ROBOTIC SURFACE FINISHING OF CURVED SURFACES: REAL-TIME
IDENTIFICATION OF SURFACE PROFILE AND CONTROL

A Thesis
by
YALUN WEN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Chair of Committee, Prabhakar Pagilla
Committee Members, Pilwon Hur
John Valasek

Head of Department, Andreas Polycarpou

May 2018

Major Subject: Mechanical Engineering

Copyright 2018 Yalun Wen

ABSTRACT

This thesis introduces a complete design framework for robotic surface finishing of curved surfaces. The system setup is subdivided into three key components: *Real-time surface registration* is accomplished by employing a proximity laser sensor mounted on the robot end-effector. The proximity sensor measurements coupled with the robot kinematics is employed to derive the coordinates of the projection points. The entire set of projection points is combined to form a grid upon which the surface normal and its normal profile are reconstructed. This surface normal profile description allows us to generate trajectories for both motion and force control. The *trajectory generation* is a variation of the time-scaling method to incorporate local surface normal information. Instead of using every local normal to form a trajectory, the trajectory is generated based on the local normal of a few sampled projection points. The local normal is estimated from the two consecutive sampled projection points to form the trajectory at the current time step while the end-effector is traveling between them. Therefore, a continuous stream of position goals is generated that would orient the robot end-effector to the local normal of the surface. Finally, this trajectory and a force set point are inputs to the force and position control loop that was synthesized using the *simultaneous force and position control* strategy; this strategy is based on the robot kinematics model using the product of exponential formula. Therefore the control strategy is robust to system uncertainties.

The integrated robotic surface finishing system consists of a UR5 robot and a custom end-effector that includes a force/torque sensor, electromechanical sander, and the laser sensor. Robot Operating System (ROS) is utilized for real-time implementation, which will enable easy migration of the developed tools to other industrial robots. The effectiveness of the strategy is evaluated by conducting a number of experiments for flat and curved surfaces that includes force regulation and surface finishing on wooden surfaces; a representative sample of results is presented and discussed to validate the proposed approach.

DEDICATION

To my mother, my father and my grandparents.

ACKNOWLEDGMENTS

Before beginning this thesis, I want to first acknowledge and thank the many people that have made it possible by contributing their support over the years.

To my adviser Dr.Pagilla, thank you for your constant guidance and support of my graduate study. Thank you for creating a free learning environment for me to absorb and grow. I came to our lab with no idea what I am capable of, and what my plans are for the MS degree. You, Shyam and Orlando guided and helped me to achieve what I have achieved, if any. Your wisdom and experience will always be with me.

To all the members of my research group: Shyam, Orlando, Mitch, Angel, Jita, Zongyao, Guoqi, Jie. You made the long hours stay in the lab a true pleasure. I owe a great many thanks to Orlando who was my mentor on the roll-to-roll machine and patiently guided me on all my research endeavors.

Finally I want to express my sincere gratitude to my parents and grandparents. Thank you for being so supportive for every decision I made over so many years even though you may never know how grateful I am and how much I love you guys.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This thesis was supervised by a committee consisting of Professor Prabhakar Pagilla (advisor) and Professor Pilwon Hur of the Department of Mechanical Engineering and Professor John Valasek of the Department of Aerospace Engineering.

This research was conducted towards the author's MS thesis. The idea of using proximity sensor for surface scanning belongs to Dr. Pagilla, experiments performed on curved surface finishing were guided by Dr. Pagilla. The author benefits a lot more from the discussions with Dr. Pagilla than a few sentences could convey. All other work conducted for this thesis was completed by the student independently.

Funding Sources

This work was made possible in part by my parents and Dr. Pagilla's generous help.

TABLE OF CONTENTS

	Page
ABSTRACT	ii
DEDICATION	iii
ACKNOWLEDGMENTS	iv
CONTRIBUTORS AND FUNDING SOURCES	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	ix
LIST OF TABLES	xi
1. INTRODUCTION AND LITERATURE REVIEW	1
1.1 Motivation	1
1.2 Thesis Overview	3
1.3 Background Information	4
1.3.1 Mechanical Surface Finishing	4
1.3.1.1 General Definition	4
1.3.1.2 General Parameters	5
1.3.1.3 Contact Stress Model	6
1.3.1.4 Material Removal Rate (MRR)	7
1.3.1.5 Tool Wear	8
1.3.2 Major Techniques for Mechanical Surface Finishing	8
1.3.2.1 Manual Polishing/Sanding	8
1.3.2.2 Computer Numerical Control (CNC)	8
1.3.2.3 Robotic Surface Finishing	9
1.3.3 Robotic Systems	10
1.3.3.1 Parallel Robots	10
1.3.3.2 Serial Robots	10
1.3.4 Polishing Paths on Free-Form Surfaces	12
1.3.5 Surface Registration	13
1.3.6 Commercially Available Automated Surface Finishing Solutions	15
1.4 Force Control	15
1.4.1 Indirect Force Control	16
1.4.1.1 Compliance Control	17
1.4.1.2 Impedance Control	18

1.4.2	Direct Force Control	19
1.4.2.1	Hybrid Position/Force Control.....	19
1.4.2.2	Natural and Artificial Constraints	20
1.4.2.3	Control Structure	20
1.4.3	Force Control Application	21
1.5	Robot Operating System (ROS)	22
1.5.1	Communications Infrastructure	22
1.5.2	Robot Specific Features	23
1.5.3	Tools	24
1.5.3.1	Rviz	24
1.5.3.2	RQT.....	25
1.5.3.3	Moveit	26
1.5.3.4	ROS-Industrial.....	27
1.6	Research Objective.....	27
1.7	Contributions	28
2.	ROBOT KINEMATICS AND CONTROL	30
2.1	Robot Kinematics	30
2.1.1	Forward Kinematics	31
2.1.1.1	UR5 Forward Kinematics Verification	33
2.1.2	Inverse Kinematics	33
2.1.2.1	UR5 Inverse Kinematics Verification.....	34
2.1.3	Jacobian Matrix.....	34
2.1.3.1	UR5 Jacobian Verification	36
2.2	Simultaneous Force/Position Control Strategy	37
2.2.1	Pose Tracking Control.....	42
2.2.2	Normal Force Control	43
2.2.3	Simultaneous Force/Position Control	44
2.2.4	Stability Analysis.....	45
2.2.4.1	Pose Tracking Stability	45
2.2.4.2	Stability of Force Control	47
3.	WORK PIECE SURFACE PROFILE IDENTIFICATION	48
3.1	Work Piece Surface Registration	50
3.1.1	Position Interpolation of Laser Sensor Projection Point.....	50
3.1.2	Surface Reconstruction.....	50
3.1.3	Surface Normal Estimation	51
3.1.4	Surface Profile Reconstruction Accuracy	51
3.2	Trajectory Generation.....	55
3.2.1	Time Scaling.....	56
3.2.1.1	Joint Space Straight Line	56
3.2.1.2	Task Space Straight Line.....	57
3.2.2	Sample Points Based Tool Path Generation	59

4. EXPERIMENTAL DESIGN AND RESULTS	62
4.1 System Description	62
4.1.1 Hardware Setup.....	62
4.1.2 Software Setup.....	68
4.2 Experimental Results	68
4.2.1 Force Regulation	69
4.2.2 Surface Finishing	73
5. SUMMARY AND FUTURE WORK	75
REFERENCES	77

LIST OF FIGURES

FIGURE	Page
1.1 Two curved bodies pressed together [1].....	7
1.2 Parallel robot	10
1.3 Serial robot	11
1.4 Tool path patterns: (a) scanning path (b) Lissajous path; (c) Peano path [2].	13
1.5 One DOF mass-spring-damper model	18
1.6 UR5 dimension from URDF [3]	24
1.7 RVIZ interface.....	25
2.1 UR5 dimension defined by URDF [3].....	31
2.2 Circular motion from Jacobian kinematics	37
2.3 Block diagram for pose tracking control	42
2.4 Block diagram for normal force control	43
2.5 Block diagram for simultaneous force/position control	45
3.1 Grid data with local surface normal	49
3.2 Sampled projection points on part.....	51
3.3 Work piece VS surface model	52
3.4 Surface normal calculation	53
3.5 Surface normal	54
3.6 Error angle VS sample distance	55
3.7 Straight line in configuration space	58
3.8 Time scaling factor VS time	60
3.9 Tool path trajectory	61

4.1	UR5 polishing system.....	63
4.2	Speedj performance on the last joint of UR5	65
4.3	FT300 noise frequency analysis via FFT	66
4.4	Low pass filter	66
4.5	Complementary filter	66
4.6	Polishing trace of random orbital sander.....	67
4.7	Flat and curved surfaces	69
4.8	Normal force regulation	70
4.9	Force plot of force/position control on flat surface	71
4.10	Force plot of force/position control on curved surface	72
4.11	Surface finishing on flat wood surface	73
4.12	Surface finishing on curved wood	74

LIST OF TABLES

TABLE	Page
2.1 UR5 DH parameters	32
2.2 UR5 safety limiting sets	33
4.1 UR5 safety limiting sets	64

1. INTRODUCTION AND LITERATURE REVIEW

1.1 Motivation

The benefits of automatic control for industrial manipulation are well documented in literature. By exploiting the repetitive nature of manufacturing and material handling process, automation has reduced labor costs and improved repeatability and efficiency. Specifically, the advancement of industry robots made significant contributions towards the automation of industry processes. Typical applications of robots include welding, painting, assembly, pick and place for printed circuit boards, packaging and labeling, palletizing, product inspection, and testing; all accomplished with high endurance, speed, and precision. In the year 2015, an estimated 1.63 million industrial robots were in operation worldwide according to International Federation of Robotics (IFR). However, the application of industry robots in surface finishing processes is rather limited due to several technical challenges.

Surface finishing could be one of the most important steps in order to obtain a finished industrial product. As opposed to machining and large material removal operations, surface finishing operations typically refer to the process of removing small amount of material and the process is usually repeated until the desired surface roughness is achieved. The objective of surface finishing then is to polish the surface until a good surface roughness or form accuracy is obtained; this is done sequentially by removing layers of materials by utilizing abrasive tools. Surface roughness and irregularities typically appear on industrial products due to tooling, forging, burr formation during casting, etc, or removing machining marks, pits, scratches and other defects [4]. Mechanical surface finishing operations, such as grinding, sanding, polishing, deburring, etc., are commonly employed to provide the work pieces with the desired surface finish in many industrial sectors, including aerospace, automobile, energy, and other heavy industries. These operations are quite labor intensive and time consuming: surface finishing could represent up to a third of production time in some industries [1]. Currently, surface finishing is predominantly performed by human

operators with manual sander or surface finishing tools. For example, the damaged turbine blade is repaired rather than replaced since it was found to be cheaper to refurbish the turbine blade. Repair and grinding of turbine blades is usually carried out manually by skilled operators. The refurbishment process of turbine blades starts by an operator manually filling gaps and cracks on the surface, and then grinding and polishing the whole surface [5].

Surface finishing with human operator introduces several drawbacks. First, surface finishing requires skilled labor and industries often have difficulties recruiting or training highly skilled manual workers, especially during times of increased productivity. Also, manual finishing is a laborious task that requires long periods of training and practice usually within a hazardous environment, such as noise and ergonomic health issues, dust, etc. The process specification and skill requirements are seldom documented for manual finishing, and the skill set may be lost when the skilled workers move away from the business. Further, the quality and cycle time for one cycle of operation strongly depends on the skill level of the operator, and the quality of the finished workpiece usually varies from job to job. In many industrial sectors and applications, finishing of surfaces takes place typically at the end of the product cycle, and thus any errors associated with surface finishing could be costly. Therefore, automation of manual finishing operations can be quite beneficial for a number of reasons as specified above. Automation of surface finishing with human supervision could provide better working environment, higher system flexibility, improved and consistent finished work piece quality. The design of automatic surface finishing solutions requires a complete understanding of the manual process. Surface finishing operations usually do not require processing forces that are as large as those of machining operations such as drilling, but one requires maintenance of steady tool-surface contact. Typical surface finishing operations, such as sanding or polishing, are repeated on the same surface progressively to improve surface roughness. Sanding or polishing is performed by rubbing of abrasive particles against a surface; a sanding pad with low grit is employed first followed by higher abrasive grit pads to smoothen the surface. During the process, the operator applies a constant contact force in the normal direction of the surface while moving the surface finishing tool tangentially along the surface; visual inspection

is usually employed to tackle the complex surface geometry to ensure the surface finishing tool is orientated properly. Some automated solutions have already been proposed to assist or replace human operators in particularly unhealthy environments, such as barrel deburring by Harper Surface Finishing Systems Inc., or robotic grinding by KUKA Gmb. However most of the existing solutions only provide limited performance for specific processes or components. These solutions typically lack the flexibility and dexterity that human operators offer: they are usually designed specifically for one type of product or workpiece. These custom automated solutions are typically expensive and are not affordable for many small and medium manufacturers who do a considerable amount of surface finishing jobs for larger industrial production. Therefore, automation of surface finishing tasks using robots is in its early stages and there are many technical challenges that need to be addressed. Humans have the ability to quickly grasp the topography of the surface, identify irregularities through visual inspection, and can handle free-form and curved surfaces while optimizing the force that needs to be applied to obtain the required surface finish. To develop a robust surface finishing robot system, one needs to understand and improve upon these aspects in a technically feasible way, which involves not only addressing robot motion and force control aspects, but also the integration of sensing data to obtain surface profile, its registration within the work cell and sequencing of motion trajectories.

1.2 Thesis Overview

In an effort to address the technical challenges and develop an integrated robotic sanding system, this thesis is organized as follows: review of the relevant literature and an introduction of various components of the robotic sanding system are presented in the remainder of Chapter 1. The kinematics and the control strategy of the robot employed for sanding application are given in Chapter 2. Chapter 2 also addresses the stability issues associated with simultaneous motion and force control involved in contact operations. Chapter 3 discusses the development of a non-contact method to obtain the surface normal profile and the generation of corresponding tool-path trajectories in real-time. Chapter 4 describes the development of the sanding system, some practical considerations, and a series of experiments conducted to validate the proposed control and surface

reconstruction strategies, and surface finishing effects. Finally, Chapter 5 gives a brief summary of the key contributions of this work and identifies areas for further investigation.

1.3 Background Information

Currently much of the research and technology development has focused on three main automated technologies: sanders mounted on industrial robots, computer numerical control (CNC) techniques, and custom devices to polish large number of workpieces. Some of the key technical challenges associated with identification of the surface for control of the robot to perform sanding operation on curved surfaces will be addressed in this thesis. The remainder of this section focuses on some of the most important technical aspects of surface finishing to develop an automated solution, and is organized to introduce relevant background information and key parameters to characterize surface finishing processes.

1.3.1 Mechanical Surface Finishing

This subsection gives a brief introduction to the mechanical surface finish processes. The general definition and some key performance parameters that can be used to evaluate the developed system are introduced in the reminder of this subsection.

1.3.1.1 General Definition

Surface finishing processes are widely used industry processes that could be employed to meet mechanical properties, design specifications, or the desired aesthetic aspects [5]. Mechanical surface finishing refers to the process of using abrasives to reduce surface roughness. For example, polishing is an abrasive operation that typically follows grinding and precedes buffing. It is the process of creating a smooth and shiny surface by rubbing it with abrasive devices, leaving a surface with a significant specular reflection. Four surface polishing techniques can be distinguished among the industries [6]:

- **Manual fishing:** A process involving a skilled operator using hand-held abrasive tools. The surface finishing tool is usually used to remove excess material after machining or casting. Manual finishing is strongly affected by the experience and judgment of the operator, which

may result in unevenly distributed surface finishing quality.

- **Mass finishing:** An automated technology that was originally used for polishing and deburring operations of large batches of small parts. Mass finishing techniques include machines such as vibrator bowl or centrifugal bowl finishers. However, they still require an experienced operator to decide the right media, compound, and finishing time.
- **Computer Numerical Control (CNC):** CNC technology is already available for grinding or polishing of surface with simple geometry. However, this type of technology is expensive and lacks the flexibility for processing of parts with complex geometries since the majority of CNC machines only have 3 degree of freedom.
- **Industrial robots equipped with surface finishing tools:** The setup imitates the way human operators performing surface finishing tasks. This technology could assist or replace the human operator in the hazardous environment with the aim for better consistency in surface finishing quality. However, the development of this technology is still in its early stage and many fundamental challenges are unsolved.

1.3.1.2 General Parameters

The commonly used parameters to characterize the efficiency and quality of surface finishing tasks are energy (power, speed, velocity), resistance (pressure, friction, material composition), and physical setup (tool type, sander size, etc) [7]. Among those parameters, perhaps the most important one is the resistance. Specifically, the maintenance of a constant contact stress (or force) between the abrasive tool and the workpiece surface. During surface finishing operations, adequate polishing force will facilitate the removal of marks, improve surface finishing quality, avoid under or over polishing and reduce the tool wear and finishing time [6, 8]. In this thesis, factors that are used to evaluate the performance of the automated surface finishing machine could be categorized as:

- abrasive material and size;

- abrasive rotation velocity;
- abrasive feed rate; and
- angle between the polishing tool and the tangential plane at the polishing spot;
- contact force between the abrasive tool and environment.

1.3.1.3 Contact Stress Model

The contact stress or force represents the level of mechanical interaction between the surface finishing tool and the environment (work piece), and the mathematical way to measure the level of mechanical interaction is expressed via the contact stress model. The contact stress is influenced by the size of the contact area, the normal contact force, and the tangential friction force [8]. Contact models can be grouped as: micro contact model and macro contact model [1, 9]. Micro contact model addresses the cut depth of the grains on the abrasive tool at micro-level, and the derivation of macro modeling is typically based on Hertzian contact theory. In Hertzian contact model, the contact between the tool and workpieces is represented as two discs pressed together, and the contact is assumed to be elastic. An illustration of the two surfaces in the Hertzian contact model is shown in Figure 1.1: The contact area is assumed to be point contact when two discs are touched, and gradually becomes an elliptical contact surface when the contact force reaches certain point. Liao presents a linearized contact stress/force model that can be used for robotic polishing [1]. Specifically, equations (1.1) to (1.3) summarize the calculation of contact stress & force.

$$P_m = \frac{2b}{3E(k')\Delta} \quad (1.1)$$

$$F = \frac{9\pi E^2(k') \Delta^2 P_m^3}{4k} \quad (1.2)$$

$$T_f = \frac{1}{3}u_k P_m \pi(a^3 + b^3) \quad (1.3)$$

where P_m denotes the contact stress, F is the variation in applied force necessary for maintaining constant contact, T_f is the friction torque resulting from the frictional force between the polishing

Figure 1.1: Two curved bodies pressed together [1]

tool and workpiece. The rest of the parameters are defined according to [9].

1.3.1.4 Material Removal Rate (MRR)

MRR is commonly defined as the volume of material removed divided by the machining time. MRR is perhaps the most important factor to consider in any material removal process, and it is strongly affected not by the contact force but by the contact stress[7]. Typically MMR increases with the the contact force, and decreases with feedrate [10]. Higher pressure will cause deeper cutting depth and hence a higher MRR. If not handled properly, high MRR may result in over-cut, or burring marks on the workpiece. Preston's model could be used to determine MRR in abrasive processes:

$$MRR = \frac{dz}{dt} = K \times P \times V \quad (1.4)$$

where P is the average pressure of contact, V is the relative velocity of the tool/part relative, and K is the Preston coefficient which is determined experimentally and depends on parameters, such as part material, abrasive, and lubrication [11]. In general, rough polishing operation will demand a large MRR with a slower feed rate; fine polishing usually requires a fast feed rate since slow feed rate could results in excessive material removal.

1.3.1.5 Tool Wear

Axinte [12] investigated the effect of the polishing method on the finishing quality, the relationship between polishing temperature and tool wear of abrasive tools during polishing. Huang [10] concluded that a high tool wear after the first operation and a decreased MRR after three cycle operation. Tool wear is an important performance factor used to evaluate surface finishing tasks, but it is generally ignored in the analysis of manual sanding processes as the humans have the ability to detect and identify tool wear visually in real-time and replace the tools as necessary.

1.3.2 Major Techniques for Mechanical Surface Finishing

This section presents the four major techniques used today for mechanical surface finishing in industry and research communities, and compares their advantages/disadvantages.

1.3.2.1 Manual Polishing/Sanding

Manual finishing is a highly skilled manufacturing process with many constraints and interactions with the environment [4]. The process is time consuming, and the quality of the finishing will strongly depend on the level of training, experience, fatigue, and expertise of the operator. Most importantly, polishing has to be performed in a hazardous environment that would threaten the health of the workers. The dust, noise, and vibration produced by the polishing process are detrimental to human health. The Health and Safety Executive (HSE) suggests that operators should take regular breaks to avoid long-term exposure to vibration. However, the manual operation is still the dominant approach adopted in the industry since it is relatively low cost, and there is no better options for small volume production or surfaces with complex geometries.

1.3.2.2 Computer Numerical Control (CNC)

Traditional CNC machines have dramatically improved the manufacturing enterprise by reducing manufacturing cycle time and increasing quality of products. However, CNC machines may not be a suitable solution for the automation of surface finishing [13]. The main advantages of CNC are the repeatability and high motion stiffness: CNC machines provide high motion ac-

curacy and is capable of enduring high contact force. Commercial solutions, such as centerless machines, are already available for grinding and heavy polishing. This approach consists of using either tools similar to those used in manual polishing or custom polishing tools (abrasive cylinder or cone attached to a metallic shaft [12]. The polishing tool is pressed against the surface of the part, with no attempt to actively control the z position of the tool. CNC machines generally don't provide force feedback control. It can be concluded that the use of CNCs for polishing may be beneficial in terms of precision and repeatability. However, this technology does not possess the dexterity as that of the industry robots for surface finishing tasks: typical CNC machines only have three DOF and perform constrained motions in the task space. Therefore, CNC machines couldn't handle free-form surfaces which typically requires six DOF motion [14, 15]. Compared to CNC machines, robots do not need to contain the workpiece within their structure, so they are capable of processing large parts. Since they can be mounted beside the object to be polished, they can be easily deployed on an assembly line. Therefore, industrial robots, typically with six DOF, are more suitable to handle with surfaces with complex geometries [16].

1.3.2.3 Robotic Surface Finishing

Industrial robots are commonly used for simple and repetitive tasks. Their applications for surface finishing tasks have been mostly restricted to deburring, chamfering and finishing of workpieces that possess a constant geometry [17, 10]. Robotic machining has certain advantages over conventional CNC machines, such as high flexibility, capability of integration with peripherals such as sensors and external actuators at low cost. To help the automated system to meet the dexterity and repeatability requirements, an integrated system must be developed around the performance parameters mentioned above to meet requirements and overcome the challenges. This implementation may require a set of techniques to reconstruct workpiece profiles, control the robot motion/force, and adjust tools orientation in real-time for optimal performance. Some of the proposed robotic polishing solutions used the the force data obtained during polishing to control the contact force and adjust the tool orientation in real-time [18, 19, 20].

1.3.3 Robotic Systems

Industrial robots may be classified as serial or parallel robots according to their structural mechanisms. This section compares the differences between two major industrial robots mechanisms, and emphasizes the reasons why serial robots are preferred over parallel robots for the automation of robotic surface finishing tasks.

1.3.3.1 Parallel Robots

Parallel robots refers to the mechanisms composed of closed kinematic chains with two rigid bodies, one is fixed to the ground that is referred to as the base platform and the other one is the moving platform which is connected to the fixed base through at least two kinematic chains [21, 19]. Most of the parallel robots are equipped with six actuators. Some may be redundantly actuated, i.e., their degree of actuation (DOA) is greater than their DOF. This setup results in high stiffness and uniformity in load distribution [22, 23]. Figure. 1.2 shows the parallel robot.

Figure 1.2: Parallel robot

1.3.3.2 Serial Robots

Serial robots are the perhaps the most commonly popular robots in many industries and could be found performing significant amount of automated manufacturing takes in aerospace, automo-

tive, and etc. A serial manipulator refers to the robots with an open kinematic chain mechanism that is made of a series of kinematic pairs connected to each other via mechanical constraints [21]. Figure 1.3 shows the typical serial robot configuration. If the robots have the same degrees of

Figure 1.3: Serial robot

freedom as the dimension of their task space (6 DOF or less), they are called fully actuated serial manipulators. Redundant serial robots refer to those robots with the number of DOF greater than the dimension of the task space. Huo specified three types of redundancy defined for robotic manipulators [24]:

- Intrinsic redundancy: the dimension of the joint space is greater than the dimension of the operational space.
- Functional redundancy: the dimension of the operational space is greater than the dimension of the task space.
- Kinematic redundancy: the dimension of the joint space is greater than the dimension of the task space.

However, there are several severe disadvantages of parallel robots namely:

- several singularity configurations within the workspace;
- small workspace;
- coupled motion and complex kinematic/dynamic relationship.

Therefore, serial industry robots are generally preferred over parallel robots.

1.3.4 Polishing Paths on Free-Form Surfaces

Perhaps one of the important sectors in automatic polishing process is the development of a proper trajectory for the surface finishing tool. The shape of the path has a significant impact on the smoothness and uniformity of the surface. So far, several methods for polishing path generation have been proposed. In [25], the CAD data of the workpiece was used to obtain the actual shape of its surfaces. Then, they produced the cutter location data from the CAD model, that presents the appropriate location and orientation of the polishing tool with respect to the workpiece surface [26, 14]. The most common path considered in many applications is the scanning path as shown in figure 1.4 (a). However, this method doesn't take into account the fixture uncertainty, and matching the part surface to that of the CAD model is generally difficult since the knowledge of the orientation of the part cannot be acquitted from above strategy.

In general, robot end-effector paths can be categorized as continuous, offset, and discontinuous [27]. In the first type, the tool follows the entire trajectory without any need to disrupt the polishing process, such as a scanning path. The offset paths made of several closed loops surround each other and the polishing tool can pass them from the inner part toward the outer one such as a spiral trajectory. In the last type, the path is neither continuous nor spiral and after finishing each path, the tool moves from its end-point to the starting-point of the next path. As mentioned by Rososhansky [27], employing the continuous path results in the fastest polishing process and encounters less computational complexity among these three categories. Tam et al. [15] discussed the use of scanning paths for continuous bi-directional scanning to facilitate the realization of

continuous polishing. Since there is no general method for direct generation of tool paths on a three dimensional surface, the indirect path generation approach is commonly adopted. The path generation usually starts with paths constructed on a two dimensional parametric plane, then this plane is mapped to a three dimensional surface using the surface model. Most of the work done by now assumes the availability of a CAD model. However, this is not true in general. Tam [15] concludes that the scanning patterns can be broadly divided into S-scanning and Z-scanning; The tool path can be divided into Lissajour patterns [2] or Peano curves [28]. Tam [15] illustrates the three different tool path patterns and they are shown in Figure 1.4:

Figure 1.4: Tool path patterns: (a) scanning path (b) Lissajous path; (c) Peano path [2].

1.3.5 Surface Registration

The contact force plays an important role in polishing processes: it should be kept constant to achieve a uniform material removal rate along the surface until the the desired profile and surface finishing are achieved. However, the maintenance of constant contact force is challenging for parts with free-form surfaces since the force has to be oriented towards the local surface normal. In cases where a part's surface geometry varies, the constant tool force without proper orientation will generate a high contact stress under a small contact area, leading to over polishing [7]. Further, a misalignment of part on the polishing tool, such as a tilt, can result in displacements and increases

tool wear. Therefore, the part surface should be described in a way that is processable in real time such that the robot end-effector is oriented properly.

Registration of the work piece with respect to the robot base frame is challenging. Often the work piece surface profile is obtained a priori by utilizing either cameras or sophisticated surface profile measuring devices. If the surface profile is identified in this manner a priori, then uncertainties associated with fixturing of the work piece in the workspace of the robot can pose difficulties in terms of setup planning, trajectory planning and subsequent real-time position/force control. In particular, this effect may be magnified for curved surfaces, as knowledge of the surface normal is needed to execute motion control tangential to the surface and force control normal to the surface. Surface registration is a difficult problem that can be divided into acquisition of surface data from sensors and 3D registration. The data acquisition sensors can be classified into contact based and non-contact based. Non-contact based sensors are considered in this thesis since contact based sensors may introduce damage if not properly handled. The non-contact based sensors can be grouped into: 1) Stereovision, 2) 3D scanners, 3) intensity imaging, and 4) proximity sensors. The stereovision method is perhaps the most studied. Shapes of the part can be acquired by modifying the intrinsic camera parameters, considering additional surface texture and geometric information, solving the correspondence problem between multiviews of a given part surface taken from different positions [20]. Despite multiview geometry being widely studied, it still presents several challenges for 3D reconstruction. The first problem is the well known correspondence problem [29]. Determining the correspondence relationship on a pixel level from multiviews is not a trivial task especially for the part lacking texture and features, such as a machined metal part. Coded structured light method could bring in extra correspondence points. However, the number of correspondence points per image is still not enough for a precise registration [30]. The second problem is the resolution of the data acquisition method. The advantage of 3D scanner is that it provides precise depth information of the surface. 3D scanners can be classified into active and passive scanners: active 3D scanners measure the distance "directly" while the passive scanners rely on the multiview geometry algorithms for depth measurement. But most active 3D

scanners are too expensive and its output rate may not be fast enough for real-time processing.

1.3.6 Commercially Available Automated Surface Finishing Solutions

Industrial companies have already introduced automated surface finishing solutions such as CNC grinding, robotic polishing, or mass finishing. However, the development of these technologies is still in its early stage and limited to only custom automated solutions are available. POLIMATIC is a European project carried out by Fraunhofer and various industrial partners for robotic polishing of free-form molds and dies. Their system includes a CAD/CAM package, force control and in-situ metrology. AESCULAP (KUKA Gmb, 2014) has introduced partial automation within their processes in collaboration with KUKA Gmb. They wanted to optimize their process by shortening the production time and reducing cost. The automated system is composed of two robots working on the finishing of a single component. ARCOS systems is a machine tool manufacture that provides industrial solutions and services for various companies. Their solutions include a complete automated cell with robotic arms, grinding-polishing-buffing-stations, and a vision system to assess the workpiece. VULCAN Europe (Vulcan Engineering, 2015) has also worked on a semi-automated grinding system called VTS, Vulcan tactile system. The robotic system is operated by a skilled worker to assist the grinding process of large foundry components with a haptic feedback device. Most of the commercially available solutions are expensive starting from \$200,000 and are not affordable for many small and medium manufacturers who do a considerable amount of surface finishing jobs for larger industrial production.

1.4 Force Control

Force control refers to the control strategy that commands the robot to apply force in a selected direction. There are a number of challenges associated with automating surface finishing operations. One of the fundamental challenges for the success of a manipulation task is the capacity to handle interaction between the manipulator and the environment (work piece). During the interaction, the environment sets the constraints that are to be followed by the robot end-effector. Thus, a purely motion control strategy would fail since the motion is constrained by the environment.

A deviation from the desired trajectory gives rise to a contact force. If the control system keeps reacting to reduce the trajectory error, the contact force would keep building up until the joint actuators are saturated or a safety protocol is broken. The contact force is the quantity that could be estimated based on environment stiffness and position control accuracy. A properly controlled contact force requires the maintenance of a compliant behaviour at the contact point, and external force measurements could provide enhanced control performance by monitoring and using the force for control system feedback. Early work on the computed torque based robot controller designs can be found in [31, 32, 33, 34]. A general review of the control methods and implementation modification can be found in [35]. The common force control problems encountered during real-time implementation are addressed in the following sectors: [36]. the constrained motion problem in Euler-Lagrange formulation [37]; the transition control between position mode and force controlled mode can be found in [36, 38]; contact control stability issue was addressed in [39]; a complete system implementation and experiments can be found in [37].

Two types of robot force control strategies are common: indirect force control and direct force control. The main difference between them is that the former achieves force control via motion control without explicit closure of a force feedback and the latter regulates the force to a set point with closure of a force feedback loop.

1.4.1 Indirect Force Control

Indirect force control achieves force control through suitable actions on the end-effector position and orientation errors. Compliant force control was first described in a series of papers by Hogan [40]. The concept of stiffness matrix for the interaction between the environment and robot end-effector is discussed in [41, 42]. Impedance and admittance control systems are inverse systems. Impedance control is based on resistance and admittance control is based on conductivity. In a dynamic relationship, if one part of the relationship is seen as an impedance, the other is admittance.

1.4.1.1 Compliance Control

Let the end-effector position and orientation is described by the vector $X_e = [p_e \ v_e]^T$, where the Euler angles are selected for the representation of orientation in the robot base frame. Let the desired position and orientation be denoted by X_d . Thus, the end-effector position and orientation error is denoted as

$$\delta x_{de} = \begin{bmatrix} \delta p_{de} \\ \delta v_{de} \end{bmatrix} \quad (1.5)$$

Assuming the environment to be elastically compliant and frictionless, the contact force is described as

$$F = K_f(p_e - p_o) \quad (1.6)$$

Where F is the end-effector force and K_f is the environment stiffness matrix which relates the force at the end-effector to deviations from the normal position. The validity of a generalized spring model can be verified from torque based position control method [43]. Similar derivation can be used to derive the generalized damper model that is similar to equation (1.6). This method is modeled by the relationship

$$F = B(\dot{x}_e - \dot{x}_o) \quad (1.7)$$

B is the damping matrix that relates the end-effector force to deviations from the nominal velocity. Whitney [44] used a 6 by 6 matrix K_f (spring model) to convert force error into joint position. Maples [45] compared the advantages of position based force control over velocity based force control. In his setup, a wrist FT sensor is used for the measurement of force and torque. The force error between the measured force and force set point is converted to the position error that is then fed into the inner position control loop.

1.4.1.2 Impedance Control

The concept of impedance relates the robot end-effector motion to the disturbance forces. The dynamic model for one DOF impedance is described as

$$m\ddot{p}_e + b\dot{p}_e + kp_e = F \quad (1.8)$$

where m is the mass, b is the damping and k is the stiffness, and F is the applied force. Taking the

Figure 1.5: One DOF mass-spring-damper model

Laplace transform of equation 1.8, we have

$$(ms^2 + bs + k)X(s) = F(s) \quad (1.9)$$

and the impedance $Z(s)$ is defined by the transfer function from the position disturbance to forces according to

$$Z(s) = F(s)/X(s) \quad (1.10)$$

Thus the impedance model is frequency dependent with low-frequency response dominated by the stiffness and a high frequency dominated by the mass. Another form force control also known as admittance control and the admittance is defined by the inverse of the impedance. The goal of impedance based force control is to maintain an impedance behavior at the contact point according to

$$M\delta\ddot{p}_{de} + B\delta\dot{p}_{de} + K\delta p_{de} = f_{ext} \quad (1.11)$$

1.4.2 Direct Force Control

The major disadvantage with indirect force control is that accuracy of such strategy is strongly influenced by the estimation of the stiffness matrix. However, a precise model of the stiffness matrix is often not available due to the different nature of contact between the sander and the environment. Thus, *direct force control* is designed to overcome this difficulty. Direct force control operates on the force error between the set and measured force, and the force error is treated as a disturbance to the system. Direct force control closes an outer force feedback loop and generates a desired reference input to the inner motion control scheme. Normally, the set force is assumed to be constant. The force profile could also be time varying but that would require a modification to the force control structure. The force and moment errors are defined as

$$\begin{aligned}\delta F &= F_d - F, \\ \delta M &= M_d - M\end{aligned}\tag{1.12}$$

where δM is the moment error and M_d is the desired moment. Then, the end-effector position can be chosen as

$$p_c = K_p^{-1}(K_{F_p}\delta F + K_{F_I} \int \delta dt)\tag{1.13}$$

where K_p and K_o are the position and orientation control gains, respectively, and K_{F_p} and K_{F_I} are the proportional and integral gains, respectively. The direct force control requires the design of outer force loop and the inner motion loop.

1.4.2.1 Hybrid Position/Force Control

In the following hybrid position/force control is briefly discussed [46]. First, the concepts of natural constraints and artificial constraints are introduced. Then, force control with inner loop closed on motion control and outer loop closure of force feedback, commonly known as hybrid position/force control, is presented.

1.4.2.2 Natural and Artificial Constraints

Natural constraints are defined to be the constraints determined by the task geometry [46]. Along each DOF of robot end-effector, the environment imposes either a velocity constraint or a force constraint. The velocity constraint does not allow translation or rotation along or about an axis while force constraint does not allow the application of force or torque along or about an axis. Under such definition, the robot end-effector can only control variables that are not subject to natural constraints, and the controllable variables are termed artificial constraints since they are determined by the control structure. An interaction task can be assigned in terms of a desired end-effector force h_d and velocity v_d . The operational space of robot end-effector can be decomposed into force and velocity controlled subspaces by specifying vectors λ_d and v_d and computing h_d and v_d as

$$h_d = S_f \lambda_d, \quad v_d = S_v v_d \quad (1.14)$$

Based on the decomposition of artificial constraints and natural constraints. Regulation of contact force control is obtained at the expense of the loss of motion control. Thus, motion control can only be achieved in the other directions of the end-effector frame. The difficulty in hybrid position force control is to blend together the inputs from force control law and motion law.

1.4.2.3 Control Structure

Hybrid position/force control refers to the control strategy that achieves the simultaneous regulation of the controlled force and motion. In hybrid position/force control, the six motion degrees of freedom of the robot end-effector are split into a number $n < 6$ motion controlled DOF and $f = 6 - n$ force controlled degrees. The majority of papers on force control refer to those constraints as “orthogonal”. Hybrid position/force control is based on the design of compliant motion. It assigns reference values for the forces and position together. Some directions are subject to constraints on the position/velocity and other directions are subject to constraints on the forces. The position/velocity constrained direction assumes a rigid stiff motion while the force constrained direction is made compliant [47]. In general, position and force cannot be controlled in the same

direction. In order to make things clear, two terms are defined: natural constraints and artificial constraints. Natural constraints are imposed by the environment along each degree of freedom which is defined by the geometry of the contact surface. Artificial constraints on the other hand are constraints imposed by the control system, which is specified by the designer. The advantage of hybrid position/force control is that the force and position control can be analyzed independently and then in the final stage the control inputs are blended and converted to joint torques. One thing is that there is no constant relationship between the applied force and position of end effector.

1.4.3 Force Control Application

Polishing requires the maintenance of steady tool-surface contact in terms of force direction and magnitude. This requirement is especially critical when dealing with work pieces that have complex geometries. There are various types of commercially available polishing machines, such as water jet polishing, reactive and non-reactive beam polishing. However, these machines are used for super fine finishing of surfaces, such as removal of tool marks. Our work is constrained to the automation of surface finishing processes using robots to remove surface irregularities, i.e, preprocess the surface for subsequent tasks such as coating or welding. Industry robots have been used to implement force control based on different objectives or applications. The main actuating variable is joint velocity or position, and the outer force loop is closed on an inner position/velocity loop. Several robotic systems in which force control methods are implemented for surface finishing have been developed. Efforts have been made to enhance tool path evenness to facilitate the realization of continuous surface finishing. However, the majority of those studies assume the availability of CAD surface model and have the tool path trajectories generated from the CAD/CAM system, which are both expensive and hard to adapt to other platforms. Mathiassen employed UR5 for force control. The control law is similar to that of Maples' [48]. An implicit hybrid controller is implemented for automated deburring [49]. The end effector follows a curved surface while applying a constant force normal to the surface. Rocco [49] proposed two velocity based implicit force trajectory tracking controller with environment compliance adaptively estimated. Ahn [8] developed an intelligent robotic polishing system for sculpted die. Du developed a robotic pol-

ishing method and system for the polishing operation of curved titanium alloy workpieces. Their system uses a robotic arm to follow a pre-planned path generated from CAM. Du also compared the effects of trajectory on the final surface roughness [50].

1.5 Robot Operating System (ROS)

ROS is an open-source project that provides a general framework for robotic applications. ROS is heavily utilized by the research community for several robotics applications due to its compatibility with other robotic platforms including industrial robots. ROS capabilities, such as advanced perception and path planning, can enable manufacturing robotic applications that were previously technically infeasible or cost prohibitive. Also, ROS provides a general platform for robotic applications such that the developed products can be easily reconfigured and migrate to other platforms.

1.5.1 Communications Infrastructure

The communication system is often one of the most critical needs to consider when implementing a new robot application. In addition to the standard message type defined by ROS, ROS built-in and well-tested messaging system saves time by managing the details of communication between distributed nodes via the anonymous publish/subscribe mechanism. The Node is used as an executable file within a ROS package. ROS nodes use a ROS client library to communicate with other nodes, and the communication mechanism is implemented in the ROS communications infrastructure. The built-in communications infrastructure is used for inter-process communication and they can be categorized by the following functions:

- publish/subscribe anonymous message passing,
- record and playback of messages,
- request/response remote procedure calls, and
- distributed parameter system.

The messaging passing system manages the communication between distributed nodes via anonymous publish/subscribe mechanisms. The structure of these message interfaces is defined in the message interface description language. Under this framework, the publish/subscribe system is anonymous and asynchronous. But ROS middleware also provides capabilities such as services so that the data can be sent in a service call. The ROS middleware is a way for tasks to share configuration information through a global key-value store. This allows easy modification of task settings, and even allows tasks to change the configuration of other tasks.

1.5.2 Robot Specific Features

ROS also provides robot specific tools/libraries such as

- standard robot messages,
- robot geometry library,
- robot description language,
- preemptable remote procedure calls.

One of the most important features introduced here is the standard robot messages. ROS introduces a set of standard message formats that cover most of the common use cases in robotics. There are message definitions for geometric concepts such as poses, transforms, for sensors such as cameras, IMUs, lasers, for navigation data like odometry, paths, and maps. However, force related standard ROS message hasn't been defined yet.

ROS provides a set of tools for describing and modeling robots so that they can be interpreted by the rest of ROS system such as `tf`, `robot_state_publisher`, and `rviz`. The format is URDF which shares the features of an XML document in which the user defines the physical properties of the robots from the lengths of limbs and sizes of wheels to the locations of sensors and visual appearance of each part of the robot. Figure 1.6 shows the UR5 dimension defined by the URDF.

ROS addresses transformation calculation problem with TF library. The TF library has been used to manage the coordinate transform data for robots with many DOF (up to one hundred) with

Figure 1.6: UR5 dimension from URDF [3]

update rates of hundreds of Hertz. The TF library allows the user to define both static and dynamic transforms. The TF library also handles the communication across the network, and the fact that the information is updated at varying rates.

1.5.3 Tools

One of the strongest features of ROS is the powerful development tools. These tools support introspecting, debugging, plotting and visualizing the state of the system being developed. The underlying publish/subscribe mechanism allows the user to spontaneously introspect the data flowing through the system.

1.5.3.1 Rviz

Rviz (ROS Visualization) is a 3D visualizer for displaying ROS messages. It provides general purpose, three-dimensional visualization of any URDF-described robot and many sensor data types such as laser scans, 3D point clouds and camera images. It also uses information from the TF library to express related sensor data in a common coordinate frame of user choice. Figure 1.7 gives the Rviz interface: The left column gives the Moveit panel, the middle column shows the

URDF model in Rviz, and the right column shows the ROS standard message types supported by Rviz.

Figure 1.7: RVIZ interface

1.5.3.2 RQT

RQT is a Qt-based graphical interface framework that allows the user to create custom interfaces/panel by composing and configuring the extensive library of built-in rqt plugins in to tabbed, split-screen, and other layouts. The user can also introduce new interface components by writing custom rqt plugins. The rqt_graph plug-in provides introspection and visualization of ROS nodes and connections between them. The rqt_plot plugin monitors the ROS messages such as encoders, voltages. For data logging and playback, ROS uses the bag format. Bag files can be created and accessed graphically via rqt_bag plugin. This plugin can record data to bags, play back selected topics from a bag, and visualize the contents of a bag, including display of images and plotting of numerical values over time.

1.5.3.3 *Moveit*

Moveit is a major library used for robot motion planning, manipulation, control and navigation. Moveit creates a primary node called move group. This node serves as an integrator: pulling all the individual components together to provide a set of ROS actions and services for users to use. Move group uses the ROS param server to get three kinds of information: 1) URDF. Move group looks for the robot description parameter on the ROS param server to get the URDF for the robot; 2) URDF; 3) Moveit Configuration. Move group will look on the ROS param server for other configuration specific to Moveit including joint limits, kinematics, motion planning, perception and other informations. Configuration files for these components are automatically generated by the Moveit setup assistant that are set by the user. Move group talks to the robot through ROS topics and actions. It communicates with the robot to get current state information (joint position, velocity and etc), to get sensor data and talk to the controllers on the robot.

Open motion planning library (OMPL) is an open-source motion planning library that primarily implements randomized motion planners. Moveit integrates directly with OMPL and uses the motion planners from that library as its primary/default set of planners. The planners in OMPL are abstract; i.e. OMPL has no concept of a robot. Instead, Moveit configures OMPL and provides the back-end for OMPL to work with problems in robots.

Moveit works with motion planner through a plugin interface. This allows moveit to communicate with and use different motion planners from multiple libraries, making moveit easily extensible, The interface to the motion planners is through a ROS action or service. The default motion planners for move_group are configured using OMPL and the moveit interface to OMPL by the moveit setup assistant.

Moveit uses a plugin infrastructure, which allows user to write their own inverse kinematics algorithms. Forward kinematics and finding Jacobian is integrated within the robotstate class itself. The default inverse kinematics plugin for moveit is configured using the KDL numerical jacobian-based solver. This plugin is automatically configured by the moveit setup assistant. Often, users choose to implement their own kinematics solvers. A popular approach to implementing such as

solver is using the IKFast package to generate the C++ codes.

Moveit performs collision checking via the configured files inside a planning scene using the Collision World object. Collision checking in moveit is mainly carried out by the FCL package. Collision checking is a very expensive operation often accounting for close to 90% of the computational expense during motion planning. The allowed collision matrix of ACM encodes a binary value corresponding to the need to check for collision between pairs of bodies (which could be on the robot or in the fixed world coordinate).

1.5.3.4 ROS-Industrial

The ROS-Industrial Open Source project began as the collaborative endeavor of Yaskawa Motoman Robotics, Southwest Research Institute, and Willow Garage to support the use of ROS for manufacturing automation. ROS-Industrial is an open-source project that extends the advanced capabilities of ROS to manufacturing automation and robotics. The ROS-Industrial community provides interfaces for common industrial manipulators, grippers, sensors, and device networks. It also provides software libraries for automatic 2D/3D sensor calibration, process path/motion planning, and developer tools like the Qt Creator ROS Plugin. The main advantage of ROS-Industrial is that it combines the relative strengths of ROS and existing technology, combining ROS high-level functionality with the low-level reliability and safety of an industrial robot controller.

1.6 Research Objective

Many industries have already benefited from the advancements in multi-axis machining to produce parts at low costs. The scope of this thesis is strongly motivated by the needs to automate the manual surface finishing processes with industrial robots. Specifically, the scope of this thesis focuses on the automation of manual sanding but the setup could be easily adapted to fit in as many surface finishing tasks as possible. Automation of surface finishing operations using robots is seen as a significant step in improving productivity and quality of finished work pieces. First, manual surface finishing can be expensive, time consuming, labor intensive and error prone. Polishing

is often a multistage process involving the use of different abrasives at each stage. Re-working or scrapping could easily result from a single mistake committed at a later stage, which would be expensive to compensate for. Second, a wide variety of products require polishing, such as aircraft frames, molds and dies, forgings and castings, turbine blades and other components with complicated geometric shapes. Mishandling of the surface finishing process could have serious implications in later stages of manufacturing. The successful implementation of a robotic surface finishing system requires a thorough study of the manual process. Surface finishing requires consistency, precision and time efficiency. Products should emerge evenly with the desired surface finish within a fair amount of cycle time, all of which could be achieved in a fraction of the time with robots over what is currently achieved by human operators. Further, it is expected that an articulated robotic system would be flexible to adapt to surface finishing of work pieces with different geometries, especially work pieces with free form surfaces. Therefore, the focus of this thesis is to develop an efficient strategy for robotic surface finishing of curved surfaces that includes real-time identification of the surface profile, control, and implementation. The goal is to design the system implementation that is flexible to allow for easy migration to other platforms with only minor modifications.

1.7 Contributions

This thesis develops an efficient strategy for robotic surface finishing of curved surfaces. The strategy includes real-time identification of the surface profile, control, and implementation. Real-time identification of the surface profile is accomplished by employing a proximity laser sensor mounted on the robot end-effector, which provides a true description of the surface profile in the robot base frame. This surface profile description allows us to generate trajectories for both motion and force control as it provides the surface normal at the selected point of the surface. The proximity sensor measurements coupled with robot kinematics are used to reconstruct the surface model and its normal profile. The normal profile calculation doesn't rely on any surface reconstruction algorithm and achieves results with adequate precision. Instead of using the whole normal profile, a trajectory is generated from a few sampled projection points that would orient the

surface finishing tool to the estimated local normal of the surface. An algorithm for simultaneous position and force control is developed for surface finishing of curved surfaces. The integrated robotic surface finishing system consists of a UR5 robot and a custom end-effector that includes a force/torque sensor, electromechanical sander, and the laser sensor. Robot Operating System (ROS) is utilized for real-time implementation of all aspects, which would enable easy migration of the system tools if other industrial robots are used. The effectiveness of the strategy is evaluated by conducting a number of experiments on flat and curved surfaces to demonstrate force regulation and surface finishing quality on wooden surfaces without any prior knowledge of the geometry of the curved surfaces; a representative sample of results will be presented and discussed.

2. ROBOT KINEMATICS AND CONTROL

This chapter¹ addresses the derivation and implementation of simultaneous force/position control strategy that was developed for robotic surface finishing tasks employing an industrial 6-DOF robot. A UR5 robot is employed as a testbed for conducting surface finishing experiments. The successful implementation of an automated robotic surface finishing system requires a through study of the finishing process and the physical setup of the robots. For surface finishing tasks, the contact force should be properly regulated for the successful execution of the tasks. When it comes to the industrial robots, they are known to follow a given motion trajectory with adequate accuracy. Thus, the resulting motion is characterized by stiffness. Stiff motion allows for good disturbance rejection in the presence of unexpected external forces and moments. However, the high stiffness character would also cause damage to the robot when the robot is in contact with the environment. Therefore, the contact force should be properly regulated to avoid damage to the robot as well as the work piece. However, precise force control functions are generally lacking in the majority of industrial robots; only safety mechanisms are available to avoid excessive forces applied on the robot. Therefore, the focus of this chapter is the development of simultaneous force/position control that allows for the robot to switch between motion and force control depending on whether the robot is in contact with the environment or not. In the reminder of the chapter, Section 2.1 presents the robot kinematics and Section 2.2 presents the control strategy.

2.1 Robot Kinematics

This section introduces the kinematic relations for serial link manipulators equipped with revolute joints. The objective of robot kinematics is to describe the motion of the manipulator without considering the forces and torques involved in the motion. Specifically, this section describes the computation of the forward/inverse kinematics and the Jacobian matrix that relates joint velocities with Cartesian velocities using the product of exponential formula [51, 3].

¹Part of the chapter is reprinted with permission from Yalun Wen and Prabhakar Pagilla, "Robotic Surface Finishing of Curved Surfaces: Real-Time Identification of Surface Profile and Control," presented at Manufacturing Science and Engineering Conference, June 18 - 22, 2018, College Station TX, ASME.

2.1.1 Forward Kinematics

The forward kinematics of a robot refers to the calculation of the position and orientation of robot end-effector given the the joint position of the robot. Figure 2.1 shows the dimension of UR5 and the coordinate frames at the robot base and end-effector. A commonly used convention for

Figure 2.1: UR5 dimension defined by URDF [3].

placing the coordinate frames at the robot joints is the Denavit-Hartenberg (DH) convention. In this method, a 4×4 matrix T_{ee}^b denotes the end-effector transformation matrix, composed of its position vector and orientation matrix, expressed in the robot base frame, which is given by

$$T_{ee}^b(\theta) = A_1 A_2 \dots A_5 A_6 = \begin{pmatrix} R_{ee}^b & p_{ee}^b \\ 0 & 1 \end{pmatrix} \quad (2.1)$$

Table 2.1: UR5 DH parameters

Joint	$a_i(m)$	$\alpha_i(rad)$	$d_i(m)$	$\theta_i(m)$
1	0	$\pi/2$	0.089	q_1
2	-0.425	0	0	q_2
3	-0.392	0	0	q_3
4	0	$\pi/2$	0.109	q_4
5	0	$-\pi/2$	0.095	q_5
6	0	0	0.082	q_6

where A_i is the homogeneous transformation that is given by

$$A_i = Rot_{Z,\theta_i} Trans_{Z,d_i} Trans_{X,a_i} Rot_{X,\alpha_i} \quad (2.2)$$

The four parameters a_i , α_i , d_i and θ_i are the DH parameters associated with link/joint i . The matrix Rot_{Z,θ_i} denotes rotation of θ_i with respect to its z-axis, and $Trans_{Z,d_i}$ denotes translation of d_i in the direction of its Z-axis. The same results can be derived from the Product Of Exponential (POE) formula [51, 3]:

$$T_{ee}^b(\theta) = e^{[S_1]\theta_1} e^{[S_2]\theta_2} \dots e^{[S_{n-1}]\theta_{n-1}} e^{[S_n]\theta_n} M \quad (2.3)$$

where $[S_i]$ is the skew symmetric representation of screw axes S_i expressed in the robot base frame and $M \in SE(3)$ is the end-effector configuration when the robot is at its home position (all joints positions are at 0). The DH parameters and screw axes of the UR5 robot are listed in Table 2.1 and ?? with all units in meter and radians. For implementation in this work, we utilized the POE formula because it allows for easy calculation of robot forward/inverse kinematics and the Jacobian; only the screw axes are needed for different robots which can be easily adapted to various industrial robots under the ROS framework. DH representation uses a minimum number of parameters to describe the robot kinematics: for a robot with n joints, the DH representation uses $3n$ parameters to describe the robot's structure and n parameters for joint positions. The POE formula requires $6n$ parameters to describe the robot's structure and n parameters for joint positions. The redun-

dancy reduces computation complexity and avoids the singularity issues commonly encountered in Euler's angle expression. Therefore, POE is preferred.

Table 2.2: UR5 safety limiting sets

Limiting Safety Function	Maximum Limit
TCP Force	250N
TCP Speed	5000mm/s
Joint Speed	30°/s
Momentum	100 kg m/s
Power	80w

2.1.1.1 UR5 Forward Kinematics Verification

The translation part of the end-effector shown on the PolyScope in ROS is used to verify the correctness of the forward kinematics implementation. If we set the joint position θ to be $\theta = \begin{bmatrix} 0.1, -1.4, 1.64, -2.00, -1.57, 1.52 \end{bmatrix}^T$, the translation part of end effector with respect to the

base frame shown on the UR polyscope is $\begin{bmatrix} -547.75 \text{ mm} \\ -167.02 \text{ mm} \\ 352.50 \text{ mm} \end{bmatrix}$. The result from our implementation

is $\begin{bmatrix} -547.83 \text{ mm} \\ -167.16 \text{ mm} \\ 314.75 \text{ mm} \end{bmatrix}$.

2.1.2 Inverse Kinematics

The Inverse Kinematics (IK) module uses the kinematics relations to determine joint positions from a given desired end-effector position and orientation T_{ee}^b . In this work, IK is derived using the Newton-Raphson method, which gives a local minimum solution. Therefore, the initial condition must be close enough to the desired solution in terms of the joint space distance. In other words,

the trajectory should be discretized suitably so that the local minimum would match the desired solution. A detailed discussion of this method can be found in [3].

2.1.2.1 UR5 Inverse Kinematics Verification

The validity of our IK implementation is verified via the comparison between the calculated IK solution to the closed form solution of UR5 [52]. The geometric IK solution is comprised of 8 solutions. The closed form solution has the shortest distance in joint space from the current robot joint position. An experiment was performed to test the correctness of IK implementation: end-effector pose was set to be $T_{end} = \begin{bmatrix} 1.1, 2.4, 1.5, 2.8, 2.2, 2.3 \end{bmatrix}^T$, and its geometric IK solution is found to be

$$q_{sol} = \begin{bmatrix} 3.90 & 6.03 & 1.10 & 1.93 & 1.25 & 5.29 \\ 3.90 & 0.80 & 5.17 & 3.08 & 1.25 & 5.291 \\ 3.90 & 5.60 & 1.47 & 5.13 & 5.02 & 2.15 \\ 3.90 & 0.72 & 4.80 & 0.40 & 5.02 & 2.16 \\ 1.10 & 2.40 & 1.50 & 2.80 & 2.20 & 2.30 \\ 1.10 & 3.82 & 4.78 & 4.37 & 2.20 & 2.30 \\ 1.10 & 2.35 & 1.08 & 0.12 & 4.08 & 5.44 \\ 1.10 & 3.38 & 5.20 & 1.25 & 4.08 & 5.44 \end{bmatrix} \quad (2.4)$$

By converting all the solutions to be in the range of $[-2\pi, 2\pi]$, the correctness of the IK solution given by Newton-Raphson method is verified.

2.1.3 Jacobian Matrix

The Jacobian matrix relates the end-effector velocity V_{ee} (three angular velocities plus three linear velocities) with the joint velocities $\dot{\theta}$. The Jacobian is one of the most important quantities in the analysis and control of robot motion. It arises in virtually every aspect of robotic manipulation: in the planning and execution of smooth trajectories, in the determination of singular configurations, etc. Two kinds of singularities are distinguished. Boundary singularities occur when the joints of the robot are fully extended and the robot is asked then to move beyond where it can

be reached. Internal singularities are caused by kinematic alignments. At internal singularities, infinite inverse kinematic solutions may exist for a given end-effector position or infinite joint velocities are required for small Cartesian motions.

We distinguish two kinds of Jacobian formulations: the space Jacobian is related to the end-effector velocity expressed in the robot base frame, and the body Jacobian is defined to be the end-effector velocity expressed in the end-effector frame. In the following, we briefly show how to compute the space Jacobian J_s using the POE formula. Similar procedure can be employed to derive the body Jacobian J_b . The space Jacobian J_s is given by

$$V_{ee} = J_s \dot{\theta}. \quad (2.5)$$

From equation (2.3),

$$\begin{aligned} \dot{T}_{ee}^b &= \frac{d}{dt} e^{[S_1]\theta_1} \dots e^{[S_n]\theta_n} M + e^{[S_1]\theta_1} \left(\frac{d}{dt} e^{[S_2]\theta_2} \right) \dots e^{[S_n]\theta_n} M + \dots \\ &= [S_1] \dot{\theta}_1 e^{[S_1]\theta_1} \dots e^{[S_n]\theta_n} M + e^{[S_1]\theta_1} [S_2] \dot{\theta}_2 e^{[S_2]\theta_2} \dots e^{[S_n]\theta_n} M + \dots \end{aligned} \quad (2.6)$$

Since $[V_{ee}] = \dot{T}_{ee}^b T_b^{ee}$ and $T_b^{ee} = M^{-1} e^{-[S_n]\theta_n} \dots e^{-[S_1]\theta_1}$, we have the following relationship

$$\begin{aligned} [V_{ee}] &= [S_1] \dot{\theta}_1 + e^{[S_1]\theta_1} [S_2] e^{-[S_1]\theta_1} \dot{\theta}_2 + \dots \\ &= J_{s_1}(\theta) \dot{\theta}_1 + \dots + J_{s_n}(\theta) \dot{\theta}_n \end{aligned} \quad (2.7)$$

where θ_i is the joint velocity at joint i . Thus, J_s is given by

$$J_s(\theta) = \begin{bmatrix} J_{s_1}(\theta) & J_{s_2}(\theta) & \dots & J_{s_n}(\theta) \end{bmatrix} \quad (2.8)$$

The Jacobian can also be used in the analysis of static force/torque relationships. The forces and moments produced when the end-effector is in contact with the environment are described by the following relationship.

$$\tau = J^T(\theta) F \quad (2.9)$$

where $J^T(\theta)$ is the transpose of the Jacobian matrix, τ is the vector of joint torques, and F is the external force. The derivation of the static force/torque relationship comes from the principle of virtual work. The same derivation can be applied for the velocity based force control law.

2.1.3.1 UR5 Jacobian Verification

The Jacobian matrix calculation within ROS is verified via the translation and rotation parts. Denote the function to calculate forward kinematics as $fw_kin(\theta)$, where θ is the vector of joint positions. The method employed in [53] is used to verify the Jacobian matrix. Let δ denote the infinitesimal displacements in the joint space. Then, we will evaluate if the following relationship in equations (2.10) and (2.13) would be satisfied.

$$fw_kin(\theta + \delta)_{translation} - fw_kin(\theta)_{translation} = J_{translation}\delta. \quad (2.10)$$

Given the joint position $\theta = \begin{bmatrix} 0.30, -2.09, 2.33, -1.93, -1.43, 0.69 \end{bmatrix}^T$, and the variant angle $\delta = \begin{bmatrix} 0.003, 0.006, 0.007, 0.005, 0.003, 0.005 \end{bmatrix}^T$. The LHS of equation (2.10) is

$$fw_kin(\theta + \delta)_{translation} - fw_kin(\theta)_{translation} = \begin{bmatrix} 8.73e^{-05} \\ -8.40e^{-05} \\ -5.51e^{-4} \end{bmatrix}. \quad (2.11)$$

The RHS of equation 2.10, which is the calculation error is

$$J_{translation}\delta_{first_three} = \begin{bmatrix} 8.68e^{-05} \\ -8.41e^{-05} \\ -5.52e^{-4} \end{bmatrix}. \quad (2.12)$$

The rotation part of the Jacobian matrix is verified by the Frobenius norm according to

$$\|A - B\|_F^2 = Tr((A - B)^T(A - B)) = 0 \quad (2.13)$$

Figure 2.2: Circular motion from Jacobian kinematics

where

$$A = (R_6^0(\theta + \delta) - R_6^0(\theta))R_6^{0T}(\theta), \quad B = J_{rotation}\delta. \quad (2.14)$$

The Frobenius difference between A and B is $8.42669e^{-12}$ which is taken to be reasonable considering the numerical precision and round-off error involved. Figure 2.2 shows the trajectory (green circle) of the UR5 end-effector with Rivz. The joint velocity is calculated using the Jacobian such that the end-effector is drawing a circle. The distance between the top trajectory to the bottom one is 4 mm, which corroborates the closeness of the Jacobian matrix calculation.

2.2 Simultaneous Force/Position Control Strategy

This section describes the simultaneous force/position control architecture in the task space of the robot. Section 2.2.1 presents the inner loop for motion control (pose tracking), Section 2.2.2 discusses the one dimensional force control strategy, and section 2.2.3 provides the simultaneous force/position control.

The objective of force control is to maintain a constant normal force at the robot end-effector. There are challenges in both the design of the control strategy as well as its implementation. First, the traditional force control implementation requires an accurate dynamic model of the robot and

a relatively precise estimation of the stiffness matrix of the environment, which are hard to derive and often the precision is not guaranteed since industrial robots are equipped with high gain gear box with unknown friction at each joint. The problem with most model-based controllers is that these controllers tend to be very sensitive to model mismatches or system disturbances (friction). Without a properly defined model, the controller may behave poorly and sometimes even instability may occur in some cases. Therefore, the robustness to unknown environment forces, caused by the vibration of surface finishing tools, and modelling errors has to be accounted for in the controller design. Second, most work in the literature on this topic assumes the actuating variable of force control is joint torque which is very rarely available as a control variable in most industrial robots for security issues. Almost all industry robots only provide access to joint velocity/position interface to users for real-time control, i.e., the controller inputs cannot be joint torques. Thus, a new controller design is required for simultaneous force/control strategy. Third, force control usually requires a faster controller rate than the servo rate of the motion controller. This factor further limits the performance of the force controller since the inner servo control loop is closed by the manufacturer and typically remains unknown to the end-user. Therefore, the objective of this section is to develop a simultaneous force/position control strategy for industrial robots with joint velocity as the controlled inputs. For this implementation, the outer control loop closes on the force/torque error which is measured by a six-axis force/torque sensors. The force error is then converted to end-effector position via a compliance model that is then controlled through inner motion control loop using joint velocity. The compliance model takes the form of a generalized spring/damper with an estimated environmental stiffness. We utilized simple PI controllers for system robustness and to ensure zero steady error in the steady state.

Compliant motion is typically referred to as the stiffness control for static behaviors and impedance control for dynamic behaviors [43]. Also, we differentiate between active versus passive compliant motion control depending on the way the reading of external force sensor is used in generating the desired compliance. Compliant motion control strategies can be categorized into two main categories: *indirect* force control and *direct* force control. *Indirect* force control seeks to create

a desired compliance/impedance relationship at the contact point. If the robot is in point contact with a plane, the normal direction of the plane is usually chosen to be the compliant direction. The control strategy is implemented by selecting the position control gains in the end-effector space. The proportional gain for the compliant motion direction is usually different from the other directions that are used for motion control. The proportional gain for the compliant direction is selected to produce a desired stiffness while the gains for the motion controlled directions are selected to achieve desired motion control bandwidth. Since the force controlled direction has faster dynamics, the proportional gains for the compliant direction are usually selected larger.

The force regulation is achieved via the maintenance of a complaint relationship between the end-effector and the environment. The compliant model does not explicitly control the contact forces. It regulates the force by composing the goal position using the environment stiffness/damping. Therefore, under the generalized spring/damper model, the force control problem becomes motion control problem and the control structure is discussed in Section 2.2.1.

During robotics surface finishing, the force set points are given in advance to be tracked precisely. The complaint model is used to convert the force error between the desired and measured contact forces to end-effector positions. Therefore, the force error is the system disturbance and the goal of the controller design is to reject this disturbance. Our approach involves designing an external force feedback loop around the inner position feedback loop. The external force controller utilized an estimated environmental stiffness to determine the desired end-effector position. If the goal is solely force regulation, then the design process stops here. However, the problem becomes more complicated when it comes to the simultaneous force/position control. The force control loop and position control loop are usually designed in parallel since they have different system dynamics and design criteria. To achieve simultaneous force/position control, a blending mechanism should be developed since the robot controller can only take one type of control input, position or velocity, in each time step otherwise it is not "simultaneous" force/position control. Therefore, the following two problems are encountered and should be addressed:

- 1) The coordinate system should be selected properly for the decomposition of system DOF:

the motion controlled axis must be perpendicular to the force controlled axis. In our setup, the end-effector frame is used: the z -axis of the end-effector frame is selected as the force controlled axis, and the other axes are position controlled. The majority of studies in the field of robotic surface finishing do not seem to address this problem adequately. The selection of the coordinate frame to decompose is not a trivial task, especially for finishing on free-form surfaces. The local normal of the free-form surface must be selected as the force controlled axis that must be one of the coordinate axis of the end-effector frame or the frames of any other moving joint. The base coordinate cannot be used since its coordinate axes are fixed in space. This means the forward/inverse kinematics, Jacobian and all related calculation must be defined based on the moving coordinate frame.

2) The control inputs for the force and position control loops should be independent to each other. Two control options are available that are based on the Jacobian transpose and the Jacobian inverse (Jacobian Kinematics). The Jacobian transpose method relates joint position to end-effector position while the Jacobian inverse method relates joint velocity to end-effector velocity which can be used to control the end-effector position via pose tracking control. Let $\Delta\theta$ denote the joint angle increment, $\Delta\vec{e}$ be the end-effector position increment, and J as the corresponding Jacobian matrix. The Jacobian transpose method is illustrated via the following relationships where $\alpha > 0$ is some appropriate constant that is also sufficiently small [54, 55]:

$$\Delta\theta = \alpha J^T \Delta\vec{e} \quad (2.15)$$

A damped least squares method could also be used for numerical stability and precision [56]. So, a similar relationship can be derived

$$\Delta\theta = J^T (JJ^T + \lambda^2 I)^{-1} \Delta\vec{e} \quad (2.16)$$

where λ is a non-zero damping constant. The Jacobian inverse method is also known as the Jaco-

bian kinematics that relates the joint velocity $\dot{\theta}$ to the end-effector velocity V_{ee} by

$$V_{ee} = J\dot{\theta}. \quad (2.17)$$

The Jacobian kinematics method is preferable over Jacobian transpose method for the following reasons:

- **Signal delay:** According to [57], commanding joint position would introduce a delay of 170 ms from the time when the command is sent until joint starts to move; joint speed based streaming of commands will result in a delay somewhere between one and two sampling periods (8 ms to 16 ms) depending on when in the control cycle the new values are read.
- **Performance stability:** Velocity based force controller provides a smoother response as opposed to position based force controller. The simplified impedance model is given by

$$F = m\ddot{p} + c\dot{p} \quad (2.18)$$

where F is the force vector measured, m is the virtual mass, c is the virtual damping, $\ddot{p}$ and $\dot{p}$ are the generalized acceleration and velocity respectively. Note that there is no position information in the above simplified impedance model. Assuming the force is the input to the system and the velocity is the output, the transfer function from force to velocity is given by

$$G(s) = \frac{1/c}{(m/c)s + 1}. \quad (2.19)$$

For a given step force input, velocity based force control behaves as a low pass filter. Consequently, it provides less vibrations to high frequency force transients. This conclusion matches with the observations that position based force control would either vibrate a lot at steady state or introduce large contact force and bouncing during contact transition, which would often trigger the built-in protective stop of the robot. Further details about the advantages of velocity based force control can be found in [58, 59].

- Singularity: In our setup, we assumed the workpiece is registered precisely within the workspace. Since the UR5 is a serial robot, we can expect singularity would not be encountered and therefore the numerical advantages provided by the Jacobian transpose method can be ignored.

Therefore, the compliant model plus pose tracking via joint velocity are used for the controller design.

2.2.1 Pose Tracking Control

In this subsection, we discuss the method of using joint velocities $\dot{\theta}$ to move the end-effector from the initial pose T_i to the desired pose T_d ; both the initial and desired transformations for the end-effector are expressed in the robot base frame. Let (R_i, p_i) and (R_d, p_d) represent the rotation matrix and translation vector of T_i and T_d , respectively. Let $\hat{\omega}$ represent the rotation axis and θ_e represent the rotation angle with respect to $\hat{\omega}$. Using the POE formula we have:

$$Rot(\hat{\omega}, \theta) = e^{[\hat{\omega}]\theta_e} = R_i R_f^T \quad (2.20)$$

where $(\hat{\omega}, \theta)$ is the exponential coordinates associated with the rotation matrix $R_i R_f^T$, $[\hat{\omega}]$ operation is the skew symmetric representation for the vector $\hat{\omega}$. Let the end-effector velocity be represented by a minimum set of coordinates $V_{ee} = \begin{bmatrix} \omega & v \end{bmatrix}^T$

Figure 2.3: Block diagram for pose tracking control

Then, the pose tracking control law to move the end-effector from the initial pose T_i to the final

pose T_d is given by

$$\dot{\theta} = J_b^{-1} \begin{bmatrix} -K_v(p_i - p_d) - K_i \int_0^\tau (p_i - p_d) d\tau \\ -K'_v(\hat{\omega}\theta_e) - K'_i \int_0^\tau \hat{\omega}\theta_e d\tau \end{bmatrix} \quad (2.21)$$

The feedback control block diagram for this type of pose tracking control is given in Figure 2.3; in the figure, K_v, K'_v, K_i, K'_i are controller gains; "Matrix log" is the inverse calculation of matrix exponential. The input to the block "Matrix log" is a rotation matrix $R \in SO(3)$ and the output is its representation in exponential coordinates $[\hat{\omega}]\theta_e$ as shown in equation (2.20) .

2.2.2 Normal Force Control

This subsection introduces the control law to maintain a constant normal force between surface and surface finishing tool. The robot is assumed to be rigid during contact, and the contact model between the surface finishing tool and environment is assumed to be compliant. A generalized spring model [47] is employed to transform force error F_e to position error according to

$$F_e = K_{fe}(P_\delta(k+1) - P_\delta(k)) = K_{fe}P_\delta \quad (2.22)$$

Where K_{fe} is the stiffness matrix, $P_\delta(k)$ is the current time step Cartesian position vector of the robot end-effector. $P_\delta(k+1)$ is the next time step Cartesian position vector which is then fed into the pose tracking loop to be converted to the joint velocity. Figure 2.4 shows the control block diagram for normal force control where P_c is the current robot end-effector position calculated via forward kinematics, P_δ is the change of Cartesian position.

Figure 2.4: Block diagram for normal force control

2.2.3 Simultaneous Force/Position Control

The controller design in this section draws idea from hybrid position force control that achieves simultaneously control of the position and force of the robot manipulator [46]. But our method is different since no dynamic model of the robot is required.

The contact between the robot end-effector and environment is assumed to be compliant. The goal of impedance control is convert the force control problem to a motion control problem. Directly impedance control acts on the errors between the set and measured forces. The typical force control structure involves designing an external loop around robot's motion controller. Figure 2.4 shows the general one-dimensional external force control diagram. The error F_e between the set force F_s and actual force F_r is fed into the force control loop, which generates an end-effector position command P_δ . The contact force acts as a disturbance on the position-control loop. Design of the external force controller is accomplished by incorporating a model of the environmental stiffness. The selection of the stiffness is critical for proper controller design. In practice, a simple linear spring model is used to connect robot force control and robot motion control. Then the operational space is decomposed into force controlled space and motion controlled space. A way to synthesize simultaneous force/motion control strategy is proposed. The proportional gains for the compliant direct is usually chosen to produce a desired contact relationship, and the gains for the other direction would be chosen to achieve the desired motion control bandwidth.

To simplify the notation for simultaneous force/position control, the z -axis of the end-effector is considered as the direction along which the force control is employed while the rest of axes are used for position tracking. The control block diagram for this strategy is given in Figure 2.5, where S is the selection matrix that determines which axis of motion is force controlled.

where F is the force vector measured, m is the virtual mass, c is the virtual damping, $\ddot{p}$ and $\dot{p}$ are the generalized acceleration and velocity respectively. Note that there is no position information in the above simplified impedance model. Assuming the force is system's input and the velocity is

Figure 2.5: Block diagram for simultaneous force/position control

the output, the transfer function from force to velocity is given by

$$G(s) = \frac{1/c}{(m/c)s + 1} \quad (2.23)$$

Therefore, for a given step force input, velocity based force control behaves like a low pass filter. Consequently, it provides less vibrations to high frequency force transients. This conclusion matches with the observations that position based force control would either vibrate a lot at steady state or introduce large contact force and bouncing during contact transition. Further details about the advantages of velocity based force control can be found in [58, 59].

2.2.4 Stability Analysis

2.2.4.1 Pose Tracking Stability

Pose tracking employs joint velocity to track a given end-effector pose. The pose P is composed of three translation coordinates p and three rotation coordinates (Euler). The proof of convergence of the translation part is trivial and this subsection addresses the stability of the rotation part. Let's define the rotation error e_r to be

$$e_r = \ln(R^{-1}R_d) \quad (2.24)$$

Where R is the current rotation matrix for the end-effector frame expressed in the robot base frame, R_d is the desired end-effector frame expressed in the robot base frame. Note that the error $e_r = 0$

when the rotation matrix R takes the value of R_d . The error dynamics is

$$\dot{e}_r = \frac{1}{R^{-1}R_d}((\dot{R}^T)R_d + R^T(\dot{R}_d)). \quad (2.25)$$

Since R_d is usually given and assumed to be constant, the above equation can be simplified to

$$\dot{e}_r = \frac{R}{R_d}((\dot{R}^T)R_d). \quad (2.26)$$

The derivative of the rotation matrix R is

$$\dot{R} = \omega_s \times R = [\omega_s]R \quad (2.27)$$

where ω_s is the angular velocity of the end-effector expressed in the robot base frame assuming R is a function of time $R(t)$. Recall in pose tracking, the skew symmetric representation of the end-effector angular velocity ω_s is derived in equation (2.20). Hence, when only a proportional controller is employed, ω_s takes the form of

$$[\omega_s] = -k_w[\hat{\omega}]\theta_e = -k_w \log^{RR_d^T} \quad (2.28)$$

where $[\omega]$ is the skew-symmetric representation of ω . Therefore,

$$\dot{R}^T = -R^T k_w \log^{R^T R_d}. \quad (2.29)$$

Substitution of equation (2.29) into (2.26), we obtain

$$\dot{e}_r = -\frac{RR^T}{R_d} k_w \log^{R^T R_d} R_d = -k_w \log^{R^T R_d} = -k_w \log^{R^{-1} R_d} = -k_w e_r \quad (2.30)$$

which provides exponential convergence of the error $e_r(t)$ to zero. Since the translation part would also converge to zero, therefore P_e would converge to zero and $\dot{P}_e$ is bounded.

2.2.4.2 Stability of Force Control

We assume the contact model between the robot and the environment is elastically compliant. That is, the force applied on the robot end-effector satisfies the following relationship:

$$F_r(t) = K_{fe}(P(t) - P_0) \quad (2.31)$$

where $P(t)$ is the end-effector position in the robot base frame and P_0 is the end-effector position corresponding to the equilibrium point (no external force), K_{fe} is the stiffness matrix. Using (2.31), we can write

$$\dot{P}(t) = \frac{\dot{F}_r}{K_{fe}} + \dot{P}_0 \quad (2.32)$$

Let F_d be the desired force set point, K_p be the proportional gain and $P_d(t)$ as the desired end-effector position in robot base frame. The control law is given by

$$\dot{P}_d(t) = -K_p F_e = -K_p (F_r - F_d). \quad (2.33)$$

Define $P_e(t) = P(t) - P_d(t)$ as the end-effector error. Then, from equation (2.32), we have

$$\begin{aligned} \dot{P}_e(t) &= \dot{P}(t) - \dot{P}_d(t) \\ \dot{P}_d(t) &= \frac{\dot{F}_r}{K_{fe}} + \dot{P}_0 - \dot{P}_e(t) \end{aligned} \quad (2.34)$$

Substituting (2.34) into (2.33), we have the following relationship

$$\begin{aligned} \frac{\dot{F}_r}{K_{fe}} + \dot{P}_0 - \dot{P}_e(t) &= -K_p F_e \\ \dot{F}_r + K_{fe} K_p F_e &= K_{fe} (\dot{P}_e(t) - \dot{P}_0) \end{aligned} \quad (2.35)$$

Assuming the force set point F_r is a constant then $\dot{F}_r$ is zero. Since P_0 is also a constant, equation (2.35) becomes

$$\dot{F}_e + K_{fe} K_p F_e = K_{fe} \dot{P}_e. \quad (2.36)$$

3. WORK PIECE SURFACE PROFILE IDENTIFICATION

In this chapter ¹, we consider a novel non-contact method to obtain the surface normal profile and to generate corresponding tool-path trajectory in real-time. Define the projection point to be the point where the laser beam hits the surface. A proximity laser sensor mounted on the end-effector is employed to obtain the distance from the sensor to the surface (projection point). The distance reading is treated as an extended robot link to form the new robot kinematics. Then the position of the projection point is calculated via the newly formed robot forward kinematics. The entire set of projection points is combined to form a grid, where the nearby points are connected by lines to calculate local surface normals. Further, this set of sampled projection points can be employed as a point cloud data for further processing. After scanning, a full set of sampled points are connected as a grid as shown in Figure 3.1. Compared to a vision system or other optical sensors that are used to obtain the surface profile, the proposed method permits the calculation of local surface normal at a specific point and the generation of corresponding tool path trajectory. Since no internal/external calibration is needed, the system setup time is significantly reduced compared to the other visual perception technique with stereo vision. Further, instead of using every local normal to form a trajectory, the trajectory is generated based on the local normal of a few sampled projection points to accelerate the processing time and reduce the cycle time. The local normal is estimated from the two consecutive sampled projection points to form the trajectory at the current time step while the end-effector is traveling between them. This method allows for higher geometry accuracy at a particular region by increasing the number of projection points in that area. Our method provides significant advantages over industry practices where path planning is based on known part geometries assuming the part location and orientation given in advance. First, our method does not require any knowledge of part geometry information. Our method also minimizes the effect of fixture uncertainty since the surface is expressed in the robot base

¹Part of the chapter is reprinted with permission from Yalun Wen and Prabhakar Pagilla, "Robotic Surface Finishing of Curved Surfaces: Real-Time Identification of Surface Profile and Control," presented at Manufacturing Science and Engineering Conference, June 18 - 22, 2018, College Station TX, ASME.

frame. Further the proposed method does not rely on matching the scanned surface with the CAD model which is generally a difficult problem since it is difficult to transform the scanned surface to the CAD model without proper knowledge of the orientation information of the part. Lastly, the trajectory is generated in real-time that incorporates the local surface normal information which is especially important for finishing of free-form surfaces.

This chapter introduces the method to identify the surface normals and path generation designed specifically for simultaneous position/force control on a curved surface. Section 3.1 discusses this novel non-contact method to identify and reconstruct the surface. Section 3.2 introduces a time scaling method for trajectory generation based on the sampled projection points obtained from our surface normal identification step.

Figure 3.1: Grid data with local surface normal

3.1 Work Piece Surface Registration

3.1.1 Position Interpolation of Laser Sensor Projection Point

This subsection calculates the transformation of the projection point T_p^b with respect to the robot base frame based on the current laser reading L_r . The laser reading is used as an extended robot link to calculate the position/orientation of the projection point via the newly formed robot forward kinematics. In our setup, while scanning the surface, the motion of the end-effector is constrained such that the xy -plane of the end-effector frame is parallel to that of the robot base frame for obtaining maximum average precision of the laser sensor. Denote the pitch angle of the projection point with respect to the robot base frame as α_p and the pitch angle of the end-effector frame with respect to the robot base as α_{ee} . The transformation from the projection point to the base frame T_p^b is given by

$$T_p^b = \begin{bmatrix} 1 & 0 & 0 & 0.076\cos(\alpha_p) \\ 0 & 1 & 0 & 0.076\sin(\alpha_p) \\ 0 & 0 & 1 & -(L_r + 0.093) \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{ee}^b \quad (3.1)$$

where 0.093 m is the distance between the laser sensor and the robot end-effector xy -plane. Figure 3.2 shows the deployment of the sampled points (blue dots). The coordinates of each point expressed in the robot base frame are calculated and then stored.

3.1.2 Surface Reconstruction

This subsection introduces the method to generate the surface profile from the sampled projection points. Denote $T_p^b = (R_p, p_p)$ where R_p and p_p are the rotation matrix and translation vector expressed in the robot base frame. The goal is to generate the surface profile of the part shown in Figure 3.3(a). First, while the robot end-effector is moving horizontally to cover the part, measurements from laser sensor are taken to calculate the coordinates of the sample points. Then, the i -th projection point's translation, p_p' , is stored. This procedure is illustrated in Figure 3.2. The dots

Figure 3.2: Sampled projection points on part

represent the sampled projection points. The method from the previous subsection provides the location information of each sampled point which is used to reconstruct the surface profile with triangulation-based linear interpolation. The raw data and its reconstructed surface are presented in Figure 3.3.

3.1.3 Surface Normal Estimation

This subsection utilizes the surface model and describes the method to generate the surface normal profile. In Figure 3.4, the goal is to calculate the local normal at point O_1 . First, use the same scanning strategy to get the coordinates of A, B, C, D, O_1 (non-collinear). In our setup, $|AO_1| = |BO_1| = |CO_1| = |DO_1| = 0.004 \text{ m}$. Denote $\overrightarrow{O_1N_1} = \overrightarrow{O_1A} \times \overrightarrow{O_1B}, \dots, \overrightarrow{O_1N_4} = \overrightarrow{O_1C} \times \overrightarrow{O_1D}$, then the local surface normal $\overrightarrow{O_1N}$ can be approximated with

$$\overrightarrow{O_1N} = \frac{\overrightarrow{O_1N_1} + \overrightarrow{O_1N_2} + \overrightarrow{O_1N_3} + \overrightarrow{O_1N_4}}{4}. \quad (3.2)$$

3.1.4 Surface Profile Reconstruction Accuracy

In this section, we discuss the accuracy of the reconstructed surface with respect to the sample distance via an example simulation. The objective is to find the proper sample distance to obtain the

(a) Curved work piece

(b) Identified work piece

Figure 3.3: Work piece VS surface model

Figure 3.4: Surface normal calculation

local surface normal with adequate precision. Please note that our local normal calculation does not rely on any reconstruction method and the reconstructed cylinder is used for demonstration only. The concept of sample point and sampled distance is illustrated in Figure 3.5 (E), where sampled point is the intersection point on each layer. In the simulation, we assume the coordinate of each sample point is known in advance. In practice, the sampled point is the same as projection point so this assumption is reasonable. The original cylinder and its local normal are shown in Figure 3.5 (A)&(C) respectively. The local normal is calculated via MATLAB command *surfnorm*. We perturb all the sample points in (A) with the truncated Gaussian with the mean of 0.0005 m (the precision of the laser sensor). Now, we have the noisy sample point as shown in (B). Then we calculate the local surface normal via our method and the results are shown in Figure 3.5 (D). Figure 3.5 (E) is the top view of the cylinder (A). Each layer consists of 20 equally spaced points, and the diameter of each layer decreases from 0.01 m to a point. Therefore, as we go from the bottom layer to the top layer, the sampled distance is decreasing. If we compare the surface normal calculated at each layer with that of the correct one, the results are shown in Figure 3.6. The error angle is defined to be the angle between the calculated surface normal and the correct

Figure 3.5: Surface normal

Figure 3.6: Error angle VS sample distance

one from MATLAB. Figure 3.6 shows the average error calculated from 20 points on each layer versus the sample distance. As we can tell, the error decreases rapidly when the distance changes from 0.05 m to 0.035 m since the dot product can only give the correct surface normal calculation close to the target point. At 0.005 m, our method gives an error of 5° . In our setup, we choose 5 mm as the sample distance and the results obtained are found to be sufficiently accurate.

3.2 Trajectory Generation

This subsection deals with the problem of how to utilize the surface grid to generate a suitable tool path. The goal of trajectory generation is to provide the robot controller with a stream of goal positions to track. The specification of the robot position as a function of time is defined as a trajectory. We will assume that the initial and final configurations are specified and the problem is to generate a path that would change the orientation of the end-effector.

Trajectory teaching, cutter location data, and off-line programming are some of the methods used in existing literature for polishing of free-form surfaces. Based on the requirements of surface finishing tasks, such methods could be error prone, skill dependent and time consuming. There-

fore, an automatic trajectory generation scheme is desirable. Upon the acquisition of the surface geometry and location information, the trajectory should incorporate the local normal information of the surface to orient the end-effector force applied on the surface for material removal. We consider a variation of the time scaling method to incorporate local surface information, which is discussed below.

3.2.1 Time Scaling

A path $\theta(s(t))$ maps a scalar path parameter s to a point in the robot configuration space. The parameter s is assumed to take the value of ‘0’ at the start of the path and the value of ‘1’ at the end. As s increases from 0 to 1, the robot moves along the path. The parameter s is called time scaling because it maps a time $t \in [0, T]$ to a value $s \in [0, 1]$. Thus, a time scaling factor ($s(t)$) and a path defines a trajectory $\theta(s(t))$. Based on this approach, the following relationships between the parameter s and joint velocity and acceleration can be derived:

$$\begin{aligned}\dot{\theta} &= \frac{d\theta}{ds} \dot{s} \\ \ddot{\theta} &= \frac{d\theta}{ds} \ddot{s} + \frac{d^2\theta}{ds^2} \dot{s}^2\end{aligned}\tag{3.3}$$

3.2.1.1 Joint Space Straight Line

The straight line in joint space takes the following form:

$$\theta(s(t)) = \theta_{start} + s(\theta_{end} - \theta_{start}) \quad s \in [0, 1]\tag{3.4}$$

and the velocity and acceleration are given by

$$\begin{aligned}\frac{d\theta}{ds} &= \theta_{end} - \theta_{start}, \\ \frac{d^2\theta}{ds^2} &= 0.\end{aligned}\tag{3.5}$$

3.2.1.2 Task Space Straight Line

The formulation of a straight line in the task space takes a similar form:

$$X(s(t)) = X_{start} + s(X_{end} - X_{start}) \quad s(t) \in [0, 1] \quad (3.6)$$

where $X(s(t))$ is the minimum set of coordinates to represent robot end-effector pose. If $X(s(t))$ is represented by a 4×4 transformation matrix from base to end effector, the POE formula can be used to rewrite equation (3.16). Taking the exponential of both sides of equation (3.16), we obtain

$$XX_s = XX_{start} \exp(\log(XX_{start}^{-1}XX_{end})s) \quad (3.7)$$

where XX_s is the exponential expression of X_s . The list of goal points generated from the above method follows the screw motion. The path is defined by

$$\begin{aligned} p(s(t)) &= p_{start} + s(p_{end} - p_{start}) \\ R(s(t)) &= R_{start} \exp(\log(R_{start}^T R_{end})s) \end{aligned} \quad (3.8)$$

where $p(s(t))$ and $R(s(t))$ are the translation vector and rotation matrix, respectively, with respect to the base frame. One advantage of time-scaling method is that the generated list of "points" discretizes the trajectory so that the Newton-Raphson method can be used for computing the inverse kinematics.

Figure 3.7 shows the configuration space which is the gray area and the straight line in the task space. In our setup, we assume a collision free environment and the configuration space is reachable. Thus we consider right figure shown in Figure 3.7 and not the left figure. A time scaling $s(t)$ of a path should ensure that the motion is appropriately smooth, and any constraints on robot

Figure 3.7: Straight line in configuration space

velocity and acceleration are satisfied. Given the relationship in equation (3.4), we can see that

$$\begin{aligned}\dot{\theta} &= \dot{s}(\theta_{end} - \theta_{start}) \\ \ddot{\theta} &= \ddot{s}(\theta_{end} - \theta_{start})\end{aligned}\tag{3.9}$$

For a straight line path in the task space parameterized by the minimum set of coordinates, replace $\theta, \dot{\theta}, \ddot{\theta}$ with $X, \dot{X}, \ddot{X}$. In our setup, a third-degree polynomial is used as the time-scaling function and it takes the form:

$$s(t) = a_0 + a_1t + a_2t^2 + a_3t^3,\tag{3.10}$$

whose derivative is

$$\dot{s}(t) = a_1 + 2a_2t + 3a_3t^2.\tag{3.11}$$

Note that we also have four constraints $s(0) = \dot{s}(0) = 0$ and $s(T) = 1, \dot{s}(T) = 0$ and we can obtain the coefficients as

$$a_0 = 0, a_1 = 0, a_2 = \frac{3}{T^2}, a_3 = -\frac{2}{T^3}.\tag{3.12}$$

Note that we can derive similar results by considering fifth degree polynomials as

$$s(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5\tag{3.13}$$

We assume the given two end-effector configurations as $T_i^b, T_d^b \in SE(3)$. The desired transformation with respect to the initial frame is

$$T_d^i = (T_i^b)^{-1}T_d^b. \quad (3.14)$$

Then, the matrix representation of the twist expressed in the initial frame is $\log(T_d^i)$. The path takes the form of

$$\theta(s(t)) = T_b^i \exp(\log(T_d^i)s(t)) \quad (3.15)$$

where T_b^i is the $SE(3)$ inverse of T_i^b . In our setup, $s(t)$ takes the form of a third degree polynomial

$$s(t) = a_0 + a_1t + a_2t^2 + a_3t^3. \quad (3.16)$$

Equation (3.16) is constrained by $s(0) = \dot{s}(0) = 0$, $s(t_f) = 1$ and $\dot{s}(t_f) = 0$, where t_f is the end time of the path. Under such constraints, a_i in equation (3.16) are given by

$$a_0 = 0, \quad a_1 = 0, \quad a_2 = \frac{3}{T^2}, \quad a_3 = -\frac{2}{T^3} \quad (3.17)$$

Figure 3.8 illustrates how the time scaling factor s varies with time.

3.2.2 Sample Points Based Tool Path Generation

This subsection calculates the pose of the robot-end-effector while it is traveling between two sampled points. The sampled points are taken from the set of scanned data and could be used to shorten the finishing cycle time and computational complexity. Given the initial and final poses, the local normal on the path connecting the two poses is approximated via the following method. In Figure 3.9, we assume that the two sampled points A and C and their local normal $\vec{N}_i$ and $\vec{N}_f$ are given. The starting point is A and the end point is C . Assume that the point B is any point on the tool path trajectory connecting A and C . The translation could be approximated with a straight line as long as its orientation is correct and the robot is performing force control. Denote the local

Figure 3.8: Time scaling factor VS time

normal at point B is $\vec{N}_m$, it can be approximated with an axis-angle relationship. Denote $\hat{w}$ as the rotation axis connecting $\vec{N}_i$ and $\vec{N}_f$ as

$$\hat{w} = \vec{N}_i \times \vec{N}_f \quad (3.18)$$

The rotation angle θ' that would transform $\vec{N}_i$ to $\vec{N}_m$, and $\hat{w}$ is approximated as

$$\theta' = \frac{|\vec{AB}|}{|\vec{AC}|} \quad (3.19)$$

Thus, the vector $\vec{N}_m$ is a result of the rotation of $\vec{N}_i$ with respect to axis $\hat{w}$ about angle θ' . Such a rotation matrix can be easily calculated using the ROS TF library.

The density of the projected points used for tool path generation depends on the sander diameter and local curvature. In our setup, the diameter of the sander is 127 mm. The number of sampled points changes along the AC direction according to the curvature of the part. Overall, 46 points were used which significantly reduces the cycle time in scanning the surface profile and subsequently reduces the finishing cycle time since it usually takes long for the robot to follow a

Figure 3.9: Tool path trajectory

trajectory composed with more goal points.

4. EXPERIMENTAL DESIGN AND RESULTS

This chapter ¹ discusses a series of experiments conducted in order to evaluate the proposed simultaneous force/position control, surface identification, and local normal calculation toward accomplishing surface finishing tasks. Surface finishing experiments were performed on both flat and free-form surfaces. The results of each experiment are presented and discussed.

4.1 System Description

A picture of the robotic surface finishing system is shown in Figure 4.1. The system consists of a 6-axis universal robot (UR5), a force/torque sensor (FT300), a proximity laser sensor (Sick DT20 Hi) and an electromechanical sander. A custom fixture (component shown in purple color) attached to the force/torque sensor is designed to hold the sensor and the sander. A more detailed description of the key hardware and software components are given in the following sections.

4.1.1 Hardware Setup

UR5 is a six joint industrial robot from Universal Robots. It is a lightweight robot (18.4 kg) with a payload capability of 5 kg and a working radius of 850 mm. Each joint is driven by a brushless servo motor with 100 : 1 zero-backlash harmonic drive reducer. The UR5 is a collaborative robot that has built-in safety mechanisms to stop the robot motion immediately if safety-related limits are exceeded. The maximum limiting safety sets are summarized in Table 4.1, where TCP is the acronym for Tool Center Point.

The UR5 is equipped with an internal controller box of type CB3 that exposes three interfaces of which a client can connect to port 30001, 30002, or 30003. Port 30001 and 30002 are robot state interfaces that provide information on the robot state such as joint position, velocity and temperature, etc. Port 30001 publishes the robot state, version information, and various messages to the GUI. But port 30002 only publishes robot state and version information. Port 30003 is the real-time interface that publishes a limited set of robot state. The controller is connected to a client

¹Part of the chapter is reprinted with permission from Yalun Wen and Prabhakar Pagilla, "Robotic Surface Finishing of Curved Surfaces: Real-Time Identification of Surface Profile and Control," presented at Manufacturing Science and Engineering Conference, June 18 - 22, 2018, College Station TX, ASME.

Figure 4.1: UR5 polishing system

(local PC) via socket connection.

Three different ways can be used to program the universal robot: PolyScope, URScript or C programming. The PolyScope is a graphical user interface that communicates with the controller at a frequency of 10 Hz, and it is mainly used for offline programming. The Universal Robot also exposes a C library that allows the user to communicate and send commands to the robot. However, this library is only available to the controller of type CB2. The URScript is a python-like language that allows the user to connect a PC to UR controller via socket connection. The URScript is sent in string format to the controller and receives data from the robot. The URScript exposes a variety of position or velocity control commands either in the Cartesian space or the joint space.

In the URScript language, there are four different ways to command robot motions: move, servo, speed and force. *Move* commands the robot end-effector to move from one Cartesian coor-

Table 4.1: UR5 safety limiting sets

Limiting Safety Function	Maximum Limit
TCP Force	250N
TCP Speed	5000mm/s
Joint Speed	30°/s
Momentum	100 kg m/s
Power	80w

dinate to another, i.e., point-to-point motion. The movement can either be linear in joint/Cartesian space, or circular in Cartesian space. The CB3 controller has a command buffer so several move commands can be executed in succession. However, a successive trajectory implemented using move command requires a blending parameter. The blending parameter blends two trajectories together but the blending point between the two consecutive trajectories may not be reached, which makes the move command unsuitable for our application. The *servo* command moves the joint to the desired position and then stops. Thus, a look ahead time and gain parameter is required for the generation of a smooth trajectory. The *speed* command moves the robot either in the joint space or Cartesian space with a constant velocity. The velocity profile takes the form of a trapezoidal shape. In other words, the robot accelerates until the desired velocity is reached and then moves at that velocity until the joint limit is reached or a new command is issued. Also, it is possible to command the robot to move compliantly in one direction while moving along the other directions, which is essentially a hybrid position/force control. However, the internal force measurements capability of the Universal Robot has low resolution, ($\pm 10N$).

The following content presents the robot responses under different commands given a precomputed trajectory using the threaded solution (ur modern driver) [57]. The reaction time of a servo-based program is around 124 ms. The actuation delay of the speed based command is around 8 ms or 16 ms. After execution of a long trajectory for 750 seconds, the speed based command gives an error of 0.18° . An experiment is performed on the last joint of UR5 to test the performance of the *speedj* command using ROS control. ROS control is introduced by the ROS community.

ROS control controllers are not meant to run as executables but rather as plugins to be loaded by the ur modern driver through the ROS ControllerManager interface. A shared library from the source code of the ROS community is available on line that allows the user to implement a custom controller. It also needs a special xml file to register the plugin to the ROS pluginlib system. The way to run ROS control is quite different from executing an executable: the controller must be called inside the ur modern driver. The controller configuration needs to be loaded to the controller manager and parameter server. Figure 4.2 shows the performance of the *speedj* command when the joint is commanded to follow a sinusoidal function. Note that the *speedj* command provides good real-time tracking performance but the joint performance may deteriorate during changes in the velocity direction.

Figure 4.2: Speedj performance on the last joint of UR5

The force-torque sensor mounted on UR5 is a FT300 from Robotiq. The force measuring range is ± 300 N and the moment measuring range is ± 30 Nm. The data output rate is 100 Hz. The standard deviation of the signal noise measured in one second for the force measurement is 0.1 N, for moment measurement it is 0.005 Nm. FT300 is connected to the computer via RS485

converter. The computer is running Ubuntu 14.04 with ROS indigo. A simple low-pass filter is designed for the FTS sensor. Discrete Fourier transform identifies main component frequency of the noise as shown in figure 4.3. and the real time performance of the low-pass filter is shown

Figure 4.3: FT300 noise frequency analysis via FFT

in Figure 4.4. A Complementary Filter (a special type of filter algorithm) is used to compare the performance of the low-pass filter and the results are shown in Figure 4.5. A fixture is designed and 3D printed for mounting the sander and laser sensor on the force torque sensor. A Dewalt random orbital sander that operates at 12,000 RPM without payload is chosen; this type of sander is used in sanding of helicopter blades. The polishing pad is of diameter 5". Random orbit sanders are also

Figure 4.4: Low pass filter

Figure 4.5: Complementary filter

commonly used as hand-held tools for sanding in which the motor delivers a random-orbit action. It is also called a dual-action sander since the rotating axis and the angle of rotation between the head and disk can be varied. It is an upgrade to the orbital sander with higher material removal rate and less visible cross-grain scratching. The sanding pattern is produced by simultaneously spinning the sanding disk and moving it in small ellipses. This ensures that no single part of the abrasive material travels the same path twice during the same rotation. As a result, the sander does not leave swirl marks. The polishing traces left on the surface after the application of a random orbital sander are shown in Figure 4.6. A Sick DT20 Hi laser sensor is used to measure the

Figure 4.6: Polishing trace of random orbital sander

distance between the mount and the machined part. The output current of the laser sensor linearly proportional to the measured distance within the range of the sensor. The current is converted to voltage to be read by an Arduino board. The Arduino board is connected via USB to local PC and ROS reads the information from Arduino via rosserial protocol. The Sick DT20 Hi laser sensor provides reliable and accurate distance measurements from 50 mm to 1000 mm. Arduino board's analog input pin can measure 0 to 5V with a 10 bit resolution.

4.1.2 Software Setup

This subsection describes the communication mechanism behind ROS. ROS is a collaborative, open source framework for robotic applications. Rosserial protocol allows the user to publish/subscribe messages from Arduino. A ROS driver exists for UR5 [57] and robotiq FT300. Similar ROS drivers are also available for many other industrial robots. In our setup, a full robot manipulation library including robot kinematics, control and surface identification was developed using the Eigen (C++), Numpy (Python) and TF libraries. One exception is that the surface interpolation was processed in MATLAB. All computations are done on a host computer. The inner joint velocity loop is embedded in the UR controller. At the scanning stage, a program in the host PC calculates the transformation from laser sensor measurement and robot forward kinematics; this transformation is used to construct the interpolated surface profile. Based on the surface profile, a tool path trajectory is generated.

The ROS driver for UR5 is configured at the start-up of the UR controller using ROS's parameter server. First, it looks for the robot's IP address which is given by the user. If connected, it then reads the parameter reverse port to determine which port to connect, the default port is port 5007. Then the internal controller reads and sends commands to the joint servo at 125 Hz, every command is evaluated at 8 ms. Also, the controller provides information to ports 30001 and 30002 which are accessed by the ROS driver to get the robot state (joint position, velocity, temperature, etc) through TCP/IP server sockets. The ROS meta-package for FT300 is open-sourced on Github. This driver provides real-time information about the current force reading. ROS reads this information in real-time and synchronizes it with ROS timestamp.

4.2 Experimental Results

A number of experiments were conducted with the proposed approach for flat and curved surfaces. The flat and curved surfaces considered are shown in Figure 4.7. For the two surfaces, experiments were conducted with the sander on and off to determine the tangential forces resulting from sander. For all the experiments, a desired normal contact force of 60 N was used, i.e.,

(a) Flat surface

(b) Curved surface

Figure 4.7: Flat and curved surfaces

$F_{set} = 60$ N. The goal was to evaluate whether the proposed approach can regulate the normal force for both flat and curved surfaces while traveling on the surface, both with sander on and sander off. The motion trajectory on the flat surface is along the points A, B, C, D , in that order. For the curved surface, the motion trajectory is along the points G, F, E .

4.2.1 Force Regulation

Figure 4.8 is an experiment conducted to evaluate normal force regulation without moving on the surface; data for both sander on and sander off conditions are shown. It is evident that the force control approach is able to regulate the force to the set point.

Evolution of the force signals for the flat surface is shown in Figure 4.9. The contact force normal to the surface reaches the desired normal force set point. It is clear that when the sander is on the signal is more noisy because of the action of the bristles on the sanding pad. Also, at the transition points when the trajectory changes direction, one can see transients in the normal force signal, especially when the sander is off. The tangential forces (x and y directions) are noisy but are constant when the sander is on. When the sander is off, the tangential forces undergo a step

Figure 4.8: Normal force regulation

change when the direction is changed, i.e., at points B and C . This may be due to the bristles on the sanding pad changing their orientation resulting in a different tangential friction coefficient.

Figure 4.10 shows the evolution of the various force signals for the curved surface. The normal contact force level reaches to within 10 percent of the set point for both sander off and sander on conditions. This indicates that the surface profile identification method as well as the simultaneous force/position control approach are both providing good results in terms of normal force regulation for curved surfaces. The tangential forces do not undergo any step changes as there is no change in direction for the motion trajectory on the curved surface.

(a) Normal forces

(b) Tangential forces

Figure 4.9: Force plot of force/position control on flat surface

(a) Normal forces

(b) Tangential forces

Figure 4.10: Force plot of force/position control on curved surface

Figure 4.11: Surface finishing on flat wood surface

4.2.2 Surface Finishing

Figure 4.11 (B) displays the finished flat wood part, and the filtered contact force in the z -direction of the end-effector is shown in Figure 4.9 (C). Figure (D) indicates that the tangential velocity of the end-effector vibrates around 0.02 m/s. Finishing a flat surface of length 0.209 m takes approximately 10 seconds. Also, the tangential velocity signal shows transients before the contact force reaches the set point (from 0 to 2 seconds) which might result from the friction force since the tangential friction is directly related to the normal contact force.

Figure 4.12 provides a comparison of the finished curved wood surface with the unfinished

surface prior to sanding. The unfinished surface displays various manufacturing defects including irregular surface roughness, cutting tool marks and cutting defects. The finished work piece takes advantage of seven local surface normal and undergoes two cycles of polishing processes using the polishing pads with 100 and 400 grains, respectively. This figure indicates the proposed method can successfully remove those surface irregularities on a free-form surface even without any prior geometric knowledge of the work piece.

Figure 4.12: Surface finishing on curved wood

5. SUMMARY AND FUTURE WORK

This thesis developed key tools, such as surface identification, tool trajectory generation, and simultaneous motion/force control algorithm, for designing a robotic surface finishing system of curved surfaces. The relevant academic literature was presented, highlighting some of the historical challenges in automating manual surface finishing processes and some of the commercially available solutions. A control strategy that can be used for surface finishing with the robot input to be the joint velocities was proposed; the strategy consists of pose tracking, force regulation and simultaneous force/position control.

A novel non-contact approach for surface registration and local normal calculation was developed and compared with the other perceptual techniques. The registered surface is represented by a grid composed of projection points acquired from a proximity laser sensor mounted on the robot end-effector. The accuracy of the reconstructed surface normal profile was verified via experiments. The acquitted surface profile was then used to modify and revise the trajectory generated using the time-scaling method. Instead of using every local normal to form a trajectory, the trajectory was generated based on the local normal of a few sampled projection points. The local normal was estimated from two consecutive sampled projection points to form the trajectory at the current time step while the end-effector is traveling between them. Therefore, a continuous stream of position goals is generated that would orient the robot end-effector to the local normal of the surface. Experiments were performed to verify the validity of the generated trajectory on free-form surfaces. Finally, an integrated system was implemented on a Universal Robot (UR5), and a number of experiments were conducted to verify the developed tools and the overall strategy for surface finishing using robots.

The following topics are suggested as future work:

- Feedrate: currently the tangential velocity of the end-effector is not directly regulated. Efforts can be made to analyze the effect of tangential velocity on the regulated force and

position control accuracy.

- **Material removal rate (MRR):** MRR is perhaps the most important design criteria for surface finishing tasks. However, the control of MRR is somewhat conflicted with the design of force control. In simultaneous force/position control, the robot is not capable of controlling the velocity in the direction of the force controlled axis. However, the control of MRR requires the robot to have at least some motion control capability over the force controlled axis. Therefore, the design should take into account the control MRR within a desired range. Extensive experiments could be performed to provide the relationship between MRR, normal force and work piece surface geometry and texture.
- **Even surface finish:** Obtaining even surface finish for free form surfaces (having both convex and concave profiles) with surface defects is challenging because of many issues, such as diameter of the tool, curvature of the surface, etc. Future research could focus on the algorithms for a smoother trajectory utilizing the scanned surface model (local curvature) and the desired surface geometry.
- **Surface finish quality:** We did not objectively evaluate the quality of the surface finish in our work. In line and real-time surface quality measurements could be useful for monitoring as well as feedback control. Future work could look at these aspects into the overall system.
- **Human machine interaction:** the automated robotic surface finishing system could use a thorough understanding of the manual operation. Efforts could be made to capture the major characteristics (variables) of human force/motion, decision making and visual inspection process. Ideas from these human trials can be then used to improve the automatic surface finishing system.

REFERENCES

- [1] L. Liao and F. Xi, "A linearized model for control of automated polishing process," in *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pp. 986–991, IEEE, 2005.
- [2] U. Cho, D. Eom, D. Lee, and J. Park, "A flexible polishing robot system for die and mould," in *23rd international symposium on industrial robots*, pp. 449–456, 1992.
- [3] K. Lynch and F. Park, *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.
- [4] A. R. A. Besari, A. S. Prabuwno, R. Zamri, and M. D. M. Palil, "Computer vision approach for robotic polishing application using artificial neural networks," in *Research and Development (SCORED), 2010 IEEE Student Conference on*, pp. 281–286, IEEE, 2010.
- [5] H. Khakpour, L. Birglenl, A. Tahan, and F. Paquet, "Experimental study on abrasive waterjet polishing of hydraulic turbine blades," in *IOP Conference Series: Earth and Environmental Science*, vol. 22, p. 052016, IOP Publishing, 2014.
- [6] E. B. Kalt, *Development of an integrated robotic polishing system*. PhD thesis, Loughborough University, 2016.
- [7] L. Liao, F. J. Xi, and K. Liu, "Modeling and control of automated polishing/deburring process using a dual-purpose compliant toolhead," *International Journal of Machine Tools and Manufacture*, vol. 48, no. 12-13, pp. 1454–1463, 2008.
- [8] J. Ahn, M. Lee, H. Jeong, S. Kim, and K. Cho, "Intelligently automated polishing for high quality surface formation of sculptured die," *Journal of Materials Processing Technology*, vol. 130, pp. 339–344, 2002.
- [9] A. Roswell, "Modeling for contact stress control in automated polishing," *Master of Applied Science Thesis, Ryerson University*, 2004.

- [10] H. Huang, Z. Gong, X. Chen, and L. Zhou, "Robotic grinding and polishing for turbine-vane overhaul," *Journal of materials processing technology*, vol. 127, no. 2, pp. 140–145, 2002.
- [11] M.-J. Tsai, J. Huang, and W. Kao, "Robotic polishing of precision molds with uniform material removal control," *International Journal of Machine Tools and Manufacture*, vol. 49, no. 11, pp. 885–895, 2009.
- [12] D. Axinte, J. Kwong, and M. Kong, "Workpiece surface integrity of ti-6-4 heat-resistant alloy when employing different polishing methods," *Journal of materials processing technology*, vol. 209, no. 4, pp. 1843–1852, 2009.
- [13] A. Guiot, S. Pattofatto, C. Tournier, and L. Mathieu, "Modeling of a polishing tool to simulate material removal," in *Advanced Materials Research*, vol. 223, pp. 754–763, Trans Tech Publ, 2011.
- [14] L. Feng-yun and L. Tian-sheng, "Development of a robot system for complex surfaces polishing based on cl data," *The International Journal of Advanced Manufacturing Technology*, vol. 26, no. 9-10, pp. 1132–1137, 2005.
- [15] H.-y. Tam, O. C.-h. Lui, and A. C. Mok, "Robotic polishing of free-form surfaces using scanning paths," *Journal of Materials Processing Technology*, vol. 95, no. 1-3, pp. 191–200, 1999.
- [16] A. F. Kenton, "Exploring options and alternatives for material removal and surface finishing," *Metal finishing*, vol. 106, no. 5, pp. 26–29, 2008.
- [17] Y. Song, W. Liang, and Y. Yang, "A method for grinding removal control of a robot belt grinding system," *Journal of Intelligent Manufacturing*, vol. 23, no. 5, pp. 1903–1913, 2012.
- [18] A. A. Akbari and S. Higuchi, "Autonomous tool adjustment in robotic grinding," *Journal of materials processing technology*, vol. 127, no. 2, pp. 274–279, 2002.
- [19] X.-J. Liu and J. Wang, "Some new parallel mechanisms containing the planar four-bar parallelogram," *The International Journal of Robotics Research*, vol. 22, no. 9, pp. 717–732, 2003.

- [20] H. Khakpour, *Design and Optimization of a Robot for Abrasive Waterjet Polishing of Hydraulic Turbine Blades*. PhD thesis, École Polytechnique de Montréal, 2014.
- [21] J. Angeles and J. Angeles, *Fundamentals of robotic mechanical systems*, vol. 2. Springer, 2002.
- [22] I. A. Bonev, J. Ryu, S.-G. Kim, and S.-K. Lee, “A closed-form solution to the direct kinematics of nearly general parallel manipulators with optimally located three linear extra sensors,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp. 148–156, 2001.
- [23] M. H. Abedinnasab and G. Vossoughi, “Analysis of a 6-dof redundantly actuated 4-legged parallel mechanism,” *Nonlinear Dynamics*, vol. 58, no. 4, p. 611, 2009.
- [24] L. Huo and L. Baron, “Kinematic inversion of functionally-redundant serial manipulators: application to arc-welding,” *Transactions of the Canadian Society for Mechanical Engineering*, vol. 29, no. 4, pp. 679–690, 2005.
- [25] F. Nagata, T. Hase, Z. Haga, M. Omoto, and K. Watanabe, “Cad/cam-based position/force controller for a mold polishing robot,” *Mechatronics*, vol. 17, no. 4-5, pp. 207–216, 2007.
- [26] F. Nagata, Y. Kusumoto, Y. Fujimoto, and K. Watanabe, “Robotic sanding system for new designed furniture with free-formed surface,” *Robotics and Computer-Integrated Manufacturing*, vol. 23, no. 4, pp. 371–379, 2007.
- [27] M. Rososhansky and F. J. Xi, “Coverage based tool-path planning for automated polishing using contact mechanics theory,” *Journal of Manufacturing Systems*, vol. 30, no. 3, pp. 144–153, 2011.
- [28] Y. Mizugaki, M. Sakamoto, and T. Sata, “Fractal path generation for a metal-mold polishing robot system and its evaluation by the operability,” *CIRP annals*, vol. 41, no. 1, pp. 531–534, 1992.
- [29] X. Armangué and J. Salvi, “Overall view regarding fundamental matrix estimation,” *Image and vision computing*, vol. 21, no. 2, pp. 205–220, 2003.

- [30] J. Batlle, E. Mouaddib, and J. Salvi, "Recent progress in coded structured light as a technique to solve the correspondence problem: a survey," *Pattern recognition*, vol. 31, no. 7, pp. 963–982, 1998.
- [31] A. K. Bejczy, "Robot arm dynamics and control," 1974.
- [32] B. Markiewicz, "Analysis of the computed torque drive method and comparison with conventional position servo for a computer-controlled manipulator," 1973.
- [33] R. Paul, "Modelling, trajectory calculation and servoing of a computer controlled arm," tech. rep., STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, 1972.
- [34] M. Raibert, "Manipulator control using the configuration space method," *Industrial Robot: An International Journal*, vol. 5, no. 2, pp. 69–73, 1978.
- [35] R. Kelly, "Pd control with desired gravity compensation of robotic manipulators: a review," *The International Journal of Robotics Research*, vol. 16, no. 5, pp. 660–672, 1997.
- [36] B. Yu and P. R. Pagilla, "Design and implementation of a robust switching control scheme for a class of constrained robot tasks," *International Journal of Systems Science*, vol. 37, no. 5, pp. 303–321, 2006.
- [37] P. R. Pagilla and B. Yu, "Robotic surface finishing processes: modeling, control, and experiments," *Journal of Dynamic Systems, Measurement, and Control*, vol. 123, no. 1, pp. 93–102, 2001.
- [38] P. R. Pagilla and B. Yu, "A stable transition controller for constrained robots," *IEEE/ASME transactions on mechatronics*, vol. 6, no. 1, pp. 65–74, 2001.
- [39] P. R. Pagilla and M. Tomizuka, "An adaptive output feedback controller for robot arms: stability and experiments," *Automatica*, vol. 37, no. 7, pp. 983–995, 2001.
- [40] N. Hogan, "Impedance control: An approach to manipulation," in *American Control Conference, 1984*, pp. 304–313, IEEE, 1984.

- [41] S. Howard, M. Zefran, and V. Kumar, "On the 6×6 cartesian stiffness matrix for three-dimensional motions," *Mechanism and machine theory*, vol. 33, no. 4, pp. 389–408, 1998.
- [42] J. Loncaric, "Normal forms of stiffness and compliance matrices," *IEEE Journal on Robotics and Automation*, vol. 3, no. 6, pp. 567–572, 1987.
- [43] B. Siciliano and L. Villani, *Robot force control*, vol. 540. Springer Science & Business Media, 2012.
- [44] D. E. Whitney, "Force feedback control of manipulator fine motions," *Journal of Dynamic Systems, Measurement, and Control*, vol. 99, no. 2, pp. 91–97, 1977.
- [45] J. Maples and J. Becker, "Experiments in force control of robotic manipulators," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, vol. 3, pp. 695–702, IEEE, 1986.
- [46] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, no. 127, pp. 126–133, 1981.
- [47] M. T. Mason, "Compliance and force control for computer controlled manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 6, pp. 418–432, 1981.
- [48] K. Mathiassen, J. E. Fjellin, K. Glette, P. K. Hol, and O. J. Elle, "An ultrasound robotic system using the commercial robot ur5," *Frontiers in Robotics and AI*, vol. 3, p. 1, 2016.
- [49] P. Rocco, G. Ferretti, and G. Magnani, "Implicit force control for industrial robots in contact with stiff surfaces," *Automatica*, vol. 33, no. 11, pp. 2041–2047, 1997.
- [50] H. Du, Y. Sun, D. Feng, and J. Xu, "Automatic robotic polishing on titanium alloy parts with compliant force/position control," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 229, no. 7, pp. 1180–1192, 2015.
- [51] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 1994.

- [52] K. P. Hawkins, “Analytic inverse kinematics for the universal robots ur-5/ur-10 arms,” tech. rep., Georgia Institute of Technology, 2013.
- [53] M. W. Spong, S. Hutchinson, M. Vidyasagar, *et al.*, *Robot modeling and control*, vol. 3. Wiley New York, 2006.
- [54] A. Balestrino, G. De Maria, and L. Sciavicco, “Robust control of robotic manipulators,” *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 2435–2440, 1984.
- [55] W. A. Wolovich and H. Elliott, “A computational technique for inverse kinematics,” in *Decision and Control, 1984. The 23rd IEEE Conference on*, vol. 23, pp. 1359–1363, IEEE, 1984.
- [56] C. Wampler and L. Leifer, “Applications of damped least-squares methods to resolved-rate and resolved-acceleration control of manipulators,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 110, no. 1, pp. 31–38, 1988.
- [57] T. T. Andersen, “Optimizing the universal robots ros driver,” tech. rep., Technical University of Denmark, Department of Electrical Engineering, 2015.
- [58] V. Duchaine and C. M. Gosselin, “General model of human-robot cooperation using a novel velocity based variable impedance control,” in *EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint*, pp. 446–451, IEEE, 2007.
- [59] A. Zelenak, C. Peterson, J. Thompson, and M. Pryor, “The advantages of velocity control for reactive robot motion,” in *ASME 2015 Dynamic Systems and Control Conference*, pp. V003T43A003–V003T43A003, American Society of Mechanical Engineers, 2015.