ACTIVE COMPLIANCE CONTROL STRUCTURE DESIGN FOR A ROBOTIC-GRINDING MACHINE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ABDÜLHAMİT DÖNDER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

SEPTEMBER 2017

Approval of the thesis:

ACTIVE COMPLIANCE CONTROL STRUCTURE DESIGN FOR A ROBOTIC-GRINDING MACHINE

Submitted by **ABDÜLHAMİT DÖNDER** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tuna Balkan
Head of Department, **Mechanical Engineering** _____

Assoc. Prof. Dr. İlhan Konukseven
Supervisor, **Mechanical Engineering Dept., METU** _____

Examining Committee Members:

Asst. Prof. Dr. Ali Emre Turgut
Mechanical Engineering Dept., METU _____

Assoc. Prof. Dr. E. İlhan Konukseven
Mechanical Engineering Dept., METU _____

Assoc. Prof. Dr. Murat Demiral
Mechanical Engineering Dept., Çankaya University _____

Asst. Prof. Dr. Ulaş Yaman
Mechanical Engineering Dept., METU _____

Asst. Prof. Dr. Yiğit Taşçıoğlu
Mechanical Engineering Dept., TOBB-University of Economics and Technology _____

Date: 07.09.2017

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Abdülhamit DÖNDER

Signature:

ABSTRACT

ACTIVE COMPLIANCE CONTROL STRUCTURE DESIGN FOR A ROBOTIC-GRINDING MACHINE

Abdülhamit DÖNDER
M.Sc., Department of Mechanical Engineering
Supervisor: Assoc. Prof. Dr. E. İlhan KONUKSEVEN

September 2017, 115 Pages

Grinding operation has an advantage of precise form shaping in machining processes. However, if the surface profile is not known before the machining process, it is hard to obtain an accurate surface profile using a grinding operation. In this work, a novel method to compensate the form shaping errors in grinding operations due to the lack of a priori knowledge of the surface profile will be presented. Grinding operation on a workpiece with an unknown surface profile is aimed. Compliance force control is implemented by means of admittance control in two degrees of freedom using a piezo actuator and a hexapod parallel manipulator. The desired force interaction between the tool and the workpiece was achieved by imposing an offset from the preset depth of cut. Additionally, tool deflection due to the grinding forces of the robotic grinding setup is taken into consideration. The deflections are computed from the grinding forces in real time and the compensation is performed by the hexapod robot in six degrees of freedom. Based on the literature review, this is the first study in which grinding on a workpiece with an unknown surface profile was performed while tool deflection due to the grinding forces was compensated. Two different control algorithms namely PID control and Active Disturbance Rejection Control were tested on a robotic grinding setup and the experiment results are discussed.

Keywords: Robotic Grinding; Active Compliance Control; Force Control; Admittance Control; Piezo Actuator; Precision Machining; Negative Compensation; PID; Angle Compensation; Tool Deflection; ADRC

ÖZ

ROBOTİK-TAŞLAMA MAKİNESİ İÇİN AKTİF UYUM KONTROLCÜ TASARIMI

Abdülhamit DÖNDER
Yüksek Lisans, Makine Mühendisliği Bölümü
Tez yöneticisi: Doç. Dr. E. İlhan KONUKSEVEN

Eylül 2017, 115 Sayfa

Taşlama işlemi hassas yüzey elde etme amacıyla yapılan bir talaşlı imalat yöntemidir. Yüzey formunun önceden bilinmediği durumlarda taşlama işlemi ile hassas yüzey elde etmek robotik taşlama için zor bir süreçtir. Bu çalışmada, yüzey formunun önceden bilinmemesinden dolayı oluşan form şekillendirme hatalarının aktif uyum kontrolcüleri ile telafi yöntemleri üzerinde durulmuş, şekli bilinmeyen bir iş parçasının robot tarafından özgün bir yöntem ile taşlanabilmesi hedeflenmiştir. Kuvvet uyum kontrolü; bir tanesi yüksek frekansta hareket edebilen piezo eyleyici, diğeri hekzapod robot ile kontrol edilen 2 serbestlik derecesindeki kontrolcü ile sağlanmıştır. Ek olarak, taşlama kuvvetlerinden dolayı oluşan kesici takım sehimi ve açısı da dikkate alınmıştır. Oluşan sehim ve açı farkı gerçek zamanlı olarak kuvvet geribeslemesi ile hesaplanıp, 6 serbestlik derecesine sahip hekzapod robot ile kompanzasyon sağlanmıştır. Yapılan literatür taramasına göre, bu çalışma, açı kompanzasyonu uygulanırken şekli bilinmeyen bir numunenin taşlama işleminin gerçekleştirildiği ilk çalışmadır. PID ve aktif bozucu giriş engelleme kontrolcüsü olmak üzere iki farklı kontrolcü denenmiş, robotik taşlama düzeneğinde yapılan deneyler ve sonuçlar üzerindeki tartışma sunulmuştur.

Anahtar Kelimeler: Robotik Taşlama; Aktif Uyum Kontrolü; Kuvvet Kontrolü; Admitans Kontrol, Piezo Eyleyici; Hassas İşleme; Negatif Kompanzasyon; PID; Açı Kompanzasyonu; Takım eğilmesi; ADRC

To my parents
and my brother

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere appreciation to my supervisor Assoc. Prof. Dr. E. İlhan Konukseven for his invaluable guidance, advice, and criticism and especially his extreme support that made this study possible.

I am thankful to Prof. Dr. Mehmet Çalışkan and Asst. Prof. Dr. Kutluk Bilge Arıkan for their valuable guidance and inspiring comments.

I would like to express my thanks to my dear friends and colleagues Kemal Açıkgöz, Payam Parvizi, Masoud Latifi-Navid, Ömer Okumuş, Musab Çağrı Uğurlu and Uğur Mengilli for their friendship and technical support throughout the thesis period.

I am grateful to my lovely family for their support, love and encouragement through all my life.

Last, but not least, I thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for their financial support:

- 2211 Yurt İçi Lisansüstü Burs Programı.
- Bilimsel ve Teknolojik Araştırma Projelerini Destekleme Programı – 114E274

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

ADRC	Active Disturbance Rejection Control
CAD	Computer Aided Design
CBN	Cubic Boron Nitride
$\hat{C}^{(a,b)}$	Transformation matrix between reference frames a and b
DoF	Degree of Freedom
e	error
$e(t)$	Gaussian noise
E	Modulus of elasticity
F_a, F_p, F_q, F_b	a, p, q, b reference frames
F_{int}	Interaction Force
F_{ref}	Reference Force
F_{t_i}	Local tangential force
F_{n_i}	Local normal force
F_{nr}	Reference normal force
F_X	Measured force in X direction
F_Y	Measured force in Y direction
f_R	feedrate in tangential direction
F/T Sensor	Force Torque Sensor
GUI	Graphical User Interface

$\widehat{H}_{FK}^{(a)}$	Transformation homogenous matrix for forward kinematics
$\widehat{H}_{HexapodNew}$	Transformation homogenous matrix for new hexapod position
$\widehat{H}_{desiredTool}$	Desired transformation homogenous matrix for the tool
I_y	Moment of inertia with respect to Y axis
K	Proportional Gain
L	Tool length
M_A	Reaction moment between tool and the spindle
$M_{zSpindle}$	Measured moment around Z axis of the spindle
md	Moving Direction
P	Grinding interaction force
PID	Proportional – Integral - Derivative
R_x	Reaction force in x direction between tool and spindle
$\widehat{R}_X$	Rotation matrix around x axis
$\widehat{R}_Y$	Rotation matrix around y axis
$\widehat{R}_{CMM}^{(a)}$	Rotation matrix between the moving plate of the hexapod robot and the spindle tip
$\widehat{R}_{Forward}$	Forward kinematics rotation matrix
rpm	Revolution per Minute
r_{tool}	Radius of the cutting tool
r_x, r_y and r_z	Distances from moving plate of the hexapod to spindle tip
T_I	Integral Time Constant
T_D	Derivative Time Constant

$T_{Forward}$	Tool forward kinematics translation
u	Controller Output
$\vec{u}_i^{(x)}$	i^{th} unit vector of reference frame x
V_{Hex}	Velocity of the hexapod
V_{Pzo}	Velocity of the piezo actuator
α_t	Rotation angle around x axis
α	Level of significance
β_t	Rotation angle around y axis
q^{-1}	Delay operator
Δy	Eccentricity of the force/torque sensor with respect to spindle axis in Y direction
Δx	Eccentricity of the force/torque sensor with respect to spindle axis in X direction
δt	Tool deflection in tangential direction
δn	Tool deflection in normal direction

CHAPTER 1

INTRODUCTION

Together with the occurrence of Industry 4.0 concept, the necessity for frequent changes in the produced parts has started to change the structure of the manufacturing systems. Due to the increased demand on customized products, the studies related to adaptive machining centers have gained recognition for the last two decades. In particular, machining of a work-piece with an unknown shape is one of the main concern of modern-day researchers since significant part of the overall cost is allocated for extracting computer aided design (CAD) models of the work-pieces and path planning studies.

Even if the CAD model of the work-piece is available, most of the time, it is hard to perform good calibration of the work-piece and the robot [1]. Additionally, due to the finite stiffness of the robotic grinding systems, the angle of the tool is affected according to the grinding forces. This is one of the major reasons for unqualified surface finishes.

In this study, grinding of a work-piece with an unknown shape was investigated, admittance control based active compliance controller was developed and implemented on the robotic – grinding setup. For implementation purposes a piezo actuator was added to the robotic grinding system. Tool angle compensation is also implemented in real-time.

For control structure, two different methods namely proportional - integral – derivative (PID) and active disturbance rejection control (ADRC) were utilized and compared. The optimization of controller parameters was done by genetic algorithm. System

identification techniques were utilized in order to estimate the system model parameters.

1.1 Automated Grinding Systems

Grinding operation is one of the most important processes in order to obtain smooth surfaces. It is an abrasive machining process. However, in manual grinding, the quality of the operation mostly depends on the skills of the operator. Therefore, automated grinding systems have gained recognition in the last decades.

The developments in path planning algorithms and control systems paved the way for robotic grinding systems.

CNC machining centers with high stiffness are commonly used in industry for grinding purposes. Their disadvantage is relatively small working ranges and high investment costs compared to industrial serial manipulators. However, the disadvantage of industrial serial manipulators is that they have relatively low machine stiffness correspondingly low accuracy[2]. Another kind of manipulator used in robotic machining is parallel manipulators. Even though their precision characteristics are better compared to serial manipulators, the limited working area they have is one of the most important drawbacks of them.

The combination of parallel and serial manipulators in robotic grinding was first proposed in [3]. This structure offers both high reachability and high precision.

In the last decade, with the advance of piezo-electric technology, piezo-actuators started to be seen in some robotic grinding applications where high frequency and precise movements are needed.

1.2 Motivation

Due to the difficuty of manual grinding operation, automated or robotized grinding cells have started to be seen in the industry. Although even an unexperienced person

can learn how to perform grinding at a certain level very fast, the robotization of this process is challenging.

In robotic - grinding, most of the robot control systems in industry require the work-piece shape or end-effector path, which can be obtained by means of off-line programming, CAD models etc.

One of the ways of reducing the task programming phase which covers the important part of the overall cost is the bringing the robotic grinding system in capability to cope with a work-piece with an unknown shape. The realization of this is significant especially when frequent changes occur in production.

One of the places where frequent changes occur in production is water jet cutting companies. Due to the nature of water jet, the cut surface is left as uneven. Therefore, these porducts are machined in milling machines after cutting operation. Once the materal is removed from water jet table, in order to put it on a milling table again, calibration between the machine and the workpiece should be performed. However, most of the time, perfect calibration cannot be reached in exchange for limited time. With the proposed approach in this work, this calibration procedure is eliminated and, most importantly, time which was spent unnecessarily will be saved.

Additionally, the burr locations are generally unknown after machining. For instance, after moulding and milling, in order to remove the burrs, proposed approach in this study, can be utilized as well.

1.3 Grinding process forces

As in the other machining applications, the generated force components in grinding has one of the most important effects in terms of surface quality. These forces are generally dependent on the quality of the cutting tool, the material of the work-piece,

spindle speed, depth of cut and feedrate. Additionally, the wear of the tool gradually affects the process forces.

As it is seen in Figure 1, the direction which is tangent to the surface is called tangential direction, and the direction perpendicular to tangential direction is called the normal direction. Therefore, grinding force in normal direction is called normal force, similarly the force in tangential direction is called tangential force.

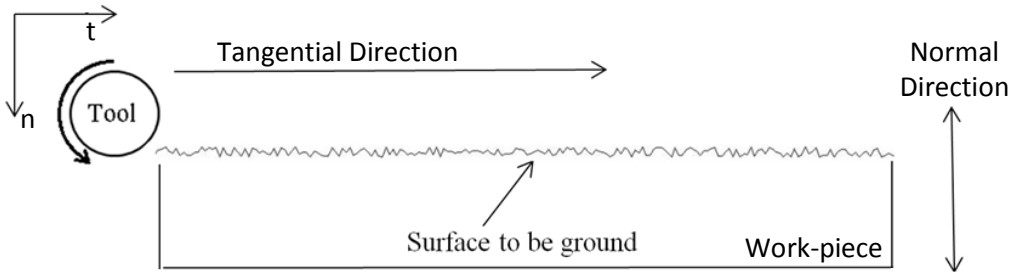


Figure 1 - Illustration of the grinding operation on a flat surface

In Figure 2 red circle represents the cutting tool, "md" is short for moving direction of the tool. Tangential force is shown by F_{t_i} and the normal force is shown by F_{n_i}.

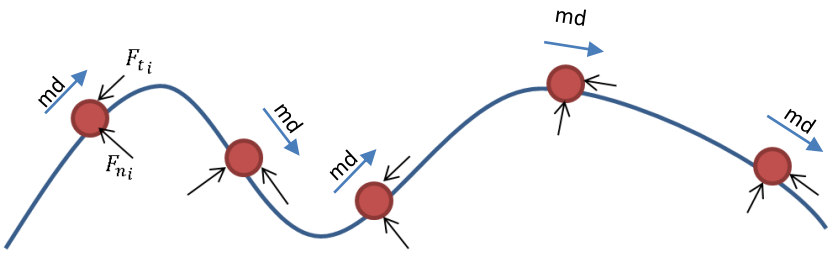


Figure 2 - Illustration of the grinding operation on a curvy surface

1.4 Force Control in Robotic Grinding

When a compliant relative motion of the work-piece and the tool is desired, force control is a kind of control strategy that can be encountered frequently.

The system is called a compliant system when the end effector trajectory is modified based on online sensor information during the process [4]. In order to apply, for instance, a constant normal force, an active compliant system is needed.

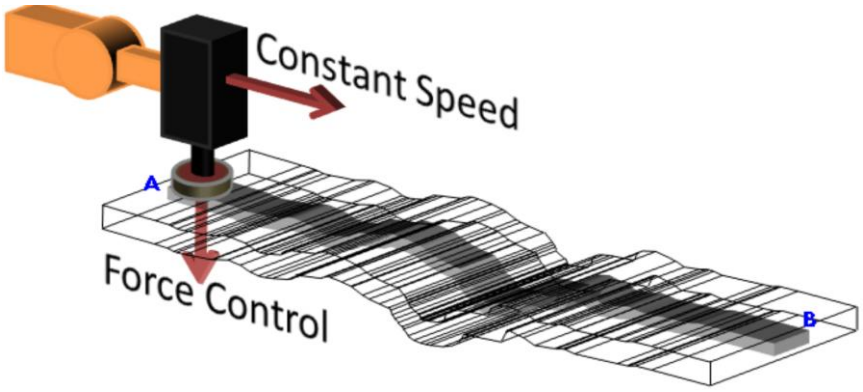


Figure 3 - Linear motion with applied force control[5]

Grinding interaction forces can affect the temperature distribution, tool wear, efficiency, material removal, therefore controlling the grinding forces is one of the ways of determining the machining quality.

One of the advantages of force control is that the force controlled robotic grinding system can track the unknown surface by trying to keep the interaction force constant (Figure 3). By doing so, grinding operation of a work-piece with an unknown shape can be performed which can reduce the task programming phase that is the main motivation of this thesis.

Another advantage of force controlled grinding is that it prevents the work-piece to be ground with reduced material removal rate as wheel wear occurs [6]. In [7] it was shown that the force control technique can reduce the average grinding force and grinding force variation. Additionally, as stated in [8] force controlled grinding requires less stringent calibration.

1.5 Machine tool stiffness: Tool deflection compensation

Grinding with constant normal force and constant tangential velocity is a well-known approach for increasing the operation accuracy and getting constant depth of cut and surface quality along work-piece. However, the mentioned approach is effective when using universal grinding machines that are stiffer than CNC type machines and the deflection of the tool and setup is negligible. In the case of robotic grinding, the stiffness of the robot and setup is approximately 30 times lower than CNC type grinding machines (this was concluded by comparing [9] and [10]). Consequently, there are considerable tool and setup deflections which have significant effect on the grinding forces. During grinding with CNC type machines when there is a flat work-piece profile and if the grinding parameters (depth of cut, spindle speed and feedrate) are constant, the grinding normal and tangential forces are expected to be constant either. But in the robotic grinding due to lower stiffness and tool-setup deflection, the grinding forces can show three different characteristics through the work-piece profile even when the grinding parameters are constant and the work-piece has flat surface profile. The mentioned three characteristics are classified in three regimes in [11]. In the first regime the grinding forces remain almost constant because the tool is able to cut the work-piece with set feedrate. In the second regime there is an almost linear increase in grinding forces because the tool cannot cut the work-piece with set feedrate and consequently tool deflection happens. In the third regime a transition between regime 1 and regime 2 happens where small tool deflection occurs followed by immediate compensation.

The mentioned difference between characteristics of the robotic grinding and CNC type grinding shows the effect of tool deflection and setup stiffness on normal and tangential forces behaviors. In grinding operation with force feedback, commonly the force sensor is mounted behind the spindle or underneath the work-piece. If tool deflection happens, an misalignment occurs between tool tip reference frame and the force sensor reference frame as shown in Figure 4.

In this case the measured normal and tangential grinding forces by the sensor are not grinding forces of the tool reference frame because of the mentioned misalignments. That's why in this study, compensation of these misalignment is considered as well.

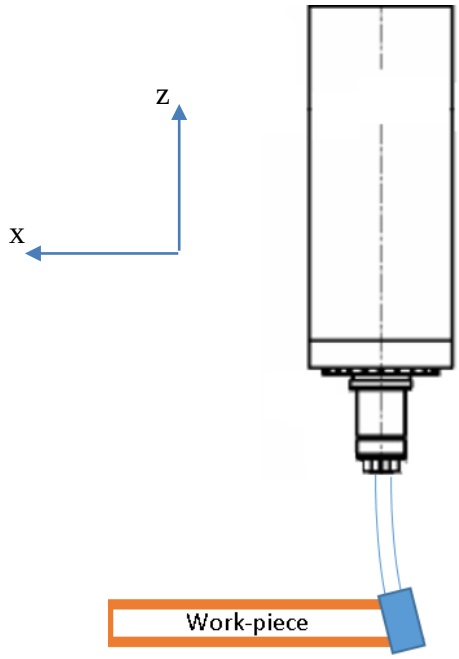


Figure 4 - Illustration of the deformation of a grinding tool

1.6 CBN Tools

CBN abrasive mounted bits (Figure 5) are frequently preferred in precision robotic grinding due to the following properties they have:

- Excellent wear resistance
- Heat dissipation
- CBN tools do not require frequent dressing operation as tool wear occurs,
- They do not require the usage of coolant [12].
- CBN grains are much harder than aluminium oxide and silicon carbide grains [12]

These are why CBN tools were used as the machining tool in the scope of this thesis.

Figure 5 - Cylindrical CBN tools [13]

1.7 Organization of the Thesis

The contents of the chapters are as follows:

Chapter 2 provides a literature survey of force control in robotic grinding, researches considering tool deflection, piezo actuators in machining and hybrid force/velocity control.

Chapter 3 explains the used experimental setup and measurement setup which was build in the scope of this thesis.

In chapter 4, two DoF hybrid velocity force control structure is discussed. In this control structure normal force and tangential velocity was tried to be kept constant and in this study this method was implemented on the robotic grinding setup.

In chapter 5, the method for modelling and optimization of controller parameters is explained and the SIMULINK model for simulation is discussed. The unknown surface profile was modeled as sinusoidal shape. Genetic algorithm was utilized for the optimization of controller parameters.

In chapter 6, the tool deflection compensation method which was developed for the hexapod robot grinding tool was given. In order to measure the amount of deflection cantilever beam theory was utilized. The tool compensation is performed in two axes.

In chapter 7, conducted experiments are explained and the used SIMULINK models together with the optimized parameters are given.

In chapter 8, the results of the conducted experiment are shown. Additionally, surface form measurements of the ground sample are given.

Finally, chapter 9 presents the discussion and the conclusion.

CHAPTER 2

LITERATURE SURVEY

2.1 Force Control in Grinding Operations

It is well known that one of the effects of the force variation is surface roughness in grinding operation[7]. Force data coming from the interaction of the tool and the workpiece is the main source of information [14]. Ref. [15] is the first study which proposes force controlled grinding.

Explicit force control which is the strategy used in this thesis, keeps the inner position control loop and implements admittance control. According to the difference between reference grinding force and the measured grinding interaction force, suitable motion of the manipulator is performed in order to obtain the desired force. Inner position loop improves the stability[16]. In contrast to this, the position is extracted from the measured force in implicit force control. Since the desired position is known, consequently the distance the robot should cover is known. Therefore, the manipulator gives the appropriate movement which corrects the current robot position[17].

In[7], a force control system for a CNC machining center was designed to reduce the grinding force variation and surface roughness. The system includes an electric hand grinder mounted on a CNC machining center, a force sensor to measure the normal grinding force, and a force control sub-system to adjust the grinding depth. The system is shown in Figure 6. Constant normal force control technique was developed by the authors.

In[17], the authors investigated simple position based force control algorithms for an industrial robot and proposed a proportional controller with positive position feedback. The grinding system they utilized is shown in Figure 7.

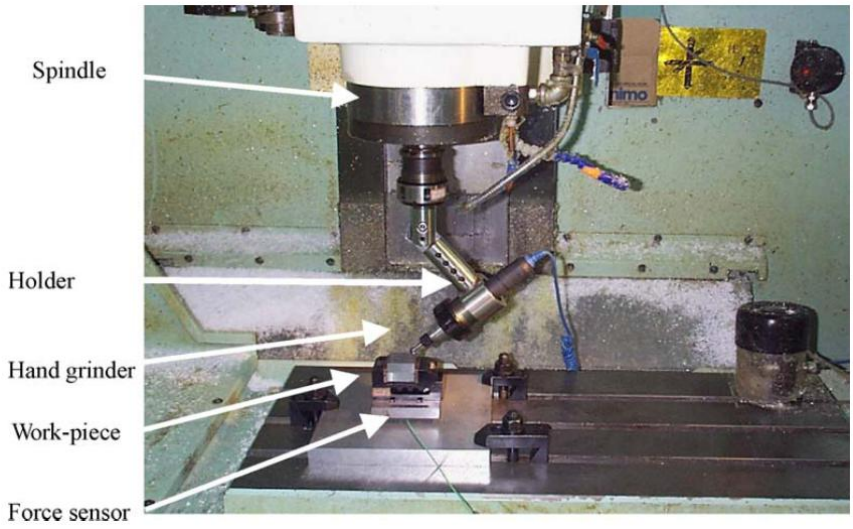


Figure 6 - Grinding system [7]

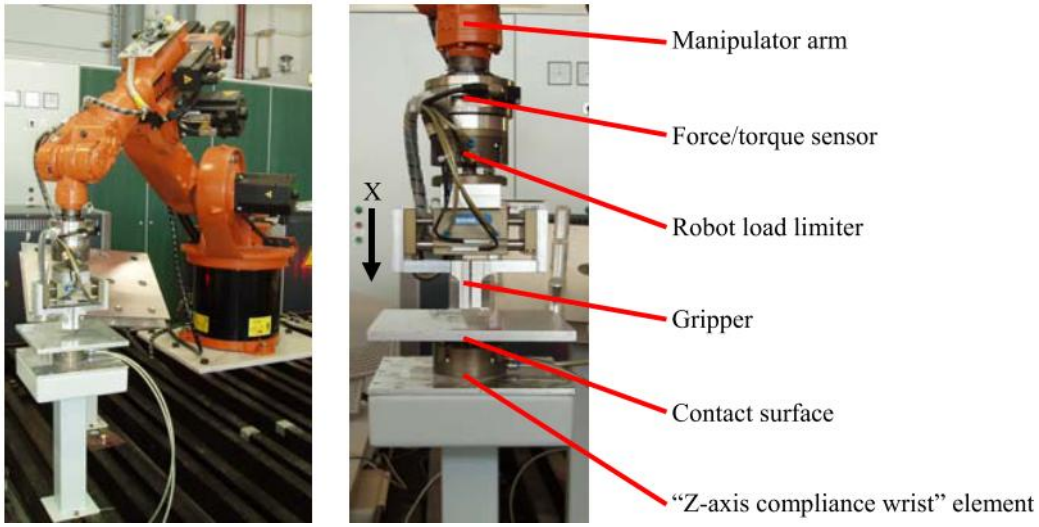


Figure 7 - Grinding system [17]

In [18], force dependent feed-rate control and orthogonal force (pressure) control was studied. In [19], a control strategy was studied in which the goal is to simultaneously track the desired motion in tangential direction and regulate the desired force normal

to the surface. In [20], an adaptive force control based deburring algorithm was developed. This algorithm is used to maintain the interaction between the work-piece and the deburring tool. The algorithm aims to make the normal force and normal velocity equal to zero. Throughout this adaptive algorithm, big burrs can be efficiently removed and damage to the work-piece under unexpected conditions can be avoided. In [21], the proposed controller allows the achievement of the decoupling of the normal force and tangential velocity control loops of robot manipulators employed in the contour tracking task of objects of unknown shape.

In [4], general properties of active force control methods have been discussed. In [16], the robot control system is based upon the external force control. They keep the original position control loop and added external force control loop as a new major loop. In [22], a model for grinding process of an automatic grinding system with grinding force control was developed and the corresponding PID controller was designed. In [23], the paper deals with the use of a hybrid force/velocity control law for the robotic deburring of planar work pieces with an unknown shape. They controlled the normal force, tangential velocity and normal velocity.

In [24], an algorithm was developed in order to control the interaction force between tool and the work-piece. A plate and a roller are used to guide the tool. These guides prevent the tool to exceed a certain depth of cut. The interaction force between the tool and the work- piece is kept constant. However due to the guiders used in this work, this is not a proper example for precise force control.

In [25] two different algorithms namely “Gradient Prediction Method” and “Progressive Stiffness Method” were designed for grinding. Contour following quality was improved. In “Gradient Prediction Method” gradient of the workpiece is estimated and force errors are corrected. “Progressive Stiffness Method” tries to keep the contact force force constant.

Additionally, in [26], which is also a publication of the author, justification of force control is given.

2.2 Piezo actuators in machining

In [27], an active tailstock was developed in order to correct the tilt errors of the rotating crankshaft during grinding. This tailstock produces a counter-tilt and it compensates the possible grinding errors caused by the rotation of the asymmetric crankshaft. The needed compensation is performed by piezo-hydraulic hybrid positioning actuator. The schematic construction of the precise alignment system is shown in Figure 8.

In [28], an active controlled palletized work-piece holding system was presented for milling operations. The active control system developed here employs piezo-actuators to control the force dynamically Figure 9.

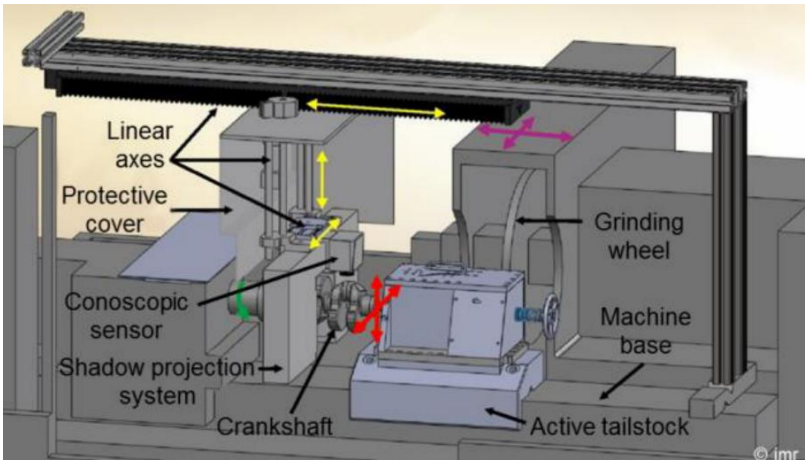


Figure 8 - Schematic construction of the precise alignment system [27]

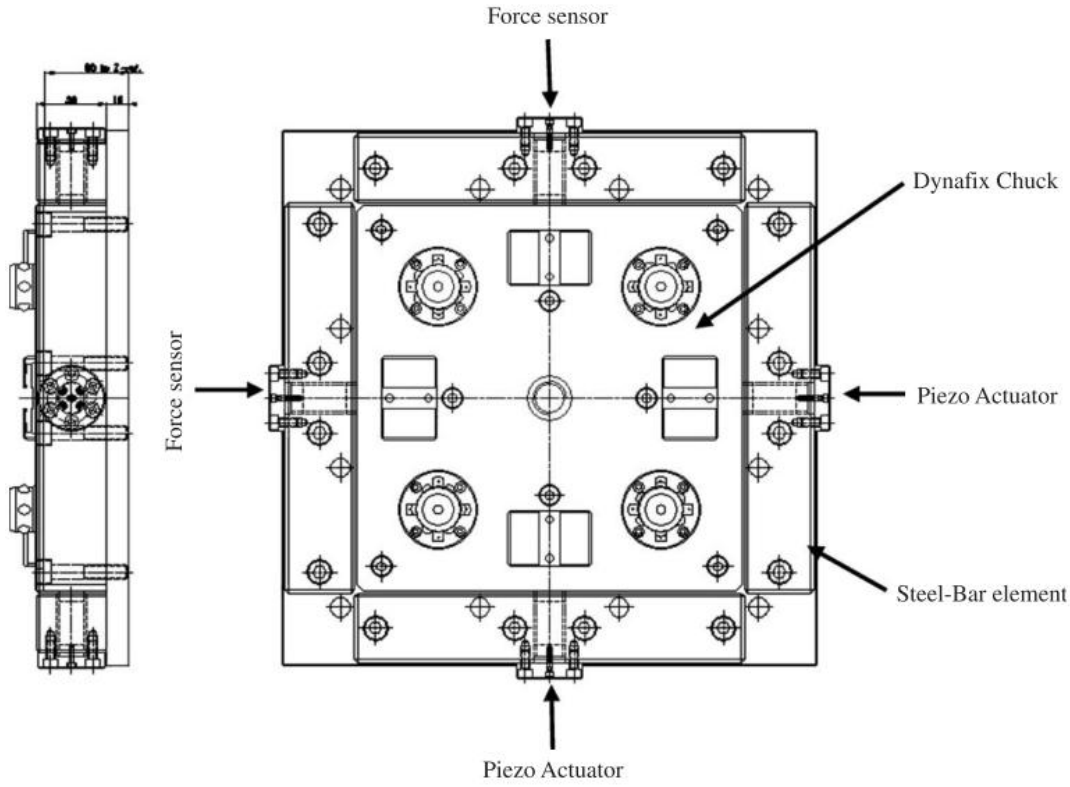


Figure 9 - Schematic design of the chuck-pallet system with active vibration control elements [28]

2.3 Passive Compliant Tools

In industry, as opposed to the method in this study, applications of passive compliance are also common. There are special deburring tools such as flex-deburr from [29] that are able to compensate the form errors passively by tracking the forms. Tracking is not performed by an active system. The tool tip can track the surface profile due to the suitable stiffness value of the used material. The mechanical design of active compensation on these tools is a hot-topic in literature [30][31].

Figure 10 - ATI's deburring tool family[32]

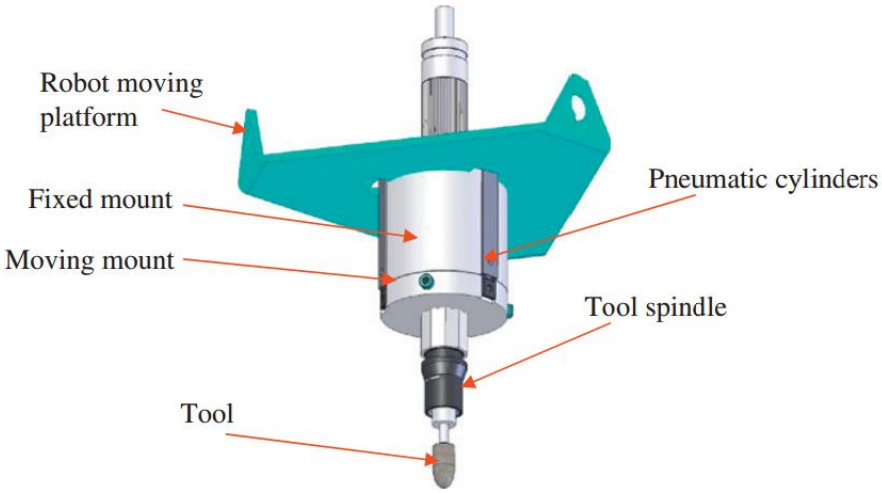


Figure 11 - Axial Compliant tool head [30]

2.4 Hybrid Force / Velocity Control

With the advance of control techniques, the researchers attached importance to machining of a work-piece with an unknown shape. In order to obtain constant depth of cut from a homogeneous material, the grinding parameters such as feedrate, spindle speed, should be invariant throughout the surface profile. These requirements can be achieved via hybrid force/velocity control [33].

In [34], the implementation of grinding of a work-piece with an unknown shape was performed. As the control algorithm hybrid force/velocity control structure was utilized. The authors dealt with the problems related to configuration dependent dynamics of the manipulator.

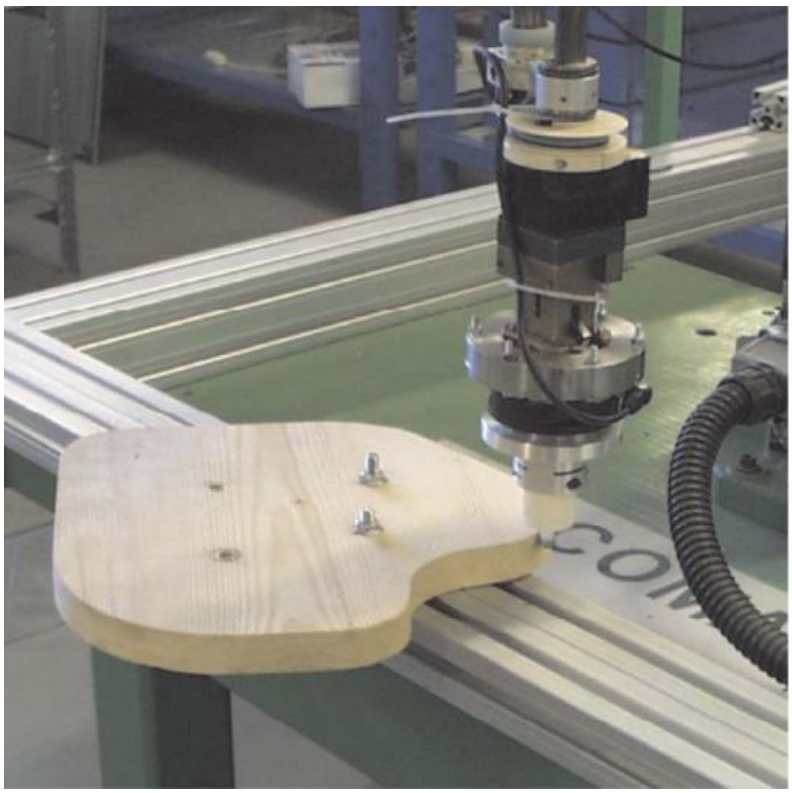


Figure 12 - Robot in contact with a wooden object [34]

In [35], the effects of elastic transmission of the robots during contour tracking of a work-piece with an unknown shape are investigated. The large force oscillations due to the elasticities in joints are compensated by an additional normal velocity feedback loop.

In [21], decoupling of normal force and tangential velocity control loops was studied. The controller was expressed as multi input – multi output, time varying, PID controller.

In [36], joint friction effects to normal force and tangential velocity variations in hybrid force / velocity controller were investigated.

Additionally, [23] and [37] are the examples of contour tracking.

2.5 Tool Deflection Compensation

There are several researches in literature related to the compensation of tool deflection effect on work-piece. In this section a review of different strategies of these studies are expressed. Most of the mentioned studies are related to the end milling operation.

Kline et al [38] proposed a method for prediction of tool and work-piece deflection amount in end milling operation based on cantilever beam theory. They used a force model and cantilever beam theory for obtaining deflection amount. Similarly, Ryu et al. [39] investigated side wall machining operation and tried to predict the errors caused by tool deflection. But, they did not express a solution for compensation of these errors. The effect of work-piece curvature on tool deflection and resulting surface errors are investigated in [40].

A method based on path correction is proposed by Law et al. [41]. Their aim was to decrease tool deflection and its effect on work-piece using optimum tool path.

Approaches for path correction in the end milling operation were presented in [42]–[45] by adding an offset to the tool path. They used cantilever beam theory in order to

calculate the amount of tool deflection. Rao et al. [46] proposed an iterative approach instead of single offset for compensation of offset error caused by tool deflection. However, they did not investigate tool angle compensation.

A method for compensation of tool angle and tool displacement during end milling operation is proposed by Yang et al. [47] where a sensor is used for detecting tool deflection amount. The strong side of their research is that they considered both tool angle and tool tip displacement by compensation of errors.

CHAPTER 3

EXPERIMENTAL SETUP

3.1 Overview of the setup

In the scope of this thesis, previously designed and partially built experimental setup was modified and used. The overall appearance of the experimental setup is shown in Figure 13.

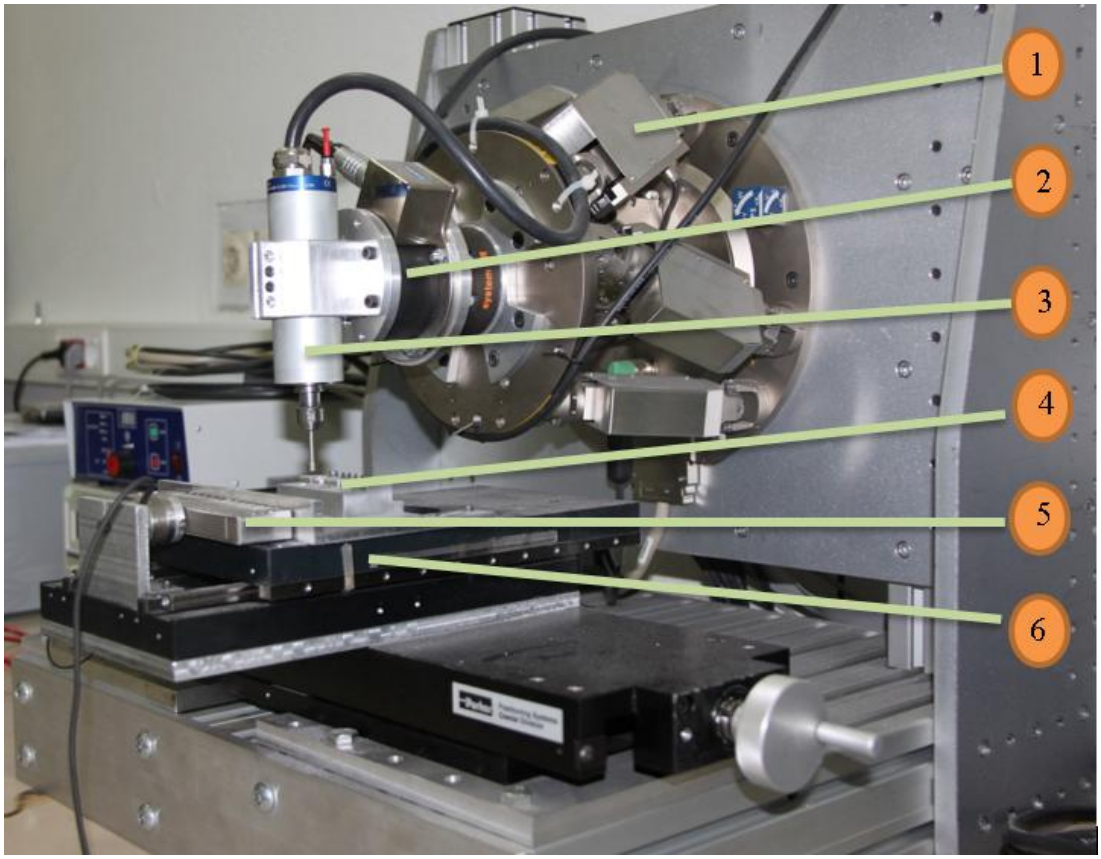


Figure 13 - Overall appearance of the experimental setup

In addition to 6-DoF parallel manipulator, the experimental setup has an additional 1 degree of freedom which is actuated by a piezo actuator. The actuator is fixed to the properly constrained table, presents a single degree of freedom in the x direction as shown in Figure 14. While performing grinding in y direction as shown in the same figure, the machining errors can be reduced by admittance control based negative compensation by the actuation of the piezo actuator.

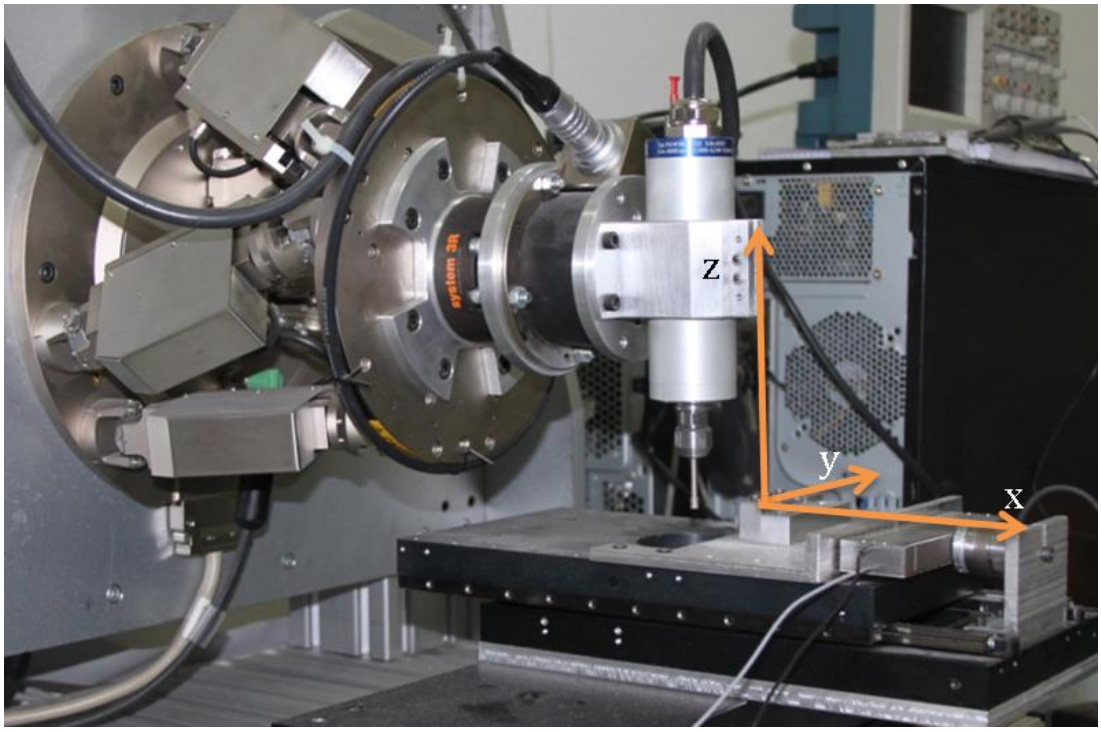


Figure 14 - Coordinate System

The robotic grinding setup components shown in Fig. 1 are:

1) Hexapod (6 DoF): PI H-824 6 DoF hexapod precision parallel positioning system

2) ATI Gamma IP60 Force / Torque Sensor

3) Spindle: BMR Typ. 222-42-MHM

4) Workpiece

5) Piezo Actuator: PI P-602 PiezoMove Flexure Actuator

6) Table (which has 1 DoF in x direction)

3.1.1 Piezo Actuator

In this work a P-602 Piezo-Move flexure-guided piezo actuator is utilized to control the movements of the table precisely (Figure 15). A piezoelectric actuator converts an electrical signal into a precisely controlled physical displacement. If displacement is prevented, a useable force will develop. The precise motion control, afforded by piezo actuators, is used to finely adjust machining tools etc. They are used in applications requiring movement or force.

In this thesis, a piezo actuator is used in order to move the machining table in one degree of freedom. Response characteristics of piezo actuator is better than the hexapod robot. That is the reason why piezo actuator was utilized.

Figure 15 - P-602 Piezo-Move flexure-guided piezo actuator

The piezo actuator is controlled by its PI E-610.S0 LVPZT motion amplifier/controller which includes PI (Proportional and Integral) controller. The working range of the piezo actuator is 1 mm and its closed loop resolution is 7nm. In this system it is used in closed loop mode thanks to the Strain Gauge sensors installed on it. Electronic connections between the piezo actuator and the driver was performed and the connection scheme is shown in Figure 16.

In order to facilitate the connection between the computer and the controller of the piezo actuator a printed circuit board was designed and produced.

The control input for the piezo actuator in our setup is voltage (0 to 10 V) and the output is position (0 to 1 mm). While giving input, it is possible to take the actual position data of the piezo actuator by strain gauge sensors installed on it. A simple SIMULINK Model which was prepared for this purpose can be seen in Figure 18.

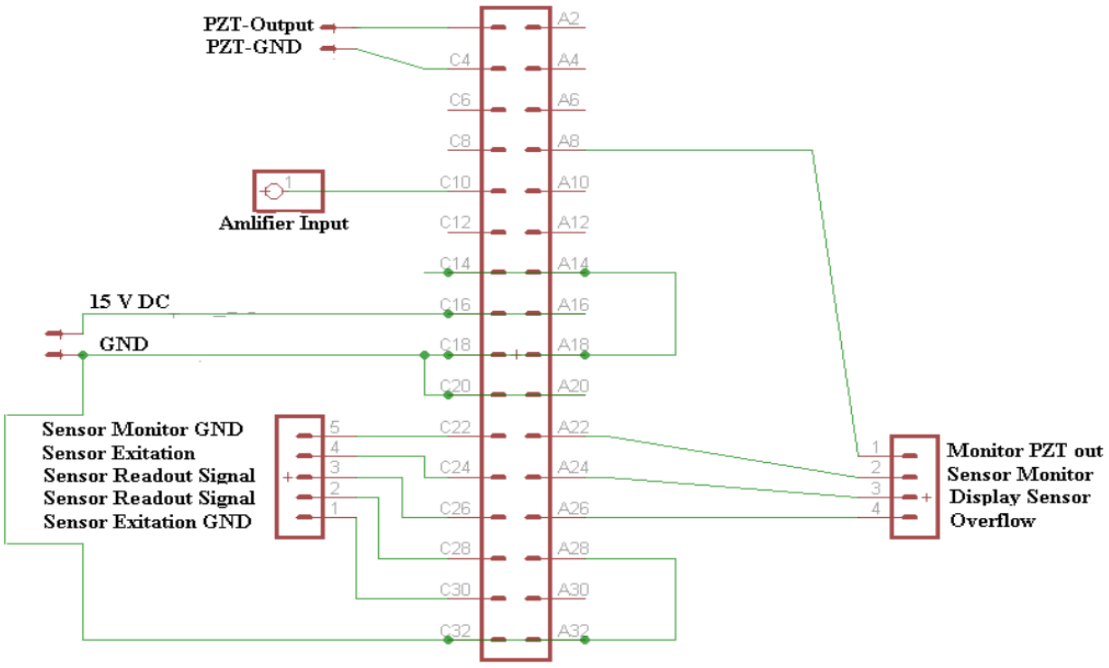


Figure 16 - Connection scheme of the piezo actuator

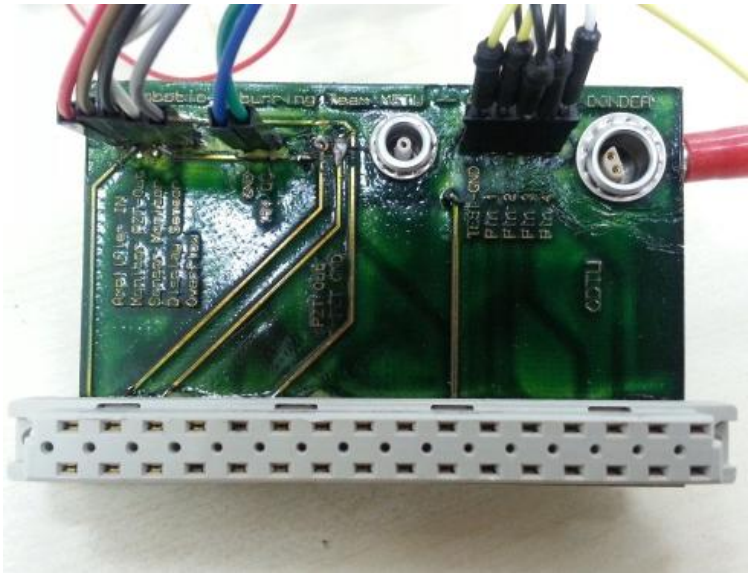


Figure 17 - Designed printed circuit board

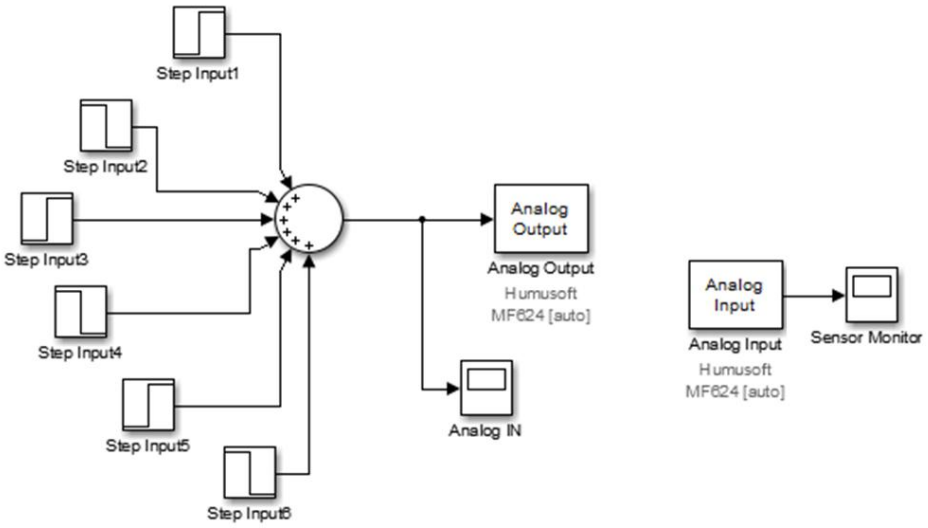


Figure 18 - SIMULINK Schema for step inputs to Piezo Actuator

In this figure, the step inputs up to 10 V are given to Piezo Actuator by using "Analog Output" block of the data acquisition card Humusoft MF624. And the

responses are read by the “Analog Input” block of the same data acquisition card. “Sensor Monitor” block shown in Figure 18 is used to show the responses of the piezo actuator against 3V, 6V and 9V (which correspond to 0.3, 0.6 and 0.9 mm inputs) are shown in Figure 19 in blue. The specifications of the piezo actuator are given in Appendix A.

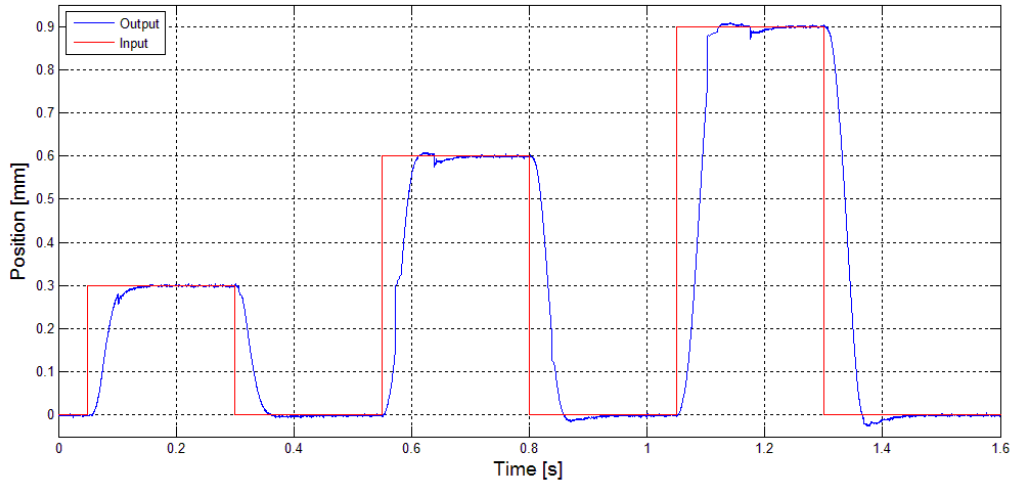


Figure 19 - 0.3, 0.6 and 0.9 mm step input responses of Piezo Actuator

3.1.2 Hexapod (Parallel Manipulator)

Hexapod is the main device which carries all the other parts of the robotic grinding experimental setup. It has 6 linear actuators connecting the platforms of the parallel manipulator. The parallel manipulator is shown in Figure 20 and the specifications are shown in Appendix A. A MATLAB SIMULINK model and a GUI was developed for controlling the hexapod robot. The used guide of the GUI is given in Appendix C.

Figure 20 - Hexapod H-824 from PI Company [48]

3.1.3 Spindle

BMR Company's Typ. 222-42-MHM Spindle – Frequency Converter couple [49] was used (Figure 21). The datasheet of the spindle is provided in Appendix A.

The frequency converter is connected to the workstation over its 15 pin D-SUB connector's RS232 pins and SIMULINK model for controlling the spindle was developed as shown in Figure 22. The MATLAB Code in the MATLAB function is given in Appendix B.

Figure 21 - Frequency converter (left) and the spindle (right) from BMR Company [49]

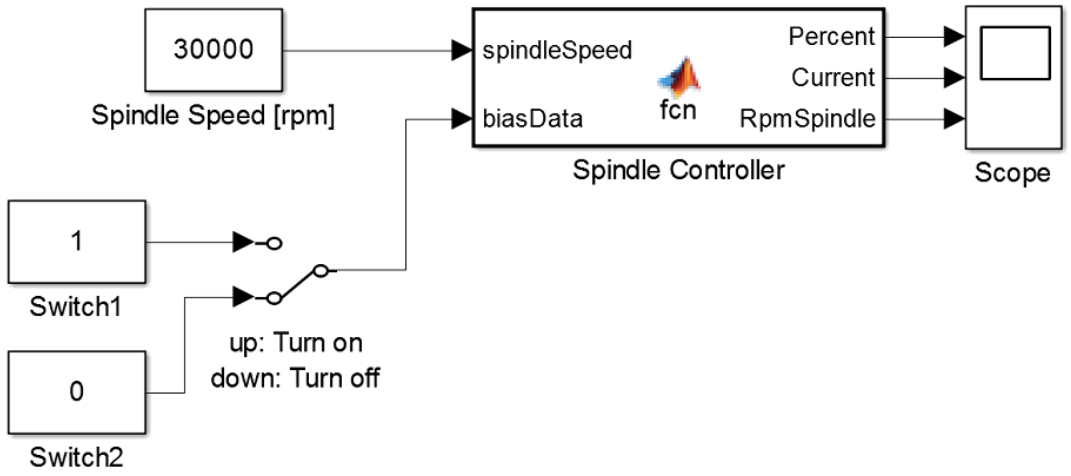


Figure 22 - Developed SIMULINK Model for controlling the Spindle

3.1.4 Multi Axis Force/Torque Sensor

In grinding operations force/torque sensors are extensively used. These sensors are used to measure the amount of force applied on parts of the machine. Additionally, by measuring the force which is applied on the tool, they can be used to check whether the contact is performed between the tool and the work-piece or not. Also if the contact is performed, the force/torque sensors can be used to measure the level of the contact.

On our robotic-grinding experimental setup, there is one force/torque sensor. This sensor is able to provide the data of the forces on 3 Cartesian basis axes and of the torques around the same axes. The transducer electronics have bandwidth of 5 kHz to 10 kHz (depending on gain settings). The force torque sensor used in this work is shown in Figure 23. The specifications are shown in Table 6.

National Instruments PCI-6052E data acquisition card is used for F/T sensor.

Figure 23 - ATI Gamma F/T sensor [29]

3.1.5 The Control Software

In order to control and drive hexapod, force/torque sensor, piezo actuator and spindle; MATLAB SIMULINK software was utilized. Used SUMULINK models are explained in the related sections. These four devices are connected to the workstation over the protocols summarized in Figure 24.

In order to control the hexapod robot, a graphical user interface was prepared by MATLAB as shown in Figure 68. Step by step user manual of this GUI is given in Appendix C.

3.2 Measurement Setup

In order to understand the amount of material removed from the surface of the work-piece and the form change, the ground amount should be measured before and after the experiment. That is why a measurement setup was built in the scope of this thesis as shown in Figure 25. The measurement system consists of a precise positioning system and a laser measurement device. In this system, laser measurement device is

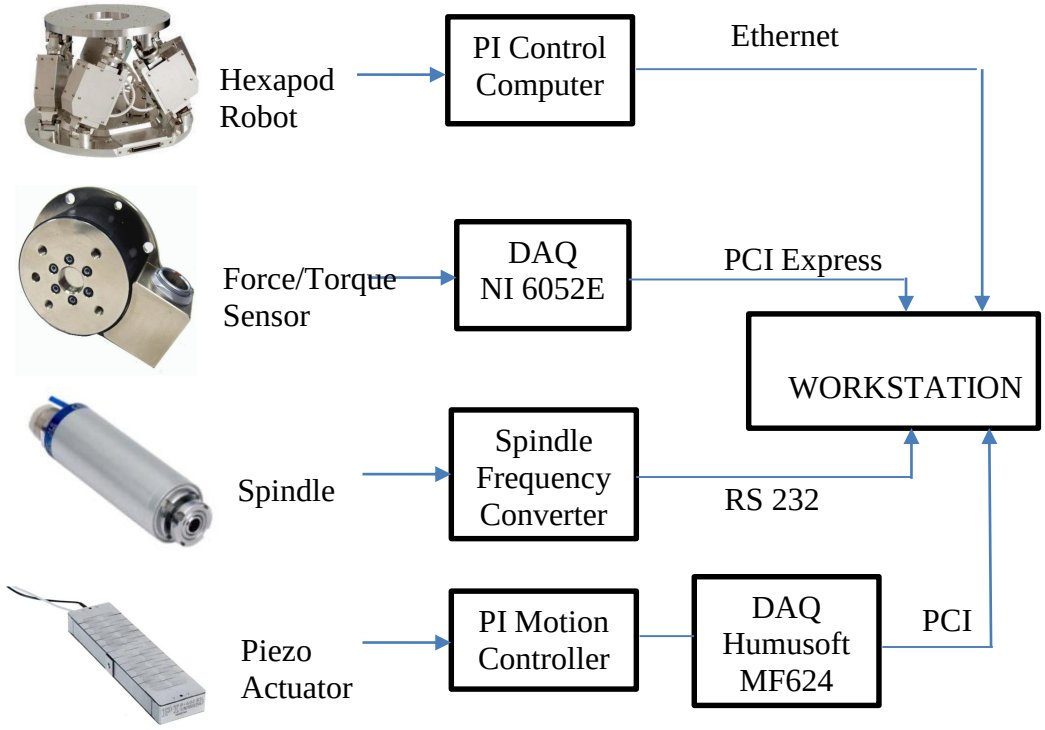


Figure 24 - Devices and Connection Protocols

located at the fixed part of the positioning system and the work-piece is passed by in front of it. In every 500 μm intervals a measurement is taken. Therefore, surface form is obtained.

A KEYENCE LK-H027 measurement device was utilized for the measurement system (Figure 26). These sensors are extensively used in the industry when precise measurement is needed. Its measurement range is 17-23 mm and the repeatability is 0.02 μm.

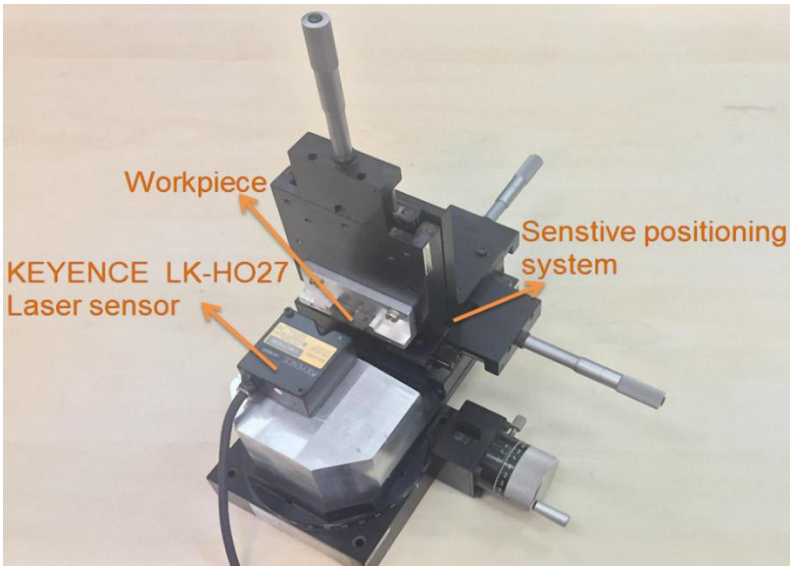


Figure 25 - Appearance of the measurement setup

Figure 26 - KEYENCE LK-H027 Laser Measurement Device [50]

CHAPTER 4

TWO DEGREE OF FREEDOM HYBRID VELOCITY FORCE CONTROL STRUCTURE

In order to obtain constant depth of cut throughout the surface profile, the grinding parameters (feedrate in tangential direction, spindle speed, interaction force in normal direction) should be kept constant. Keeping spindle speed constant is not a problem since generally spindles work with their speed controllers. However, keeping the feedrate constant in tangential direction and obtaining the constant interaction force in normal direction on a varying surface can be quite problematic. In this study a hybrid controller which controls feedrate in local tangential direction and grinding interaction force in local normal direction was developed. While feedrate compensation was performed with 6 DOF hexapod robot, normal force compensation was performed by high frequency piezo actuator. Local normal and tangential directions are shown in Figure 2.

In order to obtain constant depth of cut from variable surface, the key strategy that should be implemented is imposing appropriate normal force and tangential velocity. That is, classical explicit hybrid force/velocity control should be implemented [33]. In order to obtain the actual local normal force from measured X and Y force components, the algorithm which is explained in [23] was utilized.

The local tangential force is as follows:

$$F_t = \frac{M_{zSpindle}}{r_{tool}} \tag{1}$$

Where:

$M_{zSpindle}$: Measured moment around Z axis of the spindle

r_{tool}: Radius of the cutting tool

However, with the used setup, measured moment around Z axis of the force/torque sensor M_z is not the moment around the axis of the spindle since the force/torque sensor has an eccentricity with respect to the spindle. Therefore, local tangential force was calculated as follows:

$$F_t = \frac{M_{zSpindle}}{r_{tool}} = \frac{M_z - Fx * \Delta y - F_Y * \Delta x}{r_{tool}} \quad (2)$$

Where:

F_X: Measured force in X direction

F_Y: Measured force in Y direction

M_z: Measured moment around Z axis of the force/torque sensor

$M_{zSpindle}$: Moment around Z axis of the spindle

Δy: Eccentricity of the force/torque sensor with respect to spindle axis in Y direction

Δx: Eccentricity of the force/torque sensor with respect to spindle axis in X direction

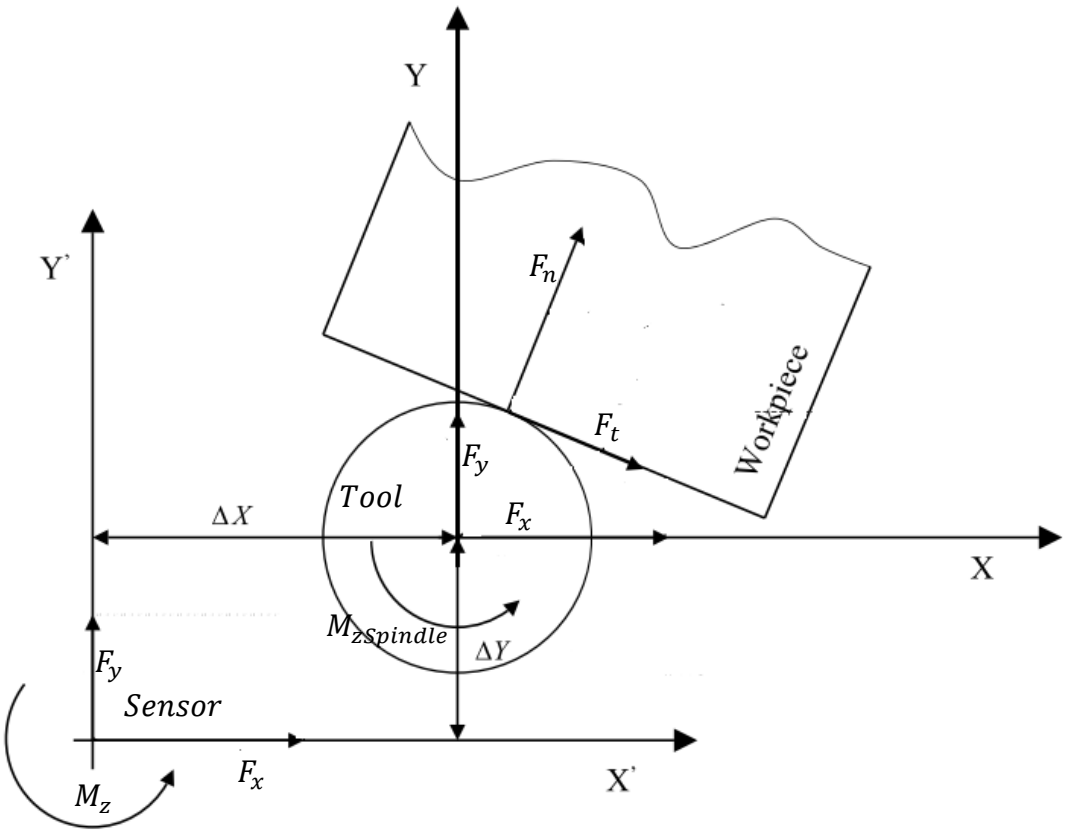


Figure 27 – The eccentricity between tool and the sensor (Modified from [23])

After calculation of F_t, the local normal force F_n is calculated by the utilization of the following equality:

$$\sqrt{F_x^2 + F_y^2} = \sqrt{F_n^2 + F_t^2} \tag{3}$$

Therefore:

$$F_n = \sqrt{F_x^2 + F_y^2 - F_t^2} \tag{4}$$

Determination of Δx and Δy was performed by the procedure explained in [23] which is simple least squares parameter estimation procedure.

While the spindle is switched off, it is traveled around the contour. During this operation forces and Z moment are collected. Since the spindle is able to roll while tracking the surface, the measured moment is only because of the X and Y force components. That is,

$$M_{zMeas.} = Fx * \Delta y + F_Y * \Delta x \tag{5}$$

After collecting N samples, following linear system can be written:

$$\begin{array}{ccc} M_{zMeas.1} & = & Fx_1 * \Delta y + F_{Y_1} * \Delta x \\ M_{zMeas.2} & = & Fx_2 * \Delta y + F_{Y_2} * \Delta x \\ \vdots & \vdots & \vdots \\ M_{zMeas.N-1} & = & Fx_{N-1} * \Delta y + F_{Y_{N-1}} * \Delta x \\ M_{zMeas.N} & = & Fx_N * \Delta y + F_{Y_N} * \Delta x \end{array} \tag{6}$$

In matrix form:

$$\boldsymbol{M}_{zMeas.} = \boldsymbol{FX} \tag{7}$$

Where;

$$\boldsymbol{M}_{zMeas.} = \begin{bmatrix} M_{zMeas.1} \\ \vdots \\ M_{zMeas.N} \end{bmatrix}, \qquad \boldsymbol{F} = \begin{bmatrix} Fx_1 & F_{Y_1} \\ \vdots & \vdots \\ Fx_N & F_{Y_N} \end{bmatrix}, \qquad \boldsymbol{X} = \begin{bmatrix} \Delta_Y \\ \Delta_X \end{bmatrix} \tag{8}$$

As a result:

$$\boldsymbol{X} = \begin{bmatrix} \Delta_Y \\ \Delta_X \end{bmatrix} = \boldsymbol{F^{pinv}} \boldsymbol{M}_{zMeas.} \tag{9}$$

Where $\boldsymbol{F^{pinv}}$is the pseudoinverse matrix of $\boldsymbol{F}$.

After this procedure Δ_X and Δ_Y were calculated as:

$$\boldsymbol{X} = \begin{bmatrix} \Delta_Y \\ \Delta_X \end{bmatrix} = \begin{bmatrix} -0.6\ mm \\ 38\ mm \end{bmatrix} \tag{10}$$

Constant velocity control is performed by the controller of the hexapod robot. However, when the piezo actuator is in action, the resultant feedrate increases since the feedrate is defined as:

$$F_R = \sqrt{V_{Hex}^2 + V_{Pzo}^2} \tag{11}$$

where:

V_{Hex}:Velocity of the hexapod

V_{Pzo}: Velocity of the piezo actuator

In order to keep the local feedrate constant, the hexapod robot arranges its velocity according to the movements of the piezo actuator. Firstly, reference local feedrate and normal force components are entered by the user. If the actual normal force is not equal to the reference normal force, the error is defined as the difference between reference normal force and the actual normal force. In the next step, the actual normal force is controlled by the movements of the piezo actuator. However, due to the movements of the piezo actuator the feedrate which is the combination of the movements of the hexapod and the piezo actuator increases. In order to keep the local feedrate constant, the controller decreases the velocity of the hexapod. After that updated actual normal force is calculated from measured X and Y force components and the moment around Z axis. Control of the feedrate of the hexapod robot is considered as an independent loop [21]. Additionally, tool deflection compensation which will be explained in Chapter 7 is considered as an independent separate loop as well.

4.1 PID Controller

PID Controllers are extensively used in industry due to their simplicity, robustness, easy implementation and their well-known tuning techniques [51], [52].

The transfer function of the PID controller can be written as follows:

$$G_{PID}(s) = K\left(1 + \frac{1}{T_I s} + T_D s\right) \tag{12}$$

Where: K is the proportional gain, T_I is the integral time constant, T_D is the derivative time constant.

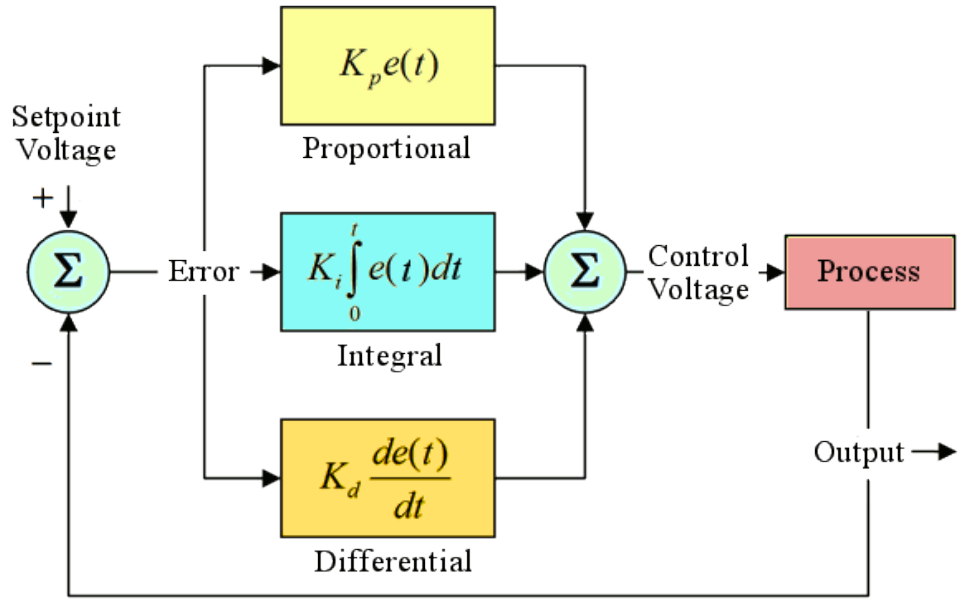


Figure 28 - Functional Diagram of a PID Control Loop [53]

The most important weaknesses of PID control are as follows [54]:

- Due to noise sensitivity, PID controller is often used without derivative(D) term
- Integral term introduces saturation and reduced stability margin due to phase lag.

4.2 Active Disturbance Rejection Controller (ADRC)

In order to eliminate the weaknesses of classical PID control, ADRC was firstly proposed in [55], [56]. It has been studied for approximately two decades. Additionally, according to the literature it is more suitable for non linear plants.

ADRC proposes following fundamental properties[54]:

1. **Set-point Jump and Tracking Differentiator**
   Generally, reference input of the system is given as a step input which is not suitable for most of the dynamic system since it results in a sudden jump of the output. In order to eliminate this drawback, it is necessary to have a transient profile that can be easily followed by the output of the system. Additionally, since the differentiation used in classical PID control is sensitive to noise, ADRC proposes following method:
   For a double integral plant:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u \end{aligned} \tag{13}$$

   A discrete-time solution for a discrete double integral plant:

$$u = fhan(v_1 - v, v_2, r_0, h_0) \tag{14}$$

$$v_1 = v_1 + hv_2 \tag{15}$$

$$v_2 = v_2 + hu \tag{16}$$

   where h is the sampling period, r_0 and h_0 controller parameters.

Additionally, $fhan(v_1, v_2, r_0, h_0)$ is:

$$d = hr_0^2, \quad a_0 = hv_2, \quad y = v_1 + a_0 \tag{17}$$

$$a_1 = \sqrt{d(d + |8y|)} \tag{18}$$

$$a_2 = a_0 + sign(y)(a_1 - d)/2 \tag{19}$$

$$s_y = (sign(y + d) - sign(y - d))/2 \tag{20}$$

$$a = (a_0 + y - a_2)s_y + a_2 \tag{21}$$

$$s_a = (sign(a + d) - sign(a - d))/2 \tag{22}$$

$$fhan = -r\left(\frac{a}{d} - sign(a)\right)s_a - rsign(a) \tag{23}$$

The most important utility of the method above is the ability to take the derivative of a noisy signal with a good signal to noise ratio and to work as a noise filter.

2. Nonlinear Feedback Combination

Following non-linear function is proposed for the combination of non-linear feedbacks:

$$fal(e, a, \delta) = \begin{cases} \dfrac{e}{\delta^{1-\alpha}} & |e| \le \delta \\ |e|^{\alpha} sign(e) & |e| \ge \delta \end{cases}$$

e presents error, α and δ are small numbers as explained in [57].

3. Total Disturbance Estimation and Rejection via Extended State Observer (ESO)

ESO provides real time feedback to eliminate the disturbance by estimating the disturbances and unmodelled dynamics of the system. This structure was designed for robustness against the variations in plant. Therefore, the necessity

for integral control which has an inherent lag that can make a closed loop control system unstable is eliminated.

The augmented variable is introduced as:

$$x_3 = a(t) = f(x_1, x_2, w(t), t) \tag{24}$$

where; x_i are the states, $w(t)$ is external disturbances and t is time.

Therefore, the non-linear state observer can be constructed as follows:

$$e = z_1 - y \tag{25}$$

$$fe = fal(e, 0.5, \delta), \quad fe_1 = fal(e, 0.25, \delta) \tag{26}$$

$$z_1 = z_1 + hz_2 - \beta_{01}e \tag{27}$$

$$z_2 = z_2 + h(z_3 + bu) - \beta_{02}fe \tag{28}$$

$$z_3 = z_3 - \beta_{03}fe_1 \tag{29}$$

where; β parameters are the observer gains, z_3 is the total action of unknown disturbances and z_1, z_2 are the estimates of the states x_1 and x_2.

Therefore, ADRC topology can be constructed as follows:

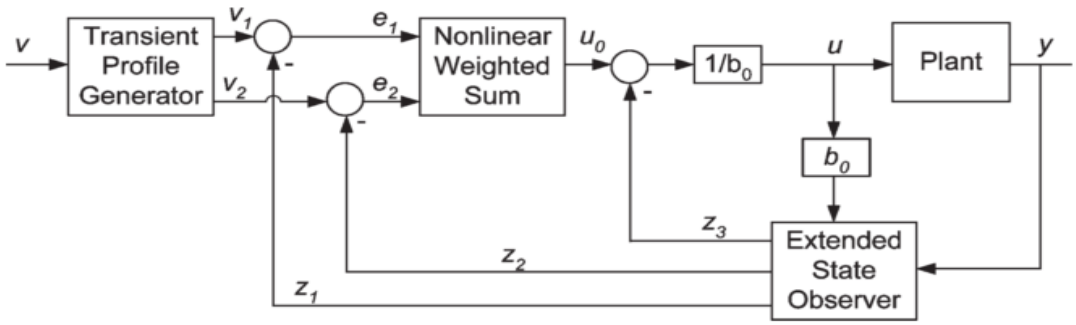


Figure 29 - ADRC Topology [54]

CHAPTER 5

MODELLING AND OPTIMIZATION OF CONTROLLER PARAMETERS

For modelling purposes MATLAB SIMULINK was used.

5.1 System Identification of the Piezo Actuator

- Data Collecting Experiments

4 experiments each of which lasted 5 minutes were conducted. While the tool traces the y direction, a flat shaped work-piece was ground. When the tool and the work-piece are in contact with a normal force of 5-20 N, 10μm step inputs were given to the piezo actuator. Therefore, response of the piezo actuator to step inputs under grinding loads were recorded.

- System Identification

After the collection of input and output data of the piezo actuator, system identification analysis was performed. For this purpose, MATLAB System Identification Toolbox was utilized[58].

In this study, transfer function model estimation is performed by ARX method[59] by MATLAB System Identification Toolbox. The model is represented by the following structure[60]:

$$y(t) = -a_1 y(t-1) - \cdots - a_{n_a} y(t-n_a) + b_1 u(t-1-n_k) + \cdots + b_{n_b} u(t-n_b-n_k + e(t) \quad (30)$$

where:

$e(t)$: Gaussian noise

$a_{n_a} and\ a_{n_b}$: Model parameters

$n_a\ and\ n_b$: The order of the polynomials of the output A(q) and the input B(q) respectively

n_k: Time delay between y(t) and u(t)

The polynomial representation of (30) is given in (31)

$$A(q)y(t) = B(q)u(t - n_k) + e(t) \tag{31}$$

Where

$$A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a} \tag{32}$$

$$A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a} \tag{33}$$

q^{-1} is the delay operator, for instance:

$$u(t = 1) = q^{-1}u(t)$$

In order to estimate A(q) and B(q), nonlinear least squares identification method was utilized.

By using each dataset (input and output couple) 3 discrete transfer functions (2^{nd}, 3^{rd}, 4^{th} order) were estimated. Then the 2^{nd} order transfer function which was estimated by the 3^{rd} dataset was selected.

$$G(z) = \frac{0.001333z}{z^2 - 1{,}949z + 0{,}9503} \tag{34}$$

5.2 Overall Model

The unknown shape of the work-piece is designed as sinusoidal. The location of the profile with respect to the tool is updated at each time instant according to the given feedrate. The parameters of the sinusoidal profile are given in Figure 30.

Differentiation of parameter D is used as tool feedrate and A is the position of the piezo actuator.

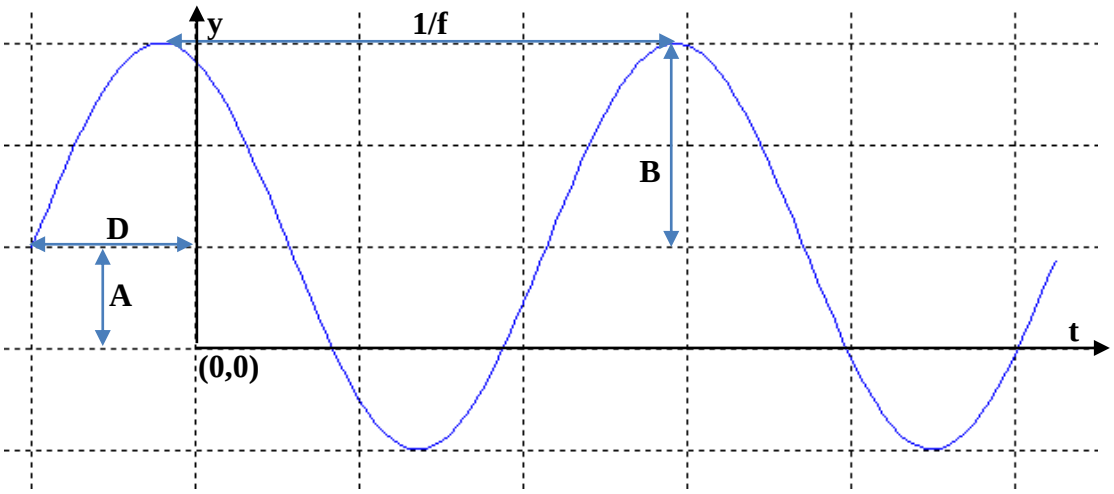


Figure 30 - y(t)=A+B sin(Ct+D) -- C is frequency(rad/sec) w=2πf

As shown in Figure 31 and Figure 32 closed loop controllers were used in the system. The controller output is the position change of the piezo actuator. The plant block is shown in Figure 33. After the position of the piezo actuator is determined, "A" parameter of sinusoidal profile is calculated in "Calculation of parameter A" block. Since this "A" parameter determines the distance between the tool and the work-piece in X direction, its inputs are tool radius, amplitude of the profile (B), initial depth of cut and initial piezo position.

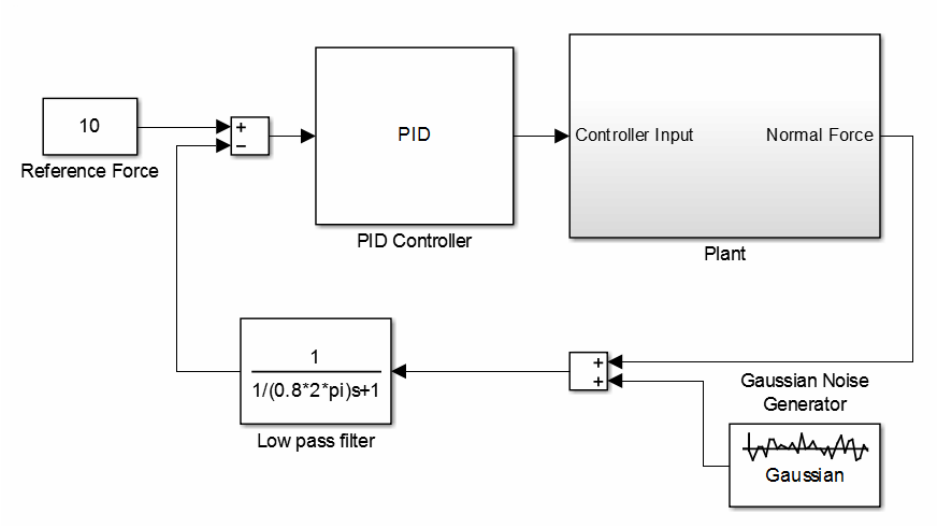


Figure 31 - PID Controller Model

"D" value of sinusoidal profile was determined according to piezo velocity. Therefore, in order to determine the piezo velocity, derivative of the piezo position was taken. After that "Calculation of hexapod velocity" block takes set feedrate and piezo velocity as inputs and calculates the hexapod velocity. By taking the integral of this velocity value, parameter "D" was reached.

After determination of all the parameters for sinusoidal profile generation, it was created and Local Depth of Cut is determined by calculating the intersection point of the tool and the surface profile. The determination of generated normal force component from the local DoC was performed by the grinding force model explained in [11]. After "Plant" block, the loops are closed by adding Gaussian Noise with a variance of 0.8 which is the variance of actual measured force data.

The animation of the grinding operation was prepared. A screenshot of it is shown in Figure 34.

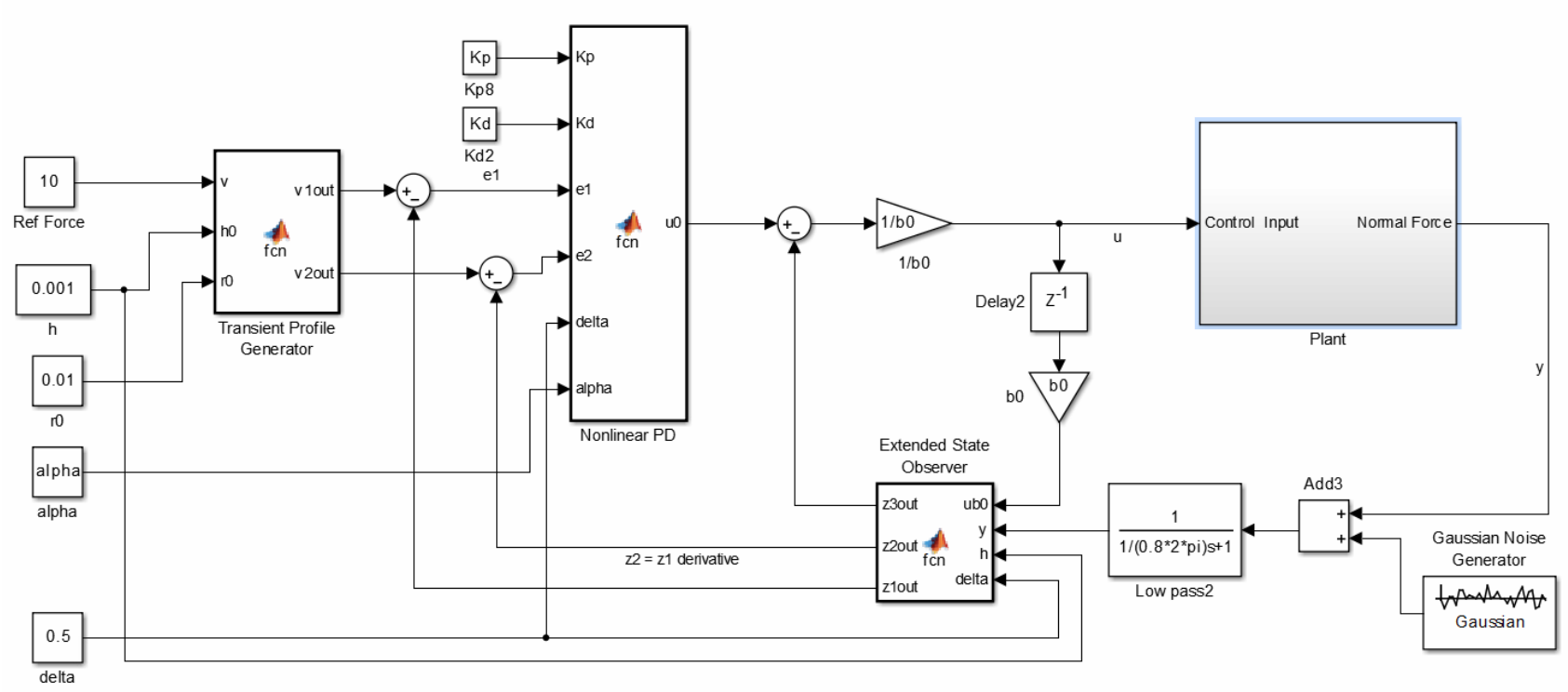


Figure 32 - Active Disturbance Rejection Controller Model

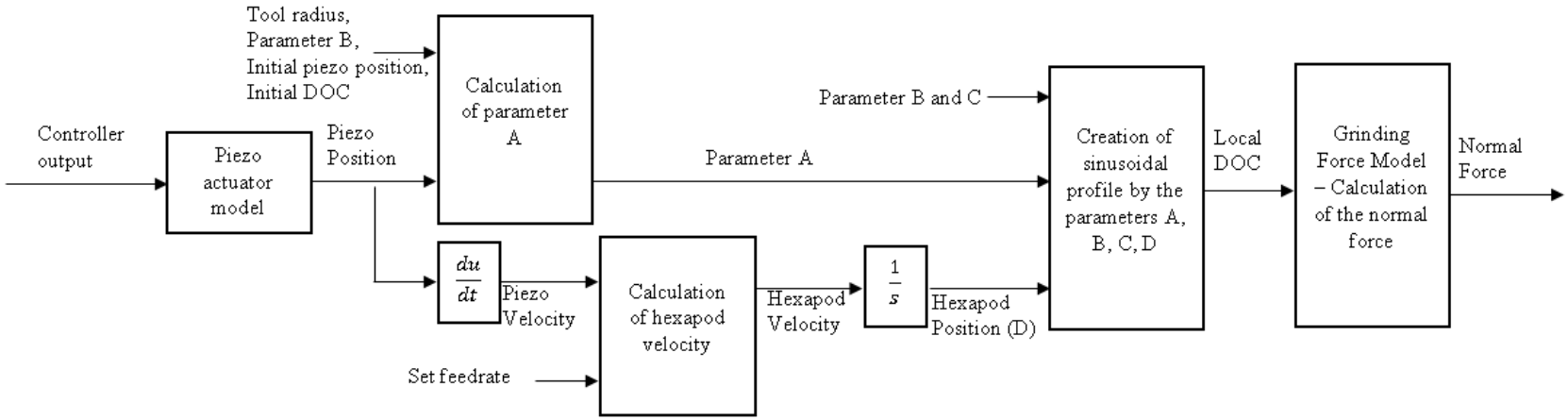


Figure 33 - Model of the plant

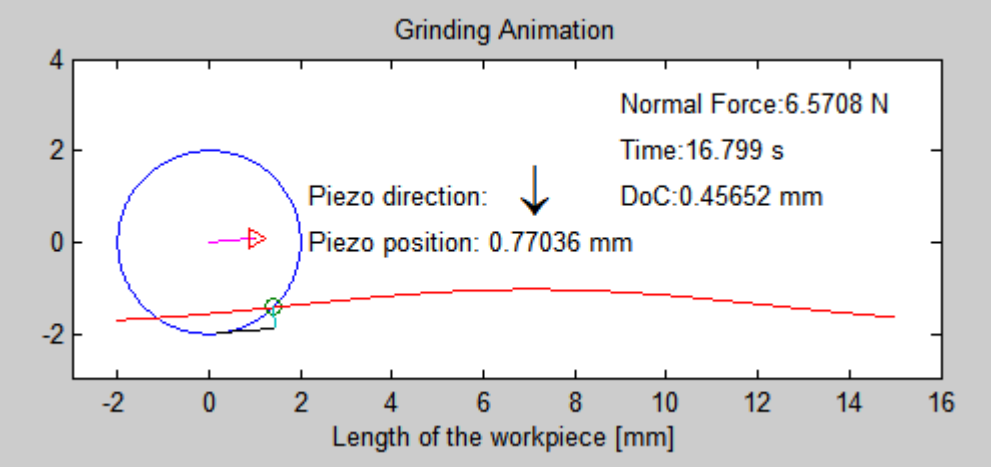


Figure 34 - Grinding animation

The blue circle represents the tool while the red curve shows the work-piece surface profile. The intersection point of the tool and the surface profile is calculated at each time step and shown by small green circle. Pink arrow represents the relative movement direction of the tool. After the determination of this direction, black line which is parallel to the movement direction is drawn at the bottom of the tool as tangent. After that the local depth of cut which is shown by green is calculated as the perpendicular distance between this black line and the intersection point of the tool and the surface profile.

The determination of generated normal force component from the local depth of cut is performed by a grinding force model [11].

5.3 Optimization of Controller Parameters by Genetic Algorithm

Genetic Algorithm is a method by which constructed and unconstructed optimization problems can be solved. At each time step, the algorithm selects some individuals in order to use them as parents in the next iteration. These selected parents are used to generate new generation at the following time step. After a certain iteration, optimal solution is approached.

The usage of this method in controller parameter tuning is also extensively used technique[61]–[63]. In this study, genetic algorithm was used for the tuning of PID controller parameters.

For the implementation of genetic algorithm, MATLAB SIMULINK 2013a Response Optimization was preferred[64].

CHAPTER 6

MACHINE – TOOL STIFFNESS: ORIENTATION COMPENSATION

When the tool interacts with the work-piece, due to the generated grinding forces, it deflects similar to Figure 35. In order to eliminate this defect, hexapod is programmed to compensate angle differences by rotating around x and y axes. The amount of rotation is determined by double integration method.

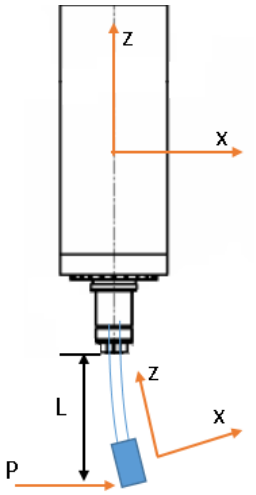


Figure 35 - Tool deflection due to the grinding force (P).

6.1 Calculation of displacement and orientation errors from force feedback: Double integration method

Assuming that the spindle and the machine is rigid, and the tool has a finite stiffness. Then the tool can be modeled as a cantilever rod:

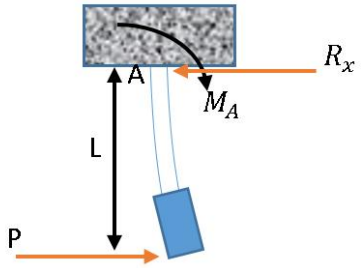


Figure 36 - The tool modeled as a cantilever rod

Cutting the tool at a certain x distance:

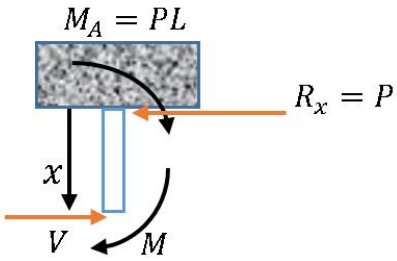


Figure 37 - Cut tool

In order to calculate the deflection and stiffness double integration method will be utilized:

$$EI_Y \frac{d^2y}{dx^2} = -M \tag{35}$$

Where

E: Modulus of Elasticity

I_Y: Moment of Inertia with respect to Y axis

y: displacement

x: The distance from point A

Therefore, following equilibrium equation can be written:

$$-Px + PL + M = 0 \tag{36}$$

Leaving M alone:

$$M = P(x - L) \tag{37}$$

Implementing double integration method:

$$EI_Y \frac{d^2y}{dx^2} = P(L - x) = PL - Px \tag{38}$$

$$EI_Y \frac{dy}{dx} = PLx - \frac{Px^2}{2} + C_1 \tag{39}$$

$$EI_Y y = \frac{PLx^2}{2} - \frac{Px^3}{6} + C_1 x + C_2 \tag{40}$$

Where, C_1 and C_2 are constants.

Implementing boundary conditions:

For $x = 0$,

$$y = 0 \rightarrow C_2 = 0 \tag{41}$$

For $x = 0$,

$$\frac{dy}{dx} = 0 \rightarrow C_1 = 0 \tag{42}$$

As a result:

$$\frac{dy}{dx} = \theta = \frac{P}{EI_Y}\left(Lx - \frac{x^2}{2}\right) \tag{43}$$

$$y = \frac{P}{2EI_Y}\left(Lx^2 - \frac{x^3}{3}\right) \tag{44}$$

Maximum deflection and maximum slope occur at the end of the tool.

For $x = L$

$$y = y_{max} = \frac{PL^3}{3EI_Y} \tag{45}$$

$$\theta = \theta_{max} = \frac{PL^2}{2EI_Y} \tag{46}$$

Moment of inertia can be calculated as follows:

$$I_Y = \frac{\pi d^4}{64} \tag{47}$$

where, “d” is the diameter of the tool.

6.2 Kinematic Calculations for the movements with respect to the tool tip.

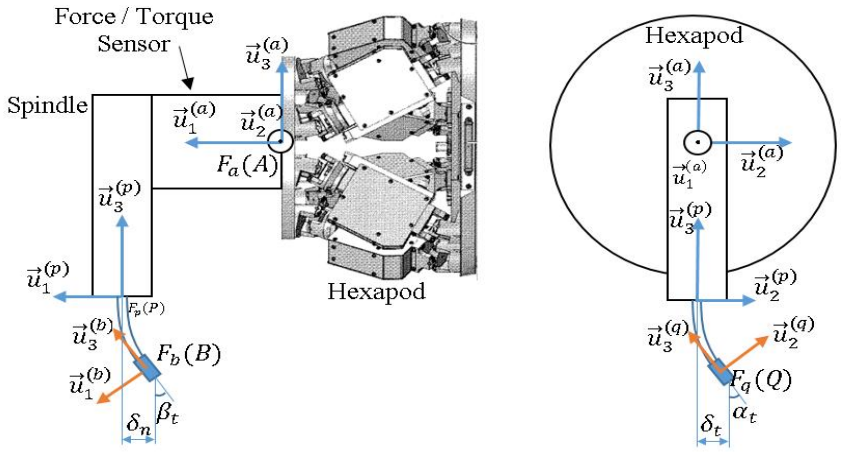


Figure 38 - Illustration of tool deflection and reference frames for kinematic calculations

Kinematic transformation matrix will be based on inertial reference frame.

$$F_a \xrightarrow{\hat{R}_{CMM}^{(a)}} F_p \xrightarrow{\vec{u}_1^{(a)},\alpha_t} F_q \xrightarrow{\vec{u}_2^{(a)},\beta_t} F_b \tag{48}$$

$$\hat{C}^{(a,b)} = \hat{C}^{(a,p)}\hat{C}^{(p,q)}\hat{C}^{(q,b)} \tag{49}$$

The transformation matrix between frames F_a and F_p is obtained from CMM measurement device ($\hat{R}_{CMM}^{(a)}$). Therefore:

$$\hat{C}^{(a,b)} = \hat{R}_{CMM}^{(a)}\, e^{\tilde{u}_1^{(a/p)},\alpha_t}\, e^{\tilde{u}_2^{(a/q)},\beta_t} \tag{50}$$

Solving the components of (50):

(51)

$$= \hat{R}_{CMM}^{(a)\ t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)}$$

$$e^{\tilde{u}_2^{(a/q)}, \beta_t} = \hat{C}^{(q,a)} e^{\tilde{u}_2^{(a/a)} \beta_t} \hat{C}^{(a,q)} \tag{52}$$

Where

$$\begin{aligned} \hat{C}^{(a,q)} &= \hat{R}_{CMM}^{(a)}\, e^{\tilde{u}_1^{(a/p)} \alpha_t} \\ &= \hat{R}_{CMM}^{(a)} \hat{R}_{CMM}^{(a)\ t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)} \\ &= e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)} \end{aligned} \tag{53}$$

$$\hat{C}^{(q,a)} = \hat{R}_{CMM}^{(a)\ t} e^{-\tilde{u}_1 \alpha_t} \tag{54}$$

Therefore (52) becomes:

$$e^{\tilde{u}_2^{(a/q)}, \beta_t} = \hat{R}_{CMM}^{(a)\ t} e^{-\tilde{u}_1 \alpha_t} e^{\tilde{u}_2^{(a/a)} \beta_t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)} \tag{55}$$

As a result, the forward kinematics transformation matrix of the robot is:

$$\begin{aligned} \hat{C}^{(a,b)} &= \hat{R}_{CMM}^{(a)}\, \hat{R}_{CMM}^{(a)\ t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)}\, \hat{R}_{CMM}^{(a)\ t} e^{-\tilde{u}_1 \alpha_t} e^{\tilde{u}_2^{(a/a)} \beta_t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)} \\ &= e^{\tilde{u}_2 \beta_t} e^{\tilde{u}_1 \alpha_t} \hat{R}_{CMM}^{(a)} \\ &= \hat{R}_y^{(a)} \hat{R}_x^{(a)} \hat{R}_{CMM}^{(a)} \end{aligned} \tag{56}$$

That is;

$$\hat{R}_{Forward} = \hat{R}_y^{(a)} \hat{R}_x^{(a)} \hat{R}_{CMM}^{(a)}$$

Rotation of the tool tip around x axis due to the generated grinding forces:

$$\hat{R}_X^{(a)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_t) & -\sin(\alpha_t) \\ 0 & \sin(\alpha_t) & \cos(\alpha_t) \end{bmatrix} \tag{57}$$

Where α_t is the rotation angle around x axis and is calculated by double integration method.

Rotation of the tool tip around y axis due to the generated grinding forces:

$$\hat{R}_y^{(a)} = \begin{bmatrix} \cos(\beta_t) & 0 & \sin(\beta_t) \\ 0 & 1 & 0 \\ -\sin(\beta_t) & 0 & \cos(\beta_t) \end{bmatrix} \tag{58}$$

Where β_t is the rotation angle around y axis and is calculated by double integration method.

Similarly, tool forward kinematics translation is:

$$T_{Forward} = \begin{bmatrix} r_x + \delta n \\ r_y + \delta t \\ r_z - L \end{bmatrix} \tag{59}$$

Where r_x, r_y and r_z are the distances from moving plate of the hexapod to spindle tip. L is the tool length, δt is the tool deflection in tangential direction due to grinding forces. δn is the tool deflection in normal direction due to grinding forces. δt and δn are calculated by double integration method.

Finally, transformation homogenous matrix for forward kinematics is:

$$\hat{H}_{FK}^{(a)} = \begin{bmatrix} \hat{R}_{Forward} & T_{Forward} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \tag{60}$$

Therefore, transformation homogenous matrix for new hexapod position and orientation is:

$$\hat{H}_{HexapodNew} = \hat{H}_{desiredTool} \hat{H}_{FK}^{(a)^{-1}} \tag{61}$$

where $\hat{H}_{desiredTool}$ is the desired transformation homogenous matrix for the tool.

The Euler angles to be given to the hexapod are calculated from $\hat{H}_{HexapodNew}$.

The rotated hexapod is shown in Figure 39. After the rotation, the reference frame of the force torque sensor is rotated as well. In order to obtain actual interaction forces, a transformation between force/torque reference frame and F_b is performed as well.

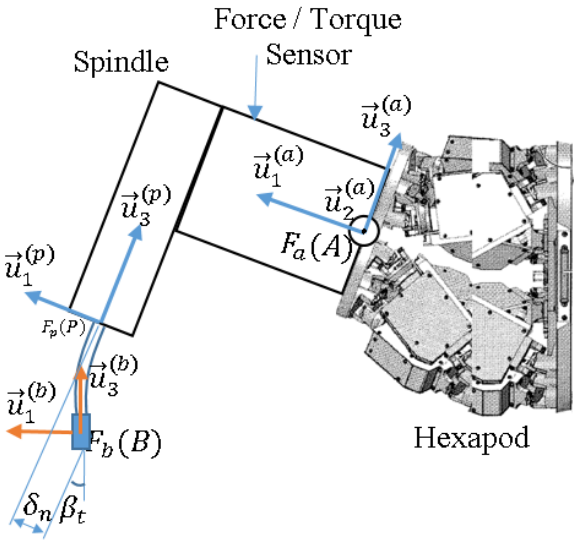


Figure 39 - Rotated hexapod (Note that the tool surface is parallel to the workpiece surface)

CHAPTER 7

EXPERIMENTAL PROCEDURE

MATLAB SIMULINK was utilized for controlling the devices of the grinding setup. The “Plant” block without its control part is shown in Figure 42. Note that hexapod orientation compensation is considered as a separate independent loop.

In Figure 42 “Control Input” is the amount of change in the piezo position. It is added to the current position of the piezo actuator in “Add1” block. After that the signal goes through a saturation block before reaching the rate limiter. The purpose of the rate limiter is to prevent piezo actuator to exceed the set feedrate value since the piezo actuator can perform high frequency movements. In the next step, the signal is given to the Humusoft MF624 DAQ by “Analog Output to Piezo1” block in order to be sent to the piezo actuator.

In order to calculate the velocity of the hexapod robot, the data which is given to the piezo actuator goes through a discrete derivative block in order to determine the velocity of the piezo actuator. After that the velocity of the hexapod robot is calculated from the set feedrate value with (11).

The generated force values are obtained by “Feedrate from F/T Sensor1” block. Then biasing is performed by block “Calculation of the biased forces and moments” Finally, the normal force is calculated with (4).

One of the most important drawbacks of water jet cutting is the inclined surface form it left as shown in Figure 53. In industry these parts are machined again in order to obtain flat surface profile. However, this process constitutes the significant amount of time due to the calibration process of the robot and workpiece. With the proposed method, significant amount of time and effort can be saved.

After water jet cutting, the material difference between upper point and the lower point of the sample is 280 μm. In order to correct this inclined surface, grinding operation is performed by keeping the normal grinding force constant. The force value is taken from a grinding model[11].

Single pass grinding operation was performed.

The used sample form is shown in Figure 40 - Sample

Figure 40 - Sample

7.1 Experiment with PID controller

The used SIMULINK Model is shown in Figure 41.

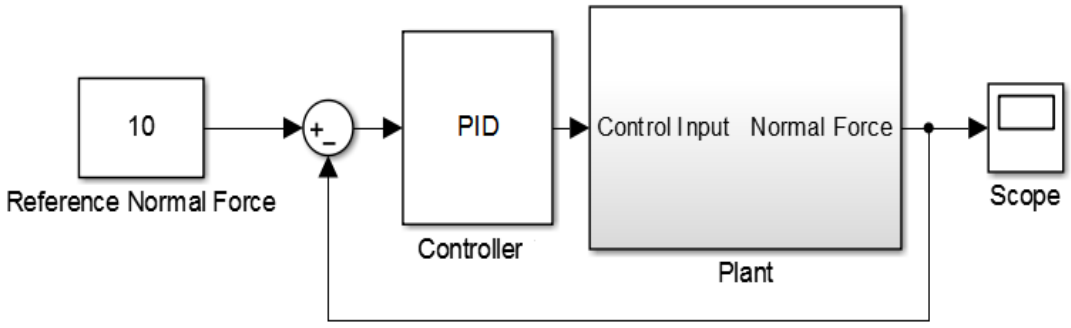


Figure 41 - Used SIMULINK Model for PID Control

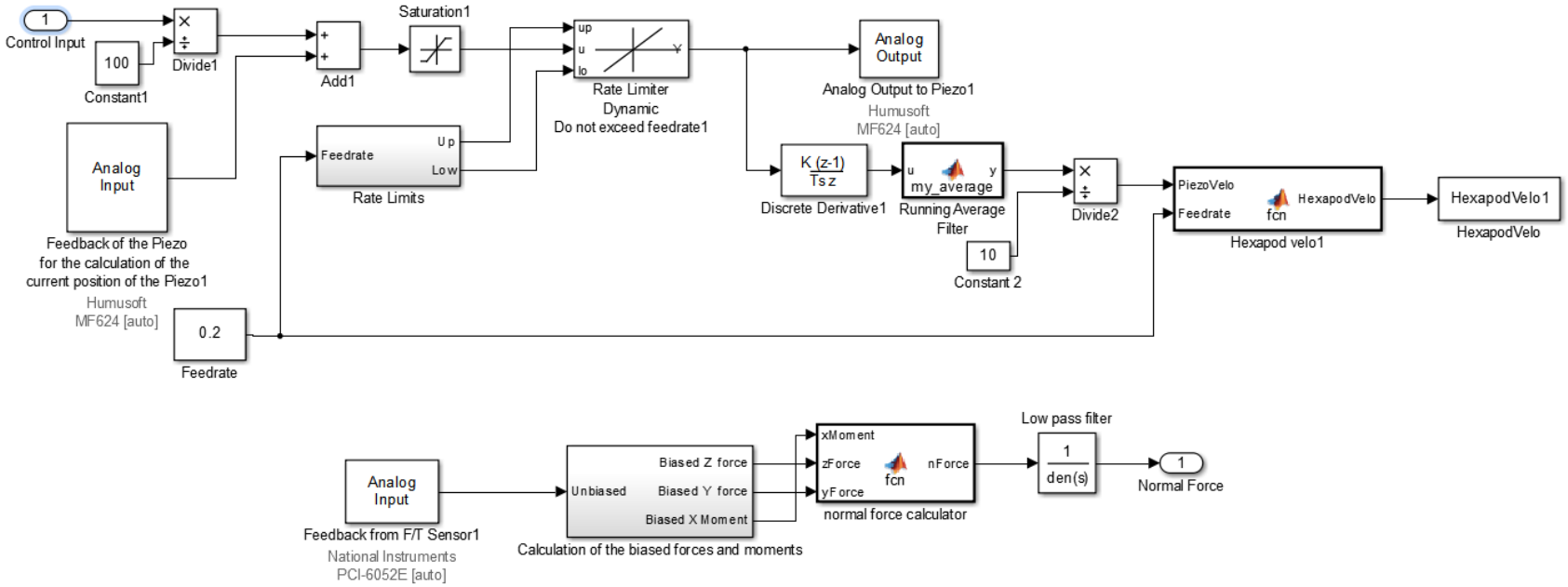


Figure 42 - Plant SIMULINK Model

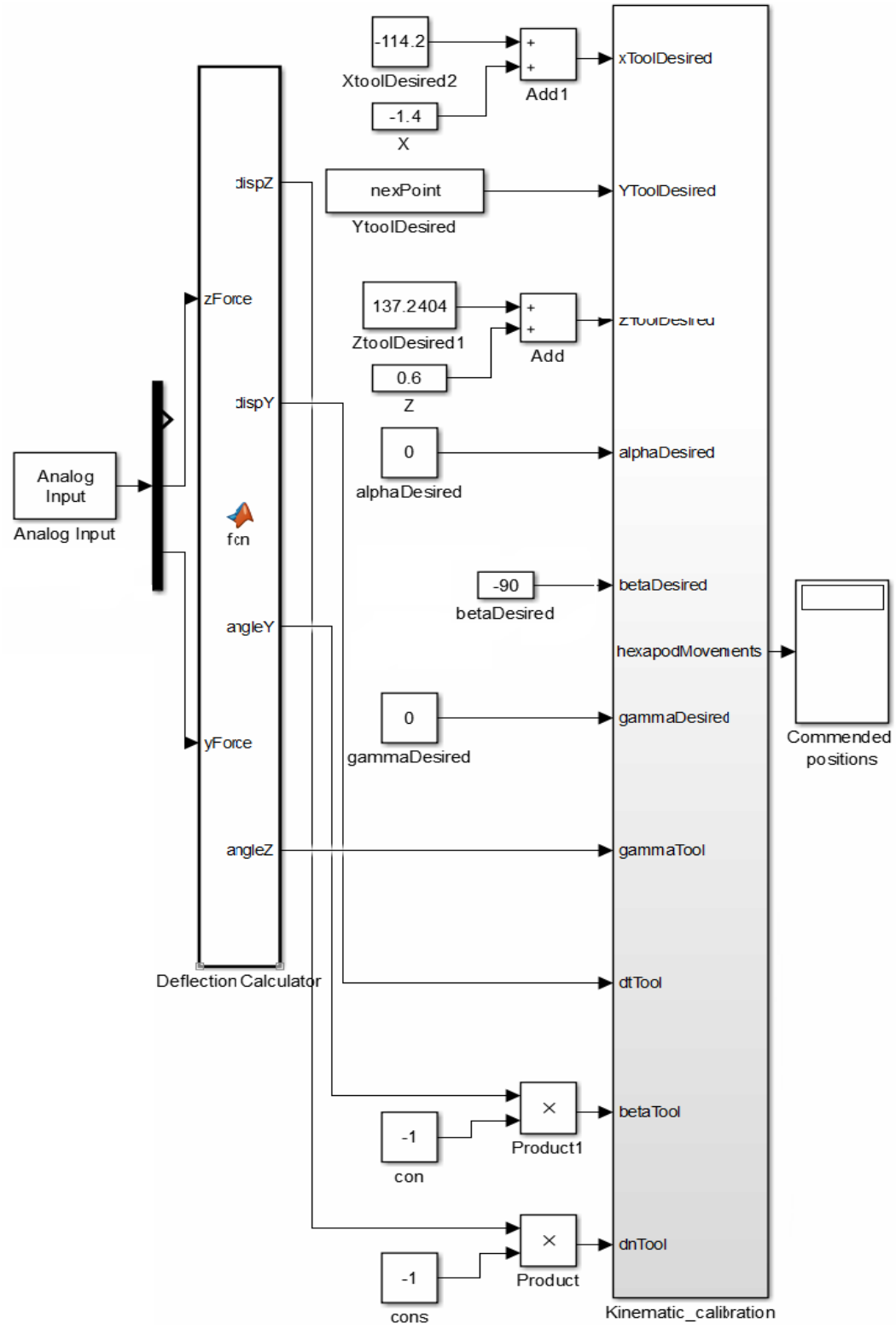


Figure 43 - Hexapod Orientation Compensation SIMULINK Model

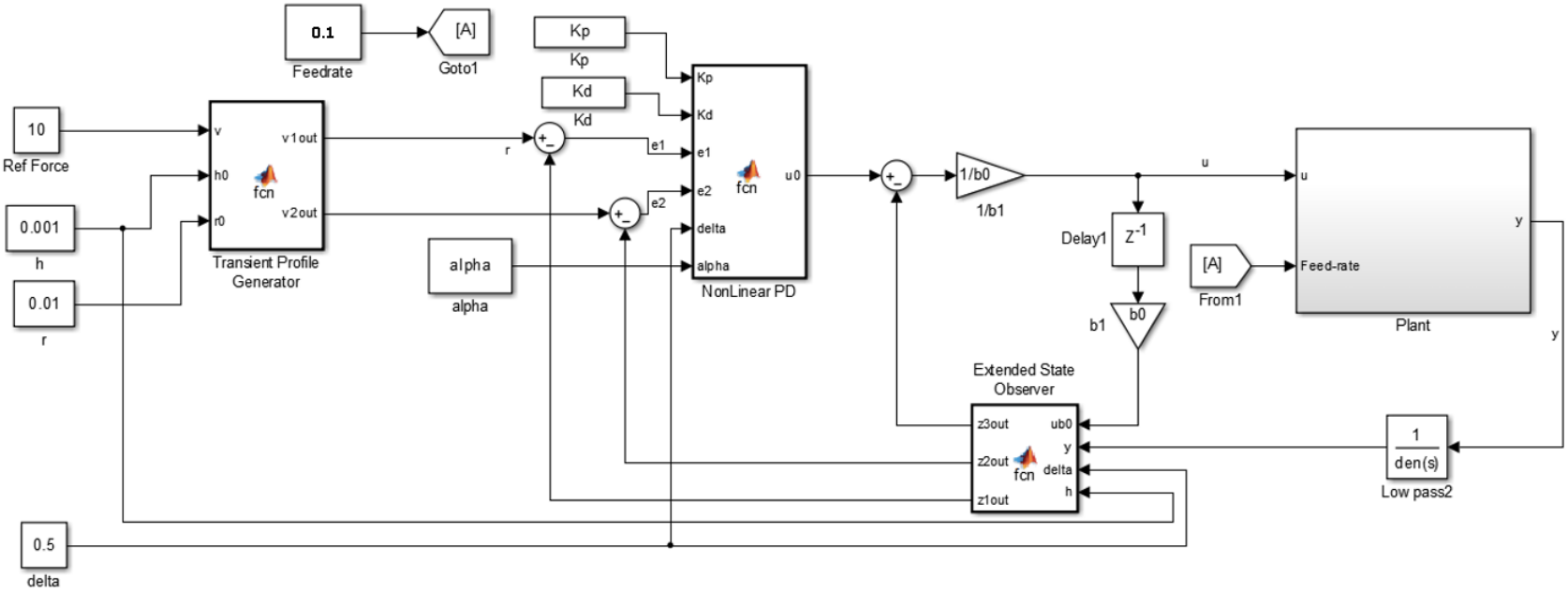


Figure 44 - Used SIMULINK Model for ADRC

The PID parameters were determined by genetic algorithm as:

- Kp: 84.06
- Kd: 14.1
- Ki: -0.01766

The experiment parameters are:

- Set feedrate: 0.1 mm/sec
- Spindle speed: 25000 rpm
- Material: 4mm ST37
- Diameter of the used CBN tool: 4mm
- Down-cut grinding operation
- Length of the tool: 23.5 mm

7.2 Experiment with ADRC

The used SIMULINK Model is shown in Figure 44.

The controller parameters were determined by genetic algorithm as:

- Kp: 61.17
- Kd: 78.64
- alpha: 0.9448
- b0: 0.543

The experiment parameters are:

- Set feedrate: 0.1 mm/sec
- Spindle speed: 25000 rpm
- Material: 4mm ST37
- Diameter of the used CBN tool: 4 mm
- Down-cut grinding operation

- Length of the tool: 23.5 mm

7.3 Contour Tracking Experiment

In order to detect the surface form of unknown profile, contour traking operation can be performed. For this purpose an experiment was conducted with the following properties:

- Used Control Method: PID
- Kp: 84.06
- Kd: 14.1
- Ki: -0.01766

The experiment parameters are:

- Set feedrate: 0.1 mm/sec
- Spindle speed: 0 rpm (freely rotatable on the workpiece)
- Material: 4mm ST37

CHAPTER 8

RESULTS OF THE EXPERIMENTS

Four experiments were conducted in order to test the performance of the controller. Three of them was grinding experiment and the remaining one was contour tracking experiment. Two of grinding experiments was by PID controller with different reference normal forces and the other one was by ADRC.

Additionally one more experiment was conducted in order to obtain desired sinusoidal surface profile.

8.1 Experiment with PID controller, Fn = 7.5 N

The variation of the local normal force when it is set to 7.5 N is shown in Figure 45.

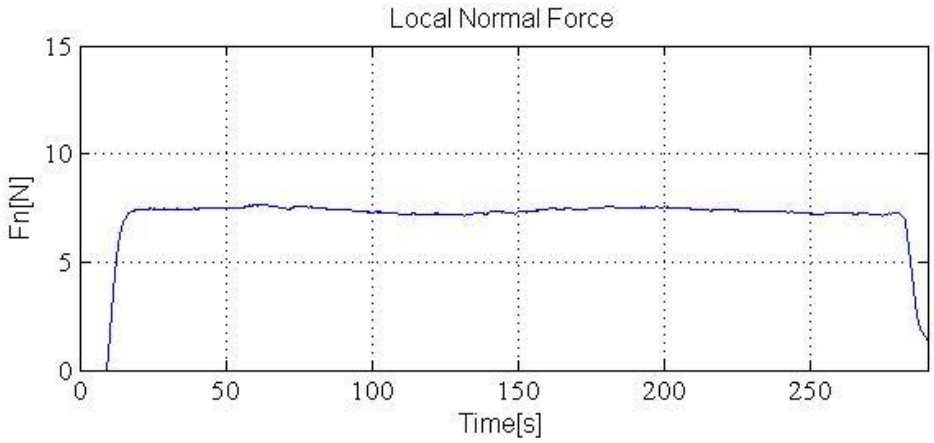


Figure 45 - Local Normal Force (PID Fn=7.5N)

The movements of the piezo actuator are shown in Figure 46

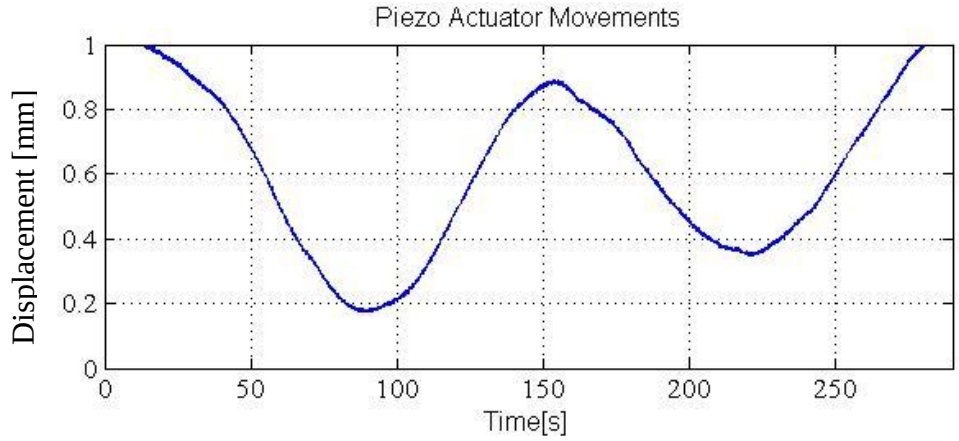


Figure 46 - Piezo Actuator Movements (PID Fn=7.5N)

The velocity of the piezo actuator after discrete derivation is shown in Figure 47.

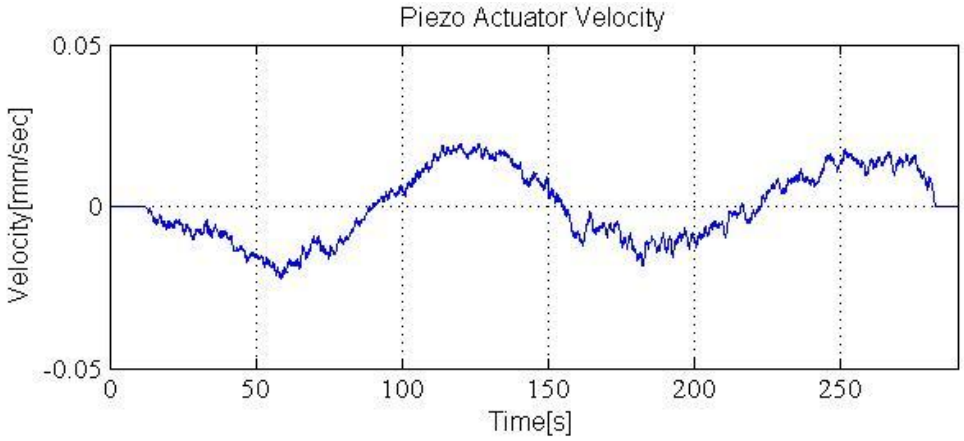


Figure 47 - Piezo Actuator Velocity (PID Fn=7.5N)

The velocity of hexapod robot is shown in Figure 48.

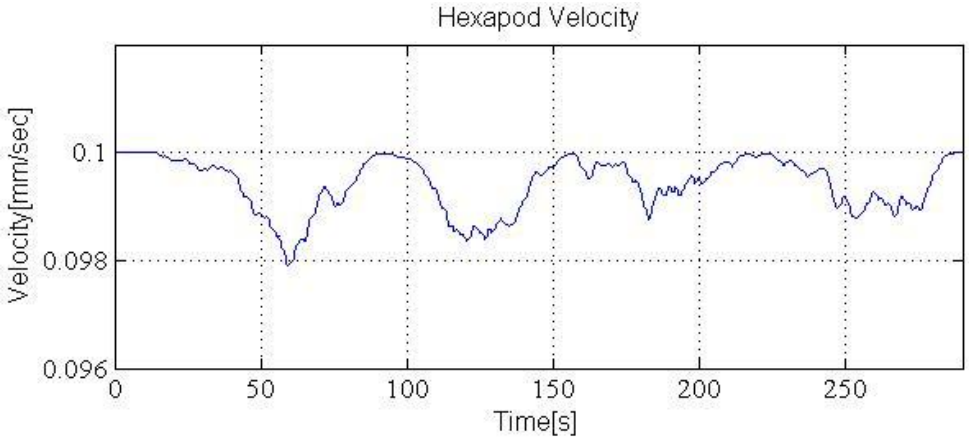


Figure 48 - Hexapod Velocity (PID Fn=7.5N)

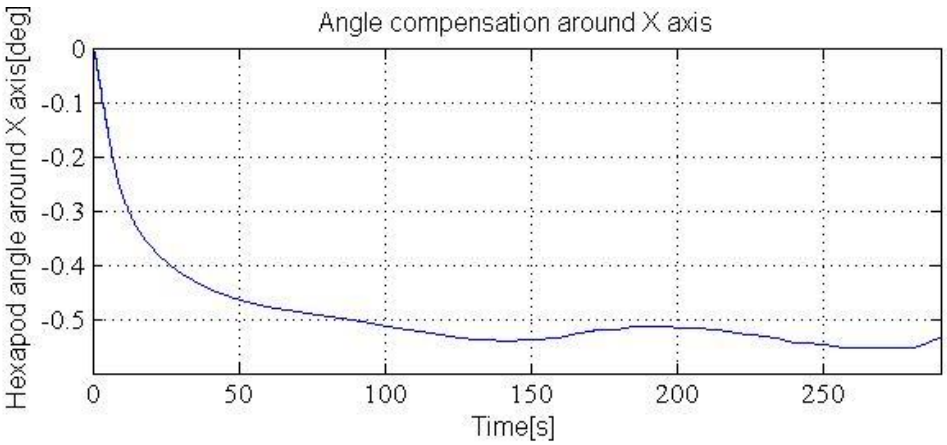


Figure 49 - Hexapod angle variation around X direction (PID Fn=7.5N)

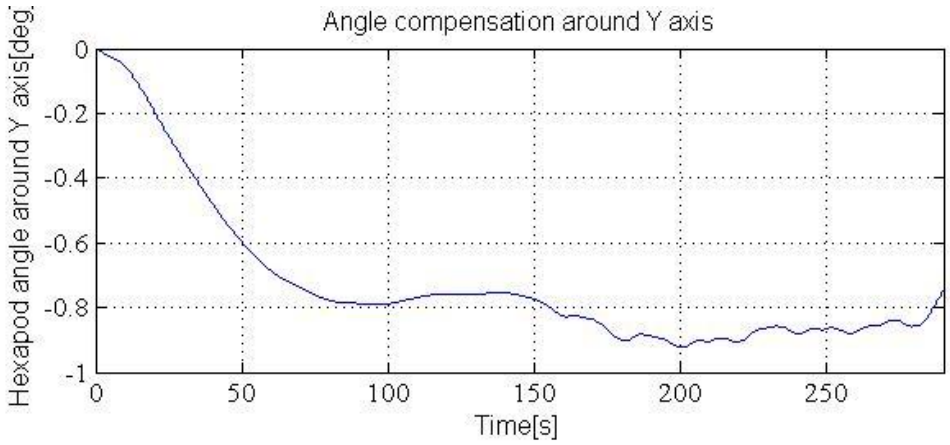


Figure 50 - Hexapod angle variation around Y direction (PID Fn=7.5N)

The change in the surface form is shown in Figure 51.

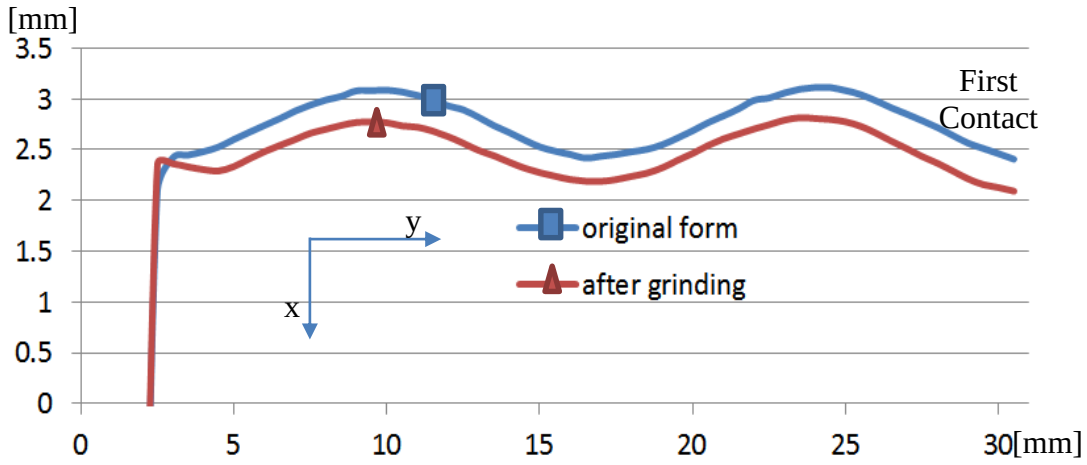


Figure 51 - Surface forms before and after grinding operation. The measurement is taken from the mid-section of the workpiece (PID Fn=7.5N)

In order to compare two surface forms, shown in Figure 51, they were superimposed and mean square error was calculated as 0.00174mm^2.

Additionally, several scans were performed in z direction in order to understand the success of angle compensation controller. After that lines were fit to the scans. Table 1 illustrates the angles of these lines with respect to the vertical axis.

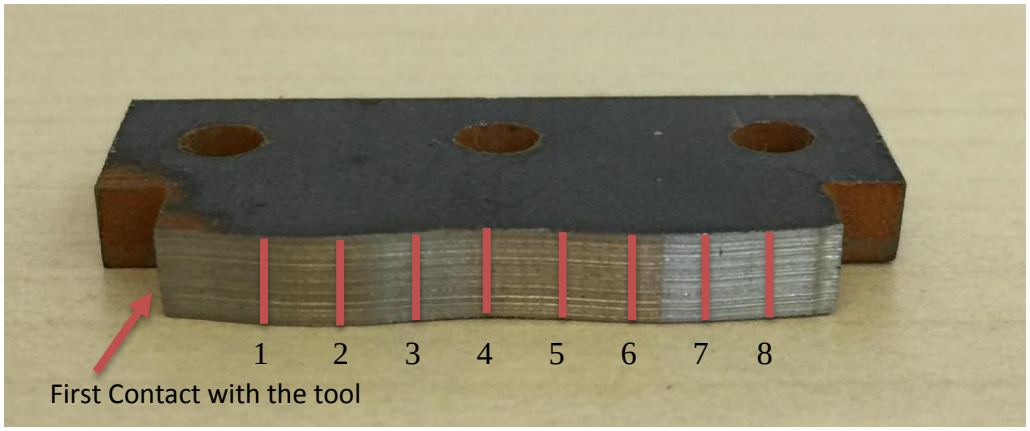


Figure 52 - Ground workpiece

Table 1 - Angles of the fit lines in Figure 52 (PID Fn=7.5N)

Numbered sections in Figure 52	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
The angle of the surface form with respect to the vertical axis (deg)	**-0,15**	**-0,17**	**-0.12**	**-0.14**	**-0.10**	**-0.06**	**-0.09**	**-0.12**

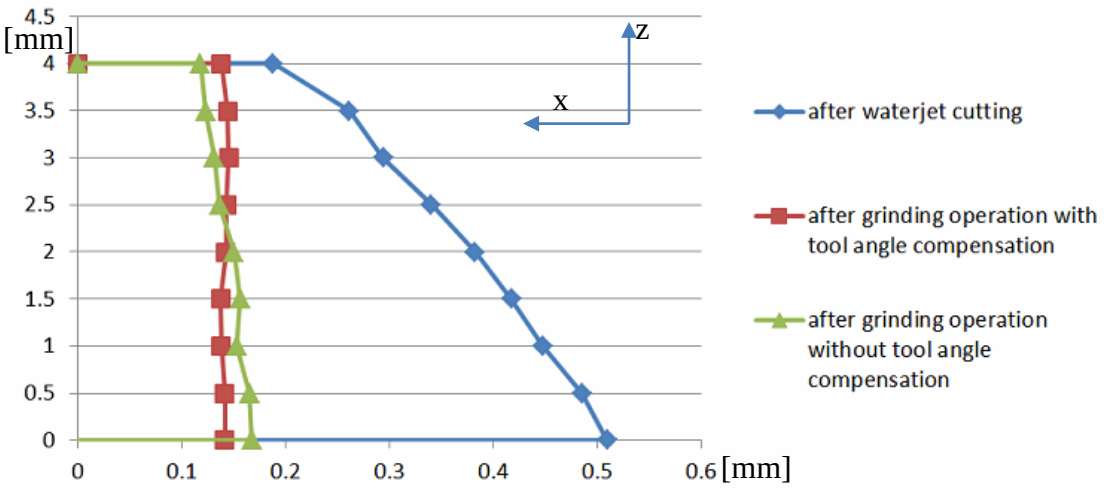


Figure 53 - Vertical scanning (z direction) of the sample surface form (PID Fn=7.5N)

As it is understood from, Figure 46, the work-piece is not well-aligned with respect to the piezo actuator in y direction. However, when Figure 51 is considered, misalignment does not seem to cause a problem which is very good advantage of the force control as it is explained in [8]. Additionally, with force control, the disadvantages due to tool wear are eliminated as well.

From piezo actuator and hexapod velocity figures, it can be seen that the local feedrate is kept constant at 0.1 mm/sec. The reaction of hexapod against the force variations can be seen in Figure 49 and Figure 50. The 4° angle (the angle after water jet cutting) with respect to the vertical axis was reduced to approximately 0.1° while it can be decreased to 0.5° without angle compensation.

8.2 Experiment with PID controller, Fn = 10 N

The variation of the local normal force when it is set to 10 N is shown in Figure 54.

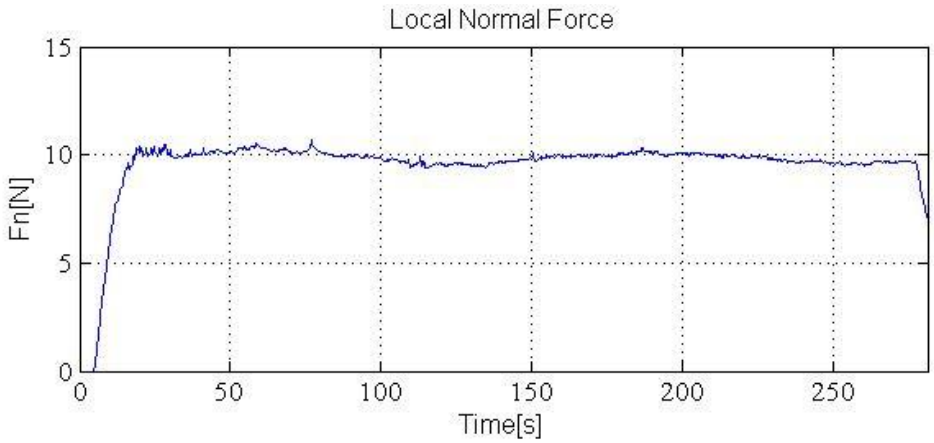


Figure 54 - Local Normal Force (PID Fn=10N)

The movements of the piezo actuator are shown in Figure 55.

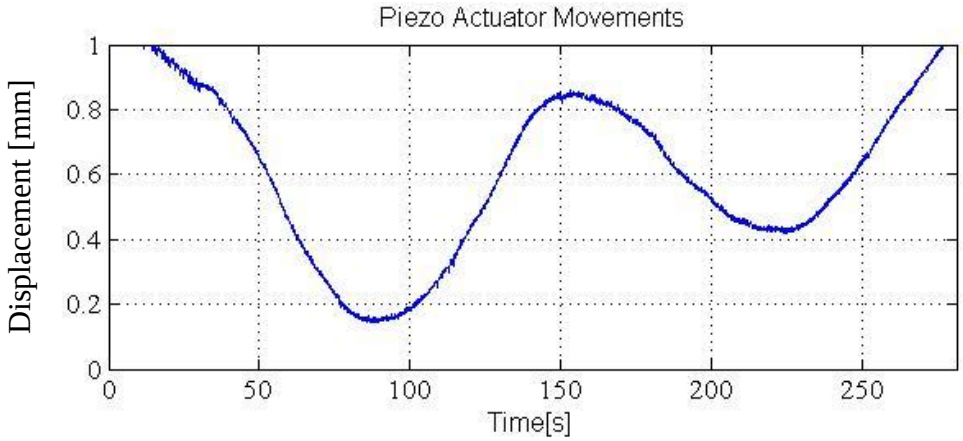


Figure 55 - Piezo Actuator Movements (PID Fn=10N)

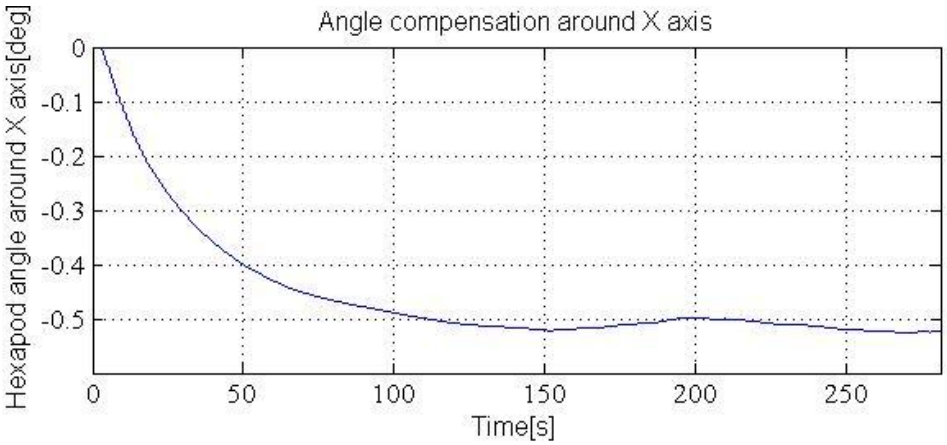


Figure 56 - Hexapod angle variation around X direction (PID Fn=10N)

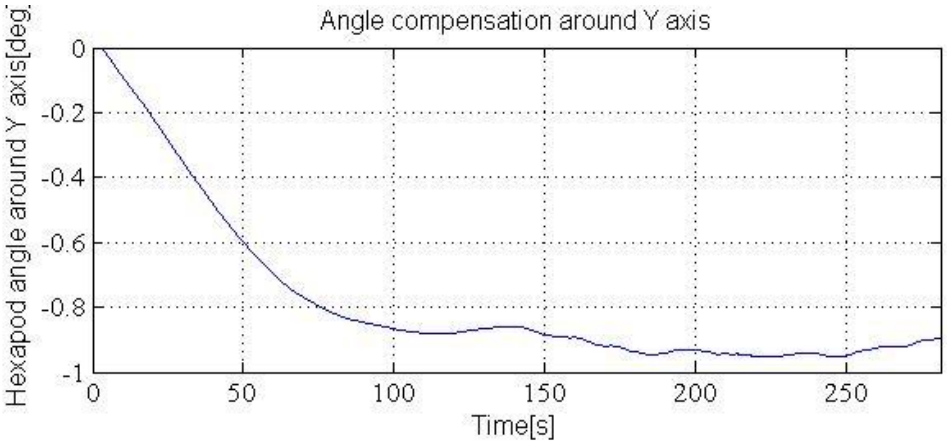


Figure 57 - Hexapod angle variation around Y direction (PID Fn=10N)

The change in the surface form is shown in Figure 58.

In order to compare two surface forms, shown in Figure 58, they were superimposed and mean square error was calculated as 0.0036mm^2.

Additionally, several scans were performed in z direction in order to understand the success of angle compensation controller. After that lines were fit to the scans. Table 2 illustrates the angles of these lines with respect to the vertical axis.

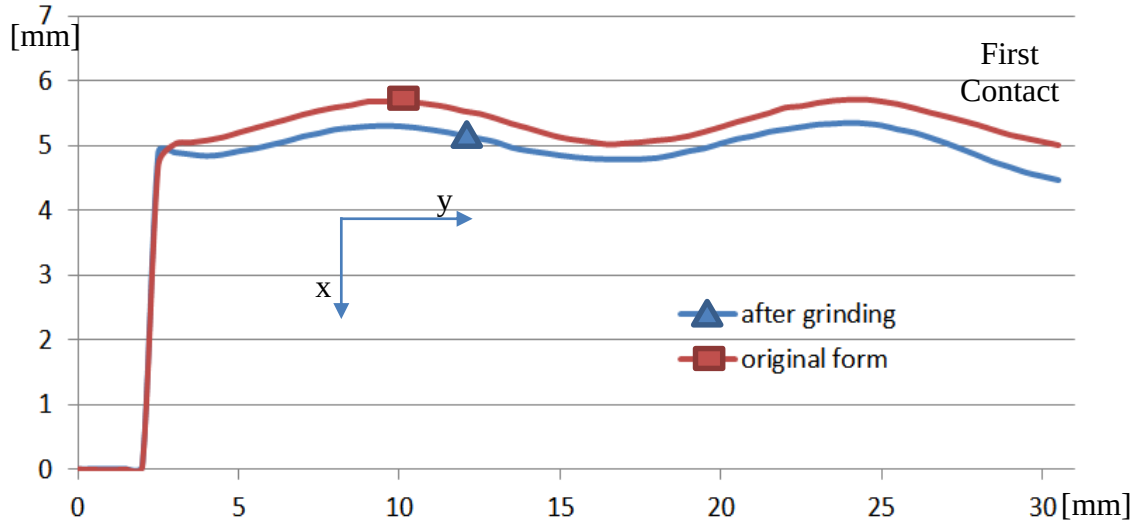


Figure 58 - Surface forms before and after grinding operation. The measurement is taken from the mid-section of the workpiece (PID Fn=10N)

Table 2 - Angles of the fit lines in Figure 52 (PID Fn=10N)

Numbered sections in Figure 52	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
The angle of the surface form with respect to the vertical axis (deg)	**-0,16**	**-0,12**	**-0.13**	**-0.11**	**-0.05**	**-0.09**	**-0.09**	**-0.10**

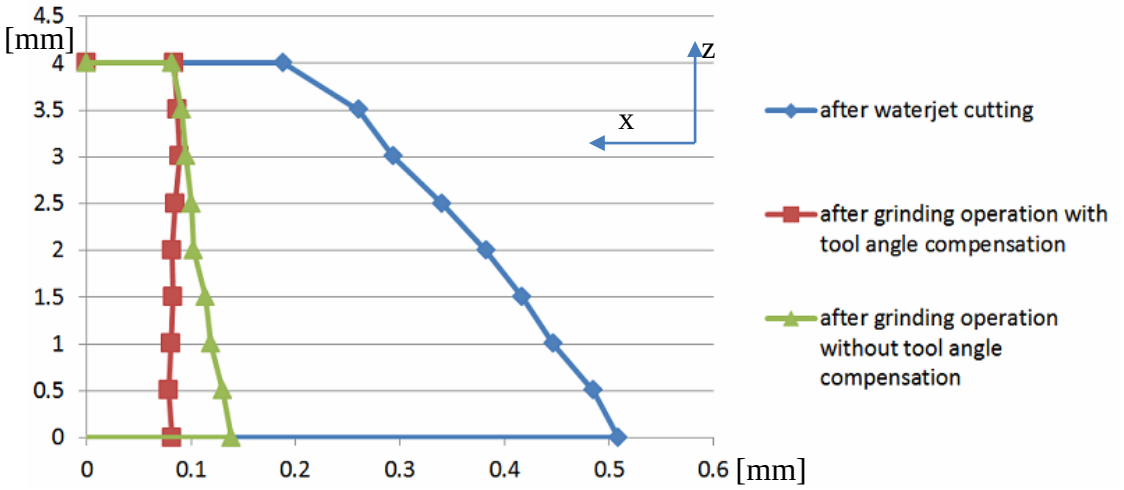


Figure 59 - Vertical scanning (z direction) of the sample surface form (PID Fn=10N)

8.3 Experiment with ADRC, Fn = 7.5 N

The variation of the local normal force when it is set to 7.5 N is shown in Figure 60.

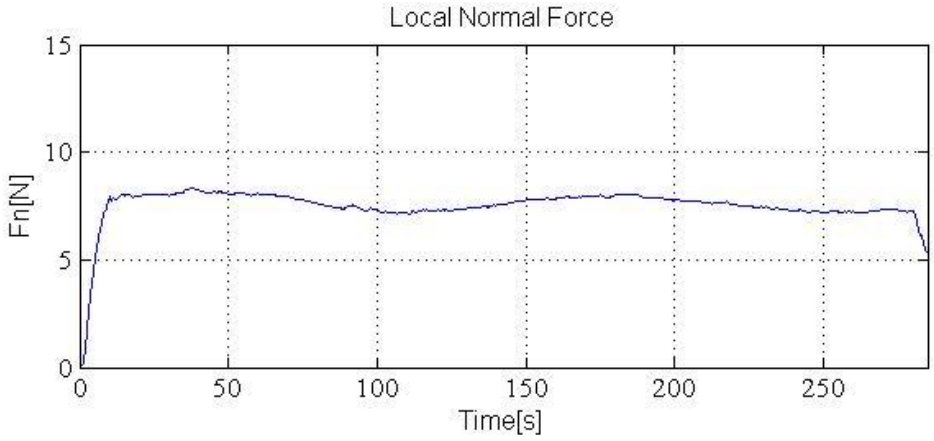


Figure 60 - Local Normal Force (ADRC Fn=7.5N)

The movements of the piezo actuator are shown in Figure 61.

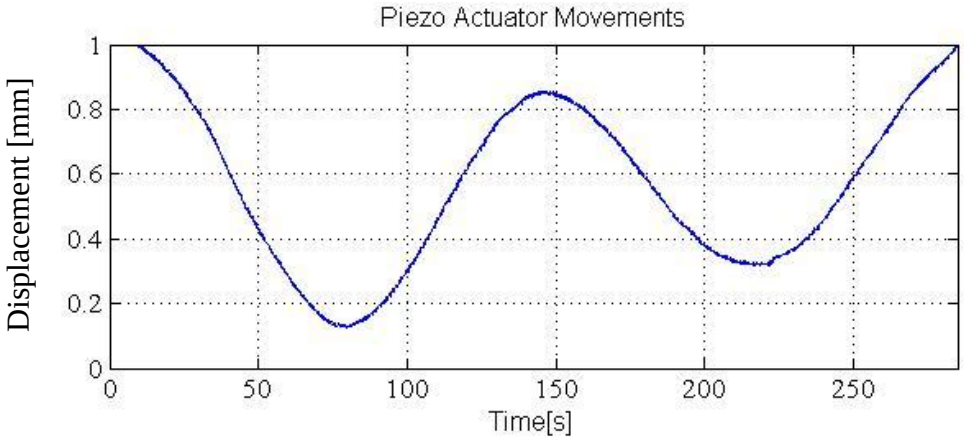


Figure 61 - Piezo Actuator Movements (ADRC Fn=7.5N)

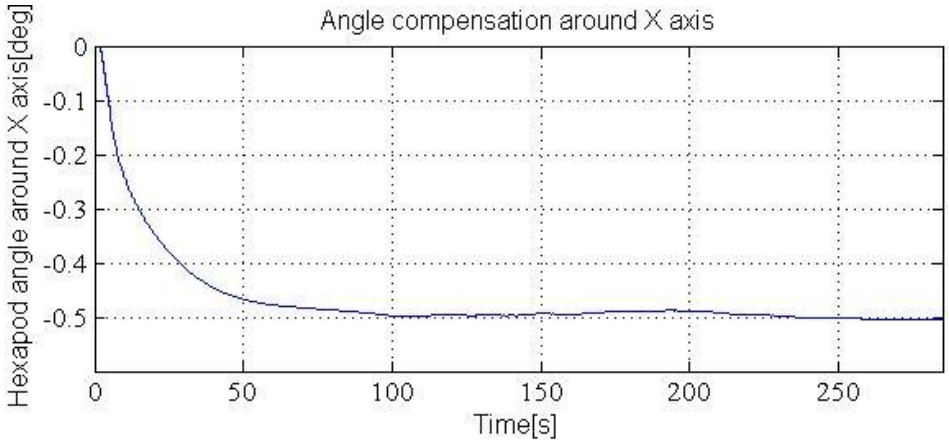


Figure 62 - Hexapod angle variation around X direction (ADRC Fn=7.5N)

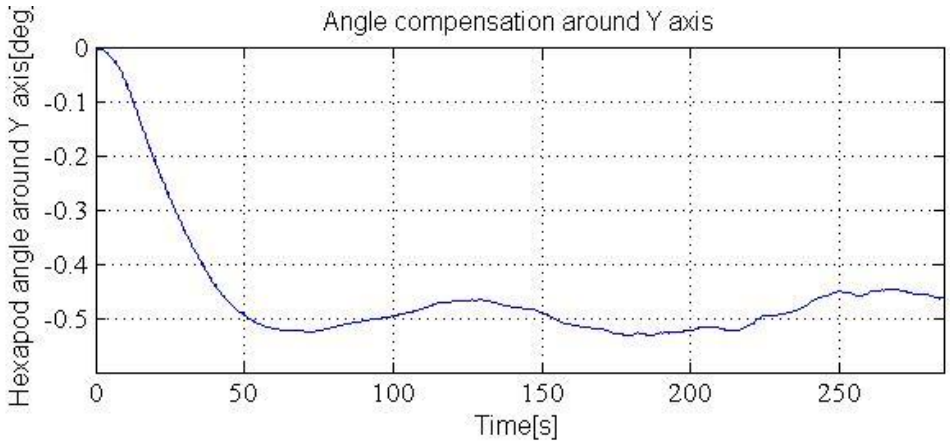


Figure 63 Hexapod angle variation around Y direction (ADRC Fn=7.5N)

The change in the surface form is shown in Figure 64.

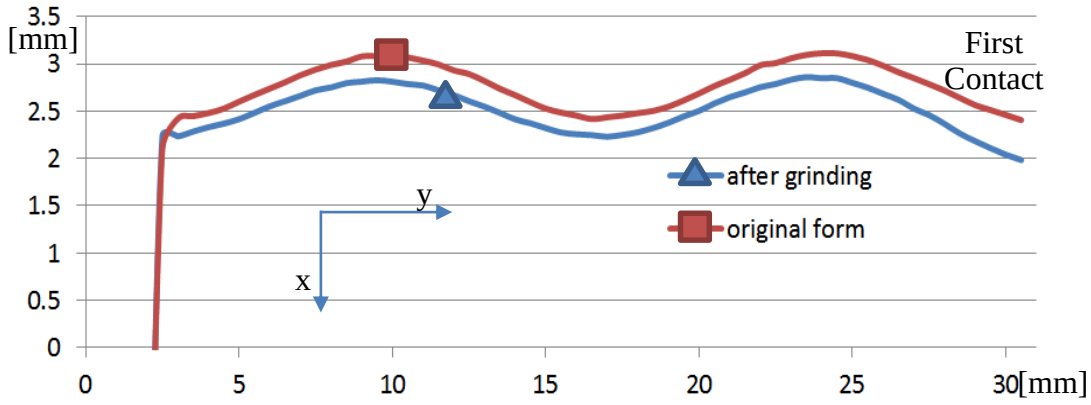


Figure 64 Surface forms before and after grinding operation. The measurement is taken from the mid-section of the workpiece (ADRC Fn=7.5N)

In order to compare two surface forms, shown in Figure 58, they were superimposed and mean square error was calculated as 0.0043 mm^2.

Additionally, several scans were performed in z direction in order to understand the success of angle compensation controller. After that lines were fit to the scans. Table 3 illustrates the angles of these lines with respect to the vertical axis.

Table 3 - Angles of the fit lines in Figure 52 (ADRC Fn=7.5N)

Numbered sections in Figure 52	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
The angle of the surface form with respect to the vertical axis (deg)	**-0,19**	**-0,16**	**-0.13**	**-0.11**	**-0.12**	**-0.09**	**-0.13**	**-0.12**

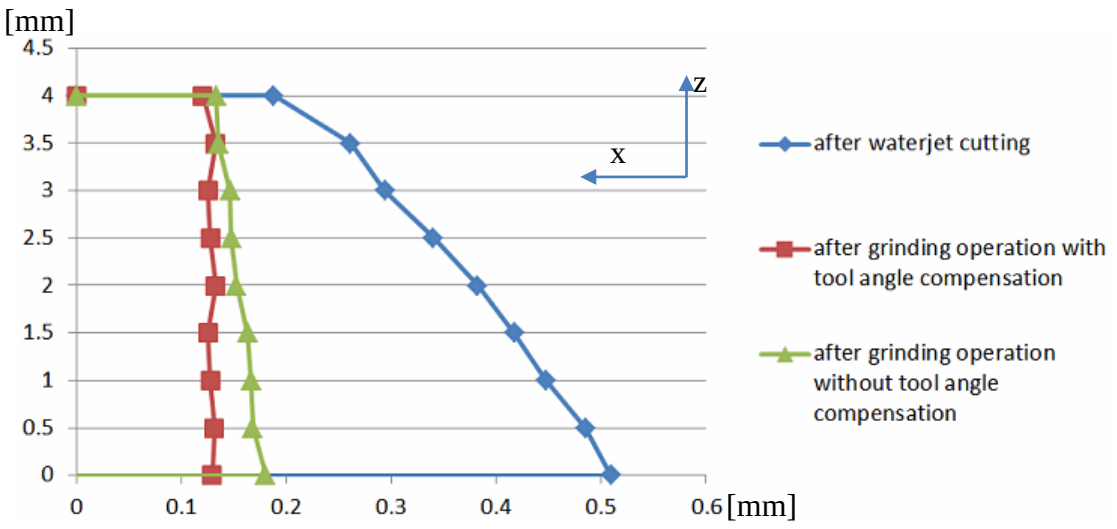


Figure 65 - Vertical scanning (z direction) of the sample surface form (ADRC Fn=7.5N)

8.4 Contour Tracking Experiment, Fn = 1 N

The two surface profiles (obtained by laser measurement and contour tracking) of the same sample are shown in Figure 66.

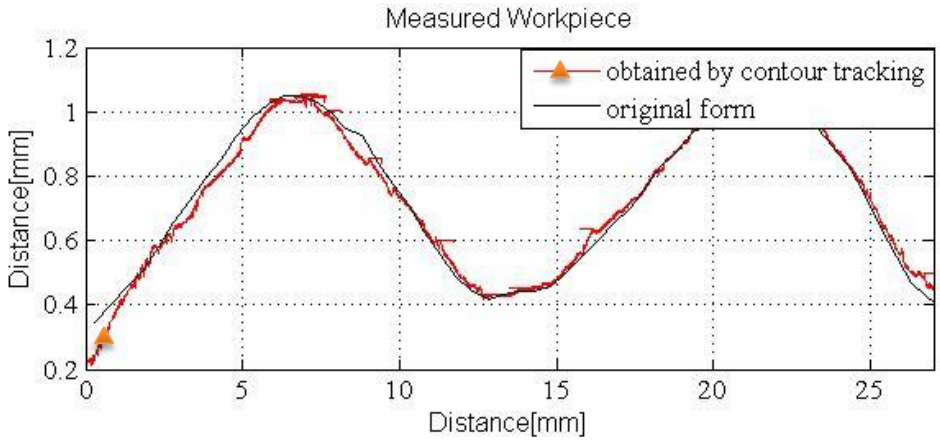


Figure 66 – Two surface profiles obtained from the same sample by 2 different measurement methods

The mean square error of two surface forms is 0.000724 mm^2

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 Conclusion

In this study, hybrid force/velocity control method was implemented to the robotic grinding experimental setup with tool angle compensation. While the local normal force is tried to be kept constant by the movements of the piezo actuator, the compensation of the local tangential velocity and tool compensation are performed by the hexapod robot. Based on literatüre review, this is the first study in which these two control algorithms were used simultaneously in real time. The results show that significant amount of path planning and calibration effort can be saved by the proposed approach. The surface angle which was created by water jet cutting was eliminated.

Two different control algorithms were tested. The mean square error of the surface profile obtained by ADRC is approximately 2.5 times greater than the surface profile obtained by PID controller. It has been shown that PID method results in better than ADRC in this application.

It was proven that contour tracking operation can be utilized in order to measure the surface form of the sample.

Additionally, it was shown that the modelling in SIMULINK environment is a useful approach for developing and testing a controller. The results obtained by the controller are in accordance with the results obtainded by simulation using SIMULINK model.

9.2 Future Work

Surface quality of the workpiece after the robotic grinding operation may not be at required level for all applications. In an attempt to obtain better surface quality and to

reduce surface roughness, another improvement would be done by changing the piezo electric component to the bottom of the workpiece. By doing so, existed system would have less movement capability along the x-axis, yet this deficiency can be atoned by the hexapod manipulator. However, placing the piezo actuator to the bottom of the worktable would allow us the vibrate workpiece along z axis during the grinding operation at a desired frequency. Advantage of movement of the z axis would be the elimination of the uneven surface of the workpiece in z direction at the end of the process.

The purpose of this study is applying robotic grinding process by preserving the initial shape of the material. By applying constant force control, it is proven that the workpiece has the same surface profile after desired amount of material is removed. Yet, a significant improvement would be adjusting the final surface shape of the workpiece to desired one after the machining operation. Applying an angle to the surface of the material during the operation would grant us the opportunity of changing the final form of the workpiece rather than preserving the initial one. This enhancement would be applied by controlling the hexapod angle during the grinding process.

Final improvement would be the investigation of the effect of the temperature to the grinding operation. Due to nature of the machining, temperature at the contact point between tool and workpiece increases. Generated heat because of the friction, has an effect on both workpiece as well as cutting tool. Generation of a model which investigates the changes on workpiece material properties or effects of high temperature on cutting performance would improve the performance of the robotic grinding application.

REFERENCES

[1] R. Bernard and S. Albright, *Robot calibration*. Springer Science \& Business Media, 1993.

[2] J. Pandremenos, C. Doukas, P. Stavropoulos, and G. Chryssolouris, "Machining with robots: A critical review," in *7th International Conference on Digital Enterprise Technology*, 2011, no. September.

[3] A. Lopes and F. Almeida, "A force-impedance controlled industrial robot using an active robotic auxiliary device," *Robot. Comput. Integr. Manuf.*, vol. 24, no. 3, pp. 299–309, 2008.

[4] J. De Schutter, "A study of active compliant motion control methods for rigit manipulators based on a generic scheme," in *Robotics and Automation. Proceedings*, 1987, vol. 4, pp. 1060–1065.

[5] M. Krantz and R. Andersson, "Robotized Polishing and Deburring with Force Feedback Control," University West, 2010.

[6] M.C. Shaw, *Principles of Abrasive Processing*. Oxford University Press, 1996.

[7] C. C. H. Liu, A. Chen, C.-C. a. C. Chen, and Y. Y.-T. Wang, "Grinding force control in an automatic surface finishing system," *J. Mater. Process. Technol.*, vol. 170, no. 1–2, pp. 367–373, 2005.

[8] J. W. J. Wang, G. Zhang, H. Z. H. Zhang, and T. Fuhlbrigge, "Force control technologies for new robotic applications," in *2008 IEEE International Conference on Technologies for Practical Robot Applications*, 2008, no. 860, pp. 143–149.

[9] S. Son, T. Kim, S. E. Sarma, and A. Slocum, "A hybrid 5-axis CNC milling machine," *Precis. Eng.*, vol. 33, no. 4, pp. 430–446, 2009.

[10] V. I. Guzeev and A. K. Nurkenov, "Researching the CNC-Machine Stiffness Impact on the Grinding Cycle Design," *Procedia Eng.*, vol. 150, pp. 815–820, 2016.

[11] M. L. Navid and E. ilhan Konukseven, "Hybrid model based on energy and experimental methods for parallel hexapod-robotic light abrasive grinding operations," *Int. J. Adv. Manuf. Technol.*, 2017.

[12] X. Chen, W. B. Rowe, and R. Cai, "Precision grinding using CBN wheels,"

Int. J. Mach. Tools Manuf., vol. 42, no. 5, pp. 585–593, 2002.

[13] "CBN Tools." [Online]. Available: http://parasdiamondco.com/Diamond & CBN Internal Grinding Pins _Wheels.html. [Accessed: 23-Feb-2017].

[14] A. A. Akbari and S. Higuchi, "Autonomous tool adjustment in robotic grinding," *J. Mater. Process. Technol.*, vol. 127, no. 2, pp. 274–279, 2002.

[15] R. S. Hahn, "Controlled-Force Grinding — A New Technique for Precision Internal Grinding," *J. Eng. Ind.*, pp. 287–293, 1964.

[16] T. Thomessen, T. K. Lien, and P. K. Sannñs, "Robot control system for grinding of large hydro power turbines," *Ind. Robot An Int. J.*, vol. 28, no. 4, pp. 328–334, 2001.

[17] A. Winkler and J. Suchý, "Position feedback in force control of industrial manipulators - An experimental comparison with basic algorithms," in *2012 IEEE International Symposium on Robotic and Sensors Environments, ROSE 2012 - Proceedings*, 2012, pp. 31–36.

[18] F. Domroes, C. Krewet, and B. Kuhlenkoetter, "Application and Analysis of Force Control Strategies to Deburring and Grinding," *Mod. Mech. Eng.*, vol. 3, no. June, pp. 11–18, 2013.

[19] P. R. Pagilla and B. Yu, "Adaptive control of robotic surface finishing processes," in *Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148)*, 2001, vol. 1, pp. 630–635.

[20] S. Lee, C. Li, D. Kim, J. Kyung, and C. Han, "The direct teaching and playback method for robotic deburring system using the adaptive-force-control," in *2009 IEEE International Symposium on Assembly and Manufacturing, ISAM 2009*, 2009, no. November, pp. 235–241.

[21] N. Pedrocchi, A. Visioli, G. Ziliani, and G. Legnani, "On the elasticity in the dynamic decoupling of hybrid force/velocity control in the contour tracking task," in *2008 IEEE/RSJ Intern. Conf. on Intell. Robots and Sys., IROS*, 2008, pp. 955–960.

[22] C. H. Liu, a. Chen, Y.-T. Wang, and C.-C. a. Chen, "Modelling and simulation of an automatic grinding system using a hand grinder," *Int. J. Adv. Manuf. Technol.*, vol. 23, pp. 874–881, 2004.

[23] G. Ziliani, A. Visioli, and G. Legnani, "A mechatronic approach for robotic deburring," *Mechatronics*, vol. 17, no. 8, pp. 431–441, 2007.

[24] J. Park, S. H. Kim, and S. Kim, "Active compliant motion control for grinding robot," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2008, vol. 17,

pp. 4285–4289.

[25] K. Kashiwagi, K. Ono, E. Izumi, T. Kurenuma, and K. Yamada, “Force controlled robot for grinding,” in *IEEE International Workshop on Intelligent Robots and Systems*, 1990, pp. 1001–1006.

[26] A. Donder and E. I. Konukseven, “Spectral Coherence Analysis between Grinding Interaction Forces and the Relative Motion of the Workpiece and the Cutting Tool,” *Int. J. Mech. Aerospace, Ind. Mechatron. Manuf. Eng.*, vol. 10, no. 8, pp. 1417–1423, 2016.

[27] B. Denkena and O. Gümmer, “Active tailstock for precise alignment of precision forged crankshafts during grinding,” in *Procedia CIRP*, 2013, vol. 12, pp. 121–126.

[28] A. Rashid and C. Mihai Nicolescu, “Active vibration control in palletised workholding system for milling,” *Int. J. Mach. Tools Manuf.*, vol. 46, no. 12–13, pp. 1626–1636, 2006.

[29] “ATI Company.” [Online]. Available: http://www.ati-ia.com/. [Accessed: 23-Feb-2017].

[30] L. Liao, F. (Jeff) Xi, and K. Liu, “Modeling and control of automated polishing/deburring process using a dual-purpose compliant toolhead,” *Int. J. Mach. Tools Manuf.*, vol. 48, no. 12–13, pp. 1454–1463, 2008.

[31] J. Kroeker, “Design , Analysis & Testing of an Enhanced Radial- Axial Hybrid Compliant Deburring Tool,” Ryerson University, 2010.

[32] “ATI’s deburring tool family.” [Online]. Available: www.ati-ia.com/products/deburr/deburring_home.aspx. [Accessed: 08-Jun-2017].

[33] M. H. Raibert and J. J. Craig, “Hybrid Position / Force Control of Manipulators,” *J. Dyn. Syst. Meas. Control*, vol. 102, no. June 1981, 1981.

[34] F. Jatta, G. Legnani, A. Visioli, and G. Ziliani, “On the use of velocity feedback in hybrid force/velocity control of industrial manipulators,” *Control Eng. Pract.*, vol. 14, no. 9, pp. 1045–1055, 2006.

[35] F. Jatta, G. Legnani, and A. Visioli, “Hybrid force / velocity control of industrial manipulators with elastic transmissions,” in *Proceedings of the 2003 IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, 2003, no. October, pp. 3276–3281.

[36] F. Jatta, G. Legnani, and A. Visioli, “Friction Compensation in Hybrid Force / Velocity Control of Industrial Manipulators,” *IEEE Trans. Ind. Electron.*, vol. 53, no. 2, pp. 604–613, 2006.

[37] T. Thomessen and T. K. Lien, “Robot Control System for Safe and Rapid Programming of Grinding Applications,” *Ind. Robot An Int. J.*, vol. 27, no. 6, pp. 437–444, 2000.

[38] W. A. Kline, R. E. Devor, and I. A. Shareef, “The Prediction of Surface Accuracy in End Milling,” in *Trans, of ASME*, 1982, vol. 104, no. 3, pp. 272–278.

[39] S. H. Ryu, H. S. Lee, and C. N. Chu, “The form error prediction in side wall machining considering tool deflection,” *Int. J. Mach. Tools Manuf.*, vol. 43, pp. 1405–1411, 2003.

[40] V. S. Rao and P. V. M. Rao, “Effect of workpiece curvature on cutting forces and surface error in peripheral milling,” in *Proceedings of the Institution of Mechanical Engineers*, 2006, vol. 220, no. 9, pp. 1399–1407.

[41] K. M. Y. Law, A. Geddam, and V. A. Ostafiev, “A process-design approach to error compensation in the end milling of pockets,” *J. Mater. Process. Technol.*, vol. 89–90, pp. 238–244, 1999.

[42] S.-H. Suh, J.-H. Cho, and J.-Y. Hascoet, “Incorporation of Tool Deflection in Tool Path Computation: Simulation and Analysis,” *J. Manuf. Syst.*, vol. 15, no. 3, pp. 190–199, 1996.

[43] B. Denkena, H. C. Möhring, and J. C. Will, “Tool deflection compensation with an adaptronic milling spindle,” in *Conference on Smart Machining Systems*, 2007.

[44] M. Soori, B. Arezoo, and M. Habibi, “Virtual machining considering dimensional, geometrical and tool deflection errors in three-axis CNC milling machines,” *J. Manuf. Syst.*, vol. 33, no. 4, pp. 498–507, 2014.

[45] M. Habibi, B. Arezoo, and M. Vahebi Nojedeh, “Tool deflection and geometrical error compensation by tool path modification,” *Int. J. Mach. Tools Manuf.*, vol. 51, no. 6, pp. 439–449, 2011.

[46] V. S. Rao and P. V. M. Rao, “Tool deflection compensation in peripheral milling of curved geometries,” *Int. J. Mach. Tools Manuf.*, vol. 46, no. 15, pp. 2036–2043, 2006.

[47] M. Y. Yang and J. G. Choi, “A tool deflection compensation system for end milling accuracy improvement,” *J. Manuf. Sci. Eng.*, vol. 120, no. 2, pp. 222–229, 1998.

[48] “PI Hexapod Robot H-824.” [Online]. Available: https://www.physikinstrumente.com/en/products/parallel-kinematic-hexapods/hexapods-with-motor-screw-drives/h-824-6-axis-hexapod-700815/.

[Accessed: 23-Feb-2017].

[49] "BMR Spindle." [Online]. Available: https://www.bmr-gmbh.de/Englisch/. [Accessed: 23-Feb-2017].

[50] "Keyence Laser." [Online]. Available: http://www.keyence.com/products/measure/laser-1d/lk-g5000/models/lk-h027/index.jsp. [Accessed: 23-Feb-2017].

[51] L. Reznik, O. Ghanayem, and A. Bourmistrov, "PID plus fuzzy controller structures as a design base for industrial applications," *Eng. Appl. Artif. Intell.*, vol. 13, no. 4, pp. 419–430, 2000.

[52] N. Minorsky, "Directional Stability and Automatically Steered Bodies," *J. Am. Soc. Nav. Eng.*, vol. 34, p. 280, 1922.

[53] "PID Control." [Online]. Available: http://tikalon.com/blog/blog.php?article=control_freaks. [Accessed: 01-Jun-2017].

[54] J. Han, "From PID to active disturbance rejection control," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 900–906, 2009.

[55] J. Han, "Active disturbance rejection control and its application," *Control Decis.*, vol. 13, pp. 19–23, 1998.

[56] J. Han, "Nonlinear design methods for control systems," in *14th IFAC World Congr.*, 1999.

[57] W. Wang and Z. Gao, "A Comparison Study of Advanced State Observer Design Techniques," in *American Control Conference 2003, Denver, Colorado*, 2003, vol. 14, pp. 4754–4759.

[58] "MATLAB System Identification Toolbox." [Online]. Available: https://www.mathworks.com/products/sysid.html. [Accessed: 15-Jun-2017].

[59] M. Verhaegen and V. Verdult, *Filtering and System Identification*. Cambridge University Press, 2007.

[60] Y. Chetouani, "Using ARX approach for modelling and prediction of the dynamics of a reactor-exchanger," in *IChemE*, 2008, no. 154.

[61] Z. Jinhua, Z. Jian, D. Haifeng, and W. Sun'an, "Self-organizing genetic algorithm based tuning of PID controllers," *Inf. Sci. (Ny).*, vol. 179, no. 7, pp. 1007–1018, 2009.

[62] Y. Mitsukura, T. Yamamoto, and M. Kaneda, "A Design of Self-Tuning PID Controllers Using a Genetic Algorithm," in *Proc. of the American Cont.*

Conf., 1999, no. June, pp. 1361–1365.

[63] T. L. Seng, M. Bin Khalid, and R. Yusof, "Tuning of a Neuro-Fuzzy Controller by Genetic Algorithm," in *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS*, 1999, vol. 29, no. 2, pp. 226–236.

[64] "Response Optimization SIMULINK." [Online]. Available: https://www.mathworks.com/help/sldo/response-optimization.html. [Accessed: 06-Jun-2017].

[65] K. Açıkgöz, "Predictıon Of The Cutting Forces for Robotic Grinding Processes with Abrasive Mounted Bits," Middle East Technical University, 2015.

APPENDIX A

SPECIFICATIONS

Table 4 - PI Company piezo actuator specifications .[65]

P-602.8S0			
Active axes	*X*	*Drive Properties*	
Motion and positioning		*Piezoceramics*	*PICMA® P-888*
Integrated sensor	*SGS*	*Electrical capacitance*	*39 µF*
Open-loop travel - 20 to 120 V	*1000 µm*	*Dynamic operating current coefficient*	*4 µA/(Hz x µm)*
Closed-loop travel	*1000 µm*	*Miscellaneous*	
Open-loop resolution	*0.5 nm*	*Operating temperature range*	*-10 to 50 °C*
Closed-loop resolution	*7 nm*	*Material*	*Stainless steel*
Closed-loop non-linearity	*1.5 %*	*Dimensions*	*126 mm x 34 mm x 14 mm*
Repeatability	*50 nm*	*Cable length*	*0.5 / 0.5 / 2 m*
Mechanical properties		*Miscellaneous*	
Stiffness in motion direction	*0.4 N/µm*		
Unloaded resonant frequency	*150 Hz*		

Table 5 - PI Company Hexapod parallel manipulator specifications.[65]

H-824.Gxx			
Active axes	$X, Y, Z, \theta_x, \theta_y, \theta_z$	*Max. Velocity* $\theta_x, \theta_y, \theta_z$	*11 mrad/s*
Motion and positioning		*Typ. Velocities for* X, Y, Z	*0.5 mm/s*
Travel Range X, Y	*±22.5 mm*	*Typ. Velocities for* $\theta_x, \theta_y, \theta_z$	*5.5 mrad/s*
Travel Range Z	*±12.5 mm*	*Mechanical Properties*	
Travel Range θ_x, θ_y	*±7.5 °*	*Stiffness* X, Y	*1.7 N/µm*
Travel Range θ_z	*±12.5 °*	*Stiffness Z*	*7 N/µm*
Single actuator design resolution	*0.007 µm*	*Load (base plate horizontal / any orientation)*	*10/5 kg (max)*
Min. incremental motion X, Y, Z	*0.3 µm*	*Holding force, de-energized (base plate horizontal / any orientation)*	*100 / 50 N (max)*
Min. incremental motion $\theta_x, \theta_y, \theta_z$	*3.5 µrad*	*Motor type*	*DC gear motor*
Backlash X, Y	*3 µm*	*Miscellaneous*	
Backlash Z	*1 µm*	*Operating temperature range*	*-10 to 50 °C*
Backlash θ_x, θ_y	*20 µrad*	*Material*	*Aluminum*

Table 6 - Multi-axis force/torque sensor specifications (ATI Gamma IP60) .[65]

$F_{x/y}$	*±1200 N*
F_z	*±4100 N*
$T_{x/y}$	*±79 Nm*
T_z	*±82 Nm*
X-axis & Y-axis forces (K_x, K_y)	$9.1x10^6$ *N/m*
Z-axis force (K_z)	$1.8x10^7$ *N/m*
X-axis & Y-axis torque (K_{tx}, K_{ty})	$1.1x10^4$ *Nm/rad*
Z-axis torque (K_{tz})	$1.6x10^4$ *Nm/rad*
F_x, F_y, T_z	*1400 Hz*
F_z, T_x, T_y	*2000 Hz*
Weight	*0.255 kg*
Diameter	*75.4 mm*
Height	*33.3 mm*

Table 7 - Spindle Specifications

221-42-MHP

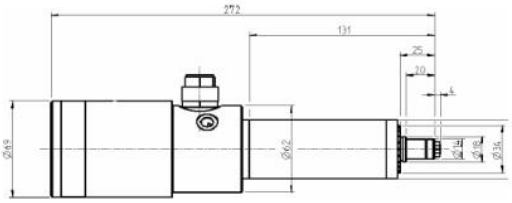
High-frequency spindle
Pneumatic direct tool change

Spindle for high-speed milling, -grinding, -drilling, -engraving

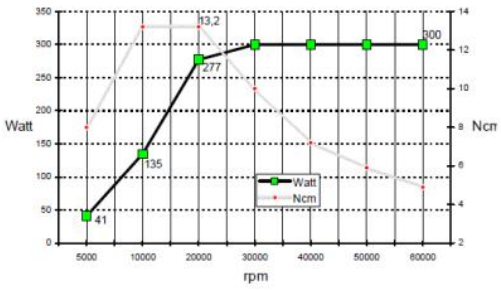
Technical specifications

- High precision hybrid ball bearings – 2 Pcs.
- Lifetime lubricated, maintenance free
- Motor: type 2
- Nominal output power: max. 0,3 kW
- Voltage: max.43 V
- Current: max. 7 A
- Frequency: max. 1000Hz
- Motor poles: 1 pair
- Rotation speed: max. 60.000 rpm
- Motor protection: thermistor
- Speed monitoring: transmitter
- Sealing-air
- Housing diameter: 42 mm
- Cooling system: non cooled
- Tool change: pneumatic direct tool change
- Clamping range: up to 6 mm (1/4")
- Coupler plug: 7-pole plastics
- Weight: 2,2 kg

Dimensions



Power-, torque- and speed diagram



BMR GmbH • elektrischer & elektronischer Gerätebau
Unterreichenbacher Straße 1• D-90455 Nürnberg-Katzwang
Tel. +49 (0)9122-631480 • Fax + 49 (0)9122-6314829
info@bmr-gmbh.de www.bmr-gmbh.de

APPENDIX B

MATLAB CODE OF SPINDLE CONTROLLER

```
%Author: Abdulhamit DONDER
function [Percent, Current, RpmSpindle]  = fcn(spindleSpeed,
biasData)
    coder.extrinsic('fwrite', 'strcat')
    coder.extrinsic('serial', 'fopen','fread','set')
    persistent s txdata_decStart txdata_decPercent
txdata_decCurrent
    persistent txdata_decDutyRpm txdata_decActRpm
txdata_decRpmSpindle
if isempty(s)

    s = serial('COM1');
    set(s,'Terminator','', 'InputBufferSize',
50000,'BaudRate',115200);
    fopen(s);
    %initialization
    rxdata_dec = zeros(3,1);
    DutyRpm = 0;
    Percent = 0;
    Current = 0;
    RpmSpindle = 0;
    ActRpm = 0;


    % Specify hex codes to be transmitted
    txdataStart = ['24'];
    %Convert to decimal format
    txdata_decStart = hex2dec(txdataStart);

    % Specify hex codes to be transmitted
    txdataDutyRpm = ['41']; %duty rpm
    %Convert to decimal format
    txdata_decDutyRpm = hex2dec(txdataDutyRpm);

    % Specify hex codes to be transmitted
    txdataActRpm = ['42']; %actual rpm of the converter
    %Convert to decimal format
    txdata_decActRpm = hex2dec(txdataActRpm);

    % Specify hex codes to be transmitted
    txdataRpmSpindle = ['43']; %rpm of the spindle
    %Convert to decimal format
```

```
    txdata_decRpmSpindle = hex2dec(txdataRpmSpindle);

    % Specify hex codes to be transmitted
    txdataPercent = ['0C';'a4';'08'];
    %Convert to decimal format
    txdata_decPercent = hex2dec(txdataPercent);

    % Specify hex codes to be transmitted
    txdataCurrent = ['0C';'b6';'0b'];
    %Convert to decimal format
    txdata_decCurrent = hex2dec(txdataCurrent);

end

if biasData == 1

    %velocity adjustment
    spindleSpeedHex = dec2hex(fix(spindleSpeed/10));
    spindleSpeed2Hex =
strcat(spindleSpeedHex(2),spindleSpeedHex(3));
    spindleSpeed3Hex = strcat('0',spindleSpeedHex(1));
    %Specify hex codes to be transmitted
    txdata = ['01';spindleSpeed2Hex;spindleSpeed3Hex];
    %Convert to decimal format
    txdata_dec = hex2dec(txdata);
    %Write using the UINT8 data format
    fwrite(s,txdata_dec,'uint8');
    %Read back data in decimal format
    rxdata_dec = fread(s, [3, 1], 'char');

    % %Write using the UINT8 data format
    fwrite(s,txdata_decStart,'uint8');
    %Read back data in decimal format
    rxdata_dec = fread(s,[3, 1],'uint8');

    % %Write using the UINT8 data format
    fwrite(s,txdata_decRpmSpindle,'uint8');
    rxdata_dec = fread(s,[3, 1],'uint8'); %read
    RpmSpindle2Hex = dec2hex(rxdata_dec(2));
    RpmSpindle3Hex = dec2hex(rxdata_dec(3));
    RpmSpindleHex = strcat(RpmSpindle3Hex,RpmSpindle2Hex);
    RpmSpindle = hex2dec(RpmSpindleHex)*10;

    % %Write using the UINT8 data format
    fwrite(s,txdata_decPercent,'uint8');
    rxdata_dec = fread(s,[3, 1],'uint8'); %read
else
    rxdata_dec = zeros(3,1);
    DutyRpm = 0;
    Percent = 0;
    Current = 0;
    RpmSpindle = 0;
```

```
    ActRpm = 0;
end
Percent2Hex = dec2hex(rxdata_dec(2));
Percent3Hex = dec2hex(rxdata_dec(3));
PercentHex = strcat(Percent3Hex,Percent2Hex);
Percent = hex2dec(PercentHex)*0.1;

% %Write using the UINT8 data format
fwrite(s,txdata_decCurrent,'uint8');
rxdata_dec = fread(s,[3, 1],'uint8'); %read
Current2Hex = dec2hex(rxdata_dec(2));
Current3Hex = dec2hex(rxdata_dec(3));
CurrentHex = strcat(Current3Hex,Current2Hex);
Current = hex2dec(CurrentHex)*0.01;
```

APPENDIX C

USER GUIDE FOR DEVELOPED HEXAPOD GRAPHICAL USER INTERFACE

This appendix gives a fully detailed explanation for using developed user interface.

1. Turn on the digital motion controller of the hexapod robot and the workstation.
2. Open the related Simulink file that is going to be used for control purposes.
3. Run the callback shown in Figure 67 in order to save the sampling time to workspace.

Figure 67 – Sampling time and calibration matrix callback

4. Run “HexVeloSabitXYZUVW.m” file.
5. The window in Figure 68 appears.
6. Click “Initialization” button.
7. Click “Connect and Reference Move” button. The hexapod will perform a reference move. Therefore, make sure that there is a safe space around the hexapod.
8. In Simulink, build the model and connect to target.
9. Click “Start Simulink” button in the GUI shown in Figure 68.
10. In order to translate the hexapod in three Cartesian axes, use the “Move to Target” buttons for related axes.
11. In order to see the current position of the hexapod robot, use “Current Position” button.

12. “Angle compensation start” button starts the angle compensation. The hexapod moves “Target Y” by compensating the tool angle errors.
13. In case of emergency, “Stop Movement” button is placed.

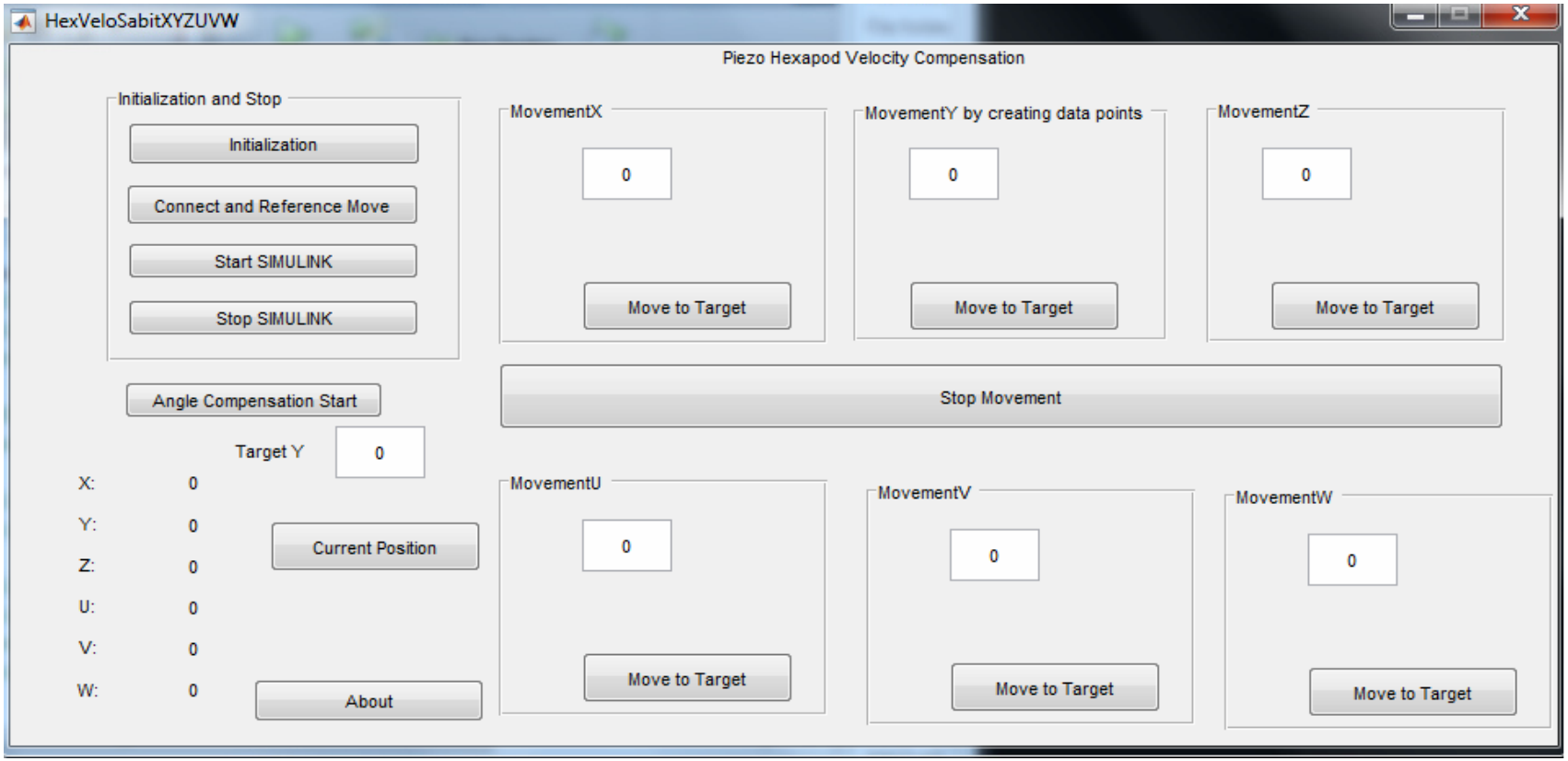


Figure 68 - Controlling hexapod robot – Developed Graphical User Interface

APPENDIX D

MATLAB CODE OF HEXAPOD CONTROLLER

```
%Author: Abdulhamit DONDER
function varargout = HexVeloSabitXYZUVW(varargin)
% HEXVELOSABITXYZUVW MATLAB code for HexVeloSabitXYZUVW.fig
%      HEXVELOSABITXYZUVW, by itself, creates a new
HEXVELOSABITXYZUVW or raises the existing
%      singleton*.
%
%      H = HEXVELOSABITXYZUVW returns the handle to a new
HEXVELOSABITXYZUVW or the handle to
%      the existing singleton*.
%
%
HEXVELOSABITXYZUVW('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in HEXVELOSABITXYZUVW.M with the
given input arguments.
%
%      HEXVELOSABITXYZUVW('Property','Value',...) creates a new
HEXVELOSABITXYZUVW or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before HexVeloSabitXYZUVW_OpeningFcn
gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to
HexVeloSabitXYZUVW_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
HexVeloSabitXYZUVW

% Last Modified by GUIDE v2.5 06-Aug-2017 09:39:56

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
```

```
                   'gui_OpeningFcn',
@HexVeloSabitXYZUVW_OpeningFcn, ...
                   'gui_OutputFcn',
@HexVeloSabitXYZUVW_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before HexVeloSabitXYZUVW is made visible.
function HexVeloSabitXYZUVW_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to HexVeloSabitXYZUVW (see
VARARGIN)

% Choose default command line output for HexVeloSabitXYZUVW
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes HexVeloSabitXYZUVW wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = HexVeloSabitXYZUVW_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double


% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in Initialization.
function Initialization_Callback(hObject, eventdata, handles)
% hObject    handle to Initialization (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

sampleTime=0.001; %Sampling time
assignin('base','sampleTime',sampleTime)
b0 = 0.5;
assignin('base','b0',b0)

CAL=[-0.02106, -0.00725, 0.22614, -13.82224, 0.08640, 13.98781;
  -0.22758, 16.33921, 0.10624, -8.03676, -0.12379, -8.05002;
  24.95057, -1.06034, 25.17707, -1.04570, 25.01689, -0.47779;
  -0.00955,   0.30094,  -0.72927,  -0.11766,   0.72374,  -0.16452;
  0.84126,  -0.03894,  -0.42088,   0.27778,  -0.42192,  -0.25142;
  0.00460,  -0.43627,   0.00710,  -0.42940,  -0.00189,  -0.43578];
assignin('base','CAL',CAL) %to matlab workspace

% --- Executes on button press in ReferenceMove.
function ReferenceMove_Callback(hObject, eventdata, handles)
% hObject    handle to ReferenceMove (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Reference stage if needed
if(~exist('Controller'))
    Controller = PI_GCS_Controller();
end;
if(~isa(Controller,'PI_GCS_Controller'))
    Controller = PI_GCS_Controller();
end;


%Set up connection between PC and Controller
ip      = '169.254.10.196';
assignin('base','ip',ip) %to matlab workspace
port    = 50000;
assignin('base','port',port) %to matlab workspace

Controller = Controller.ConnectTCPIP(ip, port);
Controller = Controller.InitializeController();
assignin('base','Controller',Controller) %to matlab workspace
%Get basic parameters

% query controller identification
identification = Controller.qIDN();
assignin('base','identification',identification) %to matlab
workspace
disp(identification);

% query axes
availableaxes = Controller.qSAI();
if(isempty(availableaxes))
    return;
end

availableaxes = regexp(availableaxes,'[\w-]+','match');
numberOfAxis = length(availableaxes);
axisname = 'x';
assignin('base','availableaxes',availableaxes)
assignin('base','numberOfAxis',numberOfAxis)
assignin('base','axisname',axisname)
axisname = 'x';
% Reference stage
if(~Controller.qFRF(axisname))
    Controller.FRF(axisname);
    bref = 0;
    while(bref)
        bref = ~Controller.IsControllerReady();
        pause(0.1);
    end
end
```

```
if(~Controller.qFRF(axisname))
    return;
end

% --- Executes on button press in startSimulink.
function startSimulink_Callback(hObject, eventdata, handles)
% hObject    handle to startSimulink (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
load_system('Main_Controllerseperated_hexapod_2015');
set_param('Main_Controllerseperated_hexapod_2015','SimulationComma
nd','Start');


block1 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Sel1'
;
block2 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Sel2'
;
block3 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Sel3'
;
block4 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Ad1';
block5 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Ad2';
block6 =
'Main_Controllerseperated_hexapod_2015/Kinematic_calibration/Ad3';
rto1 = get_param(block1, 'RuntimeObject');
rto2 = get_param(block2, 'RuntimeObject');
rto3 = get_param(block3, 'RuntimeObject');
rto4 = get_param(block4, 'RuntimeObject');
rto5 = get_param(block5, 'RuntimeObject');
rto6 = get_param(block6, 'RuntimeObject');
assignin('base','rto1',rto1) %to matlab workspace
assignin('base','rto2',rto2) %to matlab workspace
assignin('base','rto3',rto3) %to matlab workspace
assignin('base','rto4',rto4) %to matlab workspace
assignin('base','rto5',rto5) %to matlab workspace
assignin('base','rto6',rto6) %to matlab workspace




block7 = 'Main_Controllerseperated_hexapod_2015/hexVelo';
rto7 = get_param(block7, 'RuntimeObject');
assignin('base','rto7',rto7) %to matlab workspace

% --- Executes on button press in stopSimulink.
function stopSimulink_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to stopSimulink (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
set_param('Main_Controllerseperated_hexapod_2015','SimulationComma
nd','Stop')

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of
edit3 as a double



% --- Executes during object creation, after setting all
properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in moveToTargetX.
function moveToTargetX_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetX (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'x';
target = str2double(get(handles.edit2,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
% while(any(Controller.IsMoving('')))
```

```
%     positionX = Controller.qPOS('x');
%     set(handles.posX, 'string',num2str(positionX));
% end
while(Controller.IsMoving(axisname))
    % Display progress
    positionX = Controller.qPOS('x');
    set(handles.posX, 'string',num2str(positionX));
    pause(0.1);

end

positionX = Controller.qPOS('x');
set(handles.posX, 'string',num2str(positionX));


% --- Executes on button press in moveToTargetY.
function moveToTargetY_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetY (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'y';
target = str2double(get(handles.edit1,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
while(Controller.IsMoving(axisname))
    % Display progress
    positionY = Controller.qPOS('y');
    set(handles.posY, 'string',num2str(positionY));
    pause(0.1);

end
positionY = Controller.qPOS('y');
set(handles.posY, 'string',num2str(positionY));


% --- Executes on button press in moveToTargetZ.
function moveToTargetZ_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetZ (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
```

```
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'z';
target = str2double(get(handles.edit3,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
while(Controller.IsMoving(axisname))
    % Display progress
    positionZ = Controller.qPOS('z');
    set(handles.posZ, 'string',num2str(positionZ));
    pause(0.1);

end
positionZ = Controller.qPOS('z');
set(handles.posZ, 'string',num2str(positionZ));

% --- Executes on button press in stopMovement.
function stopMovement_Callback(hObject, eventdata, handles)
% hObject    handle to stopMovement (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace
result = 1;
assignin('base','result',result) %to matlab workspace
Controller.HLT('x');
Controller.HLT('y');
Controller.HLT('z');


function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of
edit2 as a double


% --- Executes during object creation, after setting all
properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in currentPos.
function currentPos_Callback(hObject, eventdata, handles)
% hObject    handle to currentPos (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace
positionX = Controller.qPOS('x');
positionY = Controller.qPOS('y');
positionZ = Controller.qPOS('z');
positionU = Controller.qPOS('u');
positionV = Controller.qPOS('v');
positionW = Controller.qPOS('w');
set(handles.posX, 'string',num2str(positionX));
set(handles.posY, 'string',num2str(positionY));
set(handles.posZ, 'string',num2str(positionZ));
set(handles.posU, 'string',num2str(positionU));
set(handles.posV, 'string',num2str(positionV));
set(handles.posW, 'string',num2str(positionW));


% --- Executes on button press in about.
function about_Callback(hObject, eventdata, handles)
% hObject    handle to about (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
a=menu('Author: Abdülhamit DÖNDER, adonder@metu.edu.tr','Close');


% --- Executes on button press in angleCompensation.
function angleCompensation_Callback(hObject, eventdata, handles)
% hObject    handle to angleCompensation (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace
rto1 = evalin('base','rto1'); %from matlab workspace
rto2 = evalin('base','rto2'); %from matlab workspace
rto3 = evalin('base','rto3'); %from matlab workspace
rto4 = evalin('base','rto4'); %from matlab workspace
rto5 = evalin('base','rto5'); %from matlab workspace
rto6 = evalin('base','rto6'); %from matlab workspace
rto7 = evalin('base','rto7'); %from matlab workspace
```

```
% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));
% working with points
trajectorySource = 1;
Controller.SPA('1',419436800,trajectorySource,null(1));

%for Y movement
targetYcompensation =
str2double(get(handles.targetYCompensation,'String'));
nextDesiredPointY=0;

initialLoopA = Controller.qPOS('');
initialLoop =
[initialLoopA(1);initialLoopA(2);initialLoopA(3);initialLoopA(4);i
nitialLoopA(5);initialLoopA(6)];
YinitialLoop= initialLoopA(2);
dataNumber = 6;
nextPoints = zeros(6,dataNumber);
result=0; %in case of emmergency it is going to be 1
assignin('base','result',result) %to matlab workspace

while result==0

    hexapodMovement1 = rto1.OutputPort(1).Data;
    hexapodMovement2 = rto2.OutputPort(1).Data;
    hexapodMovement3 = rto3.OutputPort(1).Data;
    hexapodMovement4 = rto4.OutputPort(1).Data;%radians
    hexapodMovement5 = rto5.OutputPort(1).Data;%radians
    hexapodMovement6 = rto6.OutputPort(1).Data;%radians
    hexapodMovements =
[hexapodMovement1,hexapodMovement2,hexapodMovement3,hexapodMovemen
t4*180/pi,hexapodMovement5*180/pi,hexapodMovement6*180/pi];
    HexVeloY = rto7.InputPort(1).Data;
    pause(0.1)

    % in order to write positions on simulink simultaneously
    positions = Controller.qPOS('');
    assignin('base','positions',positions); %to matlab workspace

set_param('Main_Controllerseperated_hexapod_2015/Pos','Value','pos
itions');



    %for Y movement (feedrate)
    difYY = targetYcompensation-YinitialLoop;
    bufferSizeY = abs(difYY) / (HexVeloY * 50/1000);
    steppp = difYY / bufferSizeY;
    prevv = YinitialLoop;
    for i=1:dataNumber
        nextPointsY(i) = prevv + steppp;
```

```
        prevv = nextPointsY(i);
    end

    for i=1:dataNumber
        if sign(steppp) == 1
            if nextPointsY(i)>targetYcompensation
                nextPointsY(i)=targetYcompensation;
            end

        elseif sign(steppp) == -1
            if nextPointsY(i)<targetYcompensation
                nextPointsY(i)=targetYcompensation;
            end
        end
    end

    for i=1:dataNumber
        nexPoint=nextPointsY(i);
        assignin('base','nexPoint',nexPoint) %to matlab workspace

set_param('Main_Controllerseperated_hexapod_2015/YtoolDesired','Va
lue','nexPoint');
    end

    YinitialLoop = YinitialLoop + dataNumber * steppp;


    difX = hexapodMovements(1)-initialLoop(1);
    difY = hexapodMovements(2)-initialLoop(2);
    difZ = hexapodMovements(3)-initialLoop(3);
    difU = hexapodMovements(4)-initialLoop(4);
    difV = hexapodMovements(5)-initialLoop(5);
    difW = hexapodMovements(6)-initialLoop(6);
    dif = [difX;difY;difZ;difU;difV;difW];

    HexVeloY
    Controller.VLS(HexVeloY); %Normally HexVeloY is the velocity
in Y direction. But due to impossibilities it is commended as the
velocity of hexapod.

    distance = (difX^2+difY^2+difZ^2)^(1/2);

    bufferSize = distance / (HexVeloY * 50/1000);

    stepp = dif / bufferSize;
    prev = initialLoop;


    for i=1:dataNumber
        nextPoints(:,i) = prev + stepp;
        prev = nextPoints(:,i);
```

```
    end

    for i=1:dataNumber
        %for X:
        if sign(stepp(1)) == 1
            if nextPoints(1,i)>hexapodMovements(1)
                nextPoints(1,i)=hexapodMovements(1);
            end

        elseif sign(stepp(1)) == -1
            if nextPoints(1,i)<hexapodMovements(1)
                nextPoints(1,i)=hexapodMovements(1);
            end
        end
        %for Y:
        if sign(stepp(2)) == 1
            if nextPoints(2,i)>hexapodMovements(2)
                nextPoints(2,i)=hexapodMovements(2);
            end

        elseif sign(stepp(2)) == -1
            if nextPoints(2,i)<hexapodMovements(2)
                nextPoints(2,i)=hexapodMovements(2);
            end
        end
        %for Z:
        if sign(stepp(3)) == 1
            if nextPoints(3,i)>hexapodMovements(3)
                nextPoints(3,i)=hexapodMovements(3);
            end

        elseif sign(stepp(3)) == -1
            if nextPoints(3,i)<hexapodMovements(3)
                nextPoints(3,i)=hexapodMovements(3);
            end
        end
        %for U:
        if sign(stepp(4)) == 1
            if nextPoints(4,i)>hexapodMovements(4)
                nextPoints(4,i)=hexapodMovements(4);
            end

        elseif sign(stepp(3)) == -1
            if nextPoints(4,i)<hexapodMovements(4)
                nextPoints(4,i)=hexapodMovements(4);
            end
        end
        %for V:
        if sign(stepp(5)) == 1
            if nextPoints(5,i)>hexapodMovements(5)
                nextPoints(5,i)=hexapodMovements(5);
            end
```

```
        elseif sign(stepp(5)) == -1
            if nextPoints(5,i)<hexapodMovements(5)
                nextPoints(5,i)=hexapodMovements(5);
            end
        end
        %for W:
        if sign(stepp(6)) == 1
            if nextPoints(6,i)>hexapodMovements(6)
                nextPoints(6,i)=hexapodMovements(6);
            end

        elseif sign(stepp(6)) == -1
            if nextPoints(6,i)<hexapodMovements(6)
                nextPoints(6,i)=hexapodMovements(6);
            end
        end
        Controller.MOV('X Y Z U V W', nextPoints(:,i));

        %for GUI
        position = Controller.qPOS('');
        set(handles.posX, 'string',num2str(position(1)));
        set(handles.posY, 'string',num2str(position(2)));
        set(handles.posZ, 'string',num2str(position(3)));
        set(handles.posU, 'string',num2str(position(4)));
        set(handles.posV, 'string',num2str(position(5)));
        set(handles.posW, 'string',num2str(position(6)));

    end
    result = evalin('base','result'); %from matlab workspace
    initialLoop = initialLoop + dataNumber * stepp;

end


function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of
edit6 as a double


% --- Executes during object creation, after setting all
properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in moveToTargetW.
function moveToTargetW_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetW (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'w';
target = str2double(get(handles.edit6,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
while(Controller.IsMoving(axisname))
    % Display progress
    positionW = Controller.qPOS('w');
    set(handles.posW, 'string',num2str(positionW));
    pause(0.1);

end
positionW = Controller.qPOS('w');
set(handles.posW, 'string',num2str(positionW));



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of
edit5 as a double
```

```
% --- Executes during object creation, after setting all
properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in moveToTargetV.
function moveToTargetV_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetV (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'v';
target = str2double(get(handles.edit5,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
while(Controller.IsMoving(axisname))
    % Display progress
    positionV = Controller.qPOS('v');
    set(handles.posV, 'string',num2str(positionV));
    pause(0.1);

end
positionV = Controller.qPOS('v');
set(handles.posV, 'string',num2str(positionV));



function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
```

```
%        str2double(get(hObject,'String')) returns contents of
edit4 as a double


% --- Executes during object creation, after setting all
properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in moveToTargetU.
function moveToTargetU_Callback(hObject, eventdata, handles)
% hObject    handle to moveToTargetU (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
Controller = evalin('base','Controller'); %from matlab workspace

% working with mov command
trajectorySource = 0;
Controller.SPA('1',419436800,trajectorySource,null(1));

axisname = 'u';
target = str2double(get(handles.edit4,'String'));

Controller.VLS(1);
Controller.MOV(axisname,target);
while(Controller.IsMoving(axisname))
    % Display progress
    positionU = Controller.qPOS('u');
    set(handles.posU, 'string',num2str(positionU));
    pause(0.1);

end
positionU = Controller.qPOS('u');
set(handles.posU, 'string',num2str(positionU));



function targetYCompensation_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to targetYCompensation (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
targetYCompensation as text
%        str2double(get(hObject,'String')) returns contents of
targetYCompensation as a double


% --- Executes during object creation, after setting all
properties.
function targetYCompensation_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to targetYCompensation (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```