61      C3 = [Center3; (Rrot-rd*abs(cos(w(3)-psi0)))*sin(omega*t)]; %Not sure
62      Center4 = (Rrot+rd*abs(cos(w(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
63      C4 = [Center4; (Rrot+rd*abs(cos(w(3)-psi0)))*sin(omega*t)]; %Not sure
64      xi= [tube_elevation(CenterToolVel, C1(1), C1(2));
65           tube_elevation(CenterToolVel, C2(1), C2(2));
66           tube_elevation(CenterToolVel, C3(1), C3(2));
67           tube_elevation(CenterToolVel, C4(1), C4(2))];
68      hCR =0.5*(tubeFunction(C1(1),C1(2))+tubeFunction(C2(1) ,C2(2)));
69      dh3 = hCR - tubeFunction(C3(1), C3(2)); %The difference in height from point
            below center of rotation to material [m]
70      dh4 = hCR - tubeFunction(C4(1), C4(2));
71
72      alpha = abs(xi(2))-sigma;
73      beta = abs(xi(1)) + sigma;
74      Mangle = pi/2 - atan(2*rd/h); %angle in disc of corner points to the center of
            rotation
75
76      kappa = 0; %measure in which the body is reflected
77
78      hC34 = r*sin(w(3)-psi0); %height of point C3 w.r.t. center of rotation [m]
79
80      tm = pi*sqrt(m/kp)/2; %approximation of duration of impact of C3 or C4 on the
            material [s]
81      Mit = 0;
82      Mjt = 0;
83      v0 = w(4)*r; %tangential velocity on the material *cos(alpha+Mangle) [m/s]
84      v1 = w(2)*r;
```

```
85      delta0 = sqrt(m/kp)*abs(v0);%sqrt(m/kp)*abs(v0); %small possible deviation,
            assuming liquid layer of 1 mm = atan(0.001/r)
86      delta1 = atan(0.001/r);%sqrt(m/kp)*abs(w(2));%;sqrt(m/kp)*abs(w(2))
87       if(dh3>0)
88           if( dh3-abs(hC34) <= delta0 && (w(3)-psi0)>0)
89              impuls = m * v0*(1 +kappa);
90              Mjt = -impuls*r/tm;%
91              w(4)= -m*v0*(1+kappa)*rd/(I0*abs(cos(w(3)-psi0)));%*abs(cos(w(3)-psi0))
92           end
93       else
94            if(dh3 - hC34<delta0 && w(4)>0)
95              impuls = m * v0*(1 +kappa);
96              Mjt = -impuls*r/tm; %
97              w(4)= -m*v0*(1+kappa)*rd/(I0*abs(cos(w(3)-psi0)));%*abs(cos(w(3)-psi0))
98           end
99       end
100      if(dh4>0)
101          if( abs(dh4)-abs(hC34) <= delta0 && (w(3)-psi0)<0)
102             impuls = m * v0 *(1 +kappa);
103             Mjt = -impuls*r/tm; %
104             w(4)= -m*v0*(1+kappa)*rd/(I0*abs(cos(w(3)-psi0))); %*abs(cos(w(3)-psi0))
105          end
106      else
107          if( dh4+ hC34<delta0 && w(4)<0)
108             impuls = m * v0*(1 +kappa);
109             Mjt = -impuls*r/tm; %
110             w(4)= -m*v0*(1+kappa)*rd/(I0*abs(cos(w(3)-psi0))); %*abs(cos(w(3)-psi0))
111          end
112      end
113
114      if(abs(w(1)-theta0-xi(1))> delta1 && w(2)~= 0) %abs(w(1)-theta0-xi(1))> delta1
             && abs(w(1)-xi(1))> delta1 &&
115             impuls = m * v1*(1 +kappa);
116             Mit = -impuls*r/tm; %
117             w(2) = -m*w(2)*(1+kappa)*(r)*rd/(I0*abs(cos(w(1)-theta0))); %*abs(cos(w
                    (1)-theta0))
118      end
119
120
121     %Constant force:
122     Fp = 10; %Pressure force for polishing, between 10-15 [N]
123     FC2= Fp*sin(beta)/sin(alpha+beta);
124     FC1= (Fp/cos(beta))*(1- cos(alpha)*sin(beta)/sin(alpha+beta));
125
126     Mi = (FC1*cos(beta+Mangle)-FC2*cos(alpha+Mangle))*r + Mit; %
127     Mj = Mjt;
128     Mk = 0;
129     %if(t ==0)
130     %    fileID = fopen('forcemoments.txt','w');
131     %else
132     %    fileID = fopen('forcemoments.txt','a');
133     %end
134     %fprintf(fileID,'%.4f %.4f  %.4f \n',FC2,FC1,Mi);
135     %fclose(fileID);
136
137     dwdt = [w(2);
```

```
138         Mi/I0- I*p*w(4)/I0 ; %+ r^2*w(2)*cp/I0theta
139         w(4);
140         Mj/I0+ I*p*w(2)/I0];
141
142 end
```

D.2. Airbearing ode function

```
1  function dwdt = odeAir8(t,w) %Full airbearing + two angle model for function1
2  %Variables:
3
4      vars = getGlobalVars;
5      p = vars(1);
6      omega = vars(2);
7      kp = vars(3);
8
9      Damping = getGlobalDamp;
10     cA = Damping(1);
11     cD = Damping(2);
12     cz = Damping(3);
13     cm = Damping(4);
14     cAZ = Damping(5);
15
16     %measurements:
17     rd = 0.020*0.5; % radius of brush in [m]
18     rb = 0.003*0.5; %radius of rod between tool and air bearing [m]
19     h = 0.010; %height brush in [m]
20     hb = 0.089; %height of air bearing [m]
21     wb = 0.05; %width of airbearing [m]
22     r = sqrt(rd^2 + (h/2)^2); % = sqrt(0.000125)= 0.01118 [m]
23     Rrot = 0.003; %radius of robot rotation [m]
24     l = 0.020; %length of rod between tool and air bearing [m]
25     A = rb^2*pi; %Area of cross section rod between tool and airbearing [m^2]
26     v = 0.001; %longitude speed vector [m/s]
27     htot=0.5*h+l+0.5*hb; %total length between center of airbearing and center of
           rotation [m]
28
29     %masses:
30     m = 0.2; %mass brush in kg
31     mrod = 0; %mass of rod in kg
32     mbearing = 0.337; %mass air bearing in kg
33     Ibearing = mrod *(l^2/12 + (hb/2)^2)+ m*((4/3)*((h/2)^2) + 4*(rd^2)+((hb/2)^2+l
           ^2))+mbearing*((hb/2)^2+(wb/2)^2); %Moment of inertia assumed from center of
           rotation in air bearing [kg m^2]
34     %masses reduced:
35     mz = mbearing;
36     mx = Ibearing/((hb/2 + l + h/2)^2) -m - mrod; %Not correct?
37
38     %Springconstants:
39     E = 2*10^11 ; % Youngs modulus for steel [N/m^2]
40     Ib = (rb^4)*pi/4; %Cross sectional moment of inertia [m^4]
41     kt = 110*10^6; % translational springconstant air bearing [N/m] (multiplied by
           6?)
42     kx = 3*E*Ib/(l^3); %translational vibration from bending stiffness rod [N/m]
43     kz = E*A/l; %Stiffness of the toolholder and rod [N/m]
```

```
44      kr = 23*10^6; %Torsional stiffness air bearing [Nm/rad]
45      ktot = kt*kr/(kr+kt*(htot^2)); %total spring constant of airbearing [N/m]
46
47      %Moments of inertia:
48      I = 0.5*m*r^2; %+ m*Rrot^2? Moment of inertia around principal axis z [kgm^2]
49      I0 = (1/12)*m*(3*r^2+h^2) ; %+ m*Rrot^2? Moment of inertia around principal axes
            x and y [kgm^2]
50
51      %Angles:
52      mu = 0.5*sqrt(2)*0.5; %Coefficient of friction, not correct probably
53      sigma = atan(mu); %angle of friction
54
55      ltD = getGlobalLt;
56      teind = getGlobalTermt;
57      StepSize= teind/(ltD-1);
58      for i = 1:(ltD-1)
59          if( StepSize*(i-1)<=t && StepSize*i>=t)
60             wD = (t-StepSize*(i-1))*getGlobalWd(i+1)/StepSize + (1-t+StepSize*(i-1))*
                   getGlobalWd(i)/StepSize ;
61          end
62      end
63
64      Center1 = Rrot*cos(omega*t) +v*t+ rd+Rrot ;
65      CenterToolLoc = [Center1 ; Rrot*sin(omega*t)];
66      Vel1 = -omega*Rrot*sin(omega*t) +v;
67      CenterToolVel = [Vel1; omega*Rrot*cos(omega*t)];
68      psi0 = pi/2;
69      rod0 = tube_elevation(CenterToolVel, r+Rrot,0);
70      theta0 = rod0;
71      C1 = rd*abs(cos(wD(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
72      C2 = -rd*abs(cos(wD(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
73      Center3 = (Rrot-rd*abs(cos(wD(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
74      C3 = [Center3; (Rrot-rd*abs(cos(wD(3)-psi0)))*sin(omega*t)]; %Not sure
75      Center4 = (Rrot+rd*abs(cos(wD(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
76      C4 = [Center4; (Rrot+rd*abs(cos(wD(3)-psi0)))*sin(omega*t)]; %Not sure
77      xi= [tube_elevation(CenterToolVel, C1(1), C1(2));
78           tube_elevation(CenterToolVel, C2(1), C2(2));
79           tube_elevation(CenterToolVel, C3(1), C3(2));
80           tube_elevation(CenterToolVel, C4(1), C4(2))];
81      hCR =0.5*(tubeFunction(C1(1),C1(2))+tubeFunction(C2(1) ,C2(2)));
82      dh3 = hCR - tubeFunction(C3(1), C3(2)); %The difference in height from point
            below center of rotation to material [m]
83      dh4 = hCR - tubeFunction(C4(1), C4(2));
84
85      alpha = abs(xi(2))-sigma;
86      beta = abs(xi(1)) + sigma;
87      Mangle = pi/2 - atan(2*rd/h); %angle in disc of corner points to the center of
            rotation
88
89      kappa = 0; %measure in which the body is reflected
90
91      hC34 = r*sin(wD(3)-psi0); %height of point C3 w.r.t. center of rotation [m]
92
93      tm = pi*sqrt(m/kp)/2; %approximation of duration of impact of C3 or C4 on the
            material [s]
94      Fit = 0;
```

```
95      Fjt = 0;
96      v0 = wD(4)*r; %tangential velocity on the material *cos(alpha+Mangle) [m/s]
97      v1 = wD(2)*r;
98      delta0 = sqrt(m/kp)*abs(v0);%sqrt(m/kp)*abs(v0); %small possible deviation,
            assuming liquid layer of 1 micron = atan(1*10^(-6)/r)
99      delta1 = atan(0.001/r);% atan(0.000001/r);sqrt(m/kp)*abs(wD(2));
100      if(dh3>0) %positive psi
101          if( dh3-abs(hC34) <= delta0 && (wD(3)-psi0)>0)
102              impuls = m * v0*(1 +kappa);
103              Fjt = abs(impuls)/tm; %
104          end
105      else %positive psi
106           if(dh3 - hC34<delta0 && wD(4)>0)
107              impuls = m * v0*(1 +kappa);
108              Fjt = impuls/tm; %
109          end
110      end
111      if(dh4>0) %negative psi
112          if( abs(dh4)-abs(hC34) <= delta0 && (wD(3)-psi0)<0)
113              impuls = m * v0 *(1 +kappa);
114              Fjt = -abs(impuls)/tm; %
115          end
116      else %negative psi
117          if( dh4+ hC34<delta0 && wD(4)<0)
118              impuls = m * v0*(1 +kappa);
119              Fjt = -impuls/tm; %
120          end
121      end
122
123      if(abs(wD(1)-theta0-xi(1))> delta1 && w(2)~= 0) % abs(wD(1)-xi(1))> delta1 &&
124              impuls = m * v1*(1 +kappa);
125              Fit = -impuls/tm; %
126      end
127
128     %Constant force:
129     Fp = 10; %Pressure force for polishing, between 10-15 [N]
130     FC2= Fp*sin(beta)/sin(alpha+beta);
131     FC1= (Fp/cos(beta))*(1- cos(alpha)*sin(beta)/sin(alpha+beta));
132     factor = 1; %400
133     Fx = Fjt*sin(alpha)/factor;
134     Fy = Fit*sin(beta)/factor;
135     Fz = Fp+(abs(Fit)*cos(beta) + abs(Fjt)*cos(alpha))/factor;
136     if(w(11)>0)%if there is no contact with the material: no normal force to work
            against gravity and the polishing pressure
137         Fp = Fp + mz*9.81*w(9);
138         Fz = Fz - m*9.81*w(11)-Fp;
139     end
140     w1 = w(2);
141     w2 = -ktot*w(1)/mx + kx*(w(3)-w(1))/mx-(cA+cD)*w(2)/mx + cD*w(4)/mx;
142     w3 = w(4);
143     w4 = -kx*(w(3)-w(1))/m - cA*(w(4)-w(2))/m - Fx/m;
144     w5 = w(6);
145     w6 = -ktot*w(5)/mx + kx*(w(7)-w(5))/mx -(cA+cD)*w(6)/mx + cD*w(8)/mx;
146     w7 = w(8);
147     w8 = -kx*(w(7)-w(5))/m - cA*(w(8)-w(6))/m -Fy/m;
148     w9 = w(10);
```

```
149     w10 = -kz*(w(9)-w(11))/mz - cz*(w(10)-w(12))/mz - cAZ*w(10)/mz - Fp/mz;
150     w11 = w(12);
151     w12 = kz*(w(9)-w(11))/m + cz*(w(10)-w(12))/m -cm*w(12)/m + Fz/m;
152
153     dwdt = [w1; w2; w3; w4; w5; w6; w7; w8; w9; w10; w11; w12];
154
155
156 end
```

D.3. Static state controller ode function

```
1  function dwdt = odeStateControl1(t,w) %Full airbearing + two angle model for
       function1
2  %Variables:
3
4       vars = getGlobalVars;
5       p = vars(1);
6       omega = vars(2);
7       kp = vars(3);
8
9       Damping = getGlobalDamp;
10      cA = Damping(1);
11      cD = Damping(2);
12      cz = Damping(3);
13
14      [A,B,C,B1,C1] = getGlobalSys;
15      Dc = getGlobalStateC;
16
17      %measurements:
18      rd = 0.020*0.5; % radius of brush in [m]
19      h = 0.010; %height brush in [m]
20      r = sqrt(rd^2 + (h/2)^2); % = sqrt(0.000125)= 0.01118 [m]
21      Rrot = 0.003; %radius of robot rotation [m]
22      v = 0.001; %longitude speed vector [m/s]
23
24      %masses:
25      m = 0.2; %mass brush in kg
26      %Angles:
27      mu = 0.5*sqrt(2)*0.5; %Coefficient of friction, not correct probably
28      sigma = atan(mu); %angle of friction
29
30      ltD = getGlobalLt;
31      teind = getGlobalTermt;
32      StepSize= teind/(ltD-1);
33      for i = 1:(ltD-1)
34          if( StepSize*(i-1)<=t && StepSize*i>=t)
35              wD = (t-StepSize*(i-1))*getGlobalWd(i+1)/StepSize + (1-t+StepSize*(i-1))*
                    getGlobalWd(i)/StepSize ;
36          end
37      end
38
39      Center1 = Rrot*cos(omega*t) +v*t+ rd+Rrot ;
40      CenterToolLoc = [Center1 ; Rrot*sin(omega*t)];
41      Vel1 = -omega*Rrot*sin(omega*t) +v;
42      CenterToolVel = [Vel1; omega*Rrot*cos(omega*t)];
```

```
43      psi0 = pi/2;
44      rod0 = tube_elevation(CenterToolVel, r+Rrot,0);
45      theta0 = rod0;
46      C1 = rd*abs(cos(wD(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
47      C2 = -rd*abs(cos(wD(1))-theta0)*CenterToolVel/norm(CenterToolVel)+CenterToolLoc;
48      Center3 = (Rrot-rd*abs(cos(wD(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
49      C3 = [Center3; (Rrot-rd*abs(cos(wD(3)-psi0)))*sin(omega*t)]; %Not sure
50      Center4 = (Rrot+rd*abs(cos(wD(3)-psi0)))*cos(omega*t) + rd+Rrot +v*t;
51      C4 = [Center4; (Rrot+rd*abs(cos(wD(3)-psi0)))*sin(omega*t)]; %Not sure
52      xi= [tube_elevation(CenterToolVel, C1(1), C1(2));
53           tube_elevation(CenterToolVel, C2(1), C2(2));
54           tube_elevation(CenterToolVel, C3(1), C3(2));
55           tube_elevation(CenterToolVel, C4(1), C4(2))];
56      hCR =0.5*(tubeFunction(C1(1),C1(2))+tubeFunction(C2(1) ,C2(2)));
57      dh3 = hCR - tubeFunction(C3(1), C3(2)); %The difference in height from point
            below center of rotation to material [m]
58      dh4 = hCR - tubeFunction(C4(1), C4(2));
59
60      alpha = abs(xi(2))-sigma;
61      beta = abs(xi(1)) + sigma;
62      Mangle = pi/2 - atan(2*rd/h); %angle in disc of corner points to the center of
            rotation
63
64      kappa = 0; %measure in which the body is reflected
65
66      hC34 = r*sin(wD(3)-psi0); %height of point C3 w.r.t. center of rotation [m]
67
68      tm = pi*sqrt(m/kp)/2; %approximation of duration of impact of C3 or C4 on the
            material [s]
69      Fit = 0;
70      Fjt = 0;
71      v0 = wD(4)*r; %tangential velocity on the material *cos(alpha+Mangle) [m/s]
72      v1 = wD(2)*r;
73      delta0 = sqrt(m/kp)*abs(v0);%sqrt(m/kp)*abs(v0); %small possible deviation,
            assuming liquid layer of 1 micron = atan(1*10^(-6)/r)
74      delta1 = atan(0.001/r);% atan(0.000001/r);sqrt(m/kp)*abs(wD(2));
75       if(dh3>0) %positive psi
76           if( dh3-abs(hC34) <= delta0 && (wD(3)-psi0)>0)
77              impuls = m * v0*(1 +kappa);
78              Fjt = abs(impuls)/tm;
79           end
80       else %positive psi
81            if(dh3 - hC34<delta0 && wD(4)>0)
82              impuls = m * v0*(1 +kappa);
83              Fjt = impuls/tm;
84           end
85       end
86       if(dh4>0) %negative psi
87           if( abs(dh4)-abs(hC34) <= delta0 && (wD(3)-psi0)<0)
88              impuls = m * v0 *(1 +kappa);
89              Fjt = -abs(impuls)/tm;
90           end
91       else %negative psi
92           if( dh4+ hC34<delta0 && wD(4)<0)
93              impuls = m * v0*(1 +kappa);
94              Fjt = -impuls/tm;
```

```
95          end
96      end
97
98      if(abs(wD(1)-theta0-xi(1))> delta1 && w(2)~= 0) % abs(wD(1)-xi(1))> delta1 &&
99              impuls = m * v1*(1 +kappa);
100             Fit = -impuls/tm;
101     end
102
103    %Constant force:
104    Fp = 10; %Pressure force for polishing, between 10-15 [N]
105    FC2= Fp*sin(beta)/sin(alpha+beta);
106    FC1= (Fp/cos(beta))*(1- cos(alpha)*sin(beta)/sin(alpha+beta));
107    factor = 1; %400
108    Fx = Fjt*sin(alpha)/factor;
109    Fy = Fit*sin(beta)/factor;
110    Fz = Fp+(abs(Fit)*cos(beta) + abs(Fjt)*cos(alpha))/factor;
111    forces = [Fx;Fy;Fz;Fp];
112
113    dwdt = (A+B*Dc*C)*w+B1*forces;
114
115
116 end
```

D.4. Calculation of H-infinity norm

```
1  function [Dc,gamma] = H_inf_ssf(A,B,B1,C1,D1)
2      %Minimize the H_inf norm to find the controller Dc for static state
3      %feedback
4      %Does there need to be a optimal solution or just feasible?
5      %define sizes:
6      n = size(A,1);
7      m = size(B,2);
8      k = size(B1,2);
9      l = size(C1,1);
10
11     %Define semidefinite programming variables
12     Y = sdpvar(n,n);
13     M = sdpvar(m,n);
14     gamma2 = sdpvar(1);
15
16     %Formulate constraints
17     F = [gamma2>0, Y>0, [(A*Y+B*M)+(A*Y+B*M)' B1 Y*(C1'); B1' -gamma2*eye(k) D1';
           C1*Y D1 -eye(l)]<0];
18
19     %Optimize (possibly extra options needed)
20     optimize(F,gamma2);
21     %save solutions for M Y and gamma2
22     Mfeasible = value(M);
23     Yfeasible = value(Y);
24     gamma2Feas = value(gamma2);
25     gamma = sqrt(gamma2Feas);
26
27     Dc = Mfeasible*inv(Yfeasible);
28
29     %possible check for negative eigenvalues
```

```
30
31 end
```

Bibliography

[1] M. Abderrahim and A.R. Whittaker. Kinematic model identification of industrial manipulators. *Robotics and Computer Integrated Manufacturing*, 16:1–8, 2000.

[2] T. Alban and H. Janocha. Dynamic calibration of industrial robots with inertial measurement systems. In *European control conference (ECC)*, 1999.

[3] C. Alici and B. Shirinzadeh. A systematic technique to estimate positioning errors for robot accuracy improvement using laser interferometry based sensing. *Mechanism and machine theory*, 40:879–906, 2005.

[4] S. Aoyagi, A. Kohama, Y. Nakata, Y. Hayano, and M. Suzuki. Improvement of robot accuracy by calculating kinematic model using a laser tracking system, compensation of non-geometric errors using neural networks and selection of optimal measuring points using genetic algorithm. In *IEEE/RJS international conference on intelligent robots and systems*, 2010.

[5] G. Baumann. *Mathematica for theoretical physics; classical mechanics and nonlinear dynamics*. Springer Science+Business Media, Inc, 2005.

[6] C.F. Beard. *Vibration and control systems*. Ellis Horwood Limited, 1988.

[7] New Way Air Bearings. Specification sheet of part nr. s305002. Specifications, 2012.

[8] R. Bernhardt and S.L. Albright. *Robot calibration*. Chapman & Hall, 1993.

[9] F. Caccavale and P. Chlacchio. Identification of dynamic parameters and feedforward control for a conventional industrial manipulator. *Control engineering practice*, 2(6):1039–1050, 1994.

[10] J.L. Caenen and J.C. Angue. Identification of geometric and non geometric parameters of robots. In *IEEE international conference on robotics and automotion*, 1990.

[11] G. Calafiore, M. Indri, and B. Bona. Robot dynamic calibration: optimal excitation trajectories and experimental parameter estimation. *Journal of robotic systems*, 18(2):55–68, 2001.

[12] P.H. Chang and H.S. Park. Time-varying input shaping technique applied to vibration reduction of an industrial robot. *Control Engineering Practice*, 13:121–130, 2005.

[13] I.M. Chen, G. Yang, C.T. Tan, and S.H. Yeo. Local poe model for robotic kinematic calibration. *Mechanism and machine theory*, 36:1215–1239, 2001.

[14] V. Duindam, A. Maccheli, S. Stramigioli, and H. Bruyninckx. *Modeling and control of complex physical systems; the port-Hamiltonian approach*. Springer-Verlag Berlin Heidelberg, 2009.

[15] A.Y. Elatta, L.P. Gen, F.L. Zhi, Y. Daoyuan, and L. Fei. An overview of robot calibration. *Information Technology Journal*, 3(1):74–78, 2004.

[16] M. Fogiel, Research, and Education Association. *The Essentials of Mechanics I*. Research and Education Association, 1989.

[17] K.S. Fu, R.C. Gonzalez, and C.S.G. Lee. *Robotics: control, sensing and intelligence*. Springer-Verlag London, 2008.

[18] M. Gautier and W. Khalil. Exciting trajectories for the identification of base inertial parameters of robots. *The international journal of robotics research*, 11(4):362–375, 1992.

[19] C. Gong, J. Yuan, and J. Ni. Nongeometric error identification and compensation for robotic system by inverse calibration. *International Journal of Machine Tools and Manufacture*, 40:2119–2137, 2000.

[20] D. Gross, W.A. Wall, W. Hauger, N. Rajapakse, and J. Schröder. *Engineering Mechanics 1 statics, 2nd edition.* Springer Science+Business Media Dordrecht, 2013.

[21] T. Haidegger, B. Benyó, L. Kovács, and Z. Benyó. Force sensing and force control for surgical robots. In *7th IFAC symposium on modelling and control in biomedical systems*, pages 401–406, 2009.

[22] R.A. Howland. *Intermediate dynamics, a linear algebraic approach.* Springer Science+Business Media, 2006.

[23] D.J. Inman. *Engineering vibrations, third edition.* Pearson Education, Inc, 2009.

[24] J.H. Jang, S.H. Kim, and K. Kwak. Calibration of geometric and non-geometric errors of an industrial robot. *Robotica*, 19:311–321, 2001.

[25] D. Jeltsema and J.M.A. Scherpen. Multidomain modeling of nonlinear networks and systems. *IEEE control systems magazine*, 29(4), 2009.

[26] J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[27] A.M. Lopes and F.G. Almeida. Acceleration-based force-impedance control of a six-dof parallel manipulator. *Industrial robot: an international journal*, 34(5):386–399, 2007.

[28] *Mitsubishi Industrial Robot RV-6SD/6SDL Series standard specifications manual (CR2D-711/CR3D-711M Controller).* Mitsubishi electric corporation, 5th edition, 2008.

[29] B.W. Mooring, Z.S. Roth, and M.R. Driels. *Fundamentals of manipulator calibration.* John Wiley & Sons, 1991.

[30] R.M. Murray, Z. Li, and S.S. Sastry. *A mathematical introduction to robotic manipulation.* CRC Press, 1994.

[31] H. Nguyen, J. Zhou, and H. Kang. A calibration method for enhancing robot accuracy through integration of an extended kalman filter algorithm and an artificial neural network. *Neurocomputing*, 151: 996–1005, 2015.

[32] S.B. Niku. *Engineering Principles in Everyday Life for Non-Engineers.* Morgan and Claypool Publishers, 2016.

[33] K. Okamura and F.C. Park. Kinematic calibration using the product-of-exponentials formula. *Robotica*, 14:415–421, 1996.

[34] G.J. Olsder, J.W. van der Woude, J.G. Maks, and D. Jeltsema. *Mathematical systems theory, 4th edition.* VSSD, 2011.

[35] A. Omodei, G. Legnani, and R. Adamini. Three methodologies for the calibration of industrial manipulators: experimental results on a scara robot. *Journal of robotic systems*, 17(6):291–307, 2000.

[36] B. Pal and B. Chaudhuri. *Robust control in power systems.* Springer Science+Business Media, 2005.

[37] I.W. Park and J.H. Kim. Estimating entire geometric parameter errors of manipulator arm using laser module and stationary camera. In *IECON 37th anual conference on IEEE industrial electronics society*, 2011.

[38] J.M. Renders, E. Rossignol, M. Becquet, and R. Hanus. Kinematic calibration and geometrical parameter identification for robots. *IEEE transactions on robotics and automation*, 7(6):721–732, 1991.

[39] D.G.G Rosa, J.F.S. Feiteira, P.A.F. de Abreu, and A.M. Lopes. Robotic system with force control for drilling operations. In *23th ABCM international congress of mechanical engineering*, 2015.

[40] Applications Laboratory SBT. Report 54 lapping and polishing basics. www.southbaytech.com.

[41] C. Scherer and S. Weiland. Linear matrix inequalities in control. lecture notes, 2004.

[42] K. Schröer. *Handbook on robot performance testing and calibration.* Fraunhofer IRB Verlag, 1998.

[43] M.W. Spong, S. Hutchinson, and M. Vidyasagar. Robot dynamics and control, second edition. lecture notes, 2004.

[44] G. Szuladzinski. *Formulas for Mechanical and Structural Shock and Impact.* CRC Press, 2009.

[45] J.W. van der Woude. Advanced system theory wi4226, first part. lecture notes, 2016.

[46] H.V. Vu and R. S. Esfandiari. *Dynamic systems; Modeling and analysis.* The McGraw-Hill Companies, Inc, 1998.

[47] C. Vuik, P. van Beek, F. Vermolen, and J. van Kan. *Numerieke methoden voor differentiaalvergelijkingen.* VSSD, 2006.

[48] D. Wang, Y. Bai, and J. Zhao. Robot manipulator calibration using neural network and a camera-based measurement system. Transactions of the institute of measurement and control, 2010.

[49] D.E. Whitney, C.A. Lozinski, and J.M. Rourke. Industrial robot forward calibration and result. *Journal of dynamic systems, measurement and control*, 108:1–8, 1986.

[50] J. Wu, J. Wang, and Z. You. An overview of dynamic parameter identification of robots. *Robotics and computer-integrated manufacturing*, 26:414–419, 2010.

[51] G. Zak, B. Benhabib, R.G. Fenton, and I. Saban. Application of the weighted least squares parameter estimation method to the robot calibration. In *Transactions of the ASME*, volume 116, pages 890–893, 2009.

[52] G. Zeng and A. Hemami. An overview of force control. *Robotica*, 15:473–482, 1997.