Smooth Trajectory Generation and Control for Precision Motion of Industrial Mechatronic Systems

（産業メカトロニクスシステムの精密動作のための滑らかな軌道生成と制御）

July, 2017

Doctor of Engineering

Simba Kenneth Renny

Abstract

A remarkable technological advancement in mechatronics, the synergistic application of electrical, electronic, computer and control engineering, which has evolved over the past three decades, has led to a novel stage of life. In response to the rapid growth of technology and the demand for precise products, the industrial community continues to require higher-accuracy and higher-speed manufacturing systems. The precision of mechatronic systems depends mostly on the ability to overcome nonlinear uncertainties, which are common and unavoidable. They result either from disturbance signals or system modelling errors. Normally, when a system is approximated by a mathematical model, non-fundamental factors such as systemic high-frequency dynamics and mechanical vibrations are ignored.

The primary reasons for the existence of mechanical vibrations in mechatronic systems are the highly-dynamic motion trajectories in the drive systems and elasticities of mechanical systems due to lightweight elements, such as gears and lead screws. Highly-dynamic motion trajectories contain a wide range of frequencies that can excite resonance frequencies of a mechatronic system. In machining complex parts or traversing complex paths, reference trajectories may include high curvatures that cause rapid changes in acceleration profiles. The motion must stop, change direction and restart at every corner to avoid this. Such a motion profile causes discontinuity, consumes time and power, introduces delay and leads to unnecessary wear in mechatronic systems. Reference trajectories should describe paths accurately, be kinematically smooth and satisfy physical limitations of mechatronic systems to guarantee smooth motion profiles. Moreover, trajectories should observe important criteria depending on specific applications.

Although many trajectory-generation approaches have been discussed in the literature, several problems persist. For example, many studies involving mobile robots have considered only the fundamental criteria for generating reference trajectories, such as travelling distance and expected arrival time. Other important criteria, for instance, local controllability, such that any changes in the trajectory affect only a limited region and the ability to arbitrarily set the first and second derivatives of positions at the starting and ending points of a path or path segment are usually ignored. These two criteria allow smooth update of the trajectory and are important for obstacle-avoidance motion planning through which the trajectory must be re-planned whenever an obstacle is encountered. In addition, in the case of numerical control systems, many studies have focused on smoothing linear interpolated tool-path points by using a parametric spline curve-fitting

technique. However, the curve-fitting technique leads to oscillations in trajectories dense in tool-path points.

In addition to smooth motion profiles, mechatronic systems require precise motion controllers that achieve high-tracking bandwidth with disturbance rejection. High-tracking bandwidth increases flexibility when tracking different trajectory profiles. Although traditional feedback controllers with high gains can achieve disturbance rejections, high gains may destabilise control systems and may impose several limitations owing to hardware properties. Because most industrial mechatronic systems perform repetitive operations over a fixed time interval, Iterative Learning Control (ILC) can be used as an effective tool for improving transient response and tracking performance. Although ILC has been widely applied to mechatronic systems, particularly feed drive systems, many studies have considered only tracking errors. However, tracking error-based controllers' exhibit poorer tracking capability for contours with high curvatures and show higher input variance than contouring controllers. Therefore, it is indispensable to further enhance system performance by considering contouring control.

Methods to generate smooth trajectories and ILC design to enhance the precision of industrial mechatronic systems are described in this thesis as follows: Introductory remarks are presented in chapter 1 followed by a review of related works and their shortcomings in chapter 2. Chapter 3 describes a method to generate smooth motion trajectories for autonomous mobile robots for both real-time and off-line applications. The method is based on piecewise quintic Bézier curves, where the Bézier subdivision technique is adopted to improve curvatures at sharp corners. The generated trajectories are controllable locally and can arbitrarily set the first and the second derivatives at the starting and the ending points. A method to generate vision-based smooth obstacle-avoidance trajectories for mobile robots is presented in chapter 4. A smooth and distance-optimal trajectory is generated in real time from an environmental top-view image, where a fisheye lens is used to capture a wide area from a low height. Chapter 5 describes a method to generate smooth motion trajectories for feed drive systems for a specified error tolerance in a reference contour. The generated trajectory considers fundamental criteria, for example, velocity, acceleration and jerk limits. This trajectory can be tracked easily by a feed drive system and renders a lower maximum contour error compared to conventional trajectories that are interpolated linearly. A novel contour error-based ILC for feed drive systems is presented in chapter 6. Experimental results verified that with the proposed controller, the maximum contour error of feed drive systems could be reduced by about 47.8 % on average compared to conventional controllers. Lastly, chapter 7 presents the concluding remarks of this thesis and prospective future works.

Acknowledgements

First and foremost, I would like to thank and praise the Almighty God for His grace and blessings. Without Him, this would not be possible.

I am grateful to my supervisor Prof. Naoki Uchiyama for accepting me into his research group and helping me pursue my PhD. I highly appreciate his support, guidance and encouragement throughout my studies.

I extend my thanks to the committee members Prof. Hideki Yanada, Prof. Takanori Miyoshi, Prof. Shigenori Sano and Prof. Tatsuhiko Sakaguchi for their great support, constructive ideas and suggestions in writing this thesis and for letting my defence be an enjoyable moment.

I am also grateful to my friends and colleagues for their contribution to the development of this thesis through valuable support and friendship.

I especially like to thank my family for their unconditional love and sacrifices that they always make on my behalf. Words can't explain how grateful I am to my parents as they have been always there for me. Their prayers and encouragement have sustained me so far. Special thanks to my children Ayumi and Isaac for being so good and patient even at hard times. Thanks to my sister Berther and brothers Mathew and Christian for being so helpful and courageous throughout my career.

Lastly, I would like to express my heartfelt gratitude to my beloved wife Bahati for loving and caring me. I can't thank her enough for all the sacrifices she made for me. Her love, patience and prayers are key reasons for my success.

Contents

List of Figures

List of Tables

To my belovely wife. . .

Chapter 1

Introduction

1.1 Motivation

The remarkable technological advancement in mechatronics, which is the synergistic application of electrical, electronic, computer and control engineering and has evolved over the past three decades, has led to a novel stage of life. By integrating advanced design methods and industrial mechatronic systems, manufacturing industries can realise high-quality products, while simultaneously guaranteeing a substantial reduction in the and cost of manufacturing. In response to the rapid growth of technology and demand for precise products, the industrial community requires higher-accuracy and higher-speed manufacturing systems. Therefore, industrial mechatronic systems, such as mobile robots and Computer Numerical Control (CNC) machine tools, must operate at high speeds while maintaining high positioning accuracy.

For high-speed performance, mechatronic systems need high-velocity/ high-feed-rate capabilities, fast controllers under time-optimal trajectories, and lightweight mechanical systems. On the contrary, to ensure high positioning accuracy, one requires stiff systems with accurate motion controllers under slow trajectories. Furthermore, trajectories are required to describe paths accurately, be smooth kinematically and satisfy the physical limitations of mechatronic systems. Contrariwise, lightweight mechanical operating at high speeds may suffer from significant vibration problems, thus degrading positioning accuracy and exhibiting large settling times.

The growing demand for solutions to achieve higher-speed operations while maintaining precision in mechatronic systems has attracted the attention of many researchers from the academic and the industrial communities and is the key driver of the contents of this thesis.

1.2 Problem Statement

The precision of mechatronic systems depends mostly on their ability to overcome nonlinear uncertainties which are common and unavoidable. They result from either disturbance signals or due to system modelling errors [1, 2]. When a system is approximated by a mathematical model, non-fundamental factors such as systemic high-frequency dynamics and mechanical vibrations are ignored.

The primary reasons for the existence of mechanical vibration in mechatronic systems are highly-dynamic motion trajectories in the drive systems and elasticities of mechanical systems due to lightweight elements, such as gears and lead screws [3, 4]. Highly-dynamic motion trajectories contain a wide range of frequencies that can excite resonance frequencies of mechatronic systems. In machining complex parts or traversing complex paths, reference trajectories may include steep curvatures that would cause rapid changes in acceleration profiles. The motion must stop, change direction and restart at every corner to avoid this. Such a motion profile causes discontinuity, consumes time and power, introduces delay and causes unnecessary wear on mechatronic systems [5].

Many researchers have drawn attention to enhancing the precision of mechatronic systems through different approaches. Common approaches include speed reduction at corners and modification of reference trajectories. Regarding trajectories, the fundamental idea is to generate at least a second-order trajectory for continuous acceleration. However, in applications such as mobile robots, the following additional criteria are necessary:

i Local controllability, such that any changes in trajectory affect only a limited region.

ii Ability to arbitrarily set the first and the second derivatives at the starting and ending points; together with (i), this allows for smooth update of the trajectory and is important for obstacle-avoidance motion planning, through which the trajectory must be re-planned whenever an obstacle is encountered.

Although many studies have proposed methods that devise up to first or second order differentiable trajectories, the important criteria mentioned in (i)-(ii) are usually not considered.

In addition to smooth motion profiles, mechatronic systems require precise motion controllers that achieve high-tracking bandwidth with disturbance rejection. High-tracking bandwidth increases flexibility when tracking different trajectory profiles. Although traditional feedback controllers with high gains can achieve disturbance rejections, high gains can destabilise control systems and have several limitations owing to the hardware properties. Because it is not always possible to achieve the desired tracking performance based on general control theory due to the presence of unmodelled dynamics and nonlinear uncertainties, intelligent controllers are required to enhance the tracking performance. Because most industrial mechatronic systems perform repetitive operations over a fixed time interval, Iterative Learning Control (ILC) can be used as an effective tool to improve transient response and tracking performance. ILC is among the intelligent controllers that imitate the human learning process and is proven to enhance the performance of uncertain dynamic systems. Although ILC has been widely applied to mechatronic systems, especially, feed drive systems, many studies have considered only tracking errors [6–8]. However, tracking error-based controllers' exhibit poorer tracking capability for contours with high curvatures and show higher input variance compared to contouring controllers [9]. Therefore, it is indispensable to further enhance system performance by considering contouring control.

Methods to generate smooth trajectories and ILC design to enhance the precision of industrial mechatronic systems are described in this thesis. The mechatronic systems under consideration are the typical differential-drive mobile robots and multi-axis feed drive systems because they are widely used in industrial applications.

1.3 Contribution

The main contributions of this thesis are as follows:

- A method to generate smooth motion trajectories for autonomous mobile robots for both real-time and off-line applications. The method is based on piecewise quintic Bézier curves, where the Bézier subdivision technique is adopted to enhance curvatures at sharp corners. The trajectory is proven to have desired properties that satisfy the kinematics of nonholonomic mobile robots. These include second-order derivatives of the trajectory for curvature continuity, localism, correlation, and distance optimality. This approach relieves a human operator from the tedious work of controlling the robot manually by using a joystick or operating multiple

robots in a clouded environment. The operator's chief task is simplified, whereby he/she only needs to assign a goal location and, if necessary, a desired safety distance between the robot and any obstacle. Hence, even an unskilled operator can control multiple robots at a time. The proposed method can be extended and applied to three-dimensional space navigation easily with slight modifications.

- A method to generate smooth motion trajectories for feed drive systems for a specified error tolerance in a reference contour. The generated trajectory considers fundamental criteria, namely, velocity, acceleration and jerk limits. This trajectory can be tracked easily by a feed drive system and renders a smaller maximum contour error compared to conventional linearly interpolated trajectories.

- A novel iterative learning contouring controller for feed drive systems. By incorporating the traditional proportional-integral-derivative feedback controller, friction compensator, disturbance observer and ILC algorithm, tracking performance can be improved dramatically.

1.4 Thesis Organisation

The rest of this thesis is organised as follows:

- Chapter 2 provides a review of the related works and their shortcomings. It describes different path planning and trajectory-generation techniques applied to mechatronic systems. Both classical and modern techniques are surveyed and their strengths and limitations are highlighted. This chapter also describes classical control algorithms and ILC for feed drive systems.

- Chapter 3 describes a method to generate smooth motion trajectories for autonomous mobile robots for both real-time and off-line applications. The method is based on piecewise quintic Bézier curves, where the Bézier subdivision technique is adopted to improve curvatures at sharp corners. This chapter is divided into six sections, where section 3.1 provides a general introduction. Section 3.2 provides a brief introduction of the kinematics of a mobile robot, defines a Bézier curve and describes its properties. The subdivision technique based on the de Casteljau algorithm is introduced as well. Section 3.3 proposes the trajectory-generation algorithm, including parameterisation and subdivision in high-curvature areas. Section 3.4 describes the

designed trajectory tracking controller. The simulation and experimental results are presented in section 3.5, followed by a summary in section 3.6. This chapter relates to publications J.4 and C.6.

- Chapter 4 describes an obstacle-avoidance trajectory generation method that provides a smooth trajectory in real time. The trajectory is generated from an environmental top-view image, where a fisheye lens is used to capture a wide area from a low height. The visibility path-planning algorithm is applied based on the detected top-surface corners of obstacles. A distance-optimal path is computed and replaced by a smooth trajectory generated based on piecewise quintic Bézier curves as described in chapter 2. This chapter has five sections, as follows: Section 4.1 contains introductory remarks. Section 4.2 addresses the steps involved in the extraction of corners by using a fisheye lens. A brief description of lens distortion and calibration is provided, and the log-polar transform used to correct the corner coordinates is discussed. In section 4.3, the configuration space based on corrected corners is generated, visibility graph is constructed and shortest path is found. The conversion of linear paths into a smooth trajectory is explained as well. Experimental results are presented in section 4.4, followed by a summary in section 4.5. This chapter relates to publications J.3 and C.5.

- Chapter 5 presents a method to generate smooth motion trajectories for feed drive systems for a specified error tolerance on a reference contour. It comprises six sections, where section 5.1 contains opening remarks, followed by a description of the proposed trajectory smoothing algorithm in section 5.2. A method to generate smooth velocity transitions is presented in section 5.3. Section 5.4 explains contouring controller design with friction compensator. The experimental results and closing remarks are presented in sections 5.5 and 5.6, respectively. This chapter relates to publication C.1.

- Chapter 6 describes a novel iterative learning contouring controller for feed drive systems. It is composed of five sections, where section 6.1 provides an introduction. Section 6.2 defines the contour error and explains the dynamics of biaxial feed drive systems. Section 6.3 provides the design of the proposed contouring controller, including a nonlinear friction model. Simulation and experimental results are given in section 6.4, followed by concluding remarks in section 6.5. This chapter relates to publications J.1, C.4 and C.2.

- Chapter 7 provides the concluding remarks of this thesis and an outline of future work.

Chapter 2

Literature Review

2.1 Motion Planning: Path and Trajectory

Motion planning can be categorised into path planning and trajectory planning/generation. The former tends to ignore dynamics and differential constraints and focuses on finding the feasible path from the start to the goal location and optimising it based on the selected criteria. By contrast, trajectory planning involves finding the control inputs yielding a trajectory that avoids obstacles, takes the system to the desired goal state and possibly, optimises a few objective functions [10].

For feed drive systems, particularly machine tools, motion planning or tool-path planning is typically performed off-line using computer-aided manufacturing systems. For mobile robots, motion planning is categorised as global (off-line) or local (online) motion planning. In global motion planning, the environment is static and is known in advance, and therefore, a complete trajectory from the start to the goal configurations can be generated before the robot starts moving [11]. On the contrary, local motion planning is applied if the environment is dynamic and unstructured. In this case, as the robot moves, sensors gather environmental information, and the control laws are consequently updated in real time. However, even for online motion planning, the robot initiates its motion based on off-line knowledge and switches to the online motion planning algorithm as it moves.

2.2 Path-Planning Techniques

Because path-planning methods for mobile robots are very different from those for feed drive systems, specifically, machine tools, a brief review of the methods used in both cases is provided in this section.

2.2.1 Path Planning for Mobile Robots

One of the most significant challenges in the case of autonomous robots is automatic motion planning. The prototypical task is to find a geometrical path for a mobile robot from one configuration to another while avoiding obstacles. Automatic motion planning is more complicated when it involves some constraints or specific optimisation criteria such as shortest path, minimum time, or minimum energy [11]. However, owing to the nature of mobile robot applications, it is necessary to plan an optimal, collision-free path while minimising travel distance, time, and energy.

Both off-line and online path-planning techniques can be further classified into classical and modern path-planning methods as in the following section [12].

2.2.2 Off-line Path-Planning Methods

Classical Path-Planning Methods

The fundamental approach to solving the path-planning problem is the configuration space, termed the C-space [13]. The C-space of a robot system is a complete specification of the position of every point in that system, that is, the space of all possible system configurations. Its dimension is equal to the degree of freedom of the robot, that is, the minimum number of parameters needed to specify the C-space. Based on the C-space as the fundamental concept, many classic path-planning techniques can be grouped into roadmap and cell decomposition [10].

***Roadmaps*:** The basic idea of roadmap methods is to create a roadmap that reflects the connectivity of the set of the configurations in which the robot does not intersect any obstacle. That is, to draw all line segments that connect a vertex of one obstacle to a vertex of another without entering the interior of any polygonal obstacle. If a continuous path can be found in the free space of the roadmap, the initial and the goal points are then connected to this path to arrive at the final solution (a free path). If more than one continuous path can be found, algorithms such as A* or Dijkstra's shortest path algorithm are often used to find the best path.

The most common types of roadmaps include the Visibility graph and the Voronoi diagram. In the visibility graph, the path of the robot is close to the obstacles, resulting in the shortest path from the start to the goal locations. The standard visibility graph is defined in a two-dimensional polygonal C-space where its nodes include the start and the goal positions, and the vertices of all polygonal obstacles. The nodes are joined by

straight lines if there is a line of sight between them and the task of a path planner is to find the shortest distance from the start to the goal positions [13], [10]. Visibility graphs are popular in robotics partly because they are simple to implement. However, this method is only efficient in sparse environments because the number of roads depends on the number of polygonal obstacles and their edges [14, 15]. However, by decomposing the configuration space, the visibility graph method is efficient for real-time path planning in large planar environments. Moreover, its implementation is simple, and the resulting path is always optimal regarding Euclidean distance and has certain advantages over other classical methods such as cell decomposition and the Voronoi diagram [16].

In contrast to the visibility graph method, the Voronoi diagram is an approach that tends to maximise the distance between the robot and the obstacles [14, 17]. It is constructed through the via points that are equidistant from the obstacles. Because this method maximises the distance from the robot to the obstacles, in sparse environments, the resulting path is always far from optimal; moreover, under limited range localisation sensors, the robot might fail to sense its surroundings [14, 18, 19].

***Cell Decomposition*:** The basic idea behind cell decomposition is to decompose the free C-space into smaller regions called cells and search for the connected route in the free space cell graph [20]. The cell decomposition method is normally classified as exact or approximate. The major difference in methodology is that, an exact decomposition of the free C-space is generated, whereas the approximate method yields a variable cell size. The cell decomposition method has been used for decades because it is simple to implement [21–25]. The difficult part of this method is obtaining a good path: if the number of cells is small, and hence, cells are large, one needs a strategy to find a path inside them: in contrast, if cells are small, a path can be constructed easily by passing through their centres. However, as the number of cells increases, the process becomes inefficient owing to exponential rises regarding memory requirements and search range [26].

Modern Path-Planning Methods

Classical path-planning approaches have several drawbacks that make them incompetent in complex environments, such as the tendency to get locked in an optimal local solution that may be far from the global optimal one and being Non-deterministic Polynomial time hard (NP-hard) under the presence of multiple obstacles [27, 28]. Several modern methods such as the Visibility-Voronoi diagram for clearance c (VV^c), Genetic Algorithm (GA),

Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO) and Simulated Annealing (SA) have been proposed in the literature to solve path-planning problems more effectively than the classical approaches.

VV^c is a hybrid between the visibility graph and the Voronoi diagram of polygons in the plane [29]. It evolves from the visibility graph to the Voronoi diagram as the parameter c grows from zero to infinity. This method can be used to for panning natural-looking paths for robots translating in a plane. Although the resulting path is short and smooth with clearance c, this method has a few drawbacks that are relatively similar to the drawbacks of the classical approaches, such as it is NP-hard in the presence of multiple obstacles.

GA is a randomised search method and optimisation tool inspired by the mechanics of natural genetics and selection [30, 31]. In path planning, the first step is to generate a population of candidate solutions called a generation, that is, a population containing alternative paths. In each generation, the fitness of every individual in the population is evaluated based on the optimisation criteria (fitness). The selected generation of candidate solutions is then used in the next iteration of the algorithm.The process continues until either the maximum number of generations has been produced or a satisfactory optimal condition has been reached.

Although GA has been used widely, it has several drawbacks. A fixed-length path with binary strings that yields a quick solution in a sparse environment was used in [27, 32–34]. However, with this approach, it takes several hours to evolve a solution in a complex environment. For solving path-planning problems in complex environments, a variable binary coded GA was presented in [35]. Its main drawback is that it may output wrong paths that do not reach the desired target location. Furthermore, in [36], an approach that initially considers all paths, even those that collide with obstacles and subjects them to a penalty function was proposed. The inclusion of invalid paths in the penalty function increases the computation time [37].

ACO, also called Artificial Ant Colony System, is an agent-based system that simulates the natural behaviour of ants and develops mechanisms of cooperation and learning [38]. This algorithm has been applied to many combinatorial optimisation problems, ranging from quadratic assignment to protein folding or routing vehicles, and many derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations [39–42]. In mobile robots, an improved method for path planning based on the Dijkstra algorithm and ACO was proposed

in [43], where the Dijkstra algorithm generates sub-optimal paths and the ACO algorithm is used to find the best solution. By contrast, SA is a probabilistic technique for approximating the global optimum of a given function and has been used for path planning too. For example, in [44], an SA-based approach was proposed to determine optimal or near-optimal paths quickly for a mobile robot in dynamic environments with static and dynamic obstacles. However, both ACO and SA are rarely used as optimisation tools in path planning owing to a few drawbacks, such as difficulties in theoretical analysis and longer time for uncertainty convergence [19, 45].

PSO is originally attributed to and was first intended for simulating social behaviour as a stylised representation of the movement of organisms in a bird flock or fish school, and it later came to be used as an optimisation method [46–49]. Compared to GA for example, PSO is easier to implement, and there are fewer parameters to adjust [46]. Moreover, it is among the widely used algorithms in path planning. Reference [50] is among the recent researches that use PSO for path planning for mobile robots in complex environments.

2.2.3 Online Path-Planning Methods

In manufacturing industries, the environment is usually highly controlled, such that mobile robots can be programmed to work based on predefined actions. On the contrary, autonomous mobile robots, which are used in applications ranging from space exploration to domestic applications, such as cleaning, operate in partially unknown or unpredictable environments. In addition, the environment might have dynamic characteristics that require continuous online modification of the robot behaviour. For this reason, since the last decade, many studies have explored novel methods for online path planning for autonomous robot navigation [51].

Artificial Potential Field (APF) and Collision Cone (CC) approaches are classical methods that have been used widely. Other methods include vector field histogram and dynamic window approaches. In modern practice, novel methods along with classical ones are usually applied to enhance navigation performance [12].

Classical Path-Planning Methods

The ATF method gained popularity in robotics since its initiation in [52]. In this approach, a robot navigates under the influence of imaginary forces called artificiality potential field in which obstacles and the target exert repulsive and attractive forces on the robot,

respectively. The resultant force determines the direction of the mobile robot by generating an obstacle avoidance path [53]. This method is simple to implement and the desired path can be found with little mathematical computation.

The major drawback of this approach is that the robot may be trapped in local minima owing to cancellation of the potential field under an equal magnitude of repulsive and attractive forces. Furthermore, when the robot is very far from the target, the attractive force becomes very large such that it can cause the robot to move too close to the obstacles. Because the target is always assumed to be far from obstacles, if the target is very close, it cannot be reached [54]. Several approaches have been proposed in the literature to overcome these problems, such as complementing influential algorithms and adaptive virtual target algorithm to escape from traps [55, 56], modification of attractive potential functions to avoid large potential fields when the robot is very far from the target, and modification of repulsive potential functions by considering the relative distance from the robot to the target to reach targets that are close to obstacles [57].

The CC approach is another widely used method for online mobile robot navigation, and it was proposed in [58]. A collision between a robot and an obstacle can be avoided if the relative velocity of the robot relative to an obstacle falls outside to the CC. This concept is sometimes called forbidden velocity map as proposed in [59]. Similarly, a velocity obstacle approach was proposed in [60], where the set of all velocities of a robot that will result in a collision with an obstacle at some moment in time is defined, assuming that the obstacle maintains its current velocity.

Modern Path-Planning Methods

Despite the effectiveness of the classic approaches, it is indispensable to enhance the navigation performance of mobile robots. In most cases, classical approaches such as PSO, ACO, and GA are often combined with modern methods. In [61], an Evolutionary Artificial Potential Field (EAPF) was proposed for real-time robot path planning, where the APF method was combined with GA to derive optimal potential field functions. The EAPF approach can help robots navigate in environments with moving obstacles. An algorithm called escape-force is applied to avoid local minima associated with EAPF.

An enhanced SA approach, which integrates two additional mathematical operators and initial path selection heuristics into the standard SA was proposed in [62] for robot path planning in dynamic environments with both static and dynamic obstacles. The enhanced SA can provide an optimal or near-optimal path solution in various dynamic

environments and requires considerably less processing time than the standard SA, SA with two additional operators and SA with heuristic initial path selection.

Another approach using the classic APF based on the consideration of a dynamic model of velocity potential field obtained by a variant of PSO was proposed in [63]. Furthermore, a combination of ACO and APF was proposed in [64] to obtain quicker solution convergence compared to the standard ACO approach.

2.2.4 Path Planning for Feed Drive System

Path planning for feed drive systems, especially, machine tools have been addressed in several studies with a focus on significant areas, such as selection of tool orientation and geometry and tool-path planning. Tool-path planning has been studied based mainly on traditional techniques, such as iso-parametric and iso-planar approaches [65–68]. Most of these planning methods are limited regarding accuracy and surface characteristics and are not optimal with respect to production time. Considering the iso-planar approach as an example, it treats the intersections between parallel planes and a surface as a tool path [66].

For the improving the traditional-based approaches, several methods have been proposed in the literature, in particular, conformal-parameterisation-based approaches for meshes [69] and the iso-scallop approach, which produces a constant scallop height on a machined surface [70, 71].

The advancement of five-axis machining and machining of nonparametric surfaces has resulted in the development of new approaches to resolve specific five-axis problems and to reduce machining time. As detailed in [72], these methods can be classified as curvature-matched machining [73–75], isophote-based methods [76–78], configuration space methods [79, 80], region-based tool-path generation [76], compound surface machining [81, 82] and methods for polyhedral models and cloud of points [83, 84].

2.3 Trajectory-Generation Approaches

The generation of smooth trajectories is crucial for motion control of mechatronics systems. It refers to the process of describing the time history of position, velocity, acceleration, and if necessary, jerk for each degree of freedom. The generated trajectories must describe the desired path accurately and obey the system's kinematic and dynamic

constrains to maintain high-tracking accuracy and avoid exciting the natural modes of the mechanical structure or the servo control system.

Standard geometric path-planning approaches yield a set of via points as described in [55]. Through traditional trajectory generation methods, via points are connected linearly, resulting in a piecewise linear path. Piecewise linear paths require a mechatronic system to stop and restart frequently, and therefore, it causes discontinuity, consumes time and power, introduces delay and leads to unnecessary wear of system parts [5].

In recent approaches, trajectory generation is usually performed using special functions, such as polynomial, trigonometric, exponential Fourier series expansion [85]. Using these approaches, many researchers have modelled motion trajectories as piecewise quadratic or cubic curves [86–91]. Although both quadratic and cubic curves satisfy the second derivative requirement, composite trajectories constructed using quadratic or cubic curves, for instance Bézier curves, are limited to the first derivative. Some studies have adopted higher-order curves, such as seventh-order Bézier curves in [92]. However, higher-order Bézier curves are complex to calculate and are numerically unstable, leading to oscillations in the resulting trajectory [93].

By contrast, parametric spline curves are used to interpolate the linear tool-path points or blending corners smoothly and have proven to provide sufficiently smooth trajectories for mechatronic systems [94, 95]. The main challenges associated with using parametric spline curves include, automatically generating a smooth trajectory using limited information and dealing with geometrical errors induced by the spline curves.

2.4 Control of Feed Drive Systems

Feed drive systems have a wide range of applications in the industrial community, including CNC machine tools and assembly robots. In machine tools, feed drives control the position and velocity of axes according to input commands to track a given reference trajectory. Tracking errors, the axial difference between the actual and the reference trajectories, occur in many industrial mechatronic systems. However, in machining applications, contour errors, the orthogonal differences between actual positions and reference trajectories or contours are the best indicators of machining precision because they affect the geometrical shape of the machined workpiece directly[96]. Both tracking-error and contour-error based controllers are applied to enhance the performance of feed drive systems. In the tracking error approach, each axis is controlled independently such that

the load disturbance or performance variance on either axis is compensated for in individual axial control loops. However, because motion trajectories are usually complex, where axes move synchronously with respect to one another to track the desired trajectory, performance deviation in either axis leads to contour error [97]. On the contrary, under contour-error based approach, the contour error is evaluated in real-time and compensated for through the corresponding control loops.

Different control approaches have been studied in the literature to enhance the performance of feed drive systems. In this section, a brief review of basic control algorithms and iterative learning control for feed drive systems is addressed.

2.4.1 Feedback Control

Feedback control refers to a controller that considers the output signal in the control loop to adjust the system performance to meet a desired output. In machine tools, all controllers have a feedback loop [98]. Proportional-Integral-Derivative (PID) feedback controllers are widely used in industrial applications owing to their simplicity in design and implementation, and good performs in most cases. Although PID control can be applied in many control problems, it has several limitations that leads to undesirable performance. Since gains are constant and there is no direct process knowledge to the controller, PID control may lead to overshoot. Moreover, PID control tracks corners and nonlinear contours poorly owing to sudden changes in the direction of motion.

2.4.2 Feedforward Control

Feedforward controllers are customarily added to the control loop to enhance the tracking performance. A practical feedforward controller for continuous path control of a CNC machine tool was proposed in [99] to reduce trajectory error parameters, specifically, radial reduction, edge unsharpness, asymmetric error, and vibration amplitude. A feedforward motion control design was developed in [100] for improving both the tracking and the contouring accuracies of motion control systems in CNC machine tools. By applying stable pole-zero cancellation to individual axes and by employing complementary zeros for all uncancelled zeros, the feedforward motion control design led to matched dynamics among all motion axes and thereby achieved highly accurate contouring and tracking

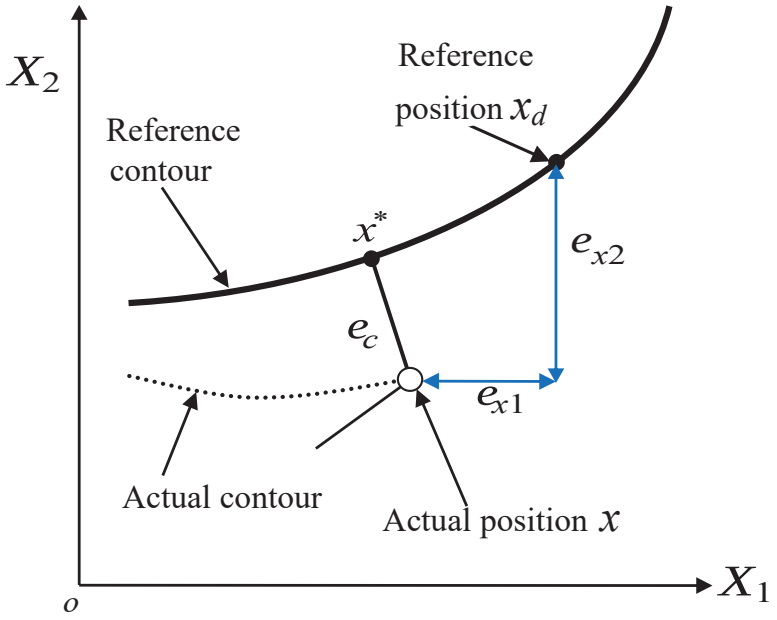


FIGURE 2.1: Definition of the contour error.

results. The main limitation of feedforward controllers is the requirement of exact knowledge of the model for controller design. In practice this knowledge is not known to the control designer, therefore the designed model may introduce position errors.

2.4.3 Cross-Coupling Control

Cross-coupling control considers a contour error based on feedback information from all axes and interpolates to find the best compensating law [98]. The altered signal is fed to the individual axes in real time. It was first introduced in [101] and extended to other approaches, especially contouring control [9, 96, 101–104]. The principle of this control algorithm is to directly reduce the contour error e_c rather than the axial errors e_{x1} and e_{x2}, that is, to position the tool at point $x^\star$ instead of x_d. Contouring control is an effective approach in machining because it provides performance comparable to that of non-contouring controllers with less input variance [9]. In addition, contouring control has better sharp-corners tracking and disturbance rejection capabilities than non-contouring controllers [98].

2.4.4 Iterative Learning Control

Iterative Learning Control is a particular form of feedforward control and an effective tool for improving the transient response and the tracking performance of uncertain dynamic systems performing repetitive operations over a fixed time interval [105]. Such systems include machine tools performing batch machining and robotic manipulators in manufacturing industries. Since it is not always possible to achieve the desired tracking performance based on general control theory owing to the presence of unmodelled dynamics and nonlinear uncertainties, ILC can be used to enhance the tracking performance of repetitive systems. In using ILC, tracking or contour errors from one iteration are used to compensate for the errors in the next iteration. Note that ILC is not independently applied because it is a feedforward controller and its application starts from the second iteration.

Although ILC has been applied widely to feed drive systems, many studies have considered only tracking error, and simulations or experiments were performed on straight, circular and non-circular trajectories [6, 7, 106, 107], and only a few studies considered cross-coupling control [8, 102]. However, as highlighted in the previous section, tracking-error-based controllers have poor sharp-corner tracking capability and higher input variance compared with contouring controllers. For this reason, it is indispensable to further enhance system performance by considering contouring control and sharp-corner trajectories.

Chapter 3

Real-Time Smooth Trajectory Generation for Mobile Robots

This chapter focuses on generating smooth trajectories for a wheeled mobile robot by using piecewise Bézier curves with properties ideally suited for this purpose. The developed algorithm generates smooth motion trajectories with C^2 continuous curvature. A teleoperated wheeled mobile robot in an indoor environment with ceiling cameras for operator visibility is considered. The motion trajectory is constrained by the operator-specified via points and path width. A method to automatically generate a trajectory based only on these two inputs is proposed and demonstrated. The Bézier subdivision method is adopted, and a quintic Bézier segment is inserted into high-curvature areas to improve the trackability of the mobile robot. The proposed algorithm can be used for real-time obstacle-avoidance trajectory generation because it allows for trajectory subdivision and arbitrary setting of the first and second derivatives at the starting and ending points. Simulation and experimental results demonstrate the effectiveness of the proposed method.

3.1 Introduction

Autonomous trajectory generation is crucial for both mobile robots and industrial machines such as cranes, CNC machines, and robot manipulators [108–110]. Autonomous mobile robots, especially, wheeled robots are potentially integrated into human environment and industries to assist or replace human workers, especially for tedious or hazardous works. Thus, intelligent wheeled robots have drawn the attention of many researchers regarding motion trajectory generation, obstacle avoidance, position control, and localisation and navigation [111–117]. Intelligent robots should be able to perform

self-localisation, mapping, trajectory planning and detection of obstacles in the workspace [118]. Real-time trajectory generation/planning is among the challenging tasks in mobile robotics; it is even more cumbersome when some objective functions and constraints such as nonholonomic constraints are involved. An ideal trajectory satisfies the following important criteria:

i Continuity, whereby the trajectory should be at least second-order differentiable at every point.

ii Local controllability, such that any changes in the trajectory affect only its limited region.

iii Ability to arbitrarily set the first and second derivatives at the starting and ending points points; together with (ii), this facilitates the smooth update of the trajectory and is necessary for obstacle avoidance motion planning, through which the trajectory must be re-planned whenever an obstacle is encountered.

Although many trajectory-generation methods have been discussed in the literature, several problems persist. One of the fundamental functions is to generate motion trajectories for robots considering several criteria and constraints such as travelling distance and expected arrival time [119]. The important criteria mentioned in (i)-(iii) are usually unconsidered and hence the resulting trajectory is typically a piecewise linear path or even a sharp path [120]. This requires a mobile robot to frequently stop, rotate and restart which causes discontinuity, leads to additional consumption of time and power, introduces delay and causes unnecessary wear on the robot parts [5].

To overcome this problem, many researchers have modelled robot trajectories as piecewise quadratic or cubic Bézier curves [86–89, 121, 122]. However, piecewise quadratic and cubic Bézier curves do not offer a continuous curvature or arbitrary setting of the second derivative at the starting and ending points of a trajectory. Although Neto et al. adopted seventh-order Bézier curves [92], higher-order Bézier curves are complex to calculate and are numerically unstable, leading to oscillations in the resulting trajectory [93].

In this chapter, a quintic Bézier curve-based algorithm that generates real-time trajectories for mobile robots is developed. This algorithm offers all the three essential criteria mentioned in (i)-(iii). It is considered that the trajectory is planned in real time by an operator, who inserts new via points by using a mouse-like device or a touch screen device. Also, the mobile robot operates in an indoor environment in which ceiling cameras

are installed; the operator uses these cameras to assigns the via points. Furthermore, a Lyapunov-theorem-based nonlinear trajectory tracking controller is designed to verify the effectiveness of the developed algorithm by conducting both simulation and experiment.

The rest of this chapter is arranged as follows: Section 3.2 provides a brief introduction to the kinematics of a mobile robot, defines a Bézier curve and describes its properties. The subdivision technique using the de Casteljau algorithm is introduced as well. Section 3.3 proposes the trajectory generation algorithm, including parameterisation and subdivision in high-curvature areas. Section 3.4 describes the designed trajectory tracking controller. The simulation and experimental results are presented in section 3.5, followed by a summary in section 3.6.

3.2 Preliminaries

3.2.1 Mobile Robot and Motion Trajectory

A typical two-wheeled differential mobile robot in a global coordinate frame, as shown in Fig. 3.1, is considered, because such robots are widely used in many applications. The linear and angular velocities are given by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{3.1}$$

where x and y are the two dimensional coordinate variables, θ is the orientation angle, and v and ω are the linear and angular velocities of the robot, respectively. The following nonholonomic constraint must be fulfilled in controller design or trajectory generation, assuming no slip in wheel motion.

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0. \tag{3.2}$$

In addition, given the maximum possible angular velocity $\dot{\phi}$ of the wheels (motors), velocity constraints are imposed. The velocity bounds are expressed as

$$|v| = \frac{r}{2}\left(\dot{\phi}_R + \dot{\phi}_L\right), \quad |\omega| = \frac{r}{2L}\left(\dot{\phi}_R - \dot{\phi}_L\right), \tag{3.3}$$

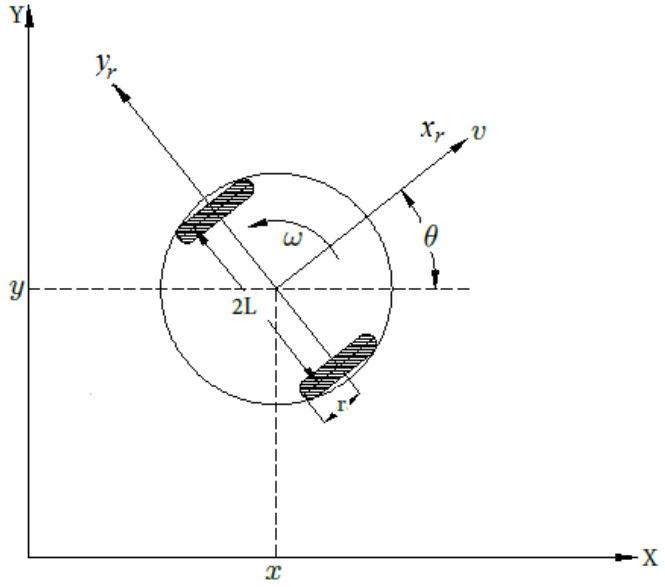


FIGURE 3.1: Mobile robot in a global coordinate frame.

where r is the wheel radius, L is the distance from the robot centre to the wheels and $\dot{\phi}_R$ and $\dot{\phi}_L$ are the angular velocities of the right and left wheels, respectively. The reference trajectory S_r of the robot is given as a function of time as follows:

$$S_r = \begin{bmatrix} x_r(t) & y_r(t) & \theta_r(t) & v_r(t) & \omega_r(t) \end{bmatrix}^{\mathrm{T}}, \tag{3.4}$$

where x_r and y_r are the reference positions in the global coordinate frame, and θ_r, v_r and ω_r are the reference angle, linear velocity and angular velocity, respectively.

$$\theta_r(t) = \arctan 2(\dot{y}_r(t), \dot{x}_r(t)), \tag{3.5}$$

$$v_r(t) = \sqrt{\dot{y}_r^2(t) + \dot{x}_r^2(t)}, \tag{3.6}$$

$$\omega_r(t) = \frac{\dot{x}_r(t)\ddot{y}_r(t) - \dot{y}_r(t)\ddot{x}_r(t)}{\dot{y}_r^2(t) + \dot{x}_r^2(t)}. \tag{3.7}$$

The curvature $C_r(t)$ of a reference trajectory at time t is given by

$$C_r(t) = \frac{\omega_r(t)}{v_r(t)} = \frac{\dot{x}_r(t)\ddot{y}_r(t) - \dot{y}_r(t)\ddot{x}_r(t)}{\left(\dot{x}_r^2(t) + \dot{y}_r^2(t)\right)^{3/2}}. \tag{3.8}$$

3.2.2 Bézier Curves

A Bézier curve is a parametric curve frequently used in graphical applications and related fields. It is defined by several control points and always passes through the initial and the final control points. Its shape can be altered by moving the control points. A two-dimensional Bézier curve of order n is represented as

$$P(t) = \sum_{i=0}^{n} \left(\frac{n!}{i!\,(n-i)!} \right) t^i (1-t)^{n-i} P_i, \quad t \in [0,1], \tag{3.9}$$

where $P(t) = [x(t), y(t)]^T$ and P_i are the two-dimensional Bézier curve and the control points, respectively. The following properties of Bézier curves render them particularly suitable for trajectory generation:

1. Bézier curves start at P_0 and end at P_n.
2. They are contained completely in the convex hull of the control points.
3. They are infinitely differentiable everywhere, and are therefore continuous at any degree (C^n-continuous, $\forall n$).
4. The vector tangential to the Bézier curve at the start (end) is parallel to the line connecting the first two (last two) control points.

The second property guarantees that, by setting all the control points within a motion area, the trajectory will always lie inside the motion area. A Bézier curve can be evaluated at a specific parameter value t and can be split at that value by using the de Casteljau algorithm [123]; new control points can be determined by the recursive application of (3.10).

$$P_i^m(t) = (1-t)P_{i-1}^{m-1} + tP_i^{m-1} \quad m = 1, 2, \ldots, n, \quad i = 0, \ldots, n-m. \tag{3.10}$$

The de Casteljau algorithm illustrated in Fig. 3.2, generates a cubic Bézier curve, which can be extended to any degree. This algorithm has the following properties:

1. The points P_i^0 are the original control points of the curve.
2. The value of the curve at t is P_n^n.
3. When the curve is split at t, it is represented as two curves with control points $(P_0^0, P_1^1, \ldots, P_n^n)$ and $(P_n^n, P_n^{n-1}, \ldots, P_n^0)$, respectively.

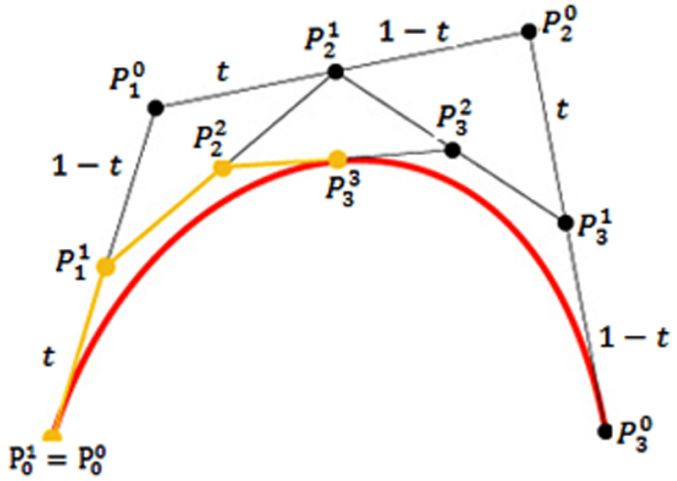


FIGURE 3.2: de Casteljau algorithm.

A Bézier curve constructed using numerous control points is numerically unstable, that is, moving one control point can alter the global shape of the curve. To overcome this problem, the entire Bézier curve is constructed from smoothly connected piecewise Bézier curves [120].

Although Bézier curves are differentiable infinitely and are therefore continuous to any degree, the continuity of long paths compiled from piecewise Bézier curves is limited by the number of control points. In this case, the continuity of the curve depends on the continuity at the intersection of the Bézier segments forming the composite Bézier curve. Furthermore, the complexity of the technique used to obtain C^n continuity is proportional to the order n. Herein, a C^2 continuity at the intersection of sequential Bézier curves, which ensures continuity of the entire trajectory, is considered.

3.2.3 Problem Statement

A teleoperated mobile robot in an indoor environment equipped with ceiling cameras for operator visibility is conceived. In this environment, the operator interacts with the robot via a wireless connection (as shown in Fig. 3.3). The physical dimensions of the objects in the environment are evaluated from the images captured using the ceiling cameras. The operator's task is to assign via points by clicking a mouse or using a touch screen device. Our objective is to develop and implement control algorithms based on corridor constraints and via points assigned by an operator in real time. The algorithm may generate off-line trajectories as well given a static map of the environment. The trajectory, which is generated automatically from the given via points and the corridor

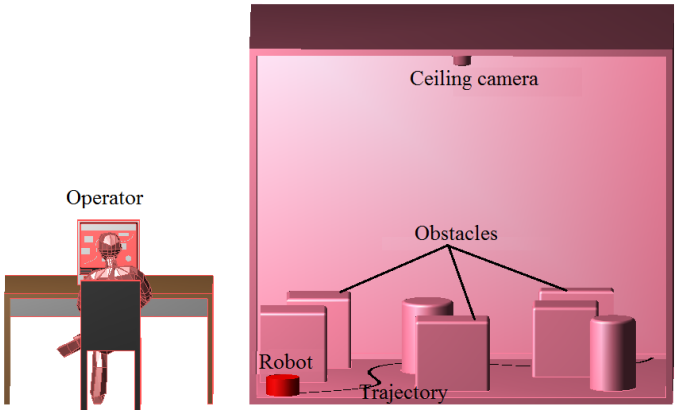


FIGURE 3.3: Teleoperated mobile robot in an indoor environment.

width, should be sufficiently smooth so that the mobile robot can track it. Simulation and experiment verify the smoothness.

3.3 Trajectory Generation

3.3.1 Trajectory-Generation Algorithm

As shown in Fig. 3.4, the trajectory is defined by two-dimensional via points W_j, $j = 0, 1, \ldots, f$ and corridor width d_l, $l = 0, 1, \ldots, f-1$, where f is the final via point (goal position). Two orthogonal unit vectors, $\hat{u}_l$ and $\hat{v}_l$, pointing in the motion direction and inside the bisector of W_{j-1}, W_j and W_{j+1}, respectively, are represented as follows:

$$\hat{u}_l = \frac{R_{\beta_j}\left[W_{j+1} - W_j\right]}{\left\|W_{j+1} - W_j\right\|}, \tag{3.11}$$

$$\hat{u}_0 = \frac{R_{\theta_0}\left[W_1 - W_0\right]}{\|W_1 - W_0\|}, \tag{3.12}$$

$$\hat{u}_f = \frac{R_{\theta_f}\left[W_j - W_{j-1}\right]}{\left\|W_j - W_{j-1}\right\|}, \tag{3.13}$$

$$\hat{v}_l = \frac{R_{\frac{\gamma_j}{2}}\left[W_{j+1} - W_j\right]}{\left\|W_{j+1} - W_j\right\|}, \tag{3.14}$$

$$R_\alpha = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}, \tag{3.15}$$

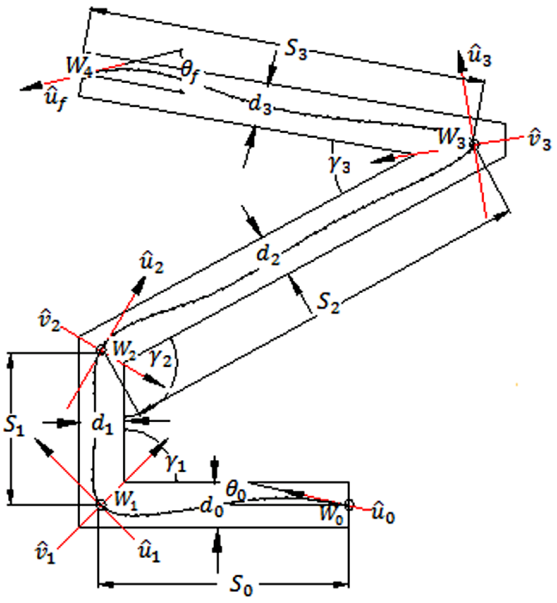


FIGURE 3.4: Robot course defined by five via points.

where $\hat{u}_0$ and $\hat{u}_f$ are unit vectors indicating the initial and the desired final robot orientations, respectively, and γ_j is the inner angle of the bisector inscribed by W_{j-1}, W_j and W_{j+1} as shown in Fig. 3.4. R_α is the rotation matrix, θ_0 and θ_f are the initial and the desired final orientations of the robot, respectively, and β_j is given by $(\pi - \gamma_j)/2$. Bézier curves are inserted between each segment defined by two consecutive via points and connected smoothly to obtain a C^2 continuous trajectory. From (5.1), the quintic Bézier curve of the k^{th} segment, $k = 1, 2, \ldots, f$ is given by

$$\begin{aligned} P(t)_k =& (1-t)^5 P_{0k} + 5t(1-t)^4 P_{1k} + 10t^2(1-t)^3 P_{2k} \\ &+ 10t^3(1-t)^2 P_{3k} + 5t^4(1-t) P_{4k} + t^5 P_{5k}, \end{aligned} \tag{3.16}$$

where $P(t)_k$, P_{ik}, for $i = 0, 1, 2, \ldots, 5$, and t are the two-dimensional Bézier curve, control points and the parameterisation variable, respectively. The first and the last points of each Bézier segment, P_{0kj} and P_{5k}, are operator-specified via points. The four inner points, P_{1k}-P_{4k}, are calculated from the initial and the final tangents (first derivatives) and the curvatures (second derivatives) of the curve. The first and the second derivatives of (5.2) at the start and the end points of each Bézier segment are given by (3.17)-(3.20), where $r_l = \frac{1}{2}\min\{d_l, d_{l+1}\}$.

$$P^{'}(0)_k = 5(P_{1k} - P_{0k}) = \hat{u}_l r_l. \tag{3.17}$$

$$P^{'}(1)_k = 5(P_{5k} - P_{4k}) = \hat{u}_{l+1} r_l. \tag{3.18}$$

$$P^{''}(0)_k = 20(P_{0k} - 2P_{1k} + P_{2k}). \tag{3.19}$$

$$P^{''}(1)_k = 20(P_{3k} - 2P_{4k} + P_{5k}). \tag{3.20}$$

Equations (3.17) and (3.18) are solved directly from the initial and the final tangents of the curve. The second derivatives, which determine the curvature in (3.19) and (3.20), require an additional technique for their solution. Here the curvatures of the quintic Bézier curve are generated using a cubic Bézier curve. However, because the cubic Bézier is not C^2 continuous, it is not applied directly; instead, the quintic Bézier curve is transformed into a cubic Bézier curve while preserving its continuity property. The cubic Bézier curve is represented as

$$Q(t) = (1-t)^3 b_0 + 3t(1-t)^2 b_1 + 3t^2(1-t) b_2 + t^3 b_3, \tag{3.21}$$

where $Q(t)$ and b_i are two-dimensional cubic Bézier curve and control points, respectively. The first and the second derivatives of (3.21) at the start and the end of the curve are given by

$$b^{'}(0) = 3(b_1 - b_0), \tag{3.22}$$

$$b^{'}(1) = 3(b_3 - b_2), \tag{3.23}$$

$$b^{''}(0) = 6(b_0 - 2b_1 + b_2), \tag{3.24}$$

$$b^{''}(1) = 6(b_1 - 2b_2 + b_3). \tag{3.25}$$

As shown in (3.22)-(3.25), if the initial and the final tangents are computed by (3.17) and (3.18), respectively, four equations and four unknowns are deduced, and the second derivative of the cubic Bézier curve is obtained easily. This idea is extended to quintic Bézier curves. Assuming zero curvature at the first and last via points, only the curvatures at the intersection of the consecutive Bézier segments S_k and S_{k+1} are considered. The

quintic Bézier segments are formulated in terms of the cubic Bézier curve by redefining the control points cp as follows:

$$cp = \begin{cases} W_{j-1},\ W_{j-1} + \frac{1}{3}\hat{u}_{l-1}r_l,\ W_j - \frac{1}{3}\hat{u}_l r_l,\ W_j, & \text{for } Q(t)_k \\ W_j,\ W_j + \frac{1}{3}\hat{u}_l r_l,\ W_{j+1} - \frac{1}{3}\hat{u}_{l+1}r_l,\ W_{j+1}, & \text{for } Q(t)_{k+1}, \end{cases} \tag{3.26}$$

where $Q(t)_k$ and $Q(t)_{k+1}$ are the Bézier curves of the segments S_k, and S_{k+1}, respectively. Therefore, the two consecutive Bézier segments are given by

$$\begin{aligned} Q(t)_k &= (1-t)^3 W_{j-1} + 3t(1-t)^2\left(W_{j-1} + \frac{1}{3}\hat{u}_{l-1}r_l\right) \\ &\quad + 3t^2(1-t)\left(W_j - \frac{1}{3}\hat{u}_l r_l\right) + t^3 W_j, \end{aligned} \tag{3.27}$$

$$\begin{aligned} Q(t)_{k+1} &= (1-t)^3 W_j + 3t(1-t)^2\left(W_j + \frac{1}{3}\hat{u}_l r_l\right) + \\ &\quad 3t^2(1-t)\left(W_{j+1} - \frac{1}{3}\hat{u}_{l+1}r_l\right) + t^3 W_{j+1}. \end{aligned} \tag{3.28}$$

By substituting (3.27) and (3.28) into (3.22)-(3.25),

$$P''(1)_k = P''(0)_{k+1} = \frac{(A+B)}{2} \tag{3.29}$$

is obtained, where

$$\begin{aligned} A &= 6W_j + 2P'(0)_k + 4P'(1)_k - 6W_{j+1}, \\ B &= -6W_{j+1} - 2P'(0)_{k+1} - 4P'(1)_{k+1} + 6W_{j+2}. \end{aligned}$$

From the designated via points and by substituting (3.29) into (3.19) and (3.20), the six required control points for the quintic Bézier segments are obtained as follows:

$$\begin{aligned}
P_{0k} &= W_j, \\
P_{1k} &= \frac{\hat{u}_l r_l}{5} + W_j, \\
P_{2k} &= \frac{P''(0)_j}{20} + 2P_{1j} - P_{0j}, \\
P_{3k} &= \frac{P''(1)_j}{20} + 2P_{4j} - W_{j+1}, \\
P_{4k} &= W_{j+1} - \frac{\hat{u}_{l+1} r_l}{5}, \\
P_{5k} &= W_{j+1}.
\end{aligned} \tag{3.30}$$

3.3.2 Subdivision for Curvature Improvement

This subdivision technique recognises that the curve can be approximated by the control polygon of a Bézier curve. As the curve is subdivided, the approximation improves. Especially, if a Bézier curve is subdivided into several segments, the arc length of the original curve exceeds the sum of the chord lengths of the segments and is less than the sum of their polygon lengths [124]. As explained in section 3.2, the robot velocities are bounded, and thus the curvature, due to the boundedness of wheel angular velocities. While generating the reference trajectory, it is important to consider a traversable curvature in 3.8. If it is above the upper bound of the robot's curvature, the robot cannot traverse smoothly and continuously. Instead, it must stop, rotate, and continue moving. These manoeuvres are time-consuming and should be avoided wherever possible. Thus, in high-curvature areas, if there is room for trajectory optimisation, the Bézier curve is subdivided and a quintic Bézier segment is inserted as explained below. Based on the curvature $C_r(t)$ in (3.8), the curve is subdivided as follows (Fig. 3.5):

If $C_r(t) > C(t)_{max}$ and $d_j > d_{j_{min}}$, then

$W_j^* = W_j + \mu \hat{v}_l$

where $C(t)_{max}$, $d_{j_{min}}$ and W_j^* are the bounded curvature, the minimum required width of the trajectory, and the reallocated via point, respectively. μ is the reallocation parameter along the unit vector $\hat{v}_l$ and is defined as $\mu \leq \frac{1}{2}\left(d_j - d_{j_{min}}\right)$

A subdivision point is introduced at $t = t^*$ in (3.10). To this end, first, the via point W_j is reallocated to W_j^* at the intersection of the two Bézier segments defining the curvature of interest with the minimum allowable corridor width (Fig. 3.5). Then, the subdivision

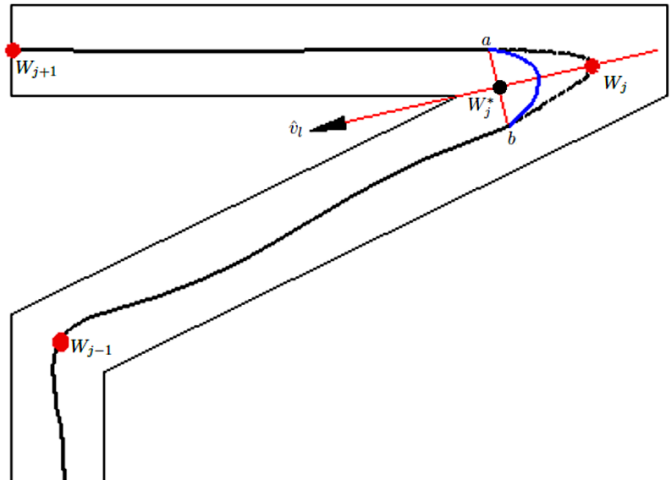


FIGURE 3.5: Subdivision and curvature improvement.

points are defined by drawing a line $\overline{ab}$ orthogonal to unit vector $\hat{v}_l$. The segments W_{j-1} and W_j and W_j and W_{j+1} are subdivided at points a and b, and a new quintic Bézier segment is inserted between a and b to ensure continuity at the junctions. Although the implementation of this step could be simplified by inserting a cubic Bézier curve, curvature discontinuity will occur because cubic curves do not provide C^2 continuity. The benefits of this technique include reduction of trajectory length and curvature improvement. If the curvature is too high, that is, at very sharp corners, the inserted segment will not respect the bounded curvature. Thus, the robot would have to stop, rotate and restart, as in conventional approaches.

3.3.3 Parameterisation for Velocity Improvement

The principal problem of interpolating path reference points using piecewise Bézier curves is the irregular temporal spacing of the reference points. Curves are commonly approximated by many short-distance segments assuming a predefined number of points/increments. The disadvantage of this approach is that points are spaced sparsely when the paths are long or overcrowded on short paths. Hence, if each segment of the curve is parameterised from 0 to 1, the parameterisation concerning the overall time parameter is not constant, and the tangent continuity ensured by Bézier's construction is lost [125]. Some studies have adopted arc length parameterisation to resolve this issue. In this approach, equidistant knots are inserted into each Bézier segment and are used to formulate a B-spline curve. Although the resulting curve is parameterised by arc length, it lacks two important properties; strong correlation and arbitrary setting of the second

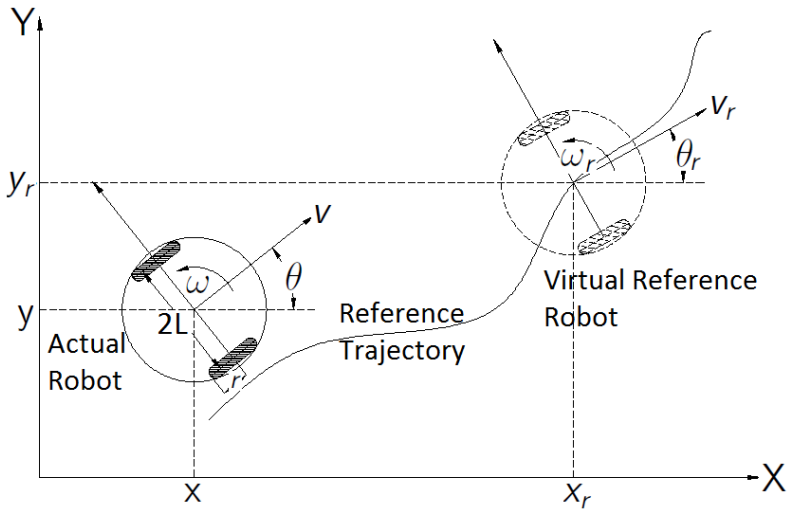


FIGURE 3.6: Trajectory tracking error of mobile robot in global coordinate frame.

derivative at the start position. Therefore, the number of points is calculated based on the length of each segment. First, the arc lengths of each segment are calculated, and the points are distributed in predefined intervals. The arc length of each segment is given by

$$S(t) = \int_0^1 \left\| P'(t) \right\| dt. \tag{3.31}$$

The re-parameterisation parameter τ is calculated from the desired velocity $v_d \leq v_r$ in (3.6), and the arc length $S(t)$ is given as follows:

$$\tau = \frac{v_d}{S(t)}, \quad \tau = [0, 1]. \tag{3.32}$$

3.4 Controller Design

A Lyapunov-theorem-based nonlinear controller for trajectory tracking is designed to verify the effectiveness of the proposed algorithm. Nonlinear controllers are preferred owing to their efficiency and wide application in nonlinear dynamic systems [126–129]. The function of this controller is to provide the mapping between the reference trajectory and the actuator commands so that the robot can achieve the tracking task. The reference position q_r and velocity $\dot{q}_r$ are determined using the above method (trajectory-generation algorithm).

The design of this controller was proposed in [130], where a linear (PID) controller was used. The idea was referred to by many researchers and extended into other control

techniques, including feedback linearisation and nonlinear controller design [131, 132]. Nonlinear control approaches take advantage of the given nonlinear system dynamics to generate high-performance controllers. In this chapter, a simple nonlinear controller is designed by selecting new control inputs, and its stability is guaranteed by Lyapunov analysis as in [130]. The tracking problem is formulated by introducing a virtual reference robot as shown in Fig. 3.6. The position-tracking error between the actual and the reference robots is expressed as

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}. \tag{3.33}$$

The error dynamics is obtained by taking the time derivative of (3.33) as

$$\begin{aligned} \dot{e_1} &= v_r \cos e_3 - v + \omega e_2, \\ \dot{e_2} &= v_r \sin e_3 - \omega e_1, \\ \dot{e_3} &= \omega_r - \omega. \end{aligned} \tag{3.34}$$

The nonlinear transformation of the velocity inputs is introduced in (3.35) and substituted into (3.34) to obtain the error dynamics in (3.36).

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} v_r \cos e_3 - v \\ \omega_r - \omega \end{bmatrix}. \tag{3.35}$$

$$\begin{aligned} \dot{e_1} &= \omega e_2 + u_1, \\ \dot{e_2} &= v_r \sin e_3 - \omega e_1, \\ \dot{e_3} &= u_2. \end{aligned} \tag{3.36}$$

An asymptotically stable controller is designed based on the Lyapunov method [133]. The following lower-bounded Lyapunov function is introduced as proposed in [130].

$$V = \frac{e_1^2}{2} + \frac{e_2^2}{2} + (1 - \cos e_3). \tag{3.37}$$

Control inputs u_1 and u_2 are selected so that the derivative of the Lyapunov function in (6.12) becomes negative semi-definite as follows:

$$\begin{aligned} u_1 &= -k_1 e_1, \\ u_2 &= -k_2 v_r e_2 - \frac{k_3 e_2^2}{\max(\sin e_3, \epsilon)} - k_4 v_r \sin e_3, \end{aligned} \tag{3.38}$$

where k_1, k_2, k_3 and k_4 are positive constants, and ϵ is a small constant, for example, 10^{-5} to avoid zero division. The function $\dot{V}$ becomes

$$\dot{V} = -k_1 e_1^2 - k_3 e_2^2 - k_4 \sin^2 e_3 \leq 0. \tag{3.39}$$

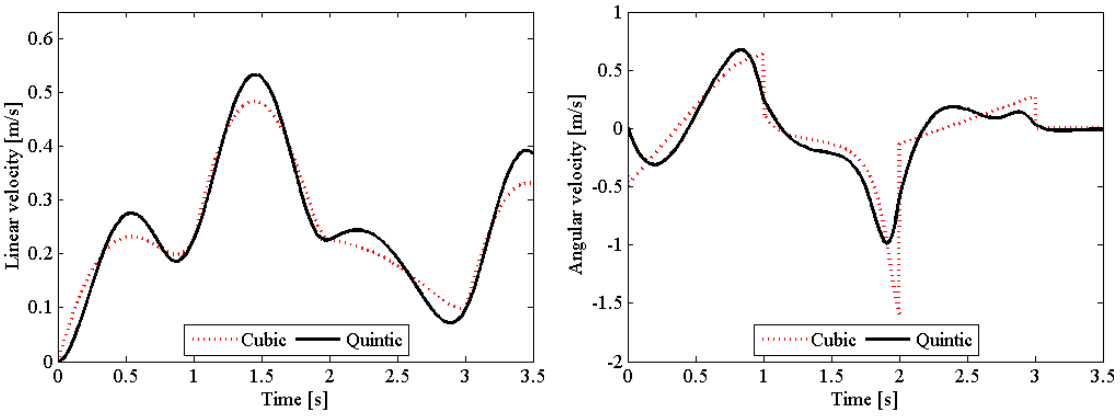
According to Barbalat's lemma for stability analysis [133], the system is concluded to be asymptotically stable.

3.5 Simulation and Experiment

The objective of conducting simulation and experiment is to verify the smoothness of the trajectory generated by the proposed algorithm, as well as the tracking ability of the designed controller. A comparison with the conventional trajectory is made to evaluate its effectiveness. In this context, the word conventional refers to linear interpolation methods that offer position continuity for trajectories with several segments but neither velocity nor curvature continuity. Although many methods have been proposed in the literature, such as quadratic and cubic piecewise Bézier curves, they do not offer curvature continuity [86–89, 121, 122]. To describe, velocity profiles of the cubic and the quintic Bézier curves in section 3.5.1 are considered.

3.5.1 Comparison with Cubic Bézier Trajectories

Although a single Bézier curve is infinitely differentiable within itself, it cannot be used to generate a long trajectory; otherwise, it will be complex to calculate and numerically unstable, leading to oscillations in the resulting trajectory [93]. Alternatively, piecewise Bézier curves are connected smoothly; therefore, the derivative property depends on the order of the affiliated curves. Cubic Bézier curves are among the curves used commonly for generating parametric trajectories. Their implementation is simple because, for a given set of via points, only the first derivatives at the start and end points of the Bézier



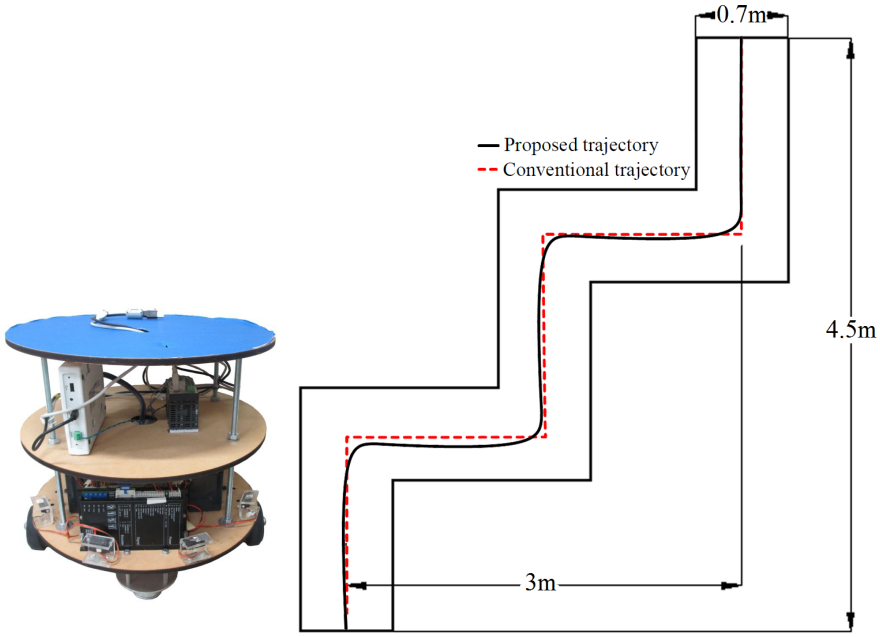
(a) Linear Velocity. (b) Angular Velocity.

FIGURE 3.7: Simulated velocity profiles of quintic and cubic Bézier curves.

curve are required to obtain all control points (equations (3.21) to (3.23)). Quintic Bézier curves require both the first and the second derivatives (equations (5.2) to (3.20)). However, the trajectories generated using piecewise cubic Bézier curves lack curvature continuity. For further elaboration, a simulation based on both the quintic and cubic Bézier curves is conducted on the same course and under the same conditions. The results pertaining to both the linear and the angular velocities are shown in Figure 4.10. Although the linear velocity profiles are relatively the same (Figure 4.10(a)), a major difference can be observed in the angular velocities (Figure 4.10(b)). The proposed method provides a smooth profile throughout the course, whereas the conventional method shows discontinuities at corners around 1 s, 2 s and 3 s. In practical situations, the robot is constrained by hardware limitations that bound the velocity as well as the acceleration. The discontinuities in angular velocity may deteriorate the tracking performance of the robot.

3.5.2 Experimental Setup

A simple differential-drive mobile robot (Fig. 3.8(a)) designed in our laboratory was used in this research. A crank course of width 0.7 m was planned such that the robot clearance would be 12 cm on both sides, and the reallocation parameter μ was set to 14 cm (Fig. 3.8(b)). Because the Bézier subdivision algorithm was applied to the proposed reference trajectory to improve curvatures at all corners, the trajectory slightly deviates from centre positions at corners. The simulation and experiment were both conducted at the maximum



(a) Mobile Robot. (b) Crank course.

FIGURE 3.8: Experimental setup: Mobile robot and trajectories used in the experiment.

linear velocity bound as shown in Table. 3.1. The aim was to verify the effectiveness of the proposed algorithm under the highest velocity condition.

The controller gains were chosen as $k_1 = 91\,\mathrm{V\,m^{-1}}$, $k_2 = 35\,\mathrm{V\,s\,m^{-2}}$, $k_3 = 30\,\mathrm{V\,s^2\,m^{-2}}$ and $k_4 = 25\,\mathrm{V\,s\,m^{-1}}$ for the simulation of both algorithms. The same gains were used in the experiment based on the proposed algorithm, whereas for the conventional one, they were set to $k_1 = 70\,\mathrm{V\,m^{-1}}$, $k_2 = 35\,\mathrm{V\,s\,m^{-2}}$, $k_3 = 30\,\mathrm{V\,s^2\,m^{-2}}$ and $k_4 = 65\,\mathrm{V\,s\,m^{-1}}$. The differences in k_1 and k_4 can be described to the fact that the conventional method has higher curvature than the proposed algorithm. Therefore, k_1 is reduced while k_4 is increased to achieve the best tracking performance.

TABLE 3.1: Robot specifications.

Robot Specification	Value	Actuator Specification	Value
Weight (kg)	17.9	Voltage (V)	24
Radius (cm)	23	Peak current (A)	5.4
Wheel Radius (cm)	5	Output power (W)	20.3
Moment of inertia	1.12	Speed (rpm)	3000
Linear velocity ($\mathrm{m\,s^{-1}}$)	3.14	Gear ratio	50
Angular velocity ($\mathrm{rad\,s^{-1}}$)	1.365	Output torque (Nm)	3.2

3.5.3 Simulation and Experimental Results

The simulated velocity profiles and tracking performance based on both methods are shown in Fig. 3.9, where (a) and (b), and (c) and (d) are the linear and angular velocities, and the tracking performances for the conventional and the proposed methods, respectively. Tracking performance is significantly equivalent in both cases; however, there is a major difference in velocity profiles. The conventional case has a higher magnitude of angular velocity owing to discontinuities at corners.

The experimental results are shown in Figs. 3.10 and 3.11, and the video captures in Fig. 3.12. The running times are 34.38 s and 31.98 s for the conventional and the proposed methods, respectively. Figure. 3.10 shows the trajectory tracking results. Figure. 3.11(a) shows the velocity profiles of both algorithms, where in the case of the proposed algorithm, it is always non-zero, while for the conventional method, it is zero at corners (at t = 7.38 s, 14.1 s, 20.88 s and 20.54 s). Although the linear velocity bounds are the same in both cases, the angular velocity reaches its upper and lower bounds with the conventional method, whereas in the case of the proposed algorithm, it is nearly half the value of (b). It is observed that the proposed velocity profiles are relatively the same in the simulation (Fig. 3.9 (a) and (b) and the experimental results (Fig. 3.11 (a) and (b)), with a slight difference at corners (at t = 8.3 s, 14.5 s, 21 s and 27.2 s). The difference in velocity profiles is mainly due to unmodelled friction. The trajectory tracking result proves that the proposed controller achieves a sufficiently good tracking performance as shown in Fig. 3.9. Compared to the simulation result in Figs. 3.9 (c) and (d), the proposed method achieves equivalent performance, whereas the conventional yields poor practical performance. These results show that the proposed method generates a trajectory smooth enough to be tracked by the mobile robot.

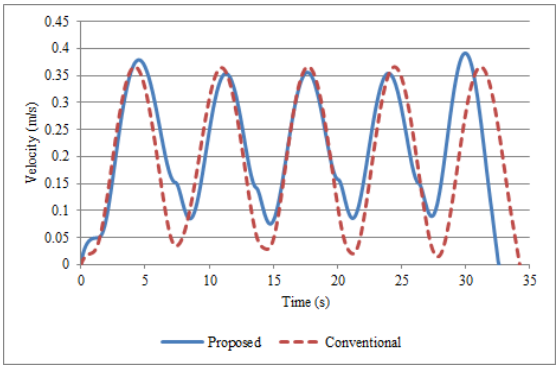
Wheel velocities of the left and the right wheels are shown in Figs. 3.11 (c) and (d), respectively. In the proposed algorithm, the wheels always spin along the motion direction, while in the conventional method, one wheel must turn in reverse to achieve a complete turn at corners. Acceleration results obtained using the proposed algorithm are relatively smoother and lower than those obtained using the conventional method, as in Figs. 3.11 (e) and (f).

3.5.4 Discussion

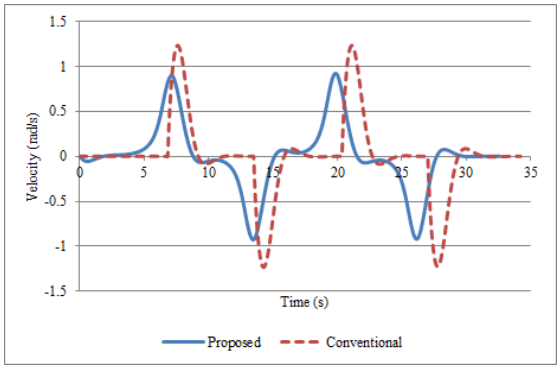
The trajectory of a nonholonomic mobile robot must observe important criteria such as continuous curvature and local controllability. The interesting properties of Bézier curves have drawn the attention of many researchers for trajectory generation. Quadratic and cubic piecewise Béziers are widely used in this area because their implementation is relatively simple compared to those of higher-order [86–89, 121, 122]. Some researchers have considered the application of quartic Bézier curves [134]; if used in a single form, it is infinitely differentiable akin to any n^{th}-order Bézier curve. However, a single Bézier curve cannot be used to generate long or complex trajectories. Instead, composite curves are used; henceforth, the trajectory continuity depends on their orders. At the very least, quintic curves (six control points) are required to guarantee a continuous curvature. For each continuity, that is, position, velocity and curvature, two control points are used at the junction of the consecutive curves. The simulation results in Fig. 4.10 show that composite low-order Bézier curves do not offer continuous curvature. In this light, a simple and efficient algorithm based on quintic Bézier curves with the adoption of Bézier subdivision for curvature improvement is proposed. The effectiveness was validated by simulation and experimental results.

3.6 Summary

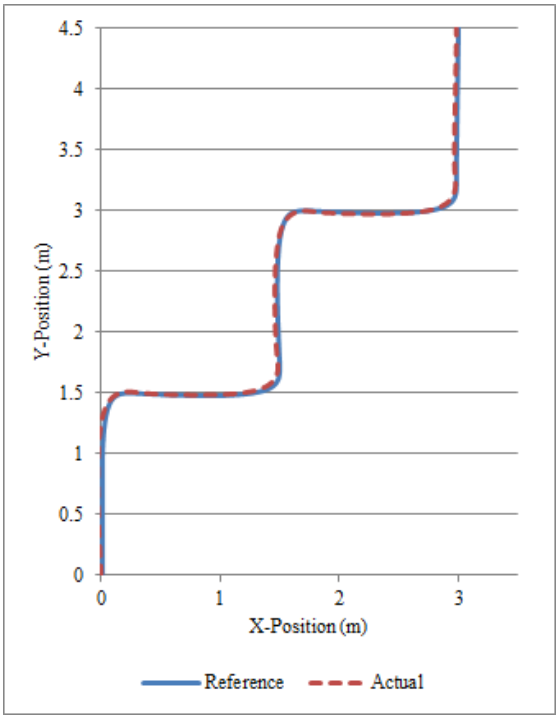
In this chapter,a simple and efficient algorithm for generating smooth trajectories by using piecewise quintic Bézier curves is demonstrated. The generated trajectory observes all the basic criteria that guarantee the smooth motion of a mobile robot. The adoption of Bézier subdivision led to a significant improvement of the curvature at sharp corners. Moreover, the proposed method allows for setting the first and the second derivatives arbitrarily at the starting and ending points of the trajectory. The combination of this feature and Bézier subdivision is suitable for trajectory generation for obstacle-avoidance. When the robot encounters an obstacle, the trajectory is subdivided and a new Bézier curve that avoids obstacles is connected smoothly. The implementation of this part is left as future work.



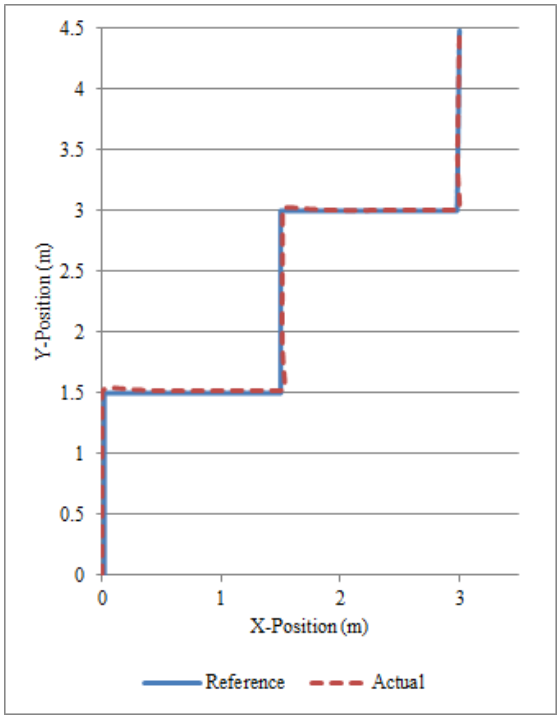
(a) Linear velocity.



(b) Angular velocity.

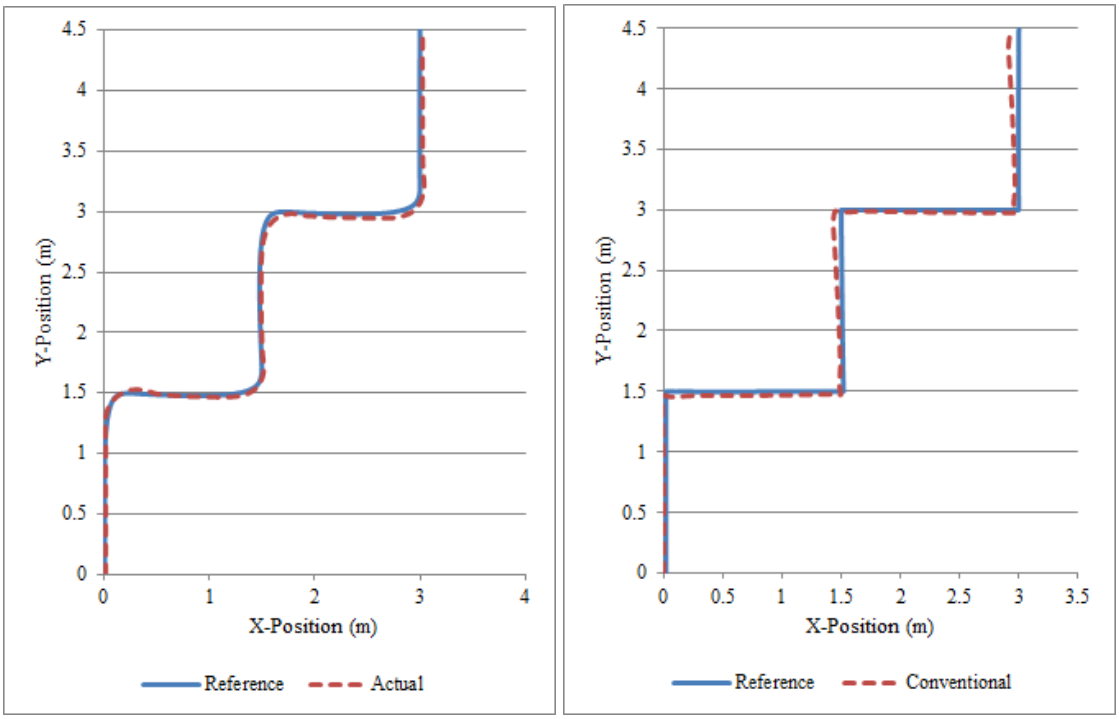


(c) Trajectory tracking based on proposed method.



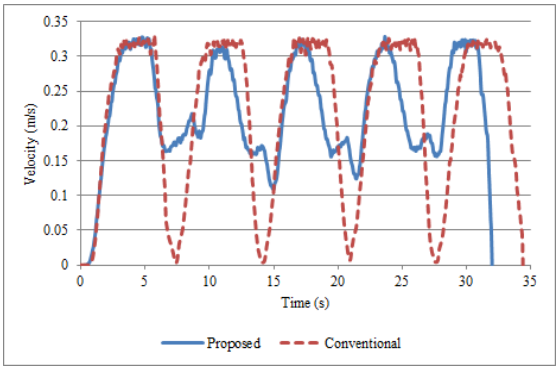
(d) Trajectory tracking based on conventional method.

FIGURE 3.9: Simulation results based on the proposed and conventional methods.

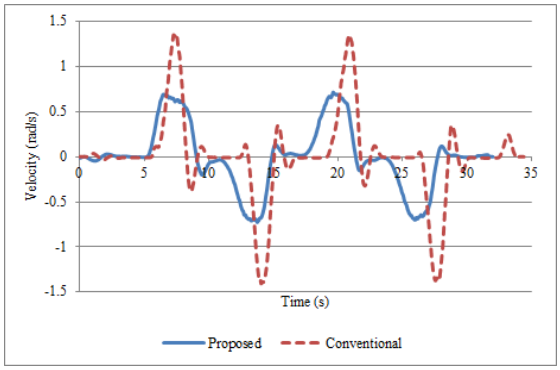


(a) Proposed method. (b) Conventional method.

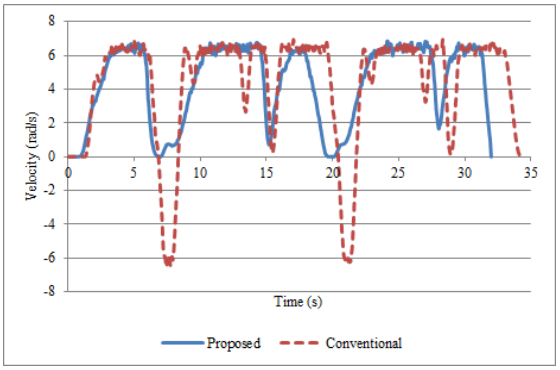
FIGURE 3.10: Experimental results showing the trajectory tracking performance using the designed controller and based on the proposed and conventional trajectories.



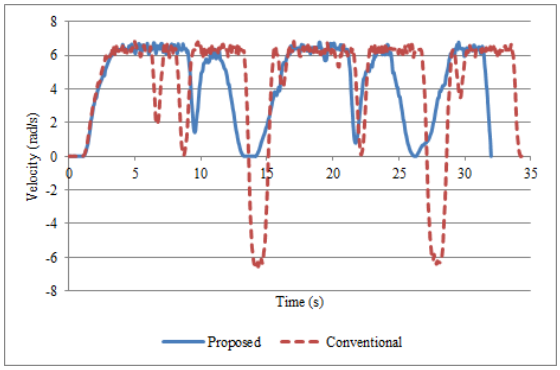
(a) Linear velocity.



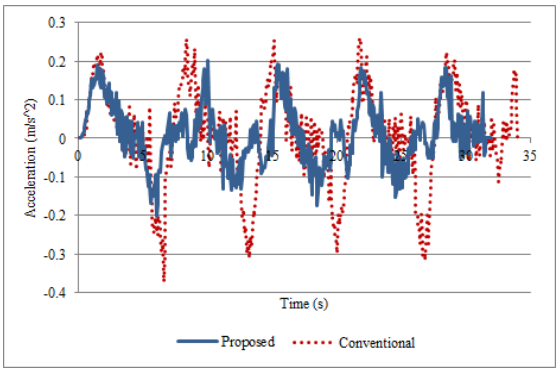
(b) Angular velocity.



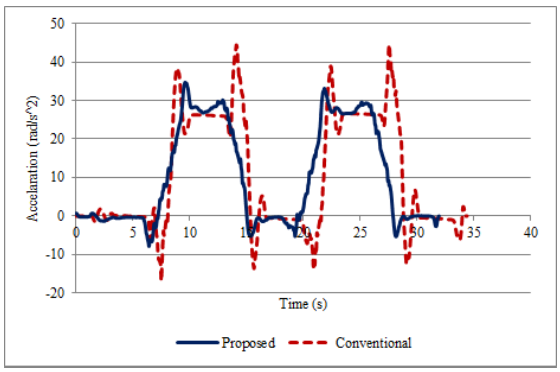
(c) Velocity of the left wheels.



(d) Velocity of the right wheels.



(e) Linear acceleration.



(f) Angular acceleration.

FIGURE 3.11: Experimental results showing velocity and acceleration profiles based on the proposed and conventional trajectories.

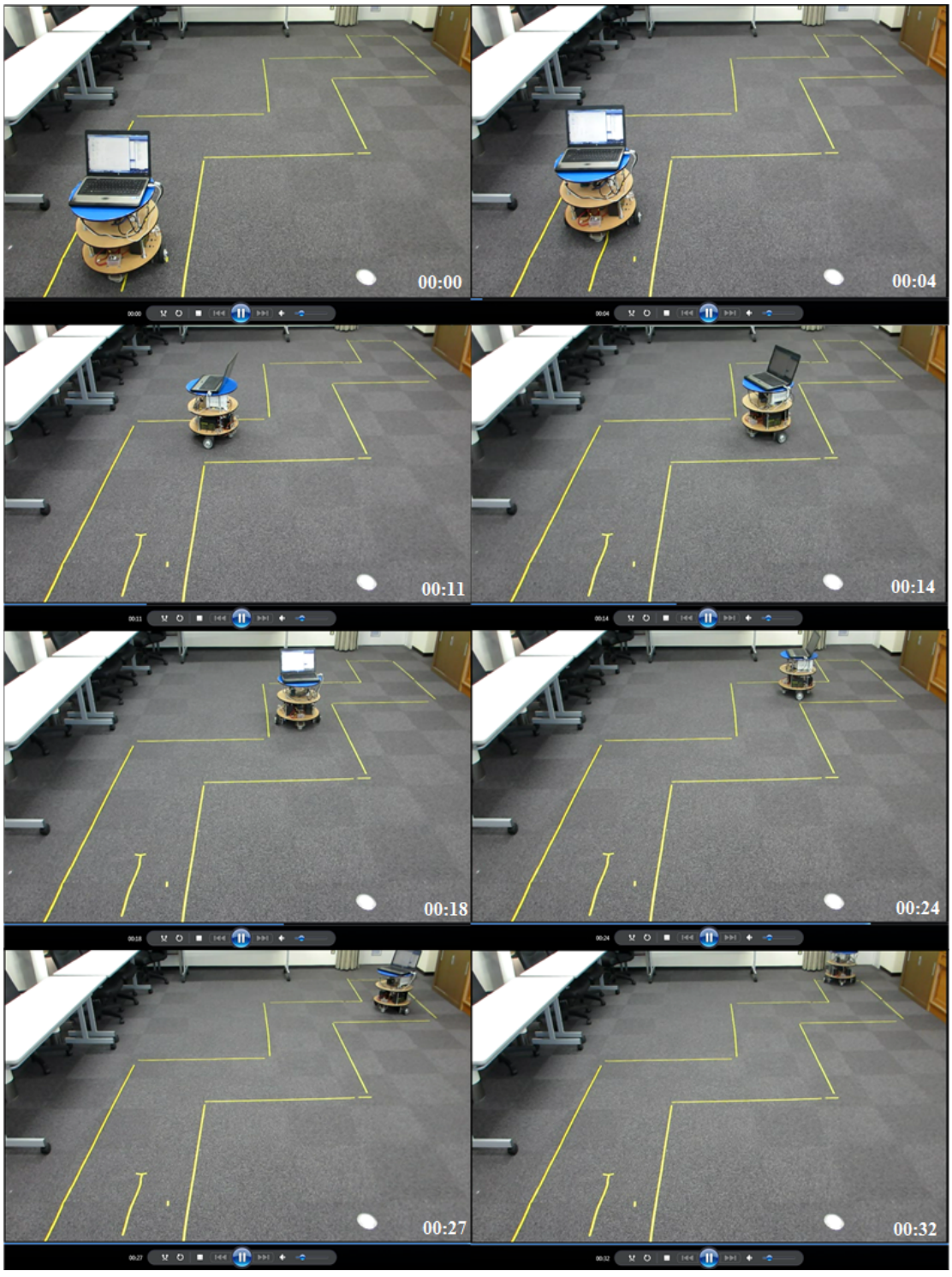


FIGURE 3.12: Experimental video capture based on the proposed controller and trajectory.

Chapter 4

Vision-Based Smooth Obstacle-Avoidance Trajectory Generation for Mobile Robots

This chapter proposes an obstacle-avoidance trajectory-generation method that provides a smooth trajectory in real time. The trajectory is generated from an environmental top-view image, where a fisheye lens is used to capture a wide area from a low height. Corners of obstacles are detected and corrected using the log-polar transform and are used to generate a simple configuration space that reduces the computation time. An optimal path is computed by using the A* algorithm and replaced by a smooth trajectory generated based on piecewise quintic Bézier curves. Based on the established goal and visual information, a method for generating the first and second derivatives at the start and the end points of each Bézier segment is proposed to generate a continuous curvature trajectory. The method is simple and easy to implement and has an average computation time of 1.17 s on a PC (CPU: 1.4 GHz) for a workspace containing five to six obstacles. Experimental results verify that the proposed method is effective for real-time motion planning of autonomous mobile robots.

4.1 Introduction

Because mobile robots normally operate in environments occupied by obstacles, they must be able to avoid obstacles. Moreover, some obstacles may move or change over time, requiring the algorithm to be updated continuously. Therefore, automatic motion planning is one of the most significant and important challenges in robotics research.

As mentioned in the previous chapter, although many path-planning methods have been discussed in the literature, several problems remain unresolved. Most studies have considered the fundamental criteria and constraints such as time delay, travelling distance, and expected arrival time. However, smoothness in terms of curvature (C^2) continuity has typically been neglected. Therefore, the resulting trajectory is usually a piecewise linear path or even a sharp path, requiring the mobile robot to stop, rotate and restart frequently [120]. Such discretised translations and rotations cause discontinuity, consume time and power, introduce delay and cause unnecessary wear of robot parts [5]. To guarantee smooth motion, many researchers have modelled robot trajectories as quadratic or cubic Bézier curves [87–89, 121, 135]. However, both quadratic and cubic piecewise Bézier curves offer neither curvature continuity nor do they allow for arbitrarily setting of a second derivatives at the starting and ending positions of the trajectory.

Chandak et al. introduced path planning for mobile robot navigation by using image processing [136]. Their method generates only the feasible path regardless of its geometry. In addition, the authors used a single normal ceiling camera whose vision was limited to a small area. To expand the operation area, they suggested the use of multiple cameras; however, they mentioned that the use of multiple cameras would increase the computation time, which would hinder real-time applications. On the other hand, Maron and Pérez showed that by decomposing the configuration space, the visibility graph method could be used efficiently for real-time path planning in large planar environments. Moreover, its implementation is simple, and the resulting path is always optimal regarding Euclidean distance and has certain advantages over other classical methods such as cell decomposition and the Voronoi diagram [16]. Furthermore, Masehian and Amin showed that the visibility graph produces a shorter path than the Voronoi diagram and the potential fields methods [137].

The objective of this study is to overcome the stated drawbacks by generating a trajectory with the shortest possible distance, without any sharp turn and with adequate safety distance from obstacles. In addition, the computation time should be minimised as much as possible to allow for real-time applications. Furthermore, the method should be autonomous to reduce the operator's input so that his/her main task becomes to assign the goal position and, if needed, the safety distance between obstacles and the robot. To this end, the visibility graph method and Bézier curves are used in this research. Image processing is employed to extract corners of obstacles from a workspace captured using a fisheye lens attached to a webcam. A top-surface image of the obstacles is obtained based

on colour filtration, and their corners are detected to match the corresponding lower-surface corners. The lower-surface corners are used to calculate the actual distances between the obstacles and robot. From these corners, the visibility graph is generated, and the shortest path is searched for, which is then transformed into a smooth trajectory by using quintic Bézier curves as proposed in chapter 3.

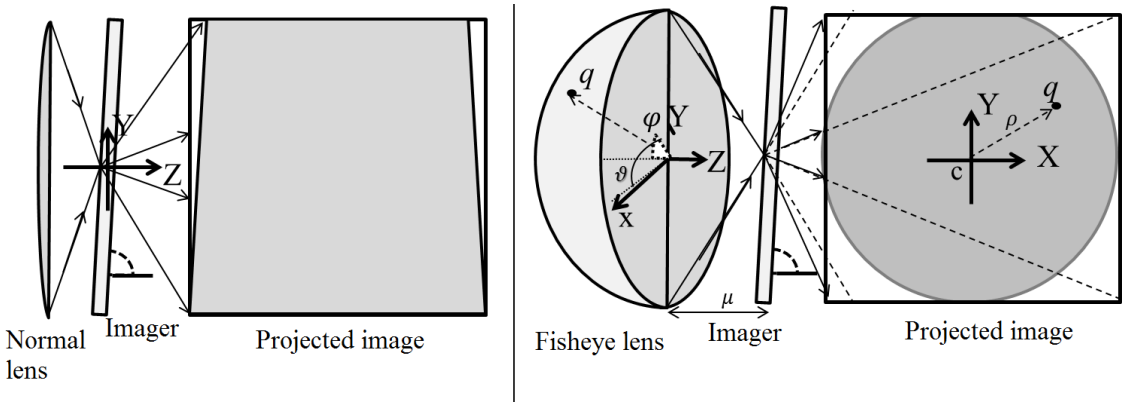
The main contribution of this chapter is the generation of smooth and obstacle-avoidance trajectories by using Bézier curves constrained by via points and path-widths (safety margins). An algorithm is proposed for generating the first and second derivatives at the start and end points of each path segment based on only the defined safety margins and the via points resulting from the visibility graph. This algorithm guarantees that the entire trajectory would be second-order differentiable. In addition, this chapter demonstrates a new approach using the Log-Polar Transform (LPT) to distinguish the top and bottom object surfaces in indoor environments captured by ceiling cameras. The LPT remaps the objects according to the distance from the centre of the environmental image. Therefore, the heights of the objects in the log-polar domain are represented equally, regardless of the objects' distribution in the environment.

The remainder of this chapter is organised as follows: Section 4.2 addresses the steps involved in the extraction of corners by using a fisheye lens. A brief description of lens distortion and calibration is provided, and the LPT used to correct the corner coordinates is discussed. In section 4.3, the configuration space based on the corrected corners is generated, visibility graph is constructed and shortest path is found. The conversion of linear paths into a smooth trajectory is explained as well. The experimental results are presented in section 4.4, followed by a summary in section 4.5.

4.2 Workspace Representation

4.2.1 Strategy Specification

Image processing has become closely associated with robotic systems, in which it is commonly used to help a robot navigate to the desired position [138–140]. Navigation information can be obtained using either a camera installed on a robot to capture the workspace concerning the current location of the robot or by using ceiling camera that captures a fixed view of the environment through which the robot is to navigate. The



(a) Conventional camera. (b) Fisheye camera.

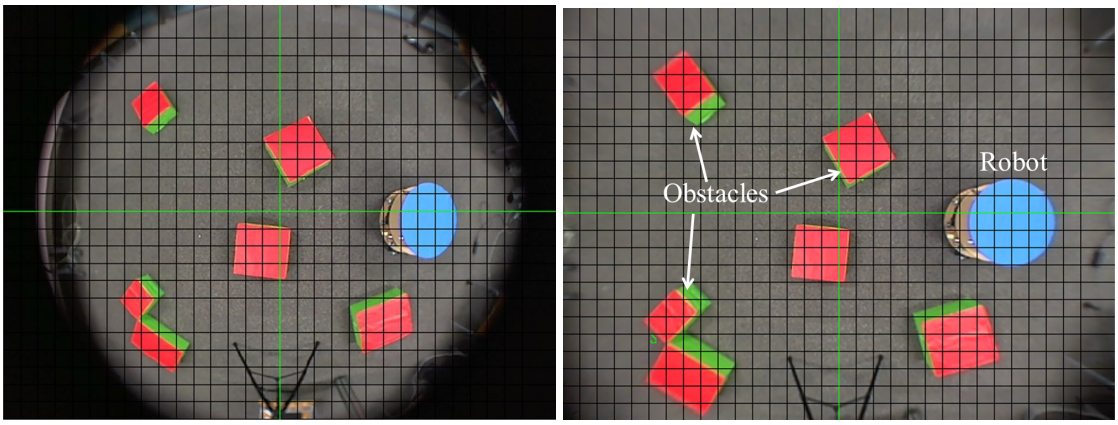
FIGURE 4.1: Camera structure.

latter strategy is considered in this study, whereby the camera is used to capture the top view of the workspace from a fixed height Λ.

4.2.2 Fisheye Distortions and Calibration Process

Conventional cameras are equipped with spherical lenses, as shown in Fig. 4.1(a), due to the complexity of manufacturing flat lenses. The angle of view is proportional to the height or distance from the area of interest; hence, at a low height of approximately 2-3 metres, the captured area is very limited. To make the proposed method applicable to various tasks, a fisheye lens is attached to the camera because its optical structure covers a wide angle of view [141]. Nevertheless, the fisheye lens causes radial and tangential distortions, which can be eliminated from an image through calibration. The radial distortion is produced by the lens shape and appears as barrel effect in an image. The tangential distortion arises due to differences in centralising the lens and the electronic chip (imager), and it appears as a tangential displacement of pixels that depends on the distance from the lens centre ρ. Equation (4.1) re-scales the pixel at the coordinates (x, y) by using the two radial distortion parameters k_1 and k_2, whereas (4.2) characterises the tangential distortion by two additional parameters ξ_1 and ξ_2 [142]. $[x,\ y]$ and $[x_c,\ y_c]$ are the two-dimensional coordinates for the fisheye lens and the projected image, respectively.

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} 1 + k_1\rho^2 + k_2\rho^4 & 0 \\ 0 & 1 + k_1\rho^2 + k_2\rho^4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \tag{4.1}$$



(a) Image by fisheye lens. (b) Undistorted image.

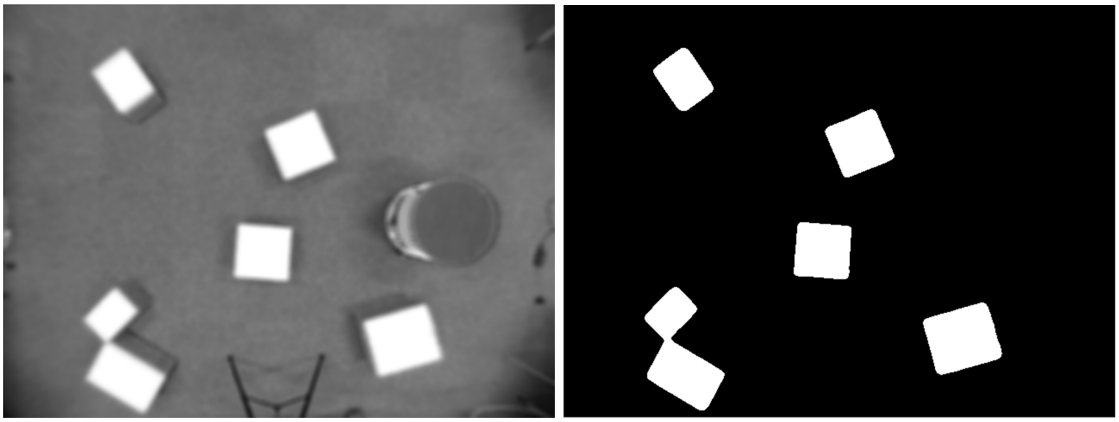
FIGURE 4.2: Calibration result.

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} x + 2\xi_1 y + \xi_2(\rho^2 + 2x^2) \\ y + 2\xi_2 x + \xi_1(\rho^2 + 2y^2) \end{bmatrix}. \tag{4.2}$$

Figure 4.1(b) shows a typical fisheye lens whose spherical shape is used to cover a wide angle of view. A particular point $q = \begin{bmatrix} q_x & q_y & q_z \end{bmatrix}^{\mathrm{T}}$ on the lens surface is projected onto the imager according to (4.3), and the distance from the centre is proportional to the angle from the viewing direction. Consequently, one can deduce that the view captured via this lens is circularly projected onto the imager as shown in Fig. 4.2(a).

$$\begin{bmatrix} q_x \\ q_y \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 + \rho\cos\vartheta & 0 \\ 0 & 1 + \rho\sin\vartheta \end{bmatrix} \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{4.3}$$

where $\vartheta = \arctan 2(q_y, q_x)$ $\rho = 2\varphi/\pi$ and $\varphi = \arctan 2(\sqrt{q_x^2 + q_y^2}, q_z)$. A calibration process is conducted to remove the distortions caused by the conventional camera as well as those caused by the attached fisheye lens [143]. In Fig. 4.2(a), the circular surface of the robot is changed into an elliptical shape due to fisheye lens distortions, and the obstacles closer to the lens edge appear to be smaller. Figure 4.2(b) shows the corresponding undistorted image after calibration; the figure shows that the obstacles are approximately normal in size regardless of their distance from the image centre and the circular shape of the robot's surface is perfectly formed.



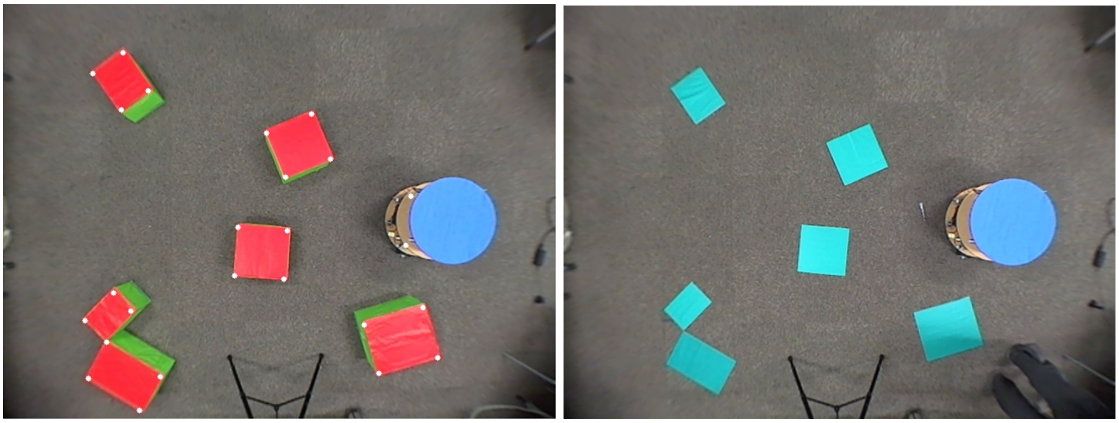
(a) Red channel. (b) Top-surface image.

FIGURE 4.3: Top surface extraction.

4.2.3 Corner Detection

The present work analyses the feasibility of the proposed method and represents a fundamental study; thus, all the obstacles are assumed to have the same height and their top surfaces have sharp corners. The top surfaces are coloured red, whereas the sides are shown in green to facilitate detection of the surface corners, thus allowing for the measurement of the distance between the obstacles and the generation of the configuration space. In general applications, although the obstacles have different shapes and heights, their top surfaces can be approximated as polygons and detected and distinguished based on a depth map generated by a stereo camera or a Microsoft Kinect. Moreover, higher-level image processing techniques should be used to address different lighting conditions.

An obstacle in the workspace can be represented by the corners of its top surface. To detect these corners, the undistorted image is decomposed into three colour channels (red, green and blue). In the red channel, the obstacles' top surfaces have bright pixels, whereas the sides and the ground are rendered in greyscale, as shown in Fig. 4.3(a). The top-surface image (Fig. 4.3(b)) can be filtered out by comparing the red channel with a threshold Ω, whereas the sides and the ground are represented by black pixels. Provided that the top surfaces are assumed to have sharp corners, it has been observed that the Harris corner detector based on eigenvalues is the best corner detection method to use [144]. A window measuring a x b is shifted along the top-surface image, and a 2 x 2 gradient variation matrix of the pixel values contained in the window is calculated. The two eigenvalues of this matrix are then computed, and the corner is found if and only if the eigenvalue is larger than a threshold δ, as shown in Fig. 4.4(a) (white points).



(a) Corner extraction. (b) Lower surface image.

FIGURE 4.4: Corner detection.

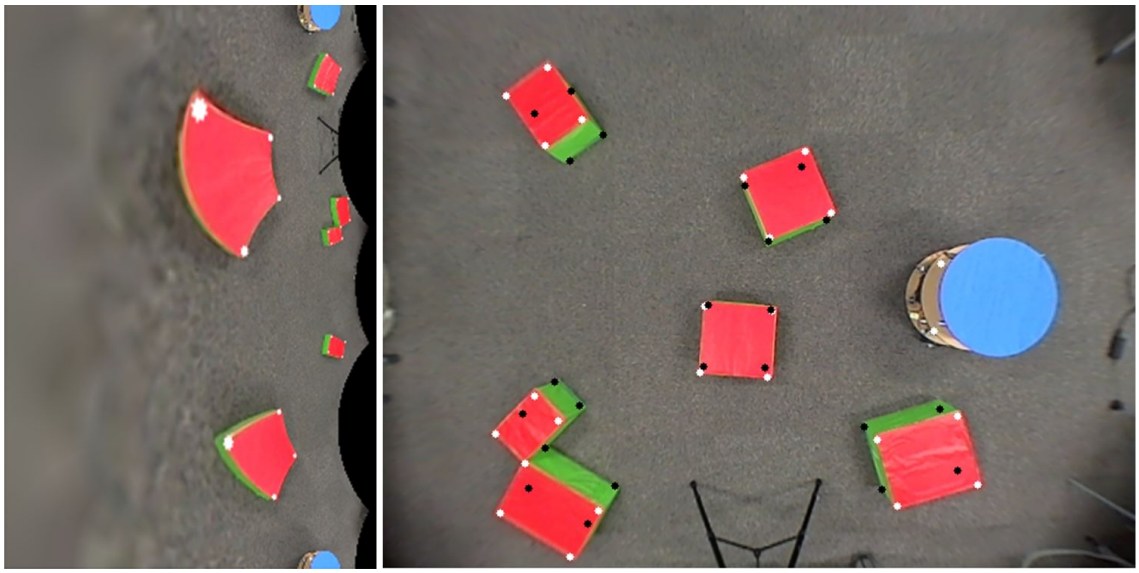
4.2.4 Corner Correction using Log-Polar Transform

Intuitively, the corners extracted from the top-surface image cannot be used to generate the configuration space because the height representation is altered radially, as shown in Fig. 4.4(a). Figure 4.4(b) highlights this issue by demonstrating the obstacles' lower surfaces, whose corners can be used to generate the configuration space. Thus, the top-surface corners must be matched with the corresponding lower corners. If perfect calibration were achieved, the height effect would vary proportionally with the distance from the centre regardless of the angular position. Hence, the matching process would displace only the corners towards the centre, making the log-polar transform the most appropriate approach for correcting the corner coordinates [145, 146].

The log-polar transform has been used for various applications in image processing. An image in a Cartesian coordinate system is exponentially sampled by a number of circular rings, which in turn are sampled by a fixed number of sectors. The coordinates of each sector are transformed into log-polar coordinates based on the angle ϑ and the radius ρ of the i^{th} circular ring. Thus, a set of pixels $I = [x, y]^{\mathrm{T}}$ in the Cartesian image plane is transformed into a set of pixels $\acute{I} = [\rho, \vartheta]^{\mathrm{T}}$ in the log-polar image, as shown in Fig. 4.5(a), by the following equation:

$$\acute{I}_{\rho,\vartheta} = \begin{bmatrix} \log\sqrt{(x^2 + y^2)} \\ \arctan\frac{y}{x} \end{bmatrix}. \tag{4.4}$$

The obstacle height Λ is assumed to be fixed so consequently, the $\log\rho$ coordinates of the top-surface corners are displaced by a fixed number of pixels λ. The displaced $\log\rho$



(a) Log-polar image. (b) Corrected corners.

FIGURE 4.5: Corner-matching process.

coordinates are transformed inversely into a Cartesian coordinate system to obtain the corrected corners, as highlighted by black points in Fig. 4.5(b), where corrected points nearly match the obstacles' lower surfaces. This relationship can be proved by focusing on the boundaries between the side surfaces (green areas) and the ground.

4.3 Trajectory Planning

4.3.1 Visibility Graph Construction

In this section, a description on how to process an input set of polygonal obstacles and construct a visibility graph is provided. A method to generate the configuration space, visibility graph and search for the shortest path from the start to goal positions is explained first. The next step is to replace the linear shortest path with piecewise Bézier curves to obtain a smooth trajectory.

Given a workspace with polygonal obstacles O_k, $k = 1, ..., m$, where m is the number of obstacles, r is robot radius and the desired safety distance from the robot to an obstacle is defined as d_c,let v_{hk} and e_{hk} be the h^{th} vertex and the h^{th} edge of an obstacle O_k, respectively, as shown in Fig. 4.6, where $h = 1$. All edges are offset by a distance $d = r + d_c$ and extended on both sides to find an intersection point v'_{hk}. For each convex vertex, a line segment $\overline{a_{hk}b_{hk}}$ orthogonal to a normal vector $\overrightarrow{v_{hk}c_{hk}}$ of magnitude d is constructed.

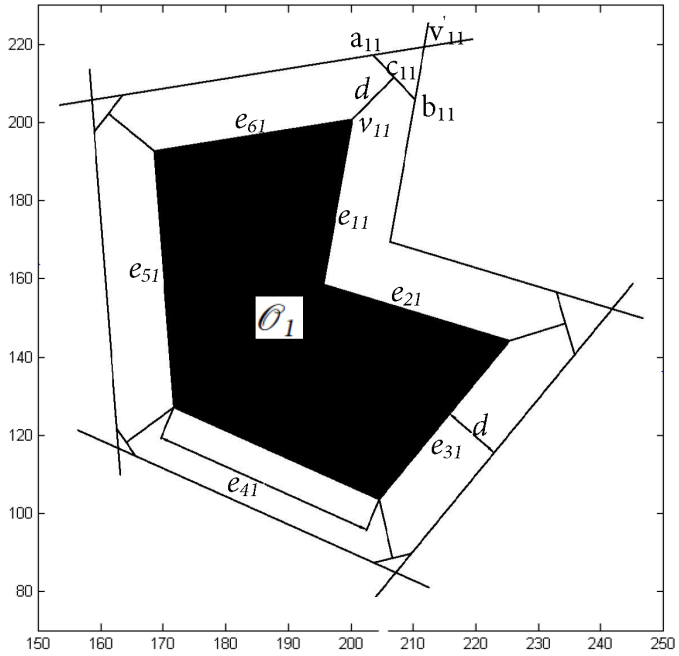


FIGURE 4.6: Configuration space construction.

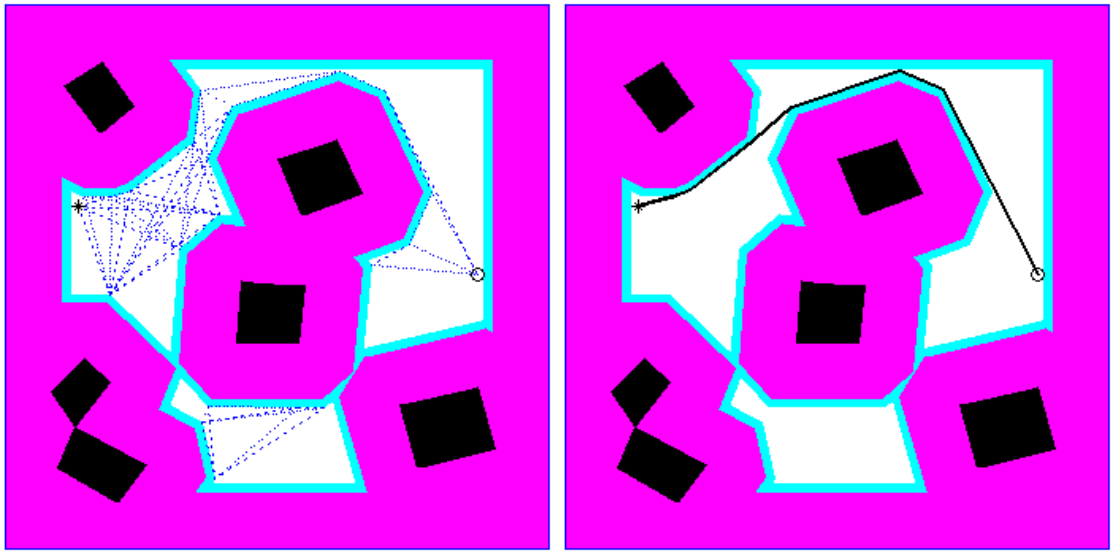
Points a_{hk} and b_{hk} are then used as nodes of O_k in the configuration space, whereas v'_{hk} is used for the concave vertex. The visibility graph is then constructed using a standard method as shown in Fig. 4.7(a).

The next step is to find the optimal path from the given start and goal positions. In this chapter, the word 'optimal' refers to the shortest Euclidean distance; however, one may apply other criteria such as minimum energy or traversal time. The A* algorithm, which features a heuristic function that makes it more efficient for node to node queries than other graph search algorithms is employed to find the shortest path [14]. The shortest path obtained here is normally a straight line with sharp turns (Fig. 4.7(b)); therefore, a smoothing algorithm is developed later in this study.

4.3.2 Smooth Trajectory Generation by Quintic Bézier Curves

Although the resulting path after smoothing is slightly longer than the one connected by straight lines, it is preferable for the actual mobile robot. Therefore, a smooth trajectory-generation algorithm using Bézier curves, as presented in chapter 3, is employed. The method is straightforward, easy to implement, and generates a smooth trajectory that can be tracked by autonomous mobile robots.

From the given visibility graph (Fig. 4.8), the nodes constructing the shortest path are first defined as the via points W_j, $j = 0, 1 \ldots, g$, where g is the final via point (goal



(a) Visibility graph. (b) Shortest path.

FIGURE 4.7: Example of visibility graph and shortest linear path.

position). The objective is to ensure that the trajectory passes through these nodes while improving curvatures to ensure smooth turning of the mobile robot. A unit vector $\hat{u}_l$, pointing in the direction of motion is introduced as follows:

$$\hat{u}_l = \frac{R_{\beta_j}\left[W_{j+1} - W_j\right]}{\left\|W_{j+1} - W_j\right\|}, \tag{4.5}$$

$$\hat{u}_0 = \frac{R_{\theta_0}\left[W_1 - W_0\right]}{\left\|W_1 - W_0\right\|}, \tag{4.6}$$

$$\hat{u}_g = \frac{R_{\theta_g}\left[W_j - W_{j-1}\right]}{\left\|W_j - W_{j-1}\right\|}, \tag{4.7}$$

$$R_\alpha = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}, \tag{4.8}$$

where $\hat{u}_0$ and $\hat{u}_g$ are unit vectors indicating the orientations of the robot at the start and goal positions, respectively, and γ_j is the inner angle of the bisector inscribed by W_{j-1}, W_j and W_{j+1} as shown in Fig. 4.8. R_α is the rotation matrix; θ_0 and θ_g are the orientations of the robot at the start and goal positions, respectively; and β_j is given by $\frac{\pi-\gamma_j}{2}$.

A quintic Bézier curve is inserted between two consecutive via points while ensuring C^2 continuity at the connecting point, thereby preserving the continuity of the entire

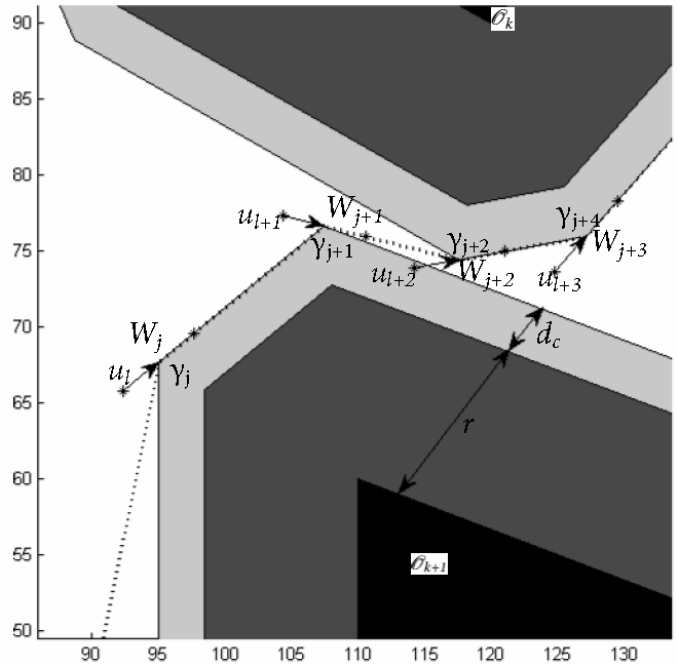


FIGURE 4.8: Variables for Bézier curve construction.

trajectory. From the definition of a Bézier curve, the quintic Bézier curve of the jth segment is given by

$$\begin{aligned} P(t)_j &= (1-t)^5 P_{0j} + 5t(1-t)^4 P_{1j} + 10t^2(1-t)^3 P_{2j} \\ &\quad + 10t^3(1-t)^2 P_{3j} + 5t^4(1-t) P_{4j} + t^5 P_{5j}. \end{aligned} \tag{4.9}$$

The start and the end points of each Bézier segment, P_{0j} and P_{5j}, are the via points W_j and W_{j+1}, respectively. The four inner points, P_{1j}-P_{4j}, are calculated from the initial and the final tangents (first derivatives), and the curvatures (second derivatives) of the curve. The first and the second derivatives of (4.9) at the start and end points of each Bézier segment are given by (4.10)-(4.13).

$$P^{'}(0)_j = 5(P_{1j} - P_{0j}) = \hat{u}_l d_c. \tag{4.10}$$

$$P^{'}(1)_j = 5(P_{5j} - P_{4j}) = \hat{u}_{l+1} d_c. \tag{4.11}$$

$$P^{''}(0)_j = 20(P_{0j} - 2P_{1j} + P_{2j}). \tag{4.12}$$

$$P^{''}(1)_j = 20(P_{3j} - 2P_{4j} + P_{5j}). \tag{4.13}$$

Equations (4.10) and (4.11) are solved directly from the initial and final tangents of the curve. Here, the initial and the final tangential vectors are chosen to be equal to $\hat{u}_l d_c$ and $\hat{u}_{l+1} d_c$, respectively, to ensure that the path is within the specified region because a Bézier curve is always in its convex hull.

The second derivatives, which determine the curvature in (4.12) and (4.13), require a further technique for their solution. Therefore, the curvatures of the quintic Bézier curve are generated using a cubic Bézier curve. Because the cubic Bézier is not C^2 continuous, it is not applied directly. Instead, the quintic Bézier curve is transformed into a cubic Bézier curve while preserving its continuity property. The cubic Bézier curve is represented as

$$Q(t) = (1-t)^3 b_0 + 3t(1-t)^2 b_1 + 3t^2(1-t)b_2 + t^3 b_3, \tag{4.14}$$

where $Q(t)$ and b_i are two-dimensional cubic Bézier curve and control points, respectively. The first and the second derivatives of (4.14) at the start and end points of the curve are given by

$$b^{'}(0) = 3(b_1 - b_0), \tag{4.15}$$

$$b^{'}(1) = 3(b_3 - b_2), \tag{4.16}$$

$$b^{''}(0) = 6(b_0 - 2b_1 + b_2), \tag{4.17}$$

$$b^{''}(1) = 6(b_1 - 2b_2 + b_3), \tag{4.18}$$

As demonstrated by (4.15) to (4.18), if the initial and final tangents are computed using (4.10) and (4.11), respectively, there are four and four unknowns, and the second derivative of the cubic Bézier curve is obtained easily. Here, this idea is extended to quintic Bézier curves. Assuming zero curvature at the first and the last via points, the curvature at the intersection of two Bézier segments $\overline{W_j W_{j+1}}$ and $\overline{W_{j+1} W_{j+2}}$ is considered. The quintic Bézier segments are formulated regarding the cubic Bézier curve by redefining the control

points cp as follows:

$$cp = \begin{cases} W_{j-1},\ W_{j-1} + \frac{1}{3}\hat{u}_{l-1}d_c,\ W_j - \frac{1}{3}\hat{u}_l d_c,\ W_j, & \text{for } Q(t)_j \\ W_j,\ W_j + \frac{1}{3}\hat{u}_l d_c,\ W_{j+1} - \frac{1}{3}\hat{u}_{l+1}d_c,\ W_{j+1}, & \text{for } Q(t)_{j+1} \end{cases}, \tag{4.19}$$

where $Q(t)_j$ and $Q(t)_{j+1}$ are Bézier curves of the segments $\overline{W_j W_{j+1}}$ and $\overline{W_{j+1} W_{j+2}}$, respectively. Therefore, the two consecutive Bézier segments are given by

$$\begin{aligned} Q(t)_j &= (1-t)^3 W_{j-1} + 3t(1-t)^2 \left(W_{j-1} + \frac{1}{3}\hat{u}_{l-1}d_c \right) \\ &\quad + 3t^2(1-t)\left(W_j - \frac{1}{3}\hat{u}_l d_c \right) + t^3 W_j, \end{aligned} \tag{4.20}$$

$$\begin{aligned} Q(t)_{j+1} &= (1-t)^3 W_j + 3t(1-t)^2 \left(W_j + \frac{1}{3}\hat{u}_l d_c \right) + \\ &\quad 3t^2(1-t)\left(W_{j+1} - \frac{1}{3}\hat{u}_{l+1}d_c \right) + t^3 W_{j+1}. \end{aligned} \tag{4.21}$$

By substituting (4.20) and (4.21) into (4.15)-(4.18),

$$P''(1)_j = P''(0)_{j+1} = \frac{(A+B)}{2} \tag{4.22}$$

is obtained, where

$$\begin{aligned} A &= 6W_j + 2P'(0)_j + 4P'(1)_j - 6W_{j+1}, \\ B &= -6W_{j+1} - 2P'(0)_{j+1} - 4P'(1)_{j+1} + 6W_{j+2}. \end{aligned}$$

From the via points of the shortest path and by substituting (4.22) into (4.12) and (4.13), the six control points required for the quintic Bézier segment are obtained as

$$\begin{aligned} &P_{0j} = W_j,\ \ P_{1j} = \frac{\hat{u}_l d_c}{5} + W_j, \\ &P_{2j} = \frac{P''(0)_j}{20} + 2P_{1j} - P_{0j},\ \ P_{3j} = \frac{P''(1)_j}{20} + 2P_{4j} - W_{j+1}, \\ &P_{4j} = W_{j+1} - \frac{\hat{u}_{l+1} r_l}{5},\ \ P_{5j} = W_{j+1}. \end{aligned} \tag{4.23}$$

Equation (4.23) is substituted into (4.9) to obtain the of each path segment from the start to the goal positions. This method produces a smooth trajectory, as shown in Fig. 4.9,

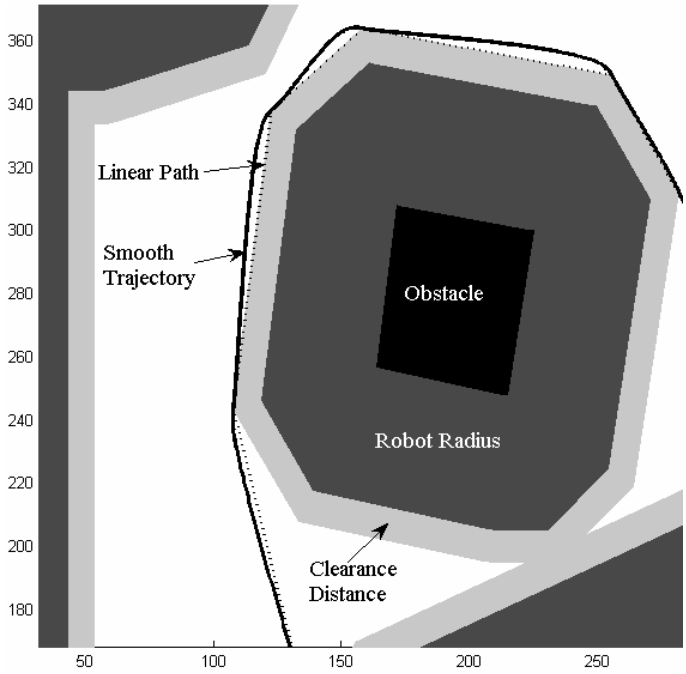
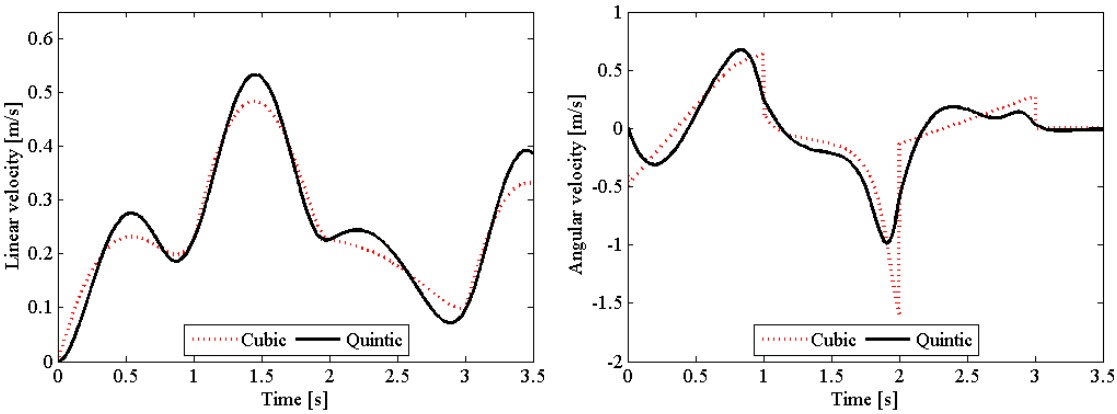


FIGURE 4.9: Smooth Trajectory and Linear Path.

which is a magnified portion of the workspace.

4.3.3 Comparison with Cubic Bézier Trajectories

Although a single Bézier curve is infinitely differentiable, a long trajectory requires higher-order Bézier curves. These curves are complex to calculate and numerically unstable, leading to oscillations in the resulting trajectory [93]. Instead, piecewise Bézier curves are used, and therefore, the continuity of the generated trajectory depends on the order of the Bézier curves. Cubic Bézier curves are among the most commonly used curves to generate parametric trajectories. Implementation is simple because, for given via points, only the first derivatives at the start and the end points of the Bézier curve are required to obtain all control points ((4.14) to (4.16)). Quintic Bézier curves require both the first and the second derivatives ((4.9) to (4.13)). However, the trajectory generated by the piecewise cubic Bézier curves lacks the curvature continuity required for mobile robots. For further elaboration, a simulation based on both the quintic and the cubic Bézier curve methods is conducted on the same course and and under the same conditions. The results of both the linear and angular velocities are shown in Fig. 4.10; although the linear velocity profiles are relatively the same (Fig. 4.10(a)), major differences can be observed in the angular velocities (Fig. 4.10(b)). The proposed method provides a smooth profile throughout the course, whereas the conventional one shows discontinuities at corners around 1 s, 2 s and



(a) Linear Velocity. (b) Angular Velocity.

FIGURE 4.10: Velocity profiles of quintic and cubic Bézier curves.

3 s. In practical situations, the robot is constrained by hardware limitations that bind the velocity and the acceleration. Therefore, the discontinuities in the angular velocity may deteriorate the tracking performance of the robot.

4.4 Experimental Setup and Results

4.4.1 Experimental Setup

A workspace measuring 2.5 x 2.5 m^2 was used to evaluate the performance of the proposed method. A fisheye lens with an 180 deg viewing angle was attached to a webcam fixed at the height of 1.58 m and positioned at the centre of the workspace. All obstacles had the same height of 19 cm, and their top surfaces were coloured red, while their sides were coloured green, as shown in Fig. 4.2. The robot was placed in each environment at different start positions, whereas the operator assigned the goal positions. A laptop computer equipped with an Intel Core 2 Duo Processor (1.40 GHz) and 4 GB RAM and running the 32-bit version of the Microsoft Windows 7 operating system was used for the experiment.

Calibration was performed to obtain an accurate representation of the obstacles, especially at the lens edges. An equivalent representation of the obstacles' lower surfaces was obtained for each workspace (as in Fig. 4.4(b)) to evaluate the accuracy of the corner correction method based on the log-polar transform. The top-surface image was extracted at a threshold Ω of 200-pixel intensity and scanned in a window measuring 5 x 5 pixels to detect the corners. By using the Harris corner detector [144], the threshold of eigenvalues

(δ) to determine whether the window contains a corner was found to be 0.00342. The log ρ coordinates of the corners were displaced in 3 pixels, whereas the robot top-surface was displaced in 8 pixels.

4.4.2 Experimental Results

Figures 4.11 to 4.13 demonstrate three workspaces constructed with different obstacle distributions. Each figure has four sub-figures. Sub-figure i shows the obstacle corners detected using the corresponding top-surface image, whereas sub-figure ii shows the lower-surface image created to observe the output of the corner-matching operation. The corrected angles and the robot position are represented by black points and a black circle, respectively. Sub-figure iii shows a visibility graph of the workspace, where black, magenta/grey and cyan/light grey represent the obstacles, area equivalent to the robot's radius and safety distance, respectively. The marks o and $*$ represent the start and the goal positions, respectively. Sub-figure iv shows the smooth optimal trajectory generated using the proposed method.

The log-polar transform successfully and sufficiently corrected the obstacles and the robot centre coordinates. Regardless of the angular position, the height representation is related to the distance from the image centre. Sub-figure ii shows the corrected corners extracted from the top-surface images of each workspace in sub-figure i, where very high matching accuracy was achieved. Moreover, the obstacles whose side surfaces do not appear in the middle of the image were corrected as well. As shown in sub-figure iv, smooth optimal trajectories were generated using the proposed method. In workspace 3 (Fig. 4.13), the obstacles were so close to the robot that no feasible path was found. The resulting trajectory was optimal and smooth enough to be tracked by the mobile robot, as shown in Fig. 4.10. In each case, the average computation time was 1.17 s, as given in Table 4.1. This computation time is faster than, for example, that indicated in [147], in which the computation times were reported to be 1.74 s and 1.09 s for 2 and 3 obstacles, respectively, by using a personal computer (CPU: 1.83 GHz).

TABLE 4.1: Computation time for generating trajectory.

	Fig. 4.11	Fig. 4.12	Fig. 4.13	Average
Computation time (s)	1.174	1.171	1.165	1.170

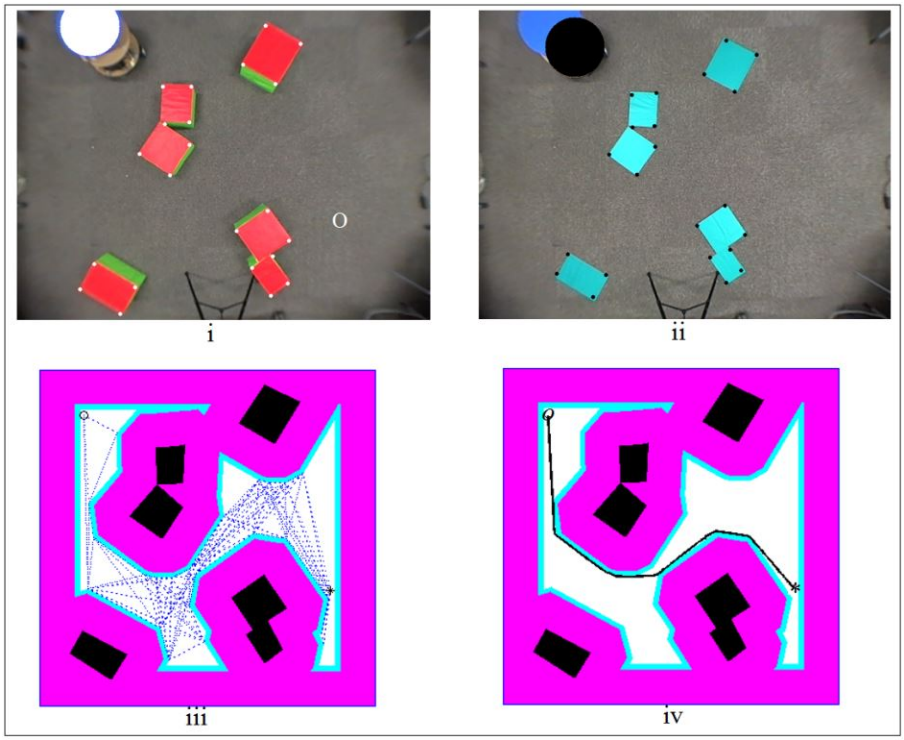


FIGURE 4.11: Experimental results for workspace 1.

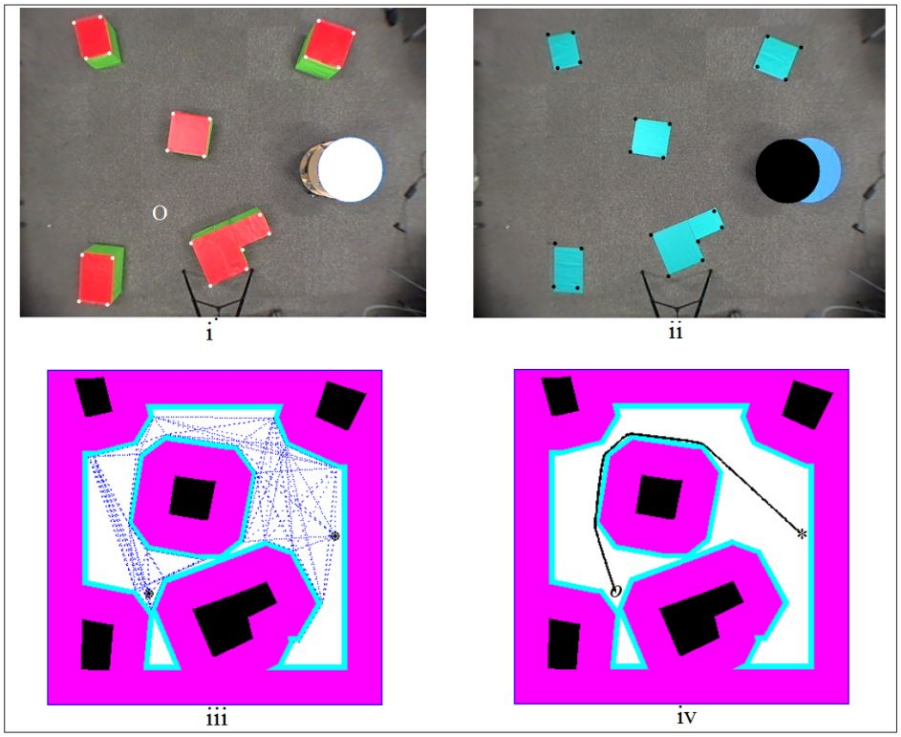


FIGURE 4.12: Experimental results for workspace 2.

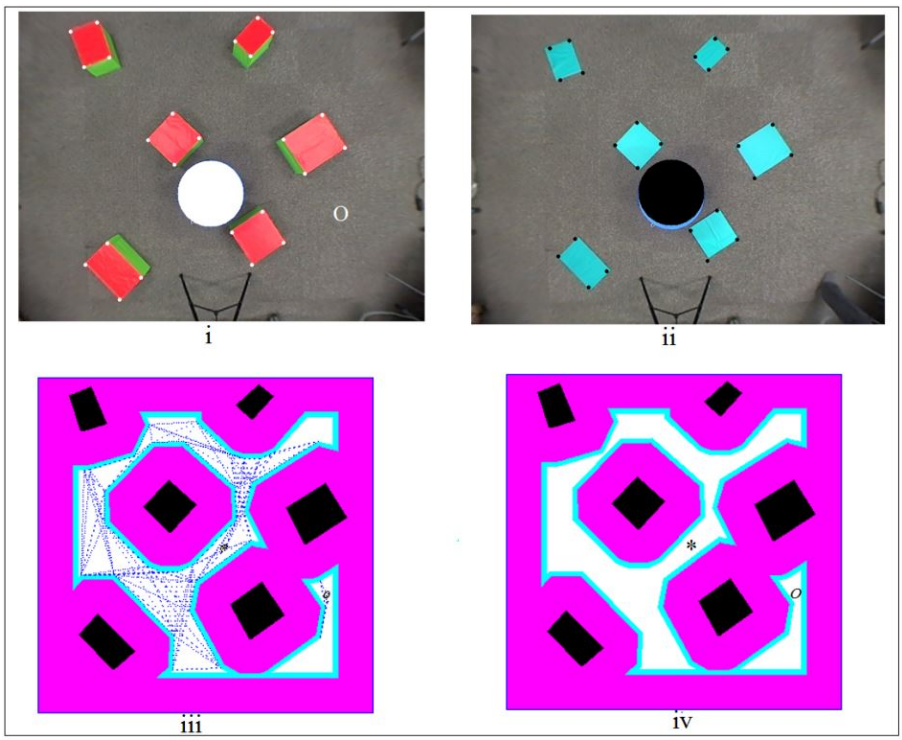


FIGURE 4.13: Experimental results for workspace 3.

4.4.3 Discussion

Because the actual representation of the obstacles' lower surfaces is required to find the shortest path, the calibration must be done perfectly. Furthermore, the corners of the obstacles' lower surfaces must be extracted accurately to calculate the shortest path. One approach is to detect the side surfaces and then shift the corners accordingly. However, this requires an additional colour detection technique because the image might be affected by noise. Moreover, because the matching process is applied independently to each top-surface corner, an independent error will occur accordingly. Therefore, the obstacles' lower surfaces will not adopt an actual geometrical shape. Although one may suggest applying the log-polar transform to the top-surface image and then detecting the corners, this approach may introduce additional distortions into the transformed image. These distortions may change the obstacles' shapes, making the detection of corners more challenging and inaccurate. The proposed method has proved to be sufficient to correct obstacles and the robot centre coordinates.

As shown in Figs. 4.11-4.13 (sub-figures iii and iv), the classical approach to constructing a configuration space was not applied. In the classical approach, the configuration space of a circular robot is constructed by enlarging polygonal obstacles by a constant width equal to the radius of the robot plus a specific safety margin. Likewise,

the configuration space of a non-circular robot that performs translation and rotation is normally constructed in a similar manner to reduce complexities. To obtain a visibility graph, arcs are approximated by many straight line segments or critical points, which lead to a large number of nodes and, in turn, a longer computation time. Therefore, the method is not effective for real-time applications.

The proposed method generates a configuration space by modifying the classical method. At corners, the configuration space of an obstacle is approximated by one line segment, as shown in Fig. 4.6. The resulting configuration space has fewer nodes compared to that in the classical method, and hence, the computation time is reduced significantly. Although there are many methods for generating smooth trajectories off-line, in real-time applications, environmental information is limited; thus, such methods cannot be applied directly. The proposed method generates smooth and optimal trajectories based on the goal position, safety margin and via points, which are calculated from the environmental top-view image.

Although the proposed algorithm offers the benefits described above, as stated in section 4.2.3, the current work is considered a fundamental study, in which implementation of the proposed algorithm is limited to obstacles of the same height and colour and uniform lighting conditions. In general applications, although obstacles have different shapes and heights, their top surfaces can be approximated as polygons and detected and distinguished by using the depth map generated by a stereo camera or a Microsoft Kinect. Moreover, higher-level image processing techniques should be used to address different lighting conditions.

4.5 Summary

A method for generating smooth and obstacle-avoidance motion trajectories for wheeled mobile robots is proposed. A ceiling camera equipped with a fisheye lens is used to capture a wide view of a given workspace. Significant use of the log-polar transform has been proven to be effective in eliminating the height effect of obstacles regardless of their distances from the centre of the workspace. The configuration space is simplified to reduce the number of nodes, and hence, the computation time is reduced significantly to allow for real-time applications. The searched optimal path is replaced by quintic Bézier curves, which are connected smoothly to obtain a curvature-continuous trajectory. The method is relatively simple and easy to implement, and its average computation time is

1.17 s on a PC (CPU: 1.4 GHz), which makes it feasible for real-time applications. The experimental results demonstrate the effectiveness of the proposed method.

Chapter 5

Smooth Trajectory Generation and Nonlinear Friction Compensation for Feed Drive Contouring Control

Generation of motion trajectories is a crucial task in feed drive control for efficient manufacturing and energy saving. They have to be smooth enough to be traversable without violating systems constraints. Literary, a number of methods, such as spline parametric trajectories have been suggested for smooth trajectories generation. However, there are still many challenges left, for example, in order to smoothly track a spline parameterised trajectory, the reference velocity as well as position have to be optimally planned, such that low-velocity should be used at the start and the end of the motion, as well as in all areas with high curvature. Additionally, mechanical systems experience friction forces which vary nonlinearly with velocities. Therefore, on implementing a smooth velocity profile, an appropriate friction compensator has to be considered to cancel out the effect of friction forces. In this study, methods for generating smooth motion trajectories using quintic Bézier curves and smooth velocity profiles based on the altered bang-bang approach are proposed. A contouring controller with a feed forward friction compensator is applied in order to smoothly track the designed trajectory by cancelling out the effect of friction forces. The performance of the proposed method was experimentally evaluated by comparing it with a conventional approach. Results have shown that the tracking performance can be greatly increased by reducing the average contour error by about 48 % without violating systems constraints.

5.1 Introduction

Due to the rapid growth of science and technology and the demand of precise products, computer numerical control (CNC) machines are widely used to manufacture complex components [148, 149]. In order to achieve a desired manufacturing quality, motion trajectories must be traversable without violation of the systems constraints such as a permissible acceleration and jerk. Generally, the trajectories have to be at least second order differentiable in order to achieve a continuous velocity and acceleration [150, 151]. Although linear trajectory interpolation is commonly used, the motion has to stop between each segment, otherwise, the system must undergo an infinite acceleration of which is practically impossible. However, motion stops consume time and power, and bring unnecessary wear on the system parts [5].

For improvement of motion trajectories, a number of methods have been proposed in the literature and most of them focus on smoothing the linear interpolated tool-path points using curve fitting technique. In this case, cubic, quadratic, and quintic spline curves are widely utilized [90, 91, 152]. For densely tool-path points, curve fitting technique exhibit oscillations in the trajectory because high-order spline curves are numerically unstable [93]. On the other hand, parametric spline curves are used to smoothly interpolate the linear tool-path points or blending corners and have proven to provide sufficiently smooth trajectories for CNC machining [94, 95].

In order to smoothly track a spline parameterised trajectory, reference velocity has to be smoothly planned, such that low-velocity values can be used during the motion start and end, and in all areas with high curvatures. Since it is well known that mechanical systems cannot instantly accelerate to high velocities and that the velocity is inversely proportional to the curvature, smooth reference velocity ensures that the motion acceleration and jerk are within the permissible range.

In addition to the velocity profile matter, mechanical systems experience friction forces which vary nonlinearly with velocities. Therefore, on implementing a smooth velocity profile, an appropriate friction compensator has to be considered to cancel out the effect of friction forces. There are abundant studies on friction compensation which aim to improve the motion performance of feed drive systems, particularly machine tools [153–155]. In [153], it was assumed that friction in feed drives of machine tool comes from many friction sources with complex nonlinear properties. The main focus was particularly on the property of the lead screw drive under insufficient lubrication

condition. The proposed nonlinear friction model includes the typical Coulomb-viscous friction and a nonlinear sinusoidal friction term for describing the lead screw property. The proposed model was experimentally verified and showed a satisfactory improvement compared to the conventional friction model.

In this study, a method for generating smooth motion trajectories using quintic Bézier curves and smooth velocity profiles based on the altered bang-bang approach is proposed. A contouring controller with a feed forward friction compensator is applied in order to smoothly track the designed trajectory by cancelling out the effect of friction forces. Finally, experiments are conducted to verify the effectiveness of the proposed method.

5.2 Bézier Smoothing Algorithm

A three-dimensional Bézier Curve (BC) of order n is represented as

$$P(\tau) = \sum_{k=0}^{n} \left(\frac{n!}{k!(n-k)!} \right) \tau^k (1-\tau)^{n-k} P_k, \quad \tau \in [0,1],$$
$$k = 0, 1, 2, \ldots, 5, \tag{5.1}$$

where $P(\tau) = [x(\tau),\ y(\tau),\ z(\tau)]^T$ and P_k are the three-dimensional Bézier curve and the control points, respectively. As shown in Fig. 5.1, the initial reference trajectory is defined in G-code fashion and smoothened by BCs, where the G-code points G_i, $i = 1, 2, \ldots$ become the inputs of the smoothing algorithm. A quintic BC is inserted between the two consecutive segments at positions B_{0_i} and B_{5_i} while ensuring curvature continuity. γ and μ_i are the tolerance in the contour error and tangential unit vectors corresponding to the motion direction, respectively. In order to satisfy the C^2 continuous condition, the ending tangent and curvature of the i^{th} segment must be similar to the starting tangent and curvature of the i^{th} Inserted Bézier Curve (IBC), respectively. Therefore, each segment is transformed into a quintic BC and smoothly connected with the inserted curve.

From (5.1), the BC of the i^{th} segment and the i^{th} IBC are as follows:

$$\begin{aligned} P(\tau)_i &= (1-\tau)^5 P_{0i} + 5\tau(1-\tau)^4 P_{1i} + 10\tau^2(1-\tau)^3 P_{2i} \\ &\quad + 10\tau^3(1-\tau)^2 P_{3i} + 5\tau^4(1-\tau) P_{4i} + \tau^5 P_{5i}, \\ B(\tau)_i &= (1-\tau)^5 B_{0i} + 5\tau(1-\tau)^4 B_{1i} + 10\tau^2(1-\tau)^3 B_{2i} \\ &\quad + 10\tau^3(1-\tau)^2 B_{3i} + 5\tau^4(1-\tau) B_{4i} + \tau^5 B_{5i}, \end{aligned} \tag{5.2}$$

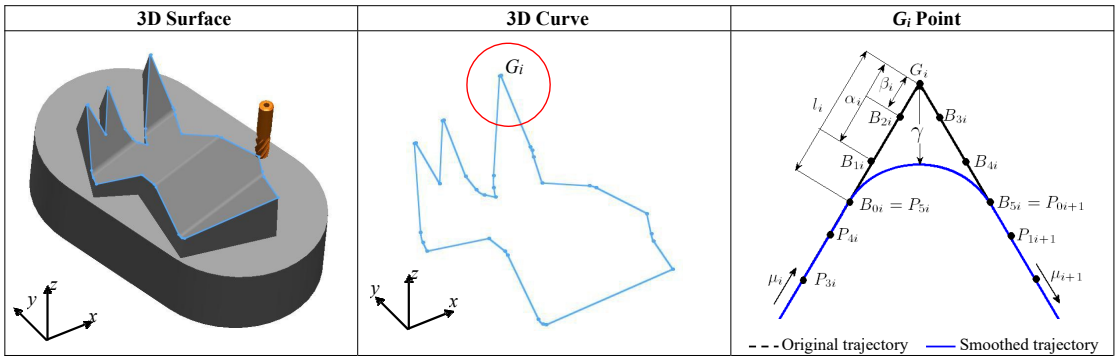


FIGURE 5.1: Proposed trajectory smoothing strategy.

where $P(\tau)_i$ and $B(\tau)_i$, P_{k_i} and B_{k_i}, and τ are the three-dimensional BC for the i^{th} segment and the i^{th} IBC, control points, and the parameterisation variable, respectively. The first and second derivatives with respect to τ for each BC in (5.2) are respectively given by

$$\begin{aligned} P^{'}(0)_i &= 5(P_{1i} - P_{0i}), \quad P^{'}(1)_i = 5(P_{5i} - P_{4i}), \\ P^{''}(0)_i &= 20(P_{0i} - 2P_{1i} + P_{2i}), \\ P^{''}(1)_i &= 20(P_{3i} - 2P_{4i} + P_{5i}), \end{aligned} \tag{5.3}$$

and

$$\begin{aligned} B^{'}(0)_i &= 5(B_{1i} - B_{0i}), \quad B^{'}(1)_i = 5(B_{5i} - B_{4i}), \\ B^{''}(0)_i &= 20(B_{0i} - 2B_{1i} + B_{2i}), \\ B^{''}(1)_i &= 20(B_{3i} - 2B_{4i} + B_{5i}). \end{aligned} \tag{5.4}$$

To satisfy the required continuity, the following must be satisfied:

$$P_{5i} = B_{0i}, \quad P^{'}(1)_i = B^{'}(0)_i, \quad P^{''}(1)_i = B^{''}(0)_i. \tag{5.5}$$

Allocation of the control points P_{0i} to P_{5i} and B_{0i} to B_{5i} is done based on the altered concept in [94] such that, for the IBC they are placed as

$$\begin{aligned} B_{0i} &= G_i - l_i\mu_i, \quad B_{1i} = G_i - \alpha_i l_i \mu_i, \\ B_{2i} &= G_i - \beta_i l_i \mu_i, \quad B_{3i} = G_i + \beta_i l_i \mu_{i+1}, \\ B_{4i} &= G_i + \alpha_i l_i \mu_{i+1}, \quad B_{1i} = G_i + l_i \mu_{i+1}, \end{aligned} \tag{5.6}$$

where α_i and β_i are the fractions of the length l_i, and they are used to design the curvature

of the IBC. Note that if the curvature is zero at the initial or final control points of the IBC, i. e. , if $2\alpha_i - \beta_i = 1$, the corresponding segments can be linearly interpolated. From (5.3) to (5.6), the control points for each segment become

$$\begin{aligned}
&P_{0i} = G_{i-1} + l_{i-1}\mu_i, \quad P_{1i} = G_{i-1} + l_{i-1}\mu_i(2-\alpha_i),\\
&P_{2i} = G_{i-1} + l_{i-1}\mu_i(\beta_i - 4\alpha_i + 4),\\
&P_{3i} = G_i - l_i\mu_i(\beta_i - 4\alpha_i + 4),\\
&P_{4i} = G_i - l_i\mu_i(2-\alpha_i), \quad P_{5i} = G_i - l_i\mu_i.
\end{aligned} \tag{5.7}$$

Since the proposed trajectory generation algorithm induces a geometrical error γ at the corner, the algorithm must guarantee that γ is within the predefined tolerance of the contour error [94]. Due to the symmetrical nature of the IBC, the maximum geometrical error occurs at the middle point of the curve, i. e. at $\tau = 0.5$ as follows:

$$\gamma = \|G_i - B(0.5)_i\|. \tag{5.8}$$

From (5.2) and (5.6), the point $B(0.5)_i$ is found by

$$B(0.5)_i = G_i + \frac{l_i}{32}(1 + 5\alpha_i + 10\beta_i)(\mu_{i+1} - \mu_i). \tag{5.9}$$

From (5.8) and (5.9), the length l_i is calculated as

$$l_i = \frac{32\gamma}{(1 + 5\alpha_i + 10\beta_i)\|\mu_i - \mu_{i+1}\|}. \tag{5.10}$$

Since μ_i and μ_{i+1} are unit vectors, (5.10) can be written as

$$l_i = \frac{32\gamma}{(1 + 5\alpha_i + 10\beta_i)\sqrt{2 - 2\cos\theta_i}}, \tag{5.11}$$

where θ_i is the inclination between the corresponding linear segments. The values of α_i and β_i are found by considering the optimal condition of the following curvature κ_i of the IBC:

$$\kappa_i = \frac{\left\|B'(\tau)_i B''(\tau)_i\right\|}{\|B'(\tau)_i\|^3}. \tag{5.12}$$

The objective is to minimize the curvature extrema so as to achieve the maximum possible cornering speed.

5.3 Smooth Velocity Transition

For velocity planing, in many cases a piecewise constant velocity can be assumed. In order to ensure a continuous overall velocity, a smooth transition between the sections with constant velocity has to be implemented. Due to physical limitations of the system and/or comfort reasons, limits on the acceleration a and the jerk j have to be met. In the following section, a bang-bang approach for the jerk with acceleration limits is explained. It is assumed that the jerk is limited to the area $j \in (-j_{\max}, j_{\max})$ and the acceleration to $a \in (-a_{\max}, a_{\max})$. As stated above, a bang-bang approach for the jerk is introduced which means that the jerk either takes its maximum or minimum value. This leads to a time optimal velocity transition under the jerk constraint. However an additional acceleration constraint might be violated in the bang-bang approach for higher changes in velocity. Therefore, a second approach is proposed, where the acceleration is limited and therefore the jerk is set to zero after reaching the acceleration limit and before reducing the acceleration again. This is called bang-bang approach with acceleration limit form here on. Note that here a velocity v greater or equal to zero at all times t is assumed.

5.3.1 Bang-Bang Approach Without Acceleration Limitation

As stated above this approach is characterized by switching the jerk only between the minimum value $-j_{\max}$ and the maximum value $j_{\max}$. Since (here) only the velocity transition from $v(t_0)$ to $v(t_\mathrm{f})$ is considered, the following two sections are considered:

$$j(t) = \begin{cases} j_{\max}, & \forall t_0 \leq t < \frac{t_\mathrm{f}-t_0}{2}, \\ -j_{\max}, & \forall \frac{t_\mathrm{f}-t_0}{2} \leq t < t_\mathrm{f}. \end{cases} \tag{5.13}$$

Using the Heaviside function $H(t)$, (5.13) can be written in closed form

$$\begin{aligned} j(t) =& H(t-t_0) j_{\max} - 2H\left(t - \frac{t_\mathrm{f}-t_0}{2}\right) j_{\max}, \\ & \forall t \in \left[t_0,\ t_f\right]. \end{aligned} \tag{5.14}$$

Note that the last term only ensures that the jerk returns to zero for $t > t_\mathrm{f}$ and does not contribute to the equation, if only $t \in [t_0, t_\mathrm{f}]$ is considered. From here on it is assumed that

$t_0 = 0$ and $t_f = 2\Delta T$, leading to

$$j(t) = H(t)j_{max} - 2H(t-\Delta T)j_{max}. \tag{5.15}$$

The acceleration a, velocity v and position s can be derived by integration:

$$a(t) = j_{max}\left\{H(t)t - 2H(t-\Delta T)(t-\Delta T)\right\}, \tag{5.16}$$

$$v(t) = v(t_0) + \frac{1}{2}j_{max}\left\{H(t)t^2 - 2H(t-\Delta T)(t-\Delta T)^2\right\}, \tag{5.17}$$

$$\begin{aligned} s(t) &= s(t_0) + v(t_0)t \\ &\quad + \frac{1}{6}j_{max}\left(H(t)t^3 - 2H(t-\Delta T)(t-\Delta T)^3\right). \end{aligned} \tag{5.18}$$

As can be seen from (5.16–5.18), $a(t_0) = a(t_f) = 0$ holds. The final velocity $v(t_f) = v_f$ of the velocity transition is known in advance and leads to

$$\Delta v = v_f - v(t_0) = j_{max}\Delta T^2. \tag{5.19}$$

From this ΔT and the actual time of the transition duration $t_f - t_0$ are calculated by

$$\Delta T = \sqrt{\frac{\Delta v}{j_{max}}}, \tag{5.20}$$

$$t_f - t_0 = 2\Delta T = 2\sqrt{\frac{\Delta v}{j_{max}}}. \tag{5.21}$$

The maximum acceleration is reached at $t = \Delta T$:

$$\max a(t) = a(\Delta T) = j_{max}\Delta T = j_{max}\sqrt{\frac{\Delta v}{j_{max}}} = \sqrt{\Delta v\, j_{max}}. \tag{5.22}$$

The path $\Delta s = s(t_f) - s(t_0)$ travelled during transition is

$$\begin{aligned} \Delta s &= v(t_0)(2\Delta T) + \frac{1}{6}j_{max} \\ &\quad \left\{H(2\Delta T)(2\Delta T)^3 - 2H(2\Delta T - \Delta T)(2\Delta T - \Delta T)^3\right\}, \\ &= 2v(t_0)\sqrt{\frac{\Delta v}{j_{max}}} + \Delta v\sqrt{\frac{\Delta v}{j_{max}}}. \end{aligned} \tag{5.23}$$

An example for the values given in table 5.1 (a) is shown in Fig. 5.2. As can be seen from

TABLE 5.1: Values for exemplary transitions.

(a) Example 1.

Variable	Value
$v(t_0)$	$0\,\mathrm{mm\,s^{-1}}$
$v(t_f)$	$8\,\mathrm{mm\,s^{-1}}$
j_{max}	$1\,\mathrm{mm\,s^{-3}}$
a_{max}	$3\,\mathrm{mm\,s^{-2}}$

(b) Example 2.

Variable	Value
$v(t_0)$	$0\,\mathrm{mm\,s^{-1}}$
$v(t_f)$	$10\,\mathrm{mm\,s^{-1}}$
j_{max}	$2\,\mathrm{mm\,s^{-3}}$
a_{max}	$3\,\mathrm{mm\,s^{-2}}$

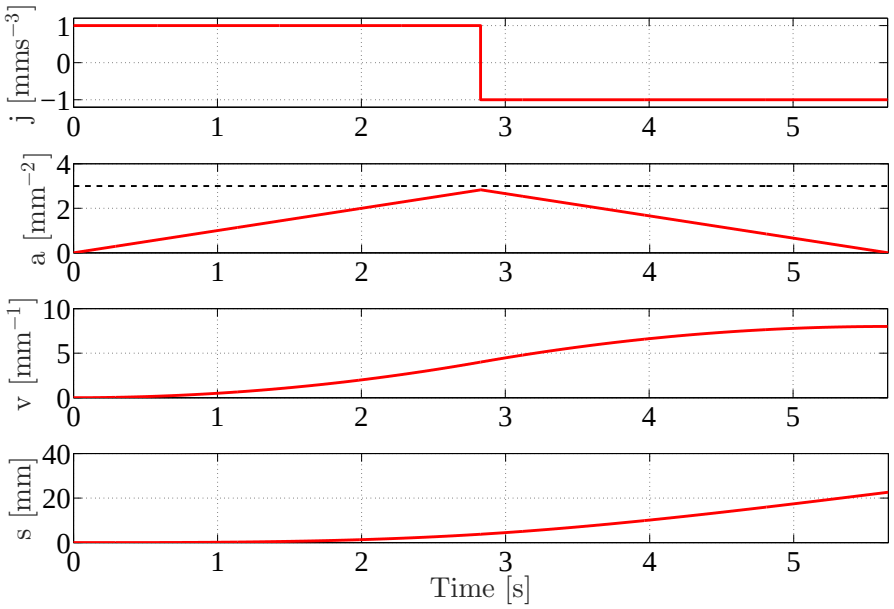


FIGURE 5.2: Velocity transition calculated by the bang-bang approach for table 5.1 (a).

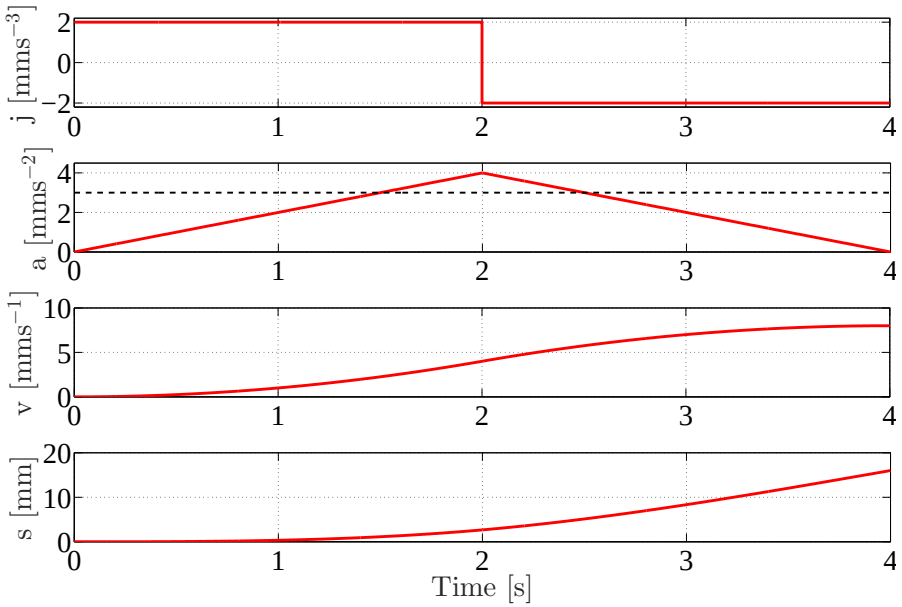


FIGURE 5.3: Velocity transition calculated by the bang-bang approach for table 5.1 (b).

this figure, the velocity transition is achieved, while obeying the jerk and acceleration limits. However in an example with higher maximum jerk, as given in table 5.1 (b), the maximum acceleration is not obeyed anymore, as shown in Fig. 5.3. Although the velocity is reached considerably faster, the acceleration peak is at $4\,\mathrm{mm\,s^{-2}}$ which is higher then the $3\,\mathrm{mm\,s^{-2}}$ maximum value defined in table 5.1 (b). To effectively limit the acceleration, another necessary approach is shown in the next section. The achievable final velocity depending on the maximum jerk and the available distance can be calculated by solving the following cubic equation which follows directly from (5.23):

$$0 = v_\mathrm{f}^3 + v_\mathrm{f}^2 v(t_0) - 3 v_\mathrm{f} v^2(t_0) + v^3(t_0) - \Delta s^2 j_\mathrm{max}. \tag{5.24}$$

5.3.2 Bang-Bang Approach with Acceleration Limitation

In this approach the third section with $j = 0\,\mathrm{mm\,s^{-3}}$ is introduced between the maximum and minimum jerk sections leading to

$$j(t) = \begin{cases} j_\mathrm{max}, & \forall t_0 \leq t < t_1, \\ -j_\mathrm{max}, & \forall t_2 \leq t < t_\mathrm{f}, \\ 0, & \text{otherwise}, \end{cases} \tag{5.25}$$

where

$$\begin{aligned} t_0 < t_1 < t_2 < t_\mathrm{f}, \\ t_1 - t_0 = \Delta T_1, \\ t_2 - t_1 = \Delta T_2, \\ t_\mathrm{f} - t_2 = \Delta T_1, \end{aligned} \tag{5.26}$$

hold. Again using Heaviside functions and ignoring the last term, and setting the jerk to zero as well as assuming $t_0 = 0$ leads to

$$j(t) = j_\mathrm{max} \left\{ H(t) - H(t - \Delta T_1) - H(t - \Delta T_1 - \Delta T_2) \right\}. \tag{5.27}$$

Integration gives

$$\begin{aligned} a(t) &= j_{\max}\{H(t)t - H(t-\Delta T_1)(t-\Delta T_1) \\ &\quad - H(t-\Delta T_1-\Delta T_2)(t-\Delta T_1-\Delta T_2)\}, \end{aligned} \tag{5.28}$$

$$\begin{aligned} v(t) &= v(t_0) + \frac{1}{2} j_{\max}\{H(t)t^2 - H(t-\Delta T_1)(t-\Delta T_1)^2 \\ &\quad - H(t-\Delta T_1-\Delta T_2)(t-\Delta T_1-\Delta T_2)^2\}, \end{aligned} \tag{5.29}$$

$$\begin{aligned} s(t) &= s(t_0) + v(t_0)t + \frac{1}{6} j_{\max}\{H(t)t^3 - H(t-\Delta T_1) \\ &\quad (t-\Delta T_1)^3 - H(t-\Delta T_1-\Delta T_2)(t-\Delta T_1-\Delta T_2)^3\}. \end{aligned} \tag{5.30}$$

Since $\max a(t) = a(\Delta T_1) = a_{\max}$, the time ΔT_1 for increasing and decreasing the acceleration is calculated from (5.28):

$$\Delta T_1 = \frac{a_{\max}}{j_{\max}}. \tag{5.31}$$

The velocity difference $\Delta v = v(t_f) - v(t_0)$ is given by

$$\begin{aligned} \Delta v &= \frac{1}{2} j_{\max}\{H(2\Delta T_1 + \Delta T_2)(2\Delta T_1 + \Delta T_2)^2 \\ &\quad - H(\Delta T_1 + \Delta T_2)(\Delta T_1 + \Delta T_2)^2 - H(\Delta T_1)(\Delta T_1)^2\}, \\ &= \frac{1}{2} j_{\max}(2\Delta T_1 + 2\Delta T_1 \Delta T_2). \end{aligned} \tag{5.32}$$

Substitution by (5.31) leads to

$$\begin{aligned} \Delta v &= j_{\max}\left(\frac{a_{\max}^2}{j_{\max}^2} + \frac{a_{\max}}{j_{\max}}\Delta T_2\right), \\ &= \frac{a_{\max}^2}{j_{\max}} + a_{\max}\Delta T_2. \end{aligned} \tag{5.33}$$

Since $v(t_f) = v_f$ is assumed to be known for the transition,

$$\Delta T_2 = \frac{\Delta v}{a_{\max}} - \frac{a_{\max}}{j_{\max}}. \tag{5.34}$$

The overall duration of the transition $t_f - t_0$ is then calculated as

$$t_f - t_0 = 2\Delta T_1 + \Delta T_2 = \frac{\Delta v}{a_{\max}} + \frac{a_{\max}}{j_{\max}}. \tag{5.35}$$

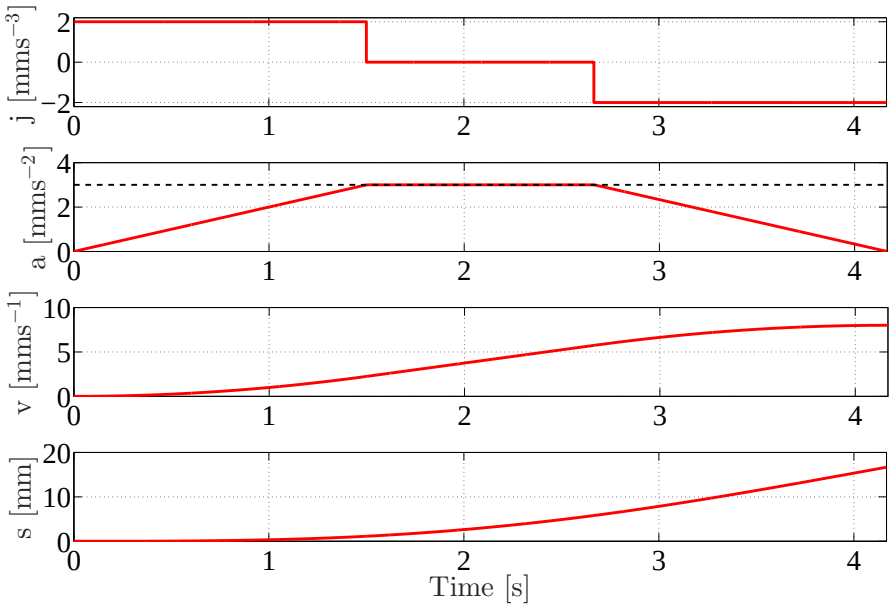


FIGURE 5.4: Velocity transition calculated by the bang-bang approach with acceleration limit for table 5.1 (b).

This is used to calculate the travelled distance $\Delta s = s(t_\mathrm{f}) - s(t_0)$:

$$\begin{aligned}\Delta s &= v(t_0)\left(2\Delta T_1 + \Delta T_2\right) + \frac{1}{6} j_\mathrm{max} \Big\{ \left(2\Delta T_1 + \Delta T_2\right)^3 \\ &\quad - \left(\Delta T_1 + \Delta T_2\right)^3 - \left(\Delta T_1\right)^3 \Big\}, \\ &= \frac{1}{2}\left(v(t_0) + v_\mathrm{f}\right)\left(\frac{\Delta v}{a_\mathrm{max}} + \frac{a_\mathrm{max}}{j_\mathrm{max}}\right). \end{aligned} \tag{5.36}$$

The result for table 5.1 (b) is shown in Fig. 5.4. Here the acceleration limit is satisfied. Note that this approach is only applicable if (5.31) and $\Delta T_2 \geq 0$ hold. For $\Delta T_2 = 0$ the equations reduce to the equations in section 5.3.1. The necessary acceleration to complete the transition within a certain distance can be determined by solving the following quadratic equation which is derived from (5.36):

$$\begin{aligned} 0 &= a_\mathrm{max}^2 \left(\frac{1}{2}\left(v(t_0) + v_\mathrm{f}\right)\right) - a_\mathrm{max} \Delta s j_\mathrm{max} \\ &\quad + \frac{1}{2}\left(v(t_0) + v_\mathrm{f}\right)\Delta v j_\mathrm{max}. \end{aligned} \tag{5.37}$$

5.3.3 Approach Selection for Smooth Velocity Transition

Since the approach in section 5.3.1 does not guarantee the acceleration limits and the approach in section 5.3.2 can only be used if the acceleration limit is actually reached, the

appropriate approach has to be selected for all velocity transitions during planning. The easiest way is to check (5.22) as follows:

- if $\max a(t)$ from (5.22) exceeds the prescribed limit, use the approach in section 5.3.2,
- otherwise, use the approach in section 5.3.1.

Note that negative velocity transitions work in the same way by setting $j_{\max}$ and $a_{\max}$ to negative values.

5.4 Contouring Controller Design with Friction Compensator

5.4.1 Modelling of Friction Compensator

As described in [153], friction force in lead-screw systems is composed of linear and nonlinear terms, where the nonlinear term is assumed to be the eccentricity between a lead screw and a nut as shown in Fig. 5.5. In precise machine tool systems, under longtime use, an infinitesimal gap between a lead-screw and a nut results in an uncertain friction value. Based on this assumption, a spring-like model to describe the friction behaviour inside the screw-nut system is proposed. The normal force N varies when the screw rotates, and the friction caused by this normal force varies depending on the angular position θ of the screw. This variation of the normal force results in a sinusoidal friction term which is described as follows:

$$f_{ec}(\theta) = k\sin(\theta - \theta_0), \tag{5.38}$$

where f_{ec}, k, θ, and θ_0 are the eccentric friction, the maximum absolute value (amplitude) of the eccentric friction in a lead screw, current angular position, and initial angular position, respectively. From the relationship between θ and x as $\theta = 2\pi x/L$, (5.38) becomes:

$$\begin{aligned} f_{ec}(x) &= \eta_2 \sin(2\pi x/L - \eta_3), \\ \eta_2 &= k, \;\; \eta_3 = \theta_0. \end{aligned} \tag{5.39}$$

From the assumption about the eccentric friction part, a nonlinear friction model that includes the Coulomb-viscous friction and nonlinear friction term f_{ec} was proposed as

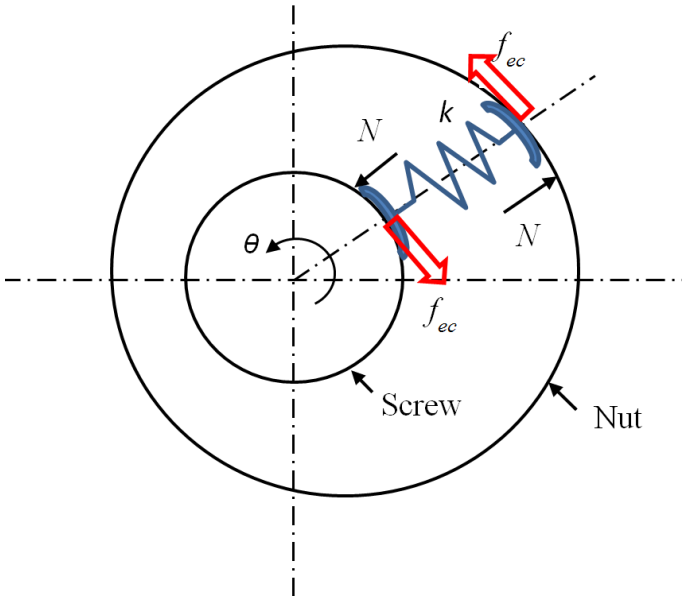


FIGURE 5.5: Modeling of eccentric phenomenon between lead screw and nut.

follows:

$$f_r(v) = f_{fli} + f_{fle} = \eta_0 \text{sgn}(v) + \eta_1 v + f_{ec}(x), \tag{5.40}$$

where $f_r(v)$, f_{fli}, and f_{fle}, $\eta_0(> 0)$, and $\eta_1(> 0)$ are the total friction force, effect of friction in the linear guide-ways and the ball screw, Coulomb force, and the viscous coefficient of the feed drive, respectively. Equation (5.40) does not include the Stribeck effect, since it only affects the low velocity region. Therefore, the eccentric friction part to precisely describe the friction behavior of a lead screw on the high velocity region is added. Eccentric friction is concerned with a part of lead screw friction f_{fle}.

5.4.2 Contouring Controller Design

In machining, the contour error is an important criterion for the quality of machining surface. Fig. 5.6 schematically explains the relationship between the tracking and contour errors. The coordinate frame Σ_w, whose axes x, y, and z correspond to the feed drive axes, is a fixed frame. The blue contour represents the desired path of the feed drive of a three-axis machine tool. The symbol $q_d = [x_d, y_d, z_d]^{\text{T}}$ denotes the desired position at time t, and is defined in Σ_w. The real position of the feed drive is assumed to be $q = [x, y, z]^{\text{T}}$, which is also defined in Σ_w. The contour error is defined as the shortest distance from q to the desired path, and it is represented by the symbol e_c. The contouring controller is concerned with reducing this error. The tracking error vector e_w, which consists of the

tracking errors in three feed drive axes, is defined as follows:

$$e_w = \begin{bmatrix} e_{wx} & e_{wy} & e_{wz} \end{bmatrix}^{\mathrm{T}} = q - q_d. \tag{5.41}$$

The approximated contour error is defined in a local coordinate frame Σ_l. Its origin is at the desired position q_d and with three axes $\mathcal{T}, \mathcal{N}, \mathcal{B}$, as shown in the figure. The axis $\mathcal{T}$ is in the tangential direction of desired path at q_d, the direction of the axis $\mathcal{N}$ is perpendicular to $\mathcal{T}$ at q_d and the axis $\mathcal{B}$ is the bi-normal component normal to $\mathcal{T}$ and $\mathcal{N}$. For the parametric trajectory, the tangential, normal, and bi-normal vectors are denoted as $\mathcal{T}$, $\mathcal{N}$, and $\mathcal{B}$, respectively, and are calculated at a time t as follows:

$$\begin{aligned}
&\mathcal{N}_{temp} = \left\{ \frac{\ddot{q}_d}{\|\ddot{q}_d\|},\ \forall \ddot{q}_d \neq 0 \right\} \text{ or } \left\{ \begin{pmatrix} 1 & 0 & 0 \end{pmatrix},\ \ddot{q}_d = 0, \right. \\
&\left. \forall \mathcal{T} \neq \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \right\} \text{ or } \left\{ \begin{pmatrix} 1 & 0 & 0 \end{pmatrix},\ \ddot{q}_d = 0,\ \text{for } \mathcal{T} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \right\}, \\
&\mathcal{T} = \frac{\dot{q}_d}{\|\dot{q}_d\|},\ \mathcal{B} = \mathcal{T} \times \mathcal{N}_{temp},\ \mathcal{N} = \mathcal{B} \times \mathcal{T},
\end{aligned} \tag{5.42}$$

where $\mathcal{N}_{temp}$ is the vector to find a plane that contains vectors $\mathcal{T}$ and $\mathcal{N}$. The normal vector $\mathcal{N}$ is calculated from vectors $\mathcal{B}$ and $\mathcal{T}$. The error between the real position and the

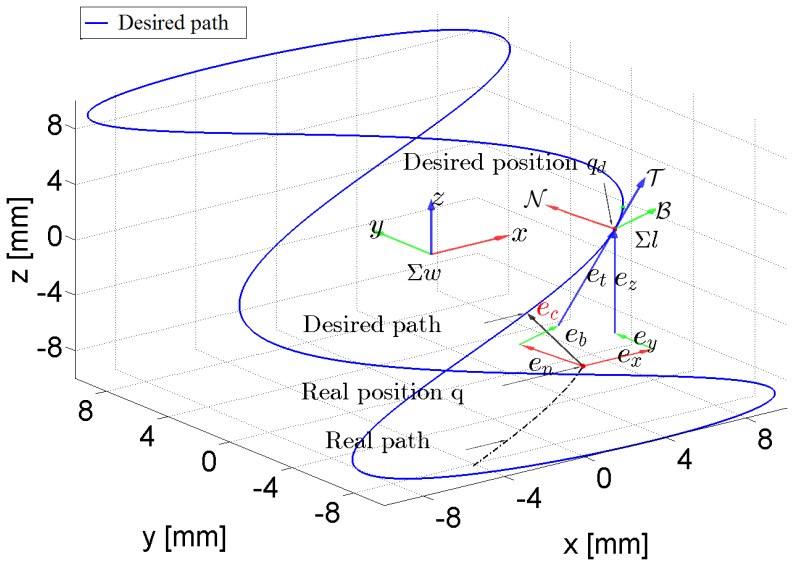


FIGURE 5.6: Definition of the contour error.

desired position can be expressed with respect to Σ_l as follows:

$$\begin{aligned} e_l &= \begin{bmatrix} e_{lt} & e_{ln} & e_{lb} \end{bmatrix}^{\mathrm{T}} = R^{\mathrm{T}} e_w, \\ R &= \begin{bmatrix} \mathcal{T} & \mathcal{N} & \mathcal{B} \end{bmatrix}, \\ e_c &\approx \sqrt{e_{ln}^2 + e_{lb}^2}, \end{aligned} \tag{5.43}$$

where $(.)_t$, $(.)_n$ and $(.)_b$ respectively correspond to $\mathcal{T}$, $\mathcal{N}$, and $\mathcal{B}$, hereinafter, $R^{\mathrm{T}}R = I$ and I is the identity matrix. The contour error e_c is used in the contouring controller design. If the controller gain with respect to e_{ln} and e_{lb} is set to be greater than that for e_{lt}, the contour error can be reduced faster than the tracking error tangential to the desired path. Therefore, the following improved contouring controller where a feed forward friction compensation is included in the controller is applied [9, 156]:

$$\begin{aligned} f_u &= M \left\{ \ddot{q}_d - R \left(K_{vl} \dot{e}_l + K_{pl} e_l + \ddot{R}^{\mathrm{T}} e_w + 2 \dot{R}^{\mathrm{T}} \dot{e}_w \right) \right\} + f_r \\ f_u &= \begin{bmatrix} f_{ux} & f_{uy} & f_{uz} \end{bmatrix}^{\mathrm{T}}, \; M = \mathrm{diag} \begin{Bmatrix} m_x & m_y & m_z \end{Bmatrix}, \\ f_r &= \begin{bmatrix} f_{rx} & f_{ry} & f_{rz} \end{bmatrix}^{\mathrm{T}}, \; K_{vl} = \mathrm{diag} \begin{Bmatrix} k_{vlt} & k_{vln} & k_{vlb} \end{Bmatrix}, \\ K_{pl} &= \mathrm{diag} \begin{Bmatrix} k_{plt} & k_{pln} & k_{plb} \end{Bmatrix}, \end{aligned} \tag{5.44}$$

where f_u, M and $\ddot{q}_d$ are the driving force vector, table mass matrix and the reference acceleration vector of the desired contour, respectively. The symbols K_{vl} and K_{pl} are the velocity and position feedback gain matrices, respectively. They are assumed to be diagonal matrices with positive constant elements.

5.5 Experiment

5.5.1 Experimental Setup

An experiment was conducted on a typical three axis feed drive system as shown in Fig. 5.7. It consists of a table coupled by three lead screws which are driven by DC servo motors. The position of the table was measured based on a 0.025 µm resolution rotary encoder attached to each servo motor, and the velocity was calculated by numerical differentiation of the position measurements. Without loss of generality, a two dimensional space was considered for the experiment. As shown in Fig. 5.8, a sharp-corner trajectory with an angle of 45° was defined in G-code fashion and smoothed by the proposed

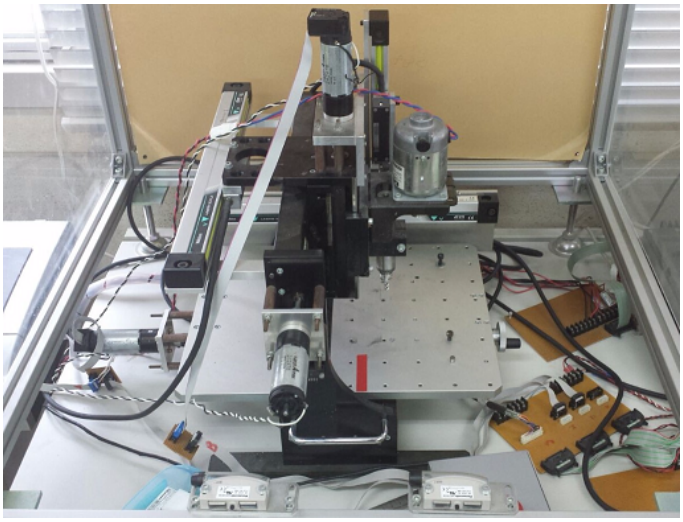


FIGURE 5.7: Experimental system.

algorithm, where the maximum contour error tolerance around the corner was limited to 80 µm. For performance evaluation, the trajectory was linearly interpolated (conventional approach) with a constant velocity of $10\,\mathrm{mm\,s^{-1}}$, while for the proposed trajectory, a smooth velocity profile was applied as proposed in section 5.3. A feedback contouring controller (FB) and the FB with a feed forward friction compensator (FBFC) were applied to both the conventional and the proposed trajectories so as to evaluate the contribution of the friction compensator (FC).

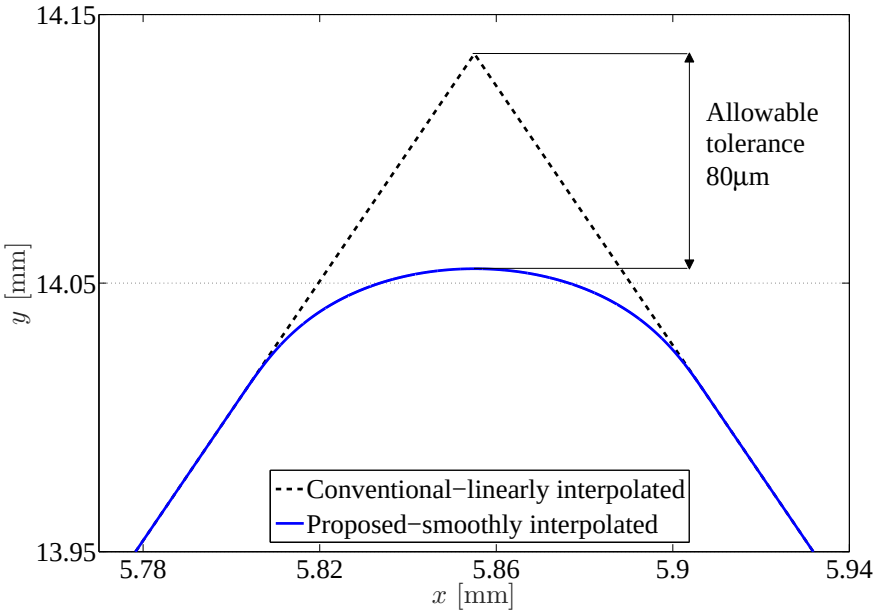


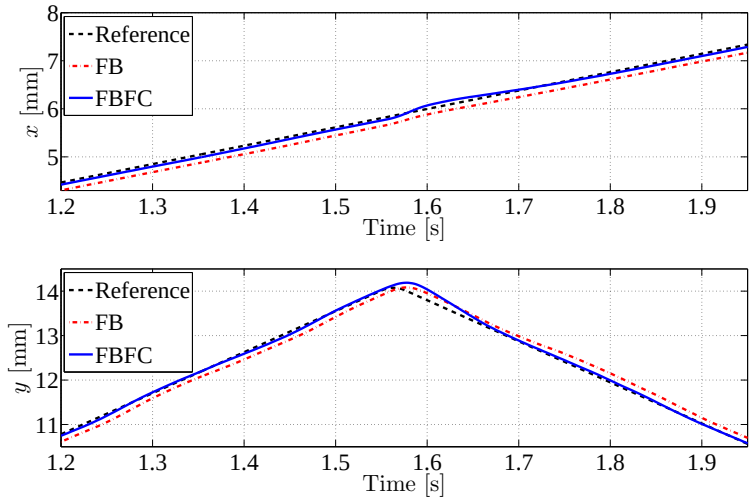
FIGURE 5.8: Zoomed portion of the designed reference trajectories.

5.5.2 Experimental Results

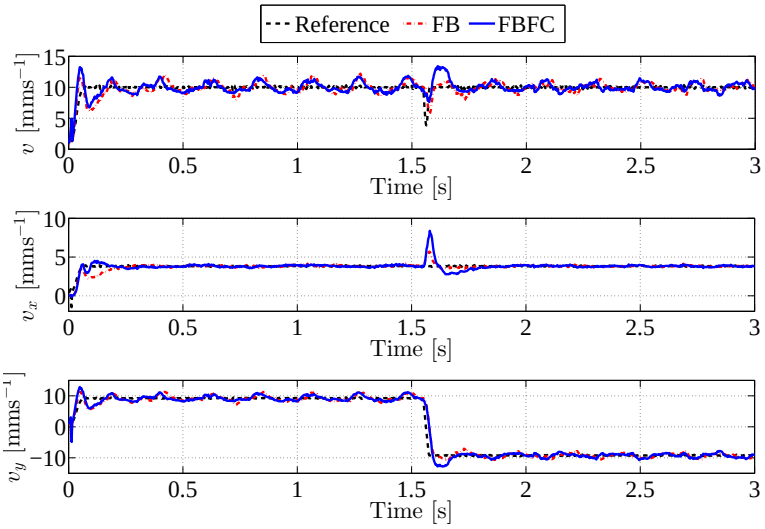
Figures 5.9 (a) and 5.10 (a) show position tracking results for the conventional and proposed trajectories, respectively. It is shown that the FC has enhanced the tracking performance in both cases except around 1.6 s in Fig. 5.9 (a) due to high acceleration at the corner. Velocity profiles for each drive axis as well as the overall system velocities for both the conventional and proposed trajectories are respectively shown in Figs. 5.9 (b) and 5.10 (b). It can be seen that for the proposed method, the maximum velocity is within the desired range ($10\,\mathrm{mm\,s^{-1}}$). However, in the conventional case, the maximum velocity of $13.4\,\mathrm{mm\,s^{-1}}$, which is beyond the desired range, occurred at the corner (around 1.6 s) due to the infinite curvature. The contour error results are respectively represented by Figs. 5.9 (c) and 5.10 (c), for the conventional and proposed methods, where e_{cx} and e_{cy} refer to components of the contour error in each drive axis. In the conventional case, although the FC shows satisfactory performance in major parts, a large contour error of about 0.259 mm occurred around the corner. Yet it is slightly less than the maximum contour error based on the FB controller only (0.262 mm). On the other hand, under the proposed method (Fig. 5.10 (c)), the maximum contour error is about 0.065 mm which is equivalent to a reduction of 75 % of the maximum contour error as compared to the conventional approach.

5.6 Summary

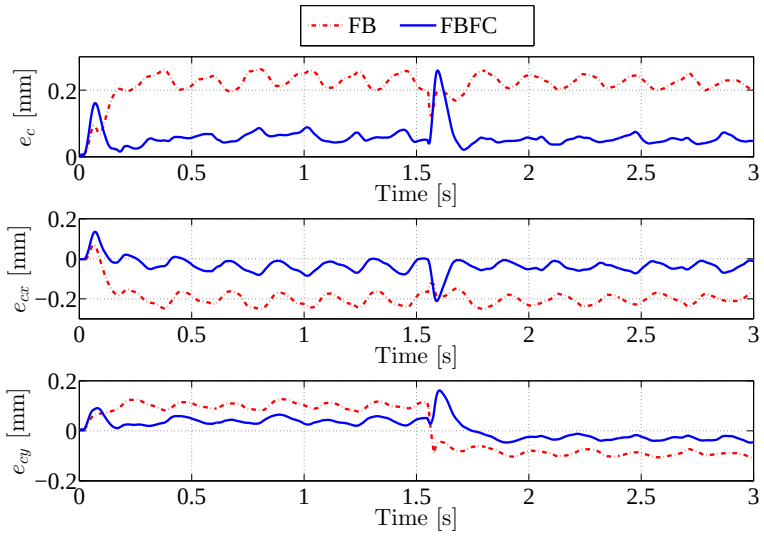
A novel approach for smooth trajectory generation and friction compensating for feed drive contouring control is proposed in this chapter. Its performance has been experimentally evaluated by comparing to the conventional approach, i. e., a linearly interpolated trajectory with a constant velocity. The proposed method has shown that, the contour error can be greatly reduced while satisfying systems constraints.



(a) Position tracking profiles.

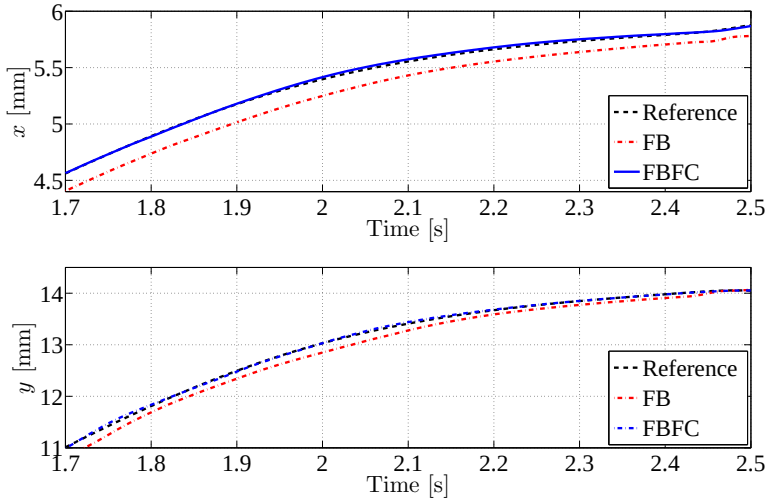


(b) Velocity profiles.

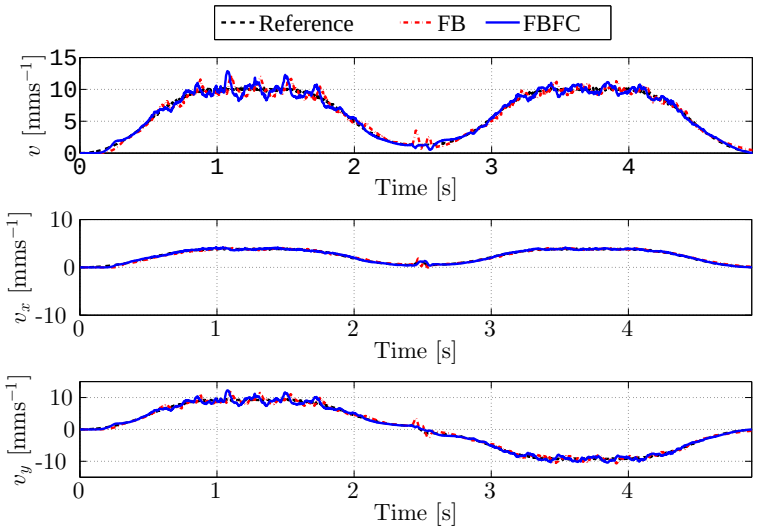


(c) Contour errors.

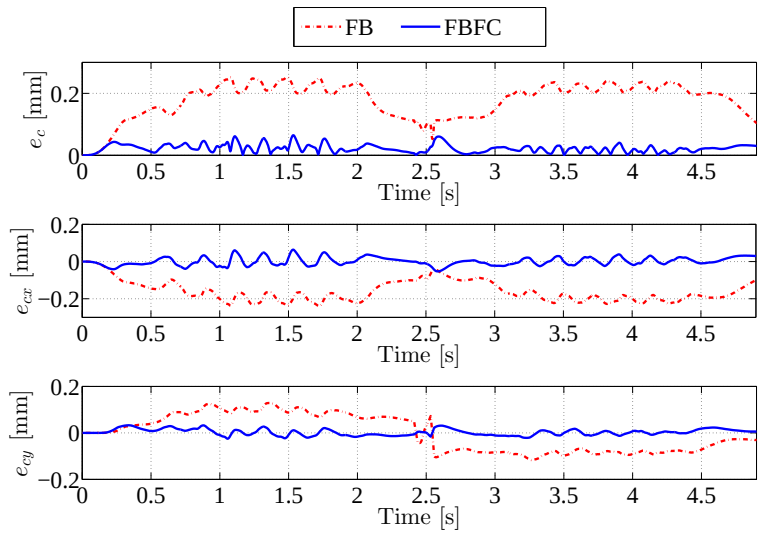
FIGURE 5.9: Experimental results for the conventional approach.



(a) Position tracking profiles.



(b) Velocity profiles.



(c) Contour errors.

FIGURE 5.10: Experimental results for the proposed approach.

Chapter 6

Iterative Learning Contouring Controller for Feed Drive Systems

In feed drive systems, particularly, machine tools, contour error is more significant than the individual axis tracking errors from the view point of enhancing precision in manufacturing and production systems. The contour error must be within the permissible tolerance of given product. In machining complex or sharp-corner products, large contour errors occur mainly owing to discontinuous trajectories and the existence of nonlinear uncertainties. Therefore, it is indispensable to design robust controllers that can enhance the tracking ability of feed drive systems. In this study, an iterative learning contouring controller consisting of a classical Proportional-Derivative (PD) controller, disturbance observer and nonlinear friction compensator is proposed. The proposed controller was evaluated experimentally by using a typical sharp-corner trajectory, and its performance was compared with that of conventional controllers. The results revealed that the maximum contour error can be reduced by about 47.8 % on average.

6.1 Introduction

The rapid growth of technology and demand for precise products have created a need for high-speed and precise production and manufacturing systems. Computer Numerical Control (CNC) machine tools are being used globally for the production of different parts ranging from pinhole sized ones such as parts of watches, cameras and computers, to larger ones such as automotive and infrastructure parts. Precision is crucial for ensuring the quality of these products of different sizes.

Nonlinear uncertainties in real control systems result from either disturbance signals or due to system modelling errors [1, 2]. These uncertainties are common and cannot be

avoided in practical settings. When a system is approximated by a mathematical model, non-fundamental factors are ignored such as high-frequency dynamics and uncertainties. These uncertainties are crucial in undermining the performance of a dynamical system. In machine tools, which deal with discontinuous trajectories programmed in G-code fashion, some uncertainties may arise at junctions, especially in sharp-corners or raster trajectories. Thus, large contour errors may occur in these areas.

Usually, products are manufactured in batches; therefore, the nature of machine tool operation is repetitive. This repetitive nature allows the design of controllers that learn from previous inputs and modify subsequent inputs to improve system performance in real time. This type of control is called Iterative Learning Control (ILC), and it has been proven to provide superior system performance [6, 102, 148, 157]. The common approach is to design independent controllers for each drive axis by feeding back the tracking errors and updating the control inputs accordingly. Given that motion trajectory profiles are normally complex, multiple axes must be moved synchronously to obtain the desired profile. Under independent axial controllers, load disturbance or performance variance of either drive axis leads to contour errors [97]. In light of this, major current approaches for improving the control performance of feed drive systems are based on contouring control [9, 96, 101–104], while a few of them are based on the tracking error of each drive axis [158].

Despite the achievements of a few previous studies, it is indispensable to further enhance system performance by considering contour errors. An ILC that considers both tracking and contour errors was designed in [159], and its feasibility was verified by simulation. Meanwhile, in [160], a friction model that considers a number of friction sources with complex nonlinear properties was proposed. The proposed model could consider nonlinearities in high-speed motion, and it was proven experimentally to be superior to existing models such as the one in [161].

In the present study, a Variable-gain Iterative Learning Contouring controller with Friction compensator and Disturbance observer (VILCFD) is proposed. compared to conventional ILCs such as the one in [159], the proposed controller achieves better performance by reducing the maximum contour error by about 47.8 % on average.

The remainder of this chapter is organised as follows: Section 6.2 defines contour error and explains the dynamics of biaxial feed drive systems. Section 6.3 describes the design of the proposed contouring controller, which includes a nonlinear friction model. Simulation and experimental results are given in section 6.4, followed by concluding

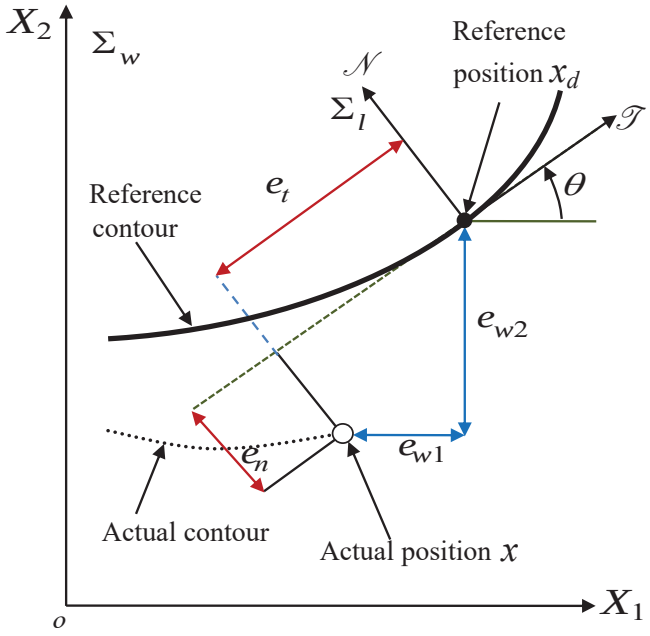


FIGURE 6.1: Definition of tracking and contour errors.

remarks in section 6.5.

6.2 Preliminaries

6.2.1 Definition of Contour Error

Here, the contour error is derived simply from the tracking error in each drive axis, as shown in Fig. 6.1. It is the perpendicular distance from the actual position to the reference contour. In contrast, the tracking error refers to the difference between the desired and actual positions of each drive axis. The desired position of a point on a machined part at sampling instant t in coordinate frame Σ_w is denoted by x_d, while x represents the actual position of the feed drive system in Σ_w. The tracking error in each drive axis is defined as

$$e_w = [e_{w1}\ e_{w2}]^{\mathrm{T}} = x_d - x. \tag{6.1}$$

The coordinate frame Σ_l is attached at x_d and its axis directional vectors are $\mathcal{T}$ and $\mathcal{N}$, which are tangential and orthogonal to the reference position x_d, respectively. Thus, the tracking error vector e_w can be expressed with respect to Σ_l as

$$e_l = \begin{bmatrix} e_t & e_n \end{bmatrix}^{\mathrm{T}} = R^{\mathrm{T}} e_w, \quad R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \tag{6.2}$$

where θ is the inclination of Σ_l to Σ_w.

6.2.2 Dynamics of Feed Drive Systems

The dynamics of a typical feed drive system is represented by the following decoupled second order system:

$$\begin{aligned}
&M_f\ddot{x} + f_r + d = f,\\
&M_f = \mathrm{diag}\left\{m_{fi}\right\}, \quad i = 1,2,\\
&x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^{\mathrm{T}}, \quad f_r = \begin{bmatrix} f_{r1} & f_{r2} \end{bmatrix}^{\mathrm{T}},\\
&d = \begin{bmatrix} d_1 & d_2 \end{bmatrix}^{\mathrm{T}}, \quad f = \begin{bmatrix} f_1 & f_2 \end{bmatrix}^{\mathrm{T}},
\end{aligned} \tag{6.3}$$

where m_{fi}, x_i, f_{ri}, d_i and f_i are the mass of the table, the position of the table, friction force, bounded disturbance and driving force on the drive axis i, respectively. Each drive axis is driven by a typical servo motor, whose dynamics is represented as follows:

$$\begin{aligned}
&H\ddot{\vartheta} + C_m\dot{\vartheta} + \tau = K_v u,\\
&H = \mathrm{diag}\left\{h_i\right\}, \quad C_m = \mathrm{diag}\left\{c_{mi}\right\}, \quad K_v = \mathrm{diag}\left\{k_{vi}\right\},\\
&\vartheta = \begin{bmatrix} \vartheta_1 & \vartheta_2 \end{bmatrix}^{\mathrm{T}}, \quad \tau = \begin{bmatrix} \tau_1 & \tau_2 \end{bmatrix}^{\mathrm{T}}, \quad u = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^{\mathrm{T}},
\end{aligned} \tag{6.4}$$

where h_i, ϑ_i, c_{mi}, τ_i, k_{vi} and u_i are the motor inertia, the rotational angle of the motor for the drive axis i, viscous friction coefficient, feed drive driving torque, torque–voltage conversion ratio, and the input voltage to the i^{th} motor, respectively. From (6.3) and (6.4), the system dynamics is described as follows:

$$\begin{aligned}
&M\ddot{x} + C\dot{x} + f_r + d = K_\mu u,\\
&M = \mathrm{diag}\left\{\frac{m_{fi}\rho_i^2 + 4\pi^2 h_i}{\rho_i^2}\right\},\\
&C = \mathrm{diag}\left\{\frac{4\pi^2 c_{mi}}{\rho_i^2}\right\},\\
&K_\mu = \mathrm{diag}\left\{\frac{2\pi k_{vi}}{\rho_i}\right\},
\end{aligned} \tag{6.5}$$

where ρ_i is the pitch of the i^{th} lead screw.

6.3 Contouring Controller Design

6.3.1 Friction Force Modelling

The friction force is among the dominating components that hinder the performance of machine tools from the viewpoints of motion accuracy and system stability. Generally, controllers with friction compensator achieve a better performance than those without it. The friction model in (6.6), which considers that friction forces in feed drive machine tools originate from multiple sources with complicated and nonlinear properties, was proposed in [160]. That it performs satisfactorily was proved experimentally, and hence, it has been employed in this study.

$$f_{ri} = \eta_{0i}\mathrm{sgn}(\dot{x}_i) + \eta_{1i}\dot{x}_i + \sum_{\sigma=1}^{n} g_{\sigma ai} \exp\left[-\left(\frac{\dot{x}_i - g_{\sigma bi}}{g_{\sigma ci}}\right)^2\right], \tag{6.6}$$

where η_{0i} and η_{1i} are the nominal Coulomb force and the viscous friction coefficient of the i^{th} drive axis, respectively. The nonlinear properties of friction are defined by the sum of n Gaussian equations in which $g_{\sigma ai}$, $g_{\sigma bi}$ and $g_{\sigma ci}$ denote the height of the Gaussian curve's peak, the position of the centre of the peak and the width of the curve for the i^{th} drive axis, respectively.

6.3.2 Feedback Controller Design

Although the dynamics of the concerned feed drive is ideally a second-order linear system, practically it has several nonlinear parameters as shown in (6.6). Thus, a nonlinear controller is required; the system dynamics is transformed into linear by selecting a suitable control input. The system is transformed as follows to take advantage of PD or Proportional-Integral-Derivative (PID) controllers typically used in industrial applications:

$$\begin{aligned} z_1 &= x, \\ z_2 &= \dot{z}_1, \\ \dot{z}_2 &= M^{-1}\left(K_\mu u - C z_2 - f_r - d\right). \end{aligned} \tag{6.7}$$

The control input

$$u = K_\mu^{-1}\left(M v + C z_2 + f_r\right), \tag{6.8}$$

leads to

$$\begin{aligned} \dot{z}_2 &= v - \gamma, \\ \gamma &= M^{-1} d. \end{aligned} \tag{6.9}$$

The variable $v = [v_1 \; v_2]^{\mathrm{T}}$ is the virtual input designed as

$$\begin{aligned} v &= \ddot{x}_d + R(K_P e_l + K_D \dot{e}_l) + \ddot{\theta} \Xi e_w - \dot{\theta}^2 e_w + 2\dot{\theta} \Xi \dot{e}_w, \\ K_P &= \mathrm{diag}\{k_{Pi}\}, \; K_D = \mathrm{diag}\{k_{Di}\}, \\ \Xi &= \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \end{aligned} \tag{6.10}$$

for contouring control [9], where k_{Pi} and k_{Di} are position and velocity feedback (FB) gains, respectively.

6.3.3 Disturbance Observer Design and Stability Analysis

Observer Design

The unknown disturbance in (6.9) is estimated as $\hat{\gamma}$ based on the altered disturbance observer in [162] as follows:

$$\begin{aligned} \dot{\hat{z}}_2 &= v - \hat{\gamma} + K_{ev}(\tilde{z}_1 + \tilde{z}_2), \\ \dot{\hat{\gamma}} &= K_{ed}(\tilde{z}_1 + \tilde{z}_2), \\ \tilde{z}_1 &= z_{1d} - z_1, \quad \tilde{z}_2 = z_{2d} - z_2, \\ K_{ev} &= \mathrm{diag}\{k_{evi}\}, \; K_{ed} = \mathrm{diag}\{k_{edi}\}, \end{aligned} \tag{6.11}$$

where $\dot{\hat{z}}_2$, and k_{evi} and k_{edi} are the estimated acceleration of the table and disturbance observer gains, respectively.

Stability Proof

Consider the following Lyapunov function $\mathcal{V}_i$ for each drive axis i:

$$\begin{aligned}
\mathcal{V}_i &= \frac{1}{2}\tilde{z}_i^{\mathrm{T}} A_i \tilde{z}_i + \frac{1}{2}\tilde{\gamma}_i^2, \\
\tilde{z}_i &= [\tilde{z}_{1i} \;\; \tilde{z}_{2i}], \\
A_i &= \begin{bmatrix} 2k_{edi}k_{evi} & k_{edi} \\ k_{edi} & k_{edi} \end{bmatrix}.
\end{aligned} \tag{6.12}$$

From (6.9) and (6.11), the derivative of $\mathcal{V}_i$ is as follows:

$$\begin{aligned}
\dot{\mathcal{V}}_i &= \dot{\tilde{z}}_i^{\mathrm{T}} A_i \tilde{z}_i + \tilde{\gamma}_i \dot{\tilde{\gamma}}_i, \\
&= \left(2k_{edi}k_{evi}\dot{\tilde{z}}_{1i} + k_{edi}\dot{\tilde{z}}_{2i}\right)\tilde{z}_{1i} \\
&\quad + \left(k_{edi}\dot{\tilde{z}}_{1i} + k_{edi}\dot{\tilde{z}}_{2i}\right)\tilde{z}_{2i} + \tilde{\gamma}_i\dot{\tilde{\gamma}}_i, \\
&= \left[2k_{edi}k_{evi}\tilde{z}_{2i} + k_{edi}\left(\dot{z}_{2di} - \dot{\hat{z}}_{2i}\right)\right]\tilde{z}_{1i} \\
&\quad + \left[k_{edi}\tilde{z}_{2i} + k_{edi}\left(\dot{z}_{2di} - \dot{\hat{z}}_{2i}\right)\right]\tilde{z}_{2i} + \tilde{\gamma}_i\dot{\tilde{\gamma}}_i, \\
&= \left[2k_{edi}k_{evi}\tilde{z}_{2i} - k_{edi}\left(\tilde{\gamma}_i + k_{evi}\tilde{z}_{1i} + k_{evi}\tilde{z}_{2i}\right)\right]\tilde{z}_{1i} \\
&\quad + \left[k_{edi}\tilde{z}_{2i} - k_{edi}\left(\tilde{\gamma}_i + k_{evi}\tilde{z}_{1i} + k_{evi}\tilde{z}_{2i}\right)\right]\tilde{z}_{2i} \\
&\quad + k_{edi}\tilde{\gamma}_i\left(\tilde{z}_{1i} + \tilde{z}_{2i}\right), \\
&= -\tilde{z}_i^{\mathrm{T}} B_i \tilde{z}_i, \\
B_i &= \begin{bmatrix} k_{edi}k_{evi} & 0 \\ 0 & k_{edi}\left(k_{evi} - 1\right) \end{bmatrix}.
\end{aligned} \tag{6.13}$$

Choosing $k_{evi} > 1$ leads to $\dot{\mathcal{V}}_i \leq 0$, and from Barbalat's lemma [133], system stability is guaranteed.

6.3.4 Application of Iterative Learning Control

A linear time-invariant Single Input Single Output (SISO) ILC system is considered to obtain insights into ILC. Considering a single axis i, its input-output relationship in a discrete-time form is represented as [163]

$$x_{ij}(t) = P_i(b^{-1})v_{ij}(t) + \lambda_i(t), \tag{6.14}$$

where t, j, P_i, b^{-1}, x_{ij} and v_{ij} and λ_i are the time index, iteration number, plant model that is assumed to be stable, delay operator defined as $b^{-1}x_i(t) \equiv x_i(t-1)$, system output and control input at an iteration j and exogenous signal that is time-varying but iteratively constant, respectively. The delay operator is introduced because this is a sampled data system in which there is a delay between the control input and the system output. In most cases of sampled data system, the delay is one; thus, an assumption is made here accordingly. The general update law is represented as [164]

$$v_{ij+1}(t) = q_i(b^{-1})\left[v_{ij}(t) + L_i(b^{-1})e_{wij}(t+1)\right], \tag{6.15}$$

where q_i and L_i are the Q-filter and the learning function, respectively. While the learning function improves the system tracking ability by improving the control input [165], the Q-filter is used to improve the system stability under the presence of high-frequency uncertainties.

The proposed control system, as shown in Fig. 6.2, includes the FB to form a control loop with previous and current signal cycles. The axes are coupled by the rotation matrix R, which transforms the tracking error of each drive axis into the contour error. Although both the tangential and the normal components of the error e_t and e_n, respectively, are used in the FB, only the normal component is used in the iterative learning function L_i. This is because L_i aims to converge the actual contour with the desired one. Elements of the normal error e_n regarding each drive axis can be represented as

$$\begin{aligned} e_{vn} &= R_n e_w, \\ R_n &= \begin{bmatrix} -\sin^2\theta & \sin\theta\cos\theta \\ -\sin\theta\cos\theta & \cos^2\theta \end{bmatrix}. \end{aligned} \tag{6.16}$$

Because of the FB controller and the use of contour error instead of tracking error and from the single axis update law in (6.15), the general update law for both axes is represented follows:

$$\begin{aligned} &v_{j+1}(t) = Q(b^{-1})\left[v_j(t) + L(b^{-1})e_{vnj}(t)\right] + Ke_{lj+1}(t), \\ &Q = \mathrm{diag}\{q_i\},\ L = \mathrm{diag}\{L_i\},\ K = \mathrm{diag}\{K_i\}, \end{aligned} \tag{6.17}$$

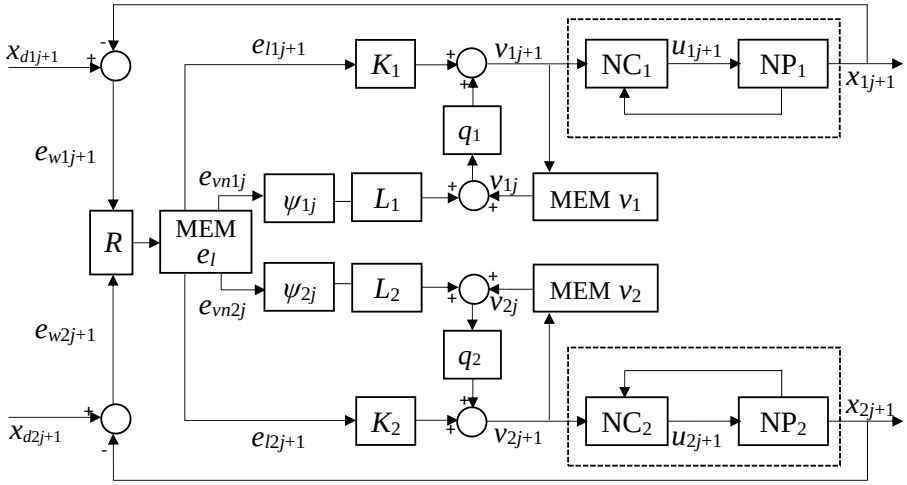


FIGURE 6.2: Block diagram for proposed control system: NP$_i$, NC$_i$, MEM v_i and MEM e_{li} and q_i are the nonlinear plant, nonlinear controller, memories of control inputs and contour errors and the input filter, respectively.

where K_i and v_j are the FB gain matrix for axis i and the system control input in the j^{th} iteration. The corresponding error terms $e_{vnj}(t)$ and $e_{lj+1}(t)$ are given by

$$\begin{aligned} e_{vnj}(t) &= R_n\left(x_d(t) - x_j(t)\right), \\ e_{lj+1}(t) &= R^{\mathrm{T}}\left(x_d(t) - x_{j+1}(t)\right), \end{aligned} \tag{6.18}$$

where the reference trajectory x_d remains unchanged iteratively. A variable PID learning function is proposed so that, the update law at sampling instant t can be written as follows:

$$\begin{aligned} v_{j+1}(t) &= Q(b^{-1})v_j(t) + K_P e_{lj+1}(t) + K_D \dot{e}_{lj+1}(t) \\ &\quad + Q(b^{-1})\Psi_{Pj}K_{PL}e_{vnj}(t) + Q(b^{-1})\Psi_{Ij}K_{IL}\sum_{t=1}^{T} e_{vnj}(t-1) \\ &\quad + Q(b^{-1})\Psi_{Dj}K_{DL}\left\{e_{vnj}(t) - e_{vnj}(t-1)\right\}, \\ K_{mL} &= \mathrm{diag}\{k_{mLi}\},\ \Psi_{mj} = \mathrm{diag}\{\psi_{mij}\},\ \ m = P,\ I,\ D, \\ \psi_{mij} &= 1 - \frac{1}{\alpha_{mi}}\mathrm{sat}\left(\frac{e_{vnij}}{\beta_{mi}}\right),\ 0 \le t \le T-1, \end{aligned} \tag{6.19}$$

where k_{mLi} and T are gains of the learning function L_i and the total number of sampling instants, respectively. ψ_{mij} is an error-dependent function with constants $\alpha_{mi} > 0$ and $\beta_{mi} > 0$.

6.3.5 Convergence Analysis

Stability in control systems is described as a property of the system to converge to an equilibrium point as time goes to infinity, for any arbitrarily small perturbation of the initial state from the equilibrium point [166]. This statement satisfies the condition of continuous-time systems; however, in the iteration domain, convergence is considered [167–169]. An ILC system is asymptotically stable in the iteration domain if and only if the control signal converges to equilibrium [170]. The point of interest is monotonic error convergence because it also implies system stability [167].

Stability analysis of ILC is carried out in the frequency domain based on a system's response to a sinusoidal input [164, 171]. Consequently, the convergence analysis of a SISO ILC system can be carried out considering its z transformation, and the monotonic error convergence condition is represented as follows:

$$\left\| \frac{e_{wij+1}(t)}{e_{wij}(t)} \right\| < 1, \quad \forall t \in T, \tag{6.20}$$

where e_{wij} and $e_{wij+1}(t)$ are the tracking errors of the drive axis i in iterations j and $j+1$, respectively. Owing to the contouring controller that includes a time-varying parameter (θ), the criteria in (6.20) cannot be applied directly. Instead, the focus is on the lifted matrix analysis, a technique that can be used for stability analysis in the time-domain approach.

Lifted Matrix

A lifted system enables one to perform convergence and stability analysis of a system in the iteration domain through matrix representation of the time-domain system dynamics [163]. By applying an impulse input to the system dynamics, the lifted system can be formed. Considering a two-dimensional system in (6.14) that evolves in both iteration

and time, the lifted form can be represented as follows [167]:

$$\underbrace{\begin{bmatrix} x_{ij}(1) \\ x_{ij}(2) \\ \vdots \\ x_{ij}(T+1) \end{bmatrix}}_{\hat{X}_{ij}} = \underbrace{\begin{bmatrix} p_i(0) & 0 & \cdots & 0 \\ p_i(1) & p_i(0) & \cdots & 0 \\ \vdots & \ddots & \ddots & 0 \\ p_i(T) & \cdots & p_i(1) & p_i(0) \end{bmatrix}}_{\hat{P}_i} \underbrace{\begin{bmatrix} v_{ij}(0) \\ v_{ij}(1) \\ \vdots \\ v_{ij}(T) \end{bmatrix}}_{\hat{V}_{ij}} + \underbrace{\begin{bmatrix} \lambda_{ij}(1) \\ \lambda_{ij}(2) \\ \vdots \\ \lambda_{ij}(T+1) \end{bmatrix}}_{\hat{\lambda}_{ij}}, \tag{6.21}$$

where $\hat{(.)}$ represents the lifted form of $(.)$. By using the lifted matrix technique, the input update law in (6.17) becomes

$$\hat{V}_{j+1} = \hat{Q}\left(\hat{V}_j + \hat{L}\hat{E}_{vnj}\right) + \hat{K}\hat{E}_{j+1}. \tag{6.22}$$

Likewise, the corresponding lifted contour errors are represented as

$$\begin{aligned} \hat{E}_{vnj} &= \hat{R}_n\left(\hat{X}_d - \hat{X}_j\right) = \hat{R}_n\left\{\hat{X}_d - \left(\hat{P}\hat{V}_j + \hat{\lambda}\right)\right\}, \\ \hat{E}_{j+1} &= R^{\mathrm{T}}\left(\hat{X}_d - \hat{X}_{j+1}\right) = \hat{R}^{\mathrm{T}}\left\{\hat{X}_d - \left(\hat{P}\hat{V}_{j+1} + \hat{\lambda}\right)\right\}. \end{aligned} \tag{6.23}$$

Substituting (6.23) into (6.22) leads to a lifted representation of the update control input and exogenous signal as follows:

$$\begin{aligned} \hat{V}_{j+1} &= \left[I + \hat{K}\hat{R}^{\mathrm{T}}\hat{P}\right]^{-1}\hat{Q}\left[I - \hat{L}\hat{R}_n\hat{P}\right]\hat{V}_j + \hat{D}, \\ \hat{D} &= \left(\hat{K}\hat{R}^{\mathrm{T}} + \hat{Q}\hat{L}\hat{R}_n\right)\left(\hat{X}_d - \hat{\lambda}\right), \end{aligned} \tag{6.24}$$

where I is the identity matrix. The term $\hat{D}$ is constant and bounded because the exogenous signal $\hat{\lambda}$ and the reference trajectory $\hat{X}_d$ remain unchanged iteratively. Thus, the control input converges if

$$\left\|\left[I + \hat{K}\hat{R}^{\mathrm{T}}\hat{P}\right]^{-1}\hat{Q}\left[I - \hat{L}\hat{R}_n\hat{P}\right]\right\|_2 = \epsilon < 1. \tag{6.25}$$

Because the input converges, the convergence of x_j and e_{lj} follows from (6.14) and (6.18), respectively. Therefore, the condition in (6.25) guarantees system stability and monotonic convergence. The convergence speed depends on the parameter ϵ which should be kept as low as possible to increase the tracking performance within a few iterations. The FB gains, K_P and K_D, are selected by considering the best performance before application of the ILC, and the Q-filter is designed depending on the desired cut-off frequency. The ILC gains ($K_{PL},\ K_{IL},\ K_{DL}$) are determined by solving the following minimisation problem in a prescribed domain of θ by using the fmincon function in MATLAB:

$$\begin{aligned} J = \min_{K_{PL},\, K_{IL},\, K_{DL}} & \epsilon(t), \quad \forall 0 \leq t \leq T, \\ \text{s.t. } & \epsilon(t) < 1. \end{aligned} \tag{6.26}$$

6.4 Simulation and Experiment

Simulation and experiment were performed to verify the effectiveness of the proposed controller. A comparison with the work in [159] was made to evaluate its performance. In this context, the compared work is referred to as a conventional method. The proposed ILC includes and an FB controller, a friction compensator and disturbance observer. Therefore, the following scenarios were considered to understand the contribution of each component:

i FB controller only.

ii FB controller with Disturbance observer (FBD).

iii FB controller with Friction compensator (FBF).

iv FB controller with Friction compensator and Disturbance observer (FBFD).

v FB controller with friction compensator, disturbance observer and ILC (FILC).

vi VILCFD.

6.4.1 Experimental Setup

A typical biaxial feed drive system (Fig. 6.3) was used for the analysis. It consisted of a table coupled by two lead screws driven by DC servo motors. The table position was

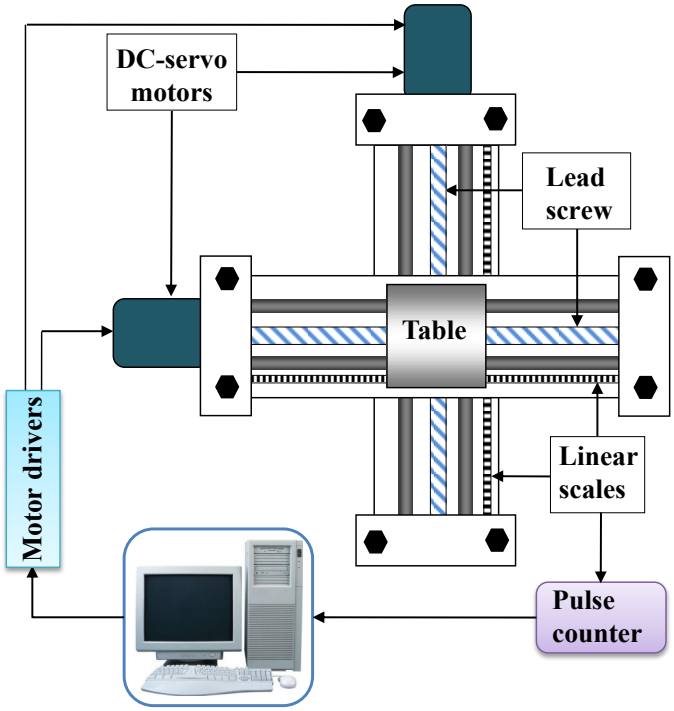


FIGURE 6.3: Biaxial feed drive system for experiment.

TABLE 6.1: System parameters.

m_{f1}	8.0 kg	$h_1,\ h_2$	0.05 kg m^2
m_{f2}	2.5 kg	$c_{m1},\ c_{m2}$	0.31 N m s rad^{-1}
$\rho_1,\ \rho_2$	0.005 m	$k_{v1},\ k_{v2}$	1.42 N m V^{-1}

measured using a rotary encoder (0.025 µm resolution) attached to each servo motor, and the velocity was calculated by numerical differentiation of the position measurements. The system was controlled by a C++ language programme on a personal computer equipped with a 1 GHz CPU, and 512 MB RAM, and running Windows OS. A counter board with four channels of 24-bit up/down counters was used to provide a fixed sampling rate of 5 ms; thus, the considered operational Nyquist frequency (ω_0) was 100 rad s^{-1}. The Q-filter was considered as unity, and the rest of parameters are given in Tables 6.1 and 6.2. The reference trajectory was defined in G-code fashion to create a typical linear interpolated sharp-corner trajectory with an angle of 90 deg. These types of trajectories are difficult to track owing to their discontinuous nature, which requires infinite acceleration at the corner. The proposed controller aims to improve tracking performance minimising of this error.

TABLE 6.2: Controller parameters.

k_{P1}, k_{P2}	5280 V mm^{-1}	k_{PL1}, k_{PL2}	281 V mm^{-1}
k_{D1}, k_{D2}	40000 V s mm^{-1}	k_{IL1}, k_{IL2}	0.31 V s^{-1} mm^{-1}
β_{m1}, β_{m2}	0.0035 mm	k_{DL1}, k_{DL2}	100 V s mm^{-1}
α_{m1}, α_{m2}	30	k_{ed1}, k_{ed2}	20 kg s^{-3}
q_1, q_2	1	k_{ev1}, k_{ev2}	40 s^{-1}

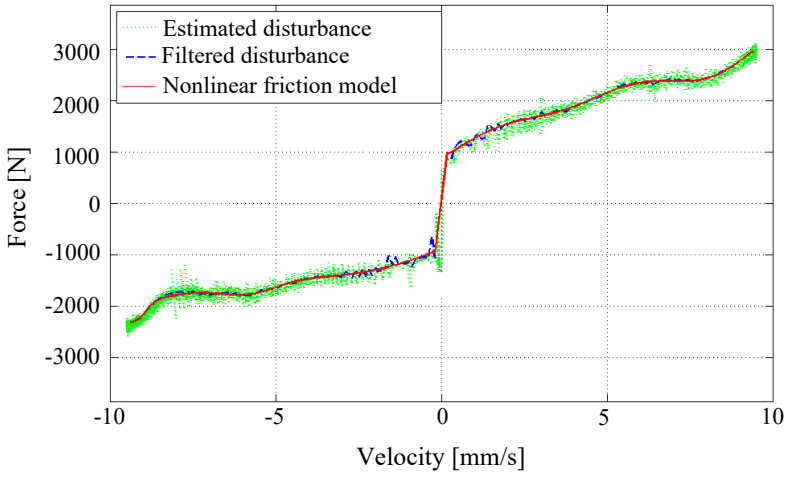


FIGURE 6.4: Measured and estimated frictions.

6.4.2 Identification of Friction Parameters

Before simulation and experiment, the friction parameters were identified using a sinusoidal reference trajectory based on the PD controller and disturbance observer, as detailed in [160]. Figure 6.4 shows the measured and the estimated friction forces in the X_1 drive axis, where the dotted, dashed and solid lines represent the disturbance measured using the disturbance observer, disturbance after applying a low-pass filter and disturbance computed using the nonlinear friction model in (6.6). The corresponding identified parameters are given in Table 6.3.

6.4.3 Simulation Results

The simulation was performed over multiple iterations until there was no further significant error reduction, and a comparison with the FB and FILC approaches was made for performance evaluation. FILC refers to the combination of the FB and constant gain ILC to form a control loop with previous and current signal cycles without application of the disturbance observer and the friction model in (6.6). In both controllers, the first iteration is represented as FB so that both FILC and VILCFD start from the second iteration.

TABLE 6.3: Identified friction model parameters.

Var.	Unit	X_1 drive axis		X_2 drive axis	
		$\dot{x}_1 \geq 0$	$\dot{x}_1 < 0$	$\dot{x}_2 \geq 0$	$\dot{x}_2 < 0$
η_0	N	768.60	-768.60	640.50	640.90
η_1	$\mathrm{N\,s\,mm^{-1}}$	102.48	102.40	140.90	140.90
n		3.00	3.00	3.00	3.00
g_{1a}	N	517.40	-378.20	518.00	-371.34
g_{1b}	$\mathrm{mm\,s^{-1}}$	1.76	-2.16	1.76	-1.96
g_{1c}	$\mathrm{mm\,s^{-1}}$	3.09	2.26	1.98	2.99
g_{2a}	N	509.30	-547.50	852.00	-236.10
g_{2b}	$\mathrm{mm\,s^{-1}}$	5.80	-5.91	5.87	-5.80
g_{2c}	$\mathrm{mm\,s^{-1}}$	1.82	1.76	2.73	1.22
g_{3a}	N	1995.90	-4472.00	1059.10	-354.97
g_{3b}	$\mathrm{mm\,s^{-1}}$	10.66	-13.90	10.20	-9.31
g_{3c}	$\mathrm{mm\,s^{-1}}$	1.87	3.65	1.74	0.47

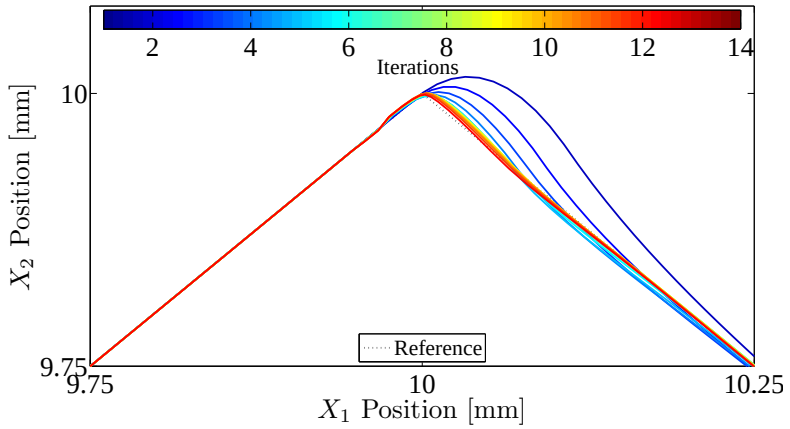


FIGURE 6.5: Simulated iterative trajectory tracking profiles for proposed controller (VILCFD).

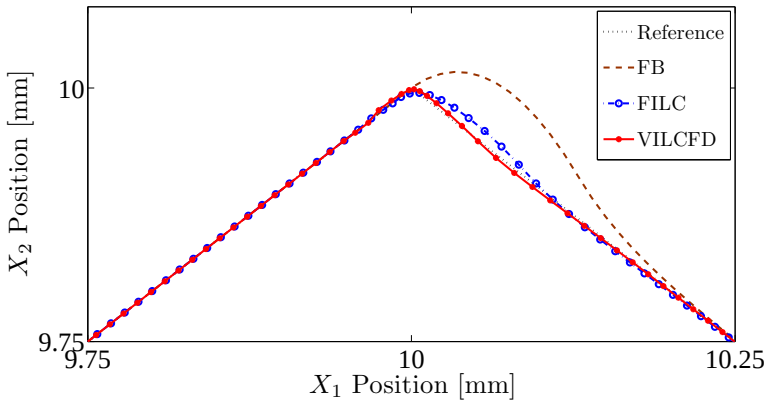


FIGURE 6.6: Simulation results of trajectory tracking.

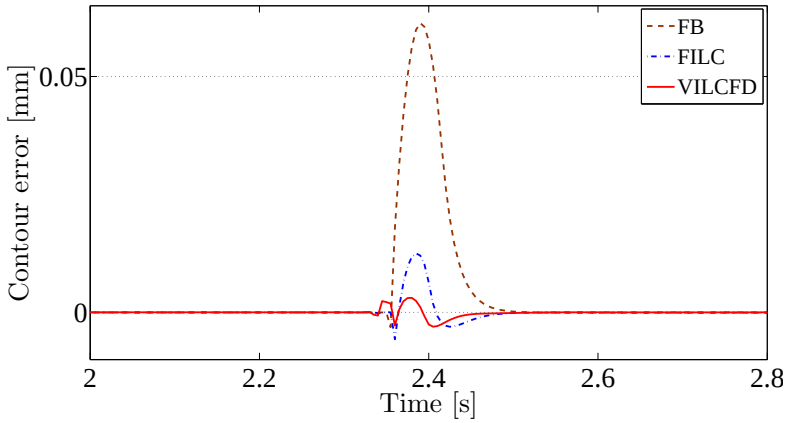
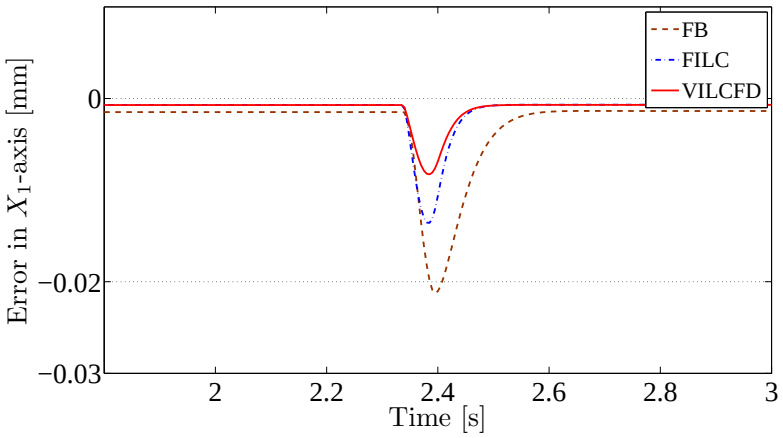
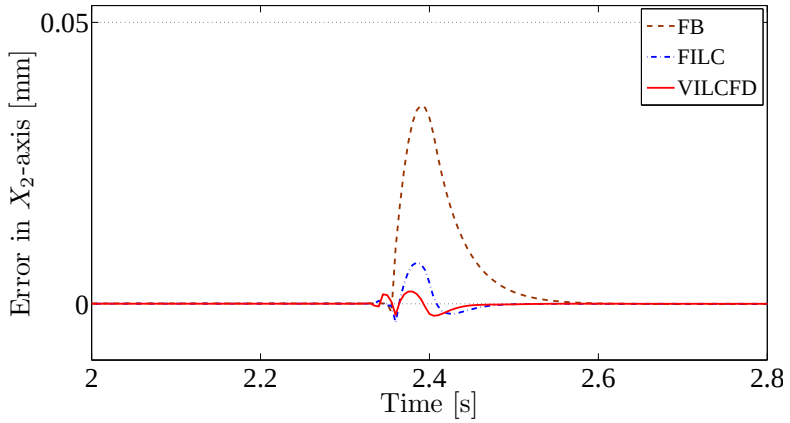


FIGURE 6.7: Simulation results of contour error.



(a) Tracking error of X_1 drive axis.



(b) Tracking error of X_2 drive axis.

FIGURE 6.8: Simulation results of tracking errors in individual drive axes.

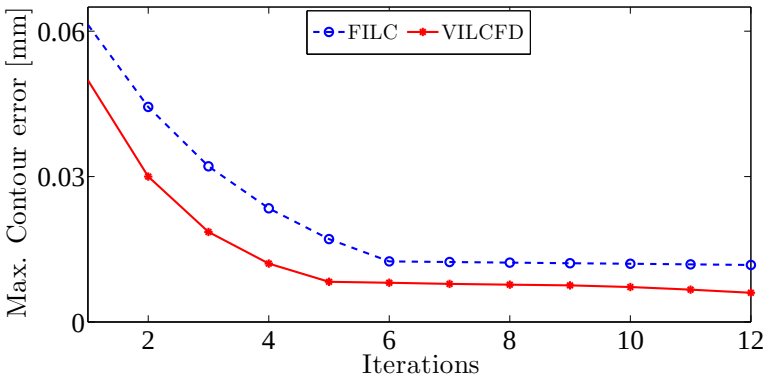


FIGURE 6.9: Simulation results of maximum contour error in each iteration.

TABLE 6.4: Summary of simulation results (µm).

Controller	Max. Tracking error X_1-axis	X_2-axis	Max. Contour error
FB	51.0	35.2	61.3
FBILC	10.4	7.3	12.5
VILCFD	2.2	2.2	3.1

Figure 6.5 shows the trajectory tracking profiles based on the proposed controller (VILCFD), and Fig. 6.6 shows the first and the final trajectory tracking profiles for three methods. It can be observed that the FB profile is the furthest from the reference, followed by FILC and VILCFD in series. The maximum contour errors results are shown in Fig. 6.7, where the maximum contour errors for the FB, FILC and VILCFD at around 2.4 s were 61.3 µm, 12.5 µm and 3.1 µm, respectively. VILCFD reduced the maximum contour error by 79.6 % and 75.2 % compared to the FB and FILC, respectively. An equivalent result was obtained for the individual drive axis tracking errors as shown in Fig. 6.8 (a) and (b). Furthermore, VILCFD showed a relatively higher convergence rate than FILC and achieved a minimal maximum contour error, as shown in Fig. 6.9. For clarity, the simulation results are summarised in Table 6.4.

6.4.4 Experimental Results

Similar to the simulation, an experiment was conducted over multiple iterations until there was no significant further convergence in the contour error. For performance evaluation, the six scenarios highlighted in section 6.4 were compared. The experimental results are shown in Figs. 6.10 - 6.14, where Fig. 6.10 shows the trajectory tracking profiles of

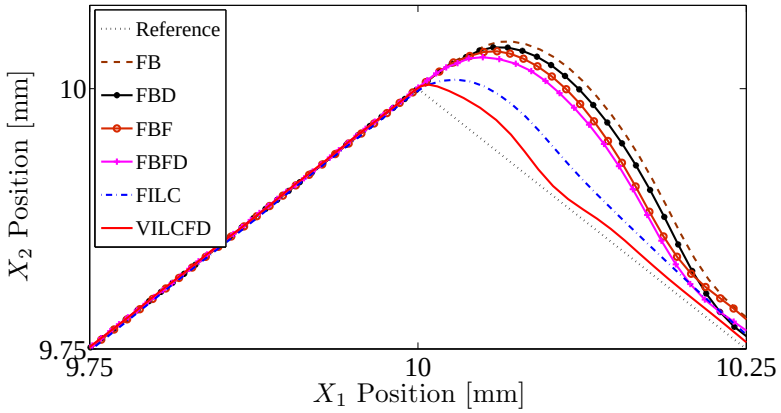


FIGURE 6.10: Experimental results of trajectory tracking.

each controller. Note that only the final results are shown for the ILCs. It can be seen that the FB controller has the worst performance, while the proposed VILCFD achieves the best performance. As shown in Fig. 6.11, the maximum contour error based on the FB controller is 95.2 µm, which decreases gradually with other controllers to 20.9 µm under VILCFD. Figure 6.12 shows the individual axial trajectory tracking errors for both axes. In the X_1 drive axis, FB has the best performance, whereas VILCFD has the worst performance. On the contrary, VILCFD has the best performance in the X_2 drive axis. This result is obvious because based on the considered trajectory, the X_1 drive axis always move along the forward direction, while the X_2 drive axis reverses its motion after the corner. However, this is not the main concern because contour error is more significant in machine tools than individual axes tracking errors [9, 172]. Although both FILC and VILCFD have relatively similar convergence rates, VILCFD achieves the smallest contour error, as shown in Fig. 6.13. Also, the results in Fig. 6.13 differ from those in Fig. 6.9 owing to the existence of time-variant uncertainties that could not be included in the minimisation problem in (6.26). The repeatability of the proposed controller (VILCFD) was verified by conducting 10 trials, and the results of maximum contour error obtained in each trial are shown in Fig. 6.14. For limpidity, the experimental results are summarised in Table 6.5.

6.4.5 Discussion

Precise machine tools are required to improve manufacturing and production systems. Because contour error is related directly to precision of workpieces, it must be as low as possible. The experimental system used here was assembled in our laboratory; therefore,

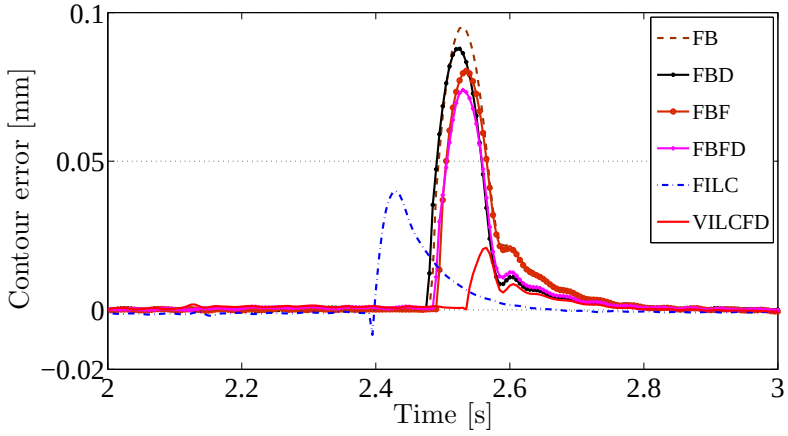
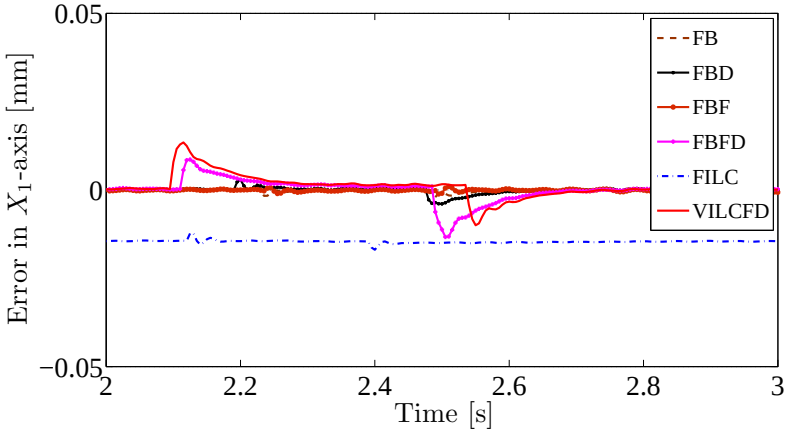
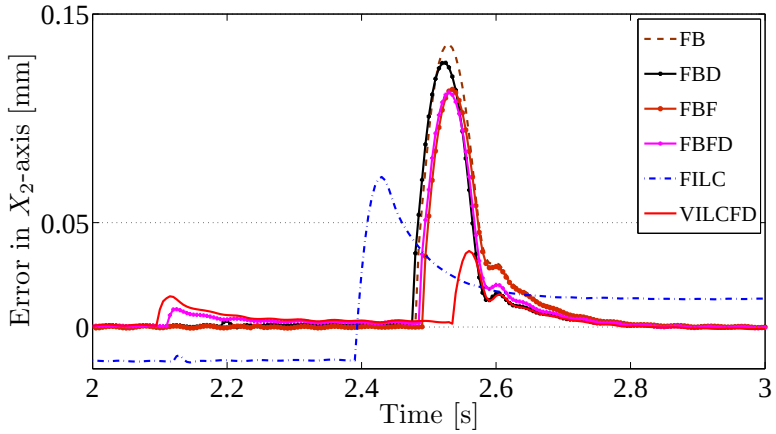


FIGURE 6.11: Experimental results of contour error.



(a) Tracking error of X_1 drive axis.



(b) Tracking error of X_2 drive axis.

FIGURE 6.12: Experimental results of tracking errors along individual drive axes.

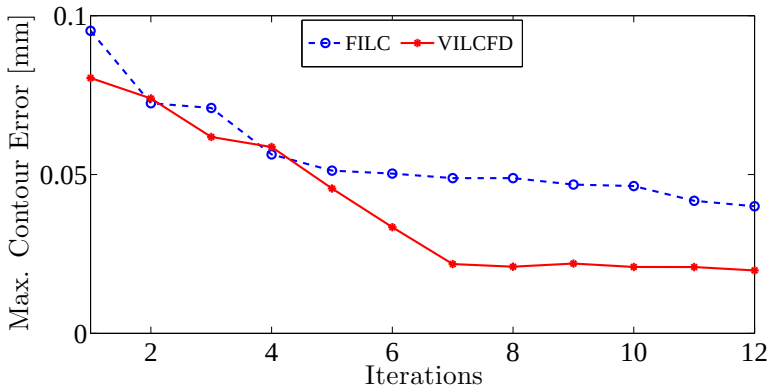


FIGURE 6.13: Experimental results of maximum contour error in each iteration.

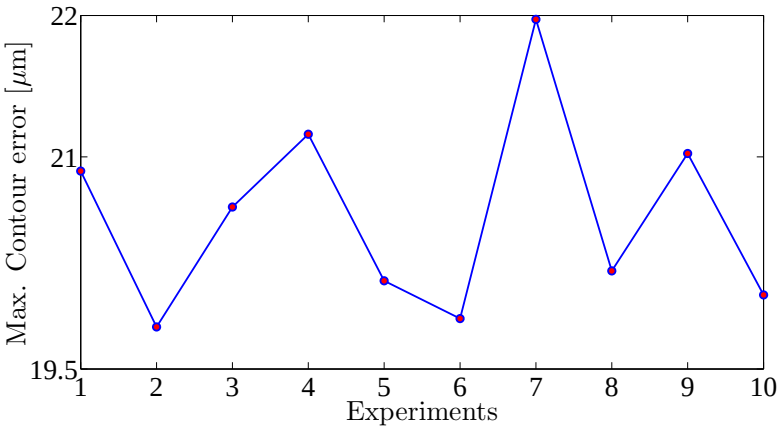


FIGURE 6.14: Experimental results of maximum contour error over 10 trials.

TABLE 6.5: Summary of experimental results (μm).

Controller	Max. Tracking error		Maximum	STD
	X_1-axis	X_2-axis	Contour error	
FB	-2.0	135.5	95.2	3.6
FBD	-3.8	126.6	92.8	3.3
FBF	1.0	113.9	87.8	3.0
FBFD	-13.3	112.5	80.4	2.5
FILC	-16.9	71.9	40.0	3.5
VILCFD	13.5	36.5	20.9	0.8

it has many uncertainties like other industrial systems. Furthermore, a rapid change in motion direction at corners poses additional uncertainties that are difficult to model. From the experimental results, it can be realised that the traditional PD controller is insufficient from the precision viewpoint. Likewise, ILC without consideration of nonlinear time-variant uncertainties cannot guarantee satisfactory performance. The disturbance observer in the proposed controller is used to estimate unknown uncertainties, such as those resulting from a sudden change in the direction of motion at the corners, in addition to other time-variant parameters [153]. In this light, it is evident that the proposed controller has superior performance, as demonstrated by the experimental results.

6.5 Summary

In this chapter, a robust iterative contouring controller is proposed for feed drive systems, and its effectiveness was demonstrated by using simulation and experimental results. It includes a traditional PD controller, nonlinear friction compensator, disturbance observer and an iterative learning controller. The proposed method showed a substantial improvement in performance by reducing the maximum contour error by 47.8 % on average. In future work, optimisation methods will be employed to increase system performance.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Owing to the strong demand for solutions to achieve high-speed motion while maintaining high precision in industrial mechatronic systems, this thesis aimed to find novel solutions for generating smooth motion trajectories and control design, specifically for mobile robots and feed drive systems. These mechatronic systems are used widely in manufacturing industries. Herein, a smooth trajectory refers to a trajectory that not only describe paths accurately but is also kinematically smooth and within the physical limitations of mechatronic systems. Also, for mobile robot applications, the trajectory should be able to set the first and the second derivatives at the starting and the ending points arbitrarily, as well as local controllability, such that any changes in the trajectory affect only limited regions.

In this thesis, methods to generate smooth trajectories are proposed for industrial mechatronic systems. The generated trajectories are smooth enough to be tracked by these systems and have all the important features stated above. The generated trajectories were evaluated experimentally by using a typical differential mobile robot and a feed drive system. Comparisons with conventional methods were made for performance evaluation. It was found that the generated trajectory rendered lower maximum velocities and accelerations at corners, shorter travel times and better tracking performance starting with the mobile robot. For feed drive systems, the generated trajectory enhanced tracking performance and decreased the maximum contour error.

Furthermore, because industrial feed drive systems operate in a repetitive fashion given that they are used for batch manufacturing, contour error-based ILC has been used to enhance the tracking performance. In fact, ILC is categorised as one of the intelligent control approaches that can guarantee the desired performance of a repetitive system.

Thus, a controller that incorporates the traditional PID controller, a friction compensator, disturbance observer and ILC, was proposed. Experimental results verified that by using the proposed controller, the maximum contour error could be reduced by about 47.8 % on average compared to conventional controllers.

7.2 Future Works

7.2.1 Energy Saving in Mechatronic Systems

Energy saving in mechatronic systems has become a major concern from the viewpoint of developing energy-efficient production systems with lower CO_2 emissions. Control theory can be used as an effective tool for reducing the energy consumption of industrial systems through software upgrades. This thesis focused on enhancing the motion precision of industrial mechatronic systems without consideration of energy consequences. Future works will include energy consumption analyses based on the proposed methods and use of optimisation techniques for saving energy.

7.2.2 Implementation of Bézier Subdivision for Obstacle Avoidance

In chapter 3 it was concluded that the proposed method for generating smooth motion trajectories for mobile robots allows for setting the first and the second derivatives arbitrarily at the starting and ending points of the trajectory. The combination of this feature and the Bézier subdivision is suitable for obstacle-avoidance trajectory generation. When a robot encounters an obstacle, the trajectory can be subdivided and a new Bézier curve that avoids obstacles can be smoothly connected. It would be beneficial to develop and analyse this concept through actual implementation.

Appendix A

List of Publications

Journal papers:

J.1. K. R. Simba, B. D. Bui, and N. Uchiyama,“Robust iterative learning contouring controller with friction compensator and disturbance observer for multi-axis feed drive systems," *ISA Transactions*. (*Submitted*)

J.2. Y. M. Hendrawan, K. R. Simba, and N. Uchiyama, “Iterative learning based trajectory generation for machine tool feed drive systems,” *Robotics and Computer-Integrated Manufacturing*. (*Submitted*)

J.3. K. R. Simba, N. Uchiyama, M. Aldibaja, and S. Sano, “Vision-based smooth obstacle avoidance motion trajectory generation for autonomous mobile robots using Bézier curves," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 231, pp. 541–554, Feb. 2017

J.4. K. R. Simba, N. Uchiyama, and S. Sano, “Real-time smooth trajectory generation for nonholonomic mobile robots using Bézier curves,” *Robotics and Computer-Integrated Manufacturing*, vol. 41, pp. 31–42, Oct. 2016

J.5. B. D. Bui, N. Uchiyama, and K. R. Simba, “Contouring control for three-axis machine tools based on nonlinear friction compensation for lead screws,” *International Journal of Machine Tools and Manufacture*, vol. 108, pp. 95–105, Sept. 2016

Conference papers:

C.1. K. R. Simba, G. Heppeler, B. D. Bui, Y. M. Hendrawan, O. Sawodny, and N. Uchiyama, "Bézier curve based trajectory generation and nonlinear friction compensation for feed drive contouring control," *World Congress of the International Federation of Automatic Control*, pp. 1980–1987, Jul. 2017

C.2. K. R. Simba, N. Uchiyama, and S. Sano, "Variable-gain iterative learning contouring control for feed drive systems," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, pp. 7227–7231, Oct. 2016

C.3. Y. M. Hendrawan, K. R. Simba, and N. Uchiyama, "Embedded iterative learning contouring controller design for biaxial feed drive systems," in *International Electronics Symposium (IES)*, (Denpasar, Indonesia), Sept. 2016

C.4. K. R. Simba, N. Uchiyama, and S. Sano, "Iterative contouring controller design for biaxial feed drive systems," in 20th Conference on Emerging Technologies & Factory Automation (ETFA), pp. 1–5, IEEE, Institute of Electrical and Electronics Engineers (IEEE), Sept. 2015

C.5. K. R. Simba, N. Uchiyama, and S. Sano, "Real-time obstacle-avoidance motion planning for autonomous mobile robots," in 4th Australian Control Conference (AUCC), pp. 267–272, Institute of Electrical and Electronics Engineers (IEEE), Nov. 2014

C.6. K. R. Simba, N. Uchiyama, and S. Sano, "Real-time trajectory generation for mobile robots in a corridor-like space using Bézier curves," in *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration*, pp. 37–41, Institute of Electrical and Electronics Engineers (IEEE), Dec. 2013

Appendix B

List of Awards

A.1. International Federation of Automatic Control Application Paper Prize (Finalist): *2017*

A.2. IEEE Industrial Electronics Society Student Paper Travel Assistance Award: *2016*

A.3. IEEE Industrial Electronics Society Best Session Presentation Award: *2016*

Bibliography

[1] F. Gao, X. Li, and Y. Li, "Control of system with large parametric uncertainty using multiple robust controllers and switching," *11th World Congress on Intelligent Control and Automation*, pp. 4408 – 4413, 2014.

[2] D. Dong, C. Chen, R. Long, B. Qi, and I. R. Petersen, "Sampling-based learning control for quantum systems with hamiltonian uncertainties," *IEEE 52nd Annual Conference on Decision and Control*, pp. 1924 – 1929, 2013.

[3] D. Verscheure, B. Paijmans, H. Van Brussel, and J. Swevers, "Vibration and motion control design and trade-off for high-performance mechatronic systems," *IEEE International Conference on Control Applications*, pp. 1115 – 1120, 2006.

[4] N. Uchiyama, S. Shigenori, and K. Haneda, "Residual vibration suppression using simple motion trajectory for mechanical systems," *SICE Journal of Control, Measurement, and System Integration*, vol. 8, no. 3, pp. 195 – 200, 2015.

[5] K. Jolly, R. S. Kumar, and R. Vijayakumar, "A bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 23 – 33, 2009.

[6] O. Elshazly, M. El-bardini, and N. El-Rabaie, "Adaptive fuzzy iterative learning controller for x-y table position control," *IEEE International Conference on Automation, Quality and Testing, Robotics*, pp. 1 – 6, 2014.

[7] T. Meng, J. Li, D. Zheng, and Z. Li, "The design of iterative learning control scheme for CNC machine tools," *IEEE International Conference on Information and Automation*, pp. 665 – 670, 2016.

[8] D.-I. Kim and S. Kim, "An iterative learning control method with application for CNC machine tools," *IEEE Transactions on Industry Applications*, vol. 32, no. 1, pp. 66 – 72, 1996.

[9] N. Uchiyama, T. Nakamura, and H. Yanagiuchi, "The effectiveness of contouring control and a design for three-dimensional machining," *International Journal of Machine Tools and Manufacture*, vol. 49, no. 11, pp. 876 – 884, 2009.

[10] H. Choset, K. M. Lynch, and S. Hutchinson, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT PR, 2005.

[11] C. E. Thomaz, M. A. C. Pacheco, and M. M. B. R. Vellasco, "Mobile robot path planning using genetic algorithms," *Foundations and Tools for Neural Modeling: International Work-Conference on Artificial and Natural Neural Networks*, pp. 671 – 679, Springer Berlin Heidelberg, 1999.

[12] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review," *International Journal of Physical Sciences*, vol. 7, no. 9, pp. 1314 – 1320, 2012.

[13] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. ACM*, vol. 22, no. 10, pp. 560 – 570, 1979.

[14] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. MIT Press Ltd, 2011.

[15] L. Li, T. Ye, M. Tan, and X.-j. Chen, "Present state and future development of mobile robot technology research," *Robot*, vol. 24, no. 5, pp. 475 – 480, 2002.

[16] O. Maron and T. Lozano-Pérez, "Visible decomposition: Real-time path planning in large planar environments," 1996.

[17] C. Ó'Dúnlaing and C. K. Yap, "A "retraction" method for planning the motion of a disc," *Journal of Algorithms*, vol. 6, no. 1, pp. 104 – 111, 1985.

[18] S. Garrido, L. Moreno, D. Blanco, and P. Jurewicz, "Path planning for mobile robot navigation using voronoi diagram and fast marching," *International Journal of Intelligent Systems and Applications in Robotics*, vol. 2, no. 1, pp. 42 – 64, 2011.

[19] E. Masehian and M. Amin-Naseri, *Composite Models for Mobile Robot Offline Path Planning*. INTECH Open Access Publisher, 2007.

[20] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, vol. C-32, no. 2, pp. 108 – 120, 1983.

[21] R. A. Brooks and T. Lozano-Perez, "A subdivision algorithm in configuration space for find path with rotation," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 2, pp. 224 – 233.

[22] O. Hachour, "The proposed genetic FPGA implementation for path planning of autonomous mobile robot," *International journal of circuits, systems and signal processing*, vol. 2, no. 2, pp. 151 – 167, 2008.

[23] M. Likhachev, D. Ferguson, G. Gordon, A. T. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm," *Proceedings of the International Conference on Automated Planning and Scheduling*, (Pittsburgh, PA), pp. 262 – 271, 2005.

[24] M. Weigl, B. Siemiàatkowska, K. Sikorski, and A. Borkowski, "Grid-based mapping for autonomous mobile robot," *Robotics and Autonomous Systems*, vol. 11, no. 1, pp. 13 – 21, 1993.

[25] D. Zhu and J.-C. Latombe, "New heuristic algorithms for efficient hierarchical path planning," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, pp. 9 – 20, 1991.

[26] G.-Z. TAN, H. HE, and A. SLOMAN, "Ant colony system algorithm for real-time globally optimal path planning of mobile robots," *Acta Automatica Sinica*, vol. 33, no. 3, pp. 279 – 285, 2007.

[27] K. Sugihara and J. Smith, "Genetic algorithms for adaptive motion planning of an autonomous mobile robot," *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 138 – 143, 1997.

[28] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," *28th Annual Symposium on Foundations of Computer Science*, pp. 49 – 60, 1987.

[29] R. Wein, J. P. van den Berg, and D. Halperin, "The visibility–voronoi complex and its applications," *Computational Geometry*, vol. 36, no. 1, pp. 66 – 87, 2007.

[30] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[31] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st ed., 1989.

[32] D. Gallardo, O. Colomina, F. Flórez, and R. Rizo, "A genetic algorithm for robust motion planning," *Tasks and Methods in Applied Artificial Intelligence*, (Berlin, Heidelberg), pp. 115 – 121, Springer Berlin Heidelberg, 1998.

[33] G. Nagib and W. Gharieb, "Path planning for a mobile robot using genetic algorithms," *International Conference on Electrical, Electronic and Computer Engineering*, pp. 185 – 189, 2004.

[34] A. Ismail, A. Sheta, and M. Al-Weshah, "A mobile robot path planning using genetic algorithm in static environment," *Journal of Computer Science*, vol. 4, no. 4, pp. 341 – 344, 2008.

[35] J. Tu and S. X. Yang, "Genetic algorithm based path planning for a mobile robot," *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 1221 – 1226, 2003.

[36] Y. Wang, D. Mulvaney, and I. Sillitoe, "Genetic-based mobile robot path planning using vertex heuristics," *IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1 – 6, 2006.

[37] P. Raja and S. Pugazhenthi, "Path planning for a mobile robot in dynamic environments," *International Journal of Physical Sciences*, vol. 6, no. 20, pp. 4721 – 4731, 2011.

[38] C. Alberto, D. Marco, and M. Vittorio, "Distributed optimization by ant colonies," *Toward a practice of autonomous systems: proceedings of the First European Conference on Artificial Life*, pp. 134 – 142, 1992.

[39] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens, "Classification with ant colony optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 651 – 665, 2007.

[40] H. Nezamabadi-pour, S. Saryazdi, and E. Rashedi, "Edge detection using ant algorithms," *Soft Computing*, vol. 10, no. 7, pp. 623 – 628, 2006.

[41] R. A. Russell and W.-C. Chiang, "Scatter search for the vehicle routing problem with time windows," *European Journal of Operational Research*, vol. 169, no. 2, pp. 606 – 622, 2006.

[42] Y. C. Liang and A. E. Smith, "An ant colony optimization algorithm for the redundancy allocation problem (RAP)," *IEEE Transactions on Reliability*, vol. 53, no. 3, pp. 417 – 423, 2004.

[43] G. Z. Tan, H. He, and S. Aaron, "Global optimal path planning for mobile robot based on improved dijkstra algorithm and ant system algorithm," *Journal of Central South University of Technology*, vol. 13, no. 1, pp. 80 – 86, 2006.

[44] H. Miao and Y. C. Tian, "Robot path planning in dynamic environments using a simulated annealing based approach," *2008 10th International Conference on Control, Automation, Robotics and Vision*, pp. 1253 – 1258, 2008.

[45] V. Selvi and D. R. Umarani, "Comparative analysis of ant colony and particle swarm optimization techniques," *International Journal of Computer Applications*, vol. 5, no. 4, pp. 1 – 6, 2010.

[46] J. Kennedy and R. Eberhart, "Particle swarm optimization," *IEEE International Conference on Neural Networks*, vol. 4, pp. 1942 – 1948, 1995.

[47] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," *IEEE International Conference on Evolutionary Computation*, pp. 69 – 73, 1998.

[48] J. F. Kennedy, J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm intelligence*. Morgan Kaufmann, 2001.

[49] J. Kennedy, "The particle swarm: social adaptation of knowledge," *IEEE International Conference on Evolutionary Computation*, pp. 303 – 308, 1997.

[50] B. Tang, Z. Zhanxia, and J. Luo, "A convergence-guaranteed particle swarm optimization method for mobile robot global path planning," *Assembly Automation*, vol. 37, no. 1, pp. 114 – 129, 2017.

[51] D. Floreano, J. Godjevac, A. Martinoli, F. Mondada, and J. D. Nicoud, "Design, control, and applications of autonomous mobile robots," *Advances in Intelligent Autonomous Systems*, (Dordrecht), pp. 159 – 186, Springer Netherlands, 1999.

[52] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90 – 98, 1986.

[53] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," *IEEE International Conference on Robotics and Automation*, pp. 1398 – 1404, 1991.

[54] J. H. Chuang and N. Ahuja, "An analytically tractable potential field model of free space and its application in obstacle avoidance," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 5, pp. 729 – 736, 1998.

[55] J.-C. Latombe, *Robot motion planning*, vol. 124. Springer Science & Business Media, 2012.

[56] G. C. Luh and W. W. Liu, "Motion planning for mobile robots in dynamic environments using a potential field immune network," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 221, no. 7, pp. 1033 – 1045, 2007.

[57] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 5, pp. 615 – 620, 2000.

[58] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: a collision cone approach," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 28, no. 5, pp. 562 – 574, 1998.

[59] B. Damas and J. Santos-Victor, "Avoiding moving obstacles: the forbidden velocity map," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4393 – 4398, 2009.

[60] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760 – 772, 1998.

[61] P. Vadakkepat, K. C. Tan, and W. Ming-Liang, "Evolutionary artificial potential fields and their application in real time robot path planning," *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 256 – 263, 2000.

[62] H. Miao and Y. C. Tian, "Dynamic robot path planning using an enhanced simulated annealing approach," *Applied Mathematics and Computation*, vol. 222, pp. 420 – 437, 2013.

[63] Z. Hong, Y. Liu, G. Zhongguo, and C. Yi, "The dynamic path planning research for mobile robot based on artificial potential field," *International Conference on Consumer Electronics, Communications and Networks*, pp. 2736 – 2739, 2011.

[64] J. W. Lee, J. J. Kim, B.-S. Choi, and J. J. Lee, "Improved ant colony optimization algorithm by potential field concept for optimal path planning," *IEEE-RAS International Conference on Humanoid Robots*, pp. 662 – 667, 2008.

[65] J. E. Bobrow, "NC machine tool path generation from CSG part representations," *Computer-Aided Design*, vol. 17, no. 2, pp. 69 – 76, 1985.

[66] S. Ding, M. Mannan, A. Poo, D. Yang, and Z. Han, "The implementation of adaptive isoplanar tool path generation for the machining of free-form surfaces," *The International Journal of Advanced Manufacturing Technology*, vol. 26, no. 7, pp. 852 – 860, 2005.

[67] J. Hwang, "Interference-free tool-path generation in the NC machining of parametric compound surfaces," *Computer-Aided Design*, vol. 24, no. 12, pp. 667 – 676, 1992.

[68] S. X. Li and R. B. Jerard, "5-axis machining of sculptured surfaces with a flat-end cutter," *Computer-Aided Design*, vol. 26, no. 3, pp. 165 – 178, 1994.

[69] J. Zhao, Q. Zou, L. Li, and B. Zhou, "Tool path planning based on conformal parameterization for meshes," *Chinese Journal of Aeronautics*, vol. 28, no. 5, pp. 1555 – 1563, 2015.

[70] Y. Koren and R. Lin, "Efficient tool-path planning for machining free-form surfaces," *Ann Arbor*, vol. 1050, pp. 48109 – 2125, 1996.

[71] K. Suresh and D. Yang, "Constant scallop-height machining of free-form surfaces," *Journal of engineering for industry*, vol. 116, no. 2, pp. 253 – 259, 1994.

[72] A. Lasemi, D. Xue, and P. Gu, "Recent development in CNC machining of freeform surfaces: A state-of-the-art review," *Computer-Aided Design*, vol. 42, no. 7, pp. 641 – 654, 2010.

[73] V. Giri, D. Bezbaruah, P. Bubna, and A. R. Choudhury, "Selection of master cutter paths in sculptured surface machining by employing curvature principle," *International Journal of Machine Tools and Manufacture*, vol. 45, no. 10, pp. 1202 – 1209, 2005.

[74] B. Lauwers, G. Kiswanto, and J. P. Kruth, "Development of a five-axis milling tool path generation algorithm based on faceted models," *CIRP Annals - Manufacturing Technology*, vol. 52, no. 1, pp. 85 – 88, 2003.

[75] Y. S. Lee and H. Ji, "Surface interrogation and machining strip evaluation for 5-axis CNC die and mold machining," *International Journal of Production Research*, vol. 35, no. 1, pp. 225 – 252, 1997.

[76] Z. Han and D. C. Yang, "Iso-phote based tool-path generation for machining free-form surfaces," *Transactions-American Society of Mechanical Engineers Journal of Manufacturing Science and Engineering*, vol. 121, pp. 656 – 664, 1999.

[77] D. Yang and Z. Han, "Interference detection and optimal tool selection in 3-axis NC machining of free-form surfaces," *Computer-Aided Design*, vol. 31, no. 5, pp. 303 – 315, 1999.

[78] S. Ding, M. Mannan, A. Poo, D. Yang, and Z. Han, "Adaptive iso-planar tool path generation for machining of free-form surfaces," *Computer-Aided Design*, vol. 35, no. 2, pp. 141 – 153, 2003.

[79] K. Morishige, K. Kase, and Y. Takeuchi, "Collision-free tool path generation using 2-dimensional c-space for 5-axis control machining," *The International Journal of Advanced Manufacturing Technology*, vol. 13, no. 6, pp. 393 – 400, 1997.

[80] B. K. Choi, D. H. Kim, and R. B. Jerard, "C-space approach to tool-path generation for die and mould machining," *Computer-Aided Design*, vol. 29, no. 9, pp. 657 – 669, 1997.

[81] D. Yang, J. J. Chuang, and T. OuLee, "Boundary-conformed toolpath generation for trimmed free-form surfaces," *Computer-Aided Design*, vol. 35, no. 2, pp. 127 – 139, 2003.

[82] D. C. Yang, J. Chuang, Z. Han, and S. Ding, "Boundary-conformed toolpath generation for trimmed free-form surfaces via coons reparametrization," *Journal of Materials Processing Technology*, vol. 138, no. 1, pp. 138 – 144, 2003.

[83] W. Sun, C. Bradley, Y. Zhang, and H. Loh, "Cloud data modelling employing a unified, non-redundant triangular mesh," *Computer-Aided Design*, vol. 33, no. 2, pp. 183 – 193, 2001.

[84] Y. Ren, H. T. Yau, and Y.-S. Lee, "Clean-up tool path generation by contraction tool method for machining complex polyhedral models," *Computers in Industry*, vol. 54, no. 1, pp. 17 – 33, 2004.

[85] L. Biagiotti and C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*. Springer Publishing Company, Incorporated, 1st ed., 2008.

[86] J. W. Choi and G. H. Elkaim, "Bézier curve for trajectory guidance," *Proceedings of the World Congress on Engineering and Computer Science*, pp. 625 – 630, 2008.

[87] J. wung Choi, R. Curry, and G. Elkaim, "Path planning based on Bézier curve for autonomous ground vehicles," *Advances in Electrical and Electronics Engineering - IAENG Special Edition of the World Congress on Engineering and Computer Science*, pp. 158 – 166, 2008.

[88] S. Wang, L. Chen, H. Hu, and K. McDonald-Maier, "Doorway passing of an intelligent wheelchair by dynamically generating Bézier curve trajectory," *IEEE International Conference on Robotics and Biomimetics*, pp. 1206 – 1211, 2012.

[89] K. Yang and S. Sukkarieh, "An analytical continuous-curvature path-smoothing algorithm," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 561 – 568, 2010.

[90] M. Zhang, W. Yan, C. Yuan, D. Wang, and X. Gao, "Curve fitting and optimal interpolation on CNC machines based on quadratic B-splines," *Science China Information Sciences*, vol. 54, no. 7, pp. 1407 – 1418, 2011.

[91] Z. Yang, L.-Y. Shen, C.-M. Yuan, and X.-S. Gao, "Curve fitting and optimal interpolation for CNC machining under confined error using quadratic B-splines," *Computer-Aided Design*, vol. 66, pp. 62 – 72, 2015.

[92] A. Neto, D. Macharet, and M. Campos, "Feasible RRT-based path planning using seventh order Bézier curves," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1445 – 1450, 2010.

[93] T. Theoharis, G. Papaioannou, N. Platis, and N. M. Patrikalakis, *Graphics and Visualization: Principles & Algorithms*. Natick, MA, USA: A. K. Peters, Ltd., 2007.

[94] B. Sencer, K. Ishizaki, and E. Shamoto, "A curvature optimal sharp corner smoothing algorithm for high-speed feed motion generation of NC systems along linear tool paths," *The International Journal of Advanced Manufacturing Technology*, vol. 76, no. 9-12, pp. 1977 – 1992, 2015.

[95] M. Chen, W.-S. Zhao, and X.-C. Xi, "Augmented taylor's expansion method for B-spline curve interpolation for CNC machine tools," *International Journal of Machine Tools and Manufacture*, vol. 94, pp. 109 – 119, 2015.

[96] A. E. K. Mohammad and N. Uchiyama, "Estimation of tool orientation contour errors for five-axis machining," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 5, pp. 271 – 277, 2013.

[97] H.-C. Ho, J.-Y. Yen, and S.-S. Lu, "A decoupled path-following control algorithm based upon the decomposed trajectory error," *International Journal of Machine Tools and Manufacture*, vol. 39, no. 10, pp. 1619–1630, 1999.

[98] Y. Koren and C. Lo, "Advanced controllers for feed drives," *CIRP Annals - Manufacturing Technology*, vol. 41, no. 2, pp. 689 – 698, 1992.

[99] K. Nagaoka and T. Sato, "Feedforward controller for continuous path control of CNC machine tools," *Advanced Technology R&D Center, Mitsubishi Electric Corporation*, vol. 7, no. 8, pp. 39 – 46, 2006.

[100] S.-S. Yeh and J.-T. Sun, "Feedforward motion control design for improving contouring accuracy of CNC machine tools," *Proceedings of the International Multi-Conference of Engineers and Computer Scientists*, vol. 1, pp. 111 – 116, 2013.

[101] Y. Koren, "Cross-coupled biaxial computer control for manufacturing systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 265–272, 1980.

[102] K. Barton and A. Alleyne, "A cross-coupled iterative learning control design for precision motion control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1218 – 1231, 2008.

[103] R. Ramesh, M. Mannan, and A. Poo, "Tracking and contour error control in CNC servo systems," *International Journal of Machine Tools and Manufacture*, vol. 45, no. 3, pp. 301–326, 2005.

[104] S.-S. Yeh and P.-L. Hsu, "A new approach to bi-axial cross-coupled control," *Proceedings of the IEEE International Conference on Control Applications*, pp. 168 – 173, 2000.

[105] H.-S. Ahn, Y. Chen, and K. L. Moore, "Iterative learning control: Brief survey and categorization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1099 – 1121, 2007.

[106] Y. Liangliang, S. Weimin, and P. Laihu, "Research on feedforward parameter optimization of linear servo system based on iterative learning of orthogonal projection," *International Conference on Information Science and Control Engineering*, pp. 889 – 892, 2015.

[107] T. Haas, N. Lanz, R. Keller, S. Weikert, and K. Wegener, "Iterative learning for machine tools using a convex optimisation approach," *Procedia CIRP*, vol. 46, pp. 391 – 395, 2016.

[108] B. Zi, J. Lin, and S. Qian, "Localization, obstacle avoidance planning and control of a cooperative cable parallel robot for multiple mobile cranes," *Robotics and Computer-Integrated Manufacturing*, vol. 34, pp. 105 – 123, 2015.

[109] K. Kaltsoukalas, S. Makris, and G. Chryssolouris, "On generating the motion of industrial robot manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 32, pp. 65 – 71, 2015.

[110] T. Martinec, J. Mlýnek, and M. Petru, "Calculation of the robot trajectory for the optimum directional orientation of fibre placement in the manufacture of composite profile frames," *Robotics and Computer-Integrated Manufacturing*, vol. 35, pp. 42 – 54, 2015.

[111] N. Uchiyama, T. Hashimoto, S. Sano, and S. Takagi, "Obstacle avoidance control for a human-operated mobile robot," *10th IEEE International Workshop on Advanced Motion Control*, pp. 468 – 473, 2008.

[112] N. Uchiyama, T. Hashimoto, S. Sano, and S. Takagi, "Model-reference control approach to obstacle avoidance for a human-operated mobile robot," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 3892 – 3896, 2009.

[113] N. Uchiyama, T. Dewi, and S. Sano, "Collision avoidance control for a human-operated four-wheeled mobile robot," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 228, no. 13, pp. 2278 – 2284, 2014.

[114] N. Uchiyama, S. Sano, and A. Yamamoto, "Sound source tracking considering obstacle avoidance for a mobile robot," *Robotica*, vol. 28, pp. 1057 – 1064, 2010.

[115] S. Liu, D. Sun, and C. Zhu, "A dynamic priority based path planning for cooperation of multiple mobile robots in formation forming," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 6, pp. 589 – 596, 2014.

[116] M. Deng, A. Inoue, K. Sekiguchi, and L. Jiang, "Two-wheeled mobile robot motion control in dynamic environments," *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 3, pp. 268 – 272, 2010.

[117] J. M. Toibero, F. Roberti, R. Carelli, and P. Fiorini, "Switching control approach for stable navigation of mobile robots in unknown environments," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 3, pp. 558 – 568, 2011.

[118] R. Luo and C. C. Lai, "Multisensor fusion-based concurrent environment mapping and moving object detection for intelligent service robotics," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4043 – 4051, 2014.

[119] I. škrjanc and G. Klančar, "Optimal cooperative collision avoidance between multiple robots based on bernstein–Bézier curves," *Robotics and Autonomous Systems*, vol. 58, no. 1, pp. 1 – 9, 2010.

[120] Y. J. Ho and J. Liu, "Collision-free curvature-bounded smooth path planning using composite Bézier curve based on voronoi diagram," *IEEE International Symposium*

on Computational Intelligence in Robotics and Automation (CIRA), pp. 463 – 468, 2009.

[121] J. Perez, J. Godoy, J. Villagra, and E. Onieva, "Trajectory generator for autonomous vehicles in urban environments," *IEEE International Conference on Robotics and Automation*, pp. 409 – 414, 2013.

[122] K. Petrinec and Z. Kovacic, "The application of spline functions and Bézier curves to agv path planning," *Proceedings of the IEEE International Symposium on Industrial Electronics*, vol. 4, pp. 1453 – 1458, 2005.

[123] J. Hoschek and D. Lasser, *Fundamentals of Computer Aided Geometric Design*. Natick, MA, USA: A. K. Peters, Ltd., 1993.

[124] J. Gravesen, "Adaptive subdivision and the length and energy of Bézier curves," *Computational Geometry*, vol. 8, no. 1, pp. 13 – 31, 1997.

[125] P. Bézier, *Numerical control : mathematics and applications*. London ; New York : J. Wiley, 1972.

[126] T. Do, T. Tjahjowidodo, M. Lau, and S. Phee, "Real-time enhancement of tracking performances for cable-conduit mechanisms-driven flexible robots," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 197 – 207, 2016.

[127] D. Hegels, T. Wiederkehr, and H. Müller, "Simulation based iterative post-optimization of paths of robot guided thermal spraying," *Robotics and Computer-Integrated Manufacturing*, vol. 35, pp. 1 – 15, 2015.

[128] C. Wang, Y. Zhao, Y. Chen, and M. Tomizuka, "Nonparametric statistical learning control of robot manipulators for trajectory or contour tracking," *Robotics and Computer-Integrated Manufacturing*, vol. 35, pp. 96 – 103, 2015.

[129] Y. Singh, V. Vinoth, Y. R. Kiran, J. K. Mohanta, and S. Mohan, "Inverse dynamics and control of a 3-DOF planar parallel (u-shaped 3-PPR) manipulator," *Robotics and Computer-Integrated Manufacturing*, vol. 34, pp. 164 – 179, 2015.

[130] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 384 – 389, 1990.

[131] G. Oriolo, A. De Luca, and M. Vendittelli, "Wmr control via dynamic feedback linearization: design, implementation, and experimental validation," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 6, pp. 835 – 852, 2002.

[132] A. De Luca, G. Oriolo, and M. Vendittelli, "Control of wheeled mobile robots: An experimental overview," *Ramsete* (S. Nicosia, B. Siciliano, A. Bicchi, and P. Valigi, eds.), vol. 270 of *Lecture Notes in Control and Information Sciences*, pp. 181 – 226, Springer Berlin Heidelberg, 2001.

[133] J. J. E. Slotine and W. Li, *Applied nonlinear control*, vol. 199. Prentice-hall Englewood Cliffs, NJ, 1991.

[134] C. Chen, Y. He, C. Bu, J. Han, and X. Zhang, "Quartic Bézier curve based trajectory generation for autonomous vehicles with curvature and velocity constraints," *IEEE International Conference on Robotics and Automation*, pp. 6108 – 6113, 2014.

[135] L. Moctezuma, A. Lobov, and J. Lastra, "Free shape paths in industrial robots," *38th Annual Conference on IEEE Industrial Electronics Society*, pp. 3739 – 3743, 2012.

[136] A. Chandak, K. Gosavi, S. Giri, S. Agrawal, and M. P. Kulkarni, "Path planning for mobile robot navigation using image processing," *International Journal of Scientific and Engineering Research*, vol. 4, no. 6, pp. 1490 – 1496, June.

[137] E. Masehian and M. R. Amin-Naseri, "A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning," *Journal of Robotic Systems*, vol. 21, no. 6, pp. 275 – 300, 2004.

[138] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robotics and Autonomous Systems*, vol. 64, pp. 1 – 20, 2015.

[139] Y. Petillot, I. Ruiz, D. Lane, Y. Wang, E. Trucco, and N. Pican, "Underwater vehicle path planning using a multi-beam forward looking sonar," *OCEANS Conference Proceedings*, vol. 2, pp. 1194 – 1199, 1998.

[140] J. Kim and Y. Do, "Moving obstacle avoidance of a mobile robot using a single camera," *Procedia Engineering*, vol. 41, pp. 911 – 916, 2012.

[141] J. J. Kumler and M. L. Bauer, "Fish-eye lens designs and their relative performance," *Current Developments in Lens Design and Optical Systems Engineering* (R. E. Fischer, R. B. Johnson, W. J. Smith, and W. H. Swantner, eds.), vol. 4093, pp. 360 – 369, 2000.

[142] J. G. Fryer and D. C. Brown, "Lens distortion for close-range photogrammetry," *Photogrammetric Engineering and Remote Sensing*, vol. 52, pp. 51 – 58, 1986.

[143] O. Faugeras, *Three-dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, MA, USA: MIT Press, 1993.

[144] K. G. Derpanis, "The harris corner detector," 2004.

[145] H. Araujo and J. Dias, "An introduction to the log-polar mapping [image sampling]," *Proceedings of Second Workshop on Cybernetic Vision*, pp. 139 – 144, 1996.

[146] W. Pan, K. Qin, and Y. Chen, "An adaptable-multilayer fractional fourier transform approach for image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 400–414, 2009.

[147] A. Saudi and J. Sulaiman, "Red-black strategy for mobile robot path planning," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 3, pp. 2215 – 2220.

[148] Y. M. Hendrawan, K. R. Simba, and N. Uchiyama, "Embedded iterative learning contouring controller design for biaxial feed drive systems," *International Electronics Symposium*, pp. 37–41, 2016.

[149] M. R. Msukwa, N. Uchiyama, and B. D. Bui, "Adaptive nonlinear sliding mode control with a nonlinear sliding surface for feed drive systems," *IEEE International Conference on Industrial Technology*, pp. 732 – 737, 2017.

[150] A. Affouard, C. Tournier, S. Lavernhe, and C. Lartigue, "Description formats of tool trajectory suited to High-Speed Machining," *International Conference on High Speed Machining*, (Nanjing, China), 2004.

[151] G. Cong and W. Yuhou, "An interpolation method based on tool orientation fitting in five-axis CNC machining," *IEEE 14th International Conference on Industrial Informatics*, pp. 213 – 218, 2016.

[152] M. T. Lin, M. C. Lee, J. C. Lee, C. Y. Lee, and Z. W. Jian, "A look-ahead interpolator with curve fitting algorithm for five-axis tool path," *IEEE International Conference on Advanced Intelligent Mechatronics*, pp. 189 – 194, 2016.

[153] B. D. Bui, N. Uchiyama, and K. R. Simba, "Contouring control for three-axis machine tools based on nonlinear friction compensation for lead screws," *International Journal of Machine Tools and Manufacture*, vol. 108, pp. 95 – 105, 2016.

[154] B. Armstrong-Hélouvry, P. Dupont, and C. C. D. Wit, "A survey of models, analysis tools and compensation methods for the control of machines with friction," *Automatica*, vol. 30, no. 7, pp. 1083 – 1138, 1994.

[155] K. Zhang, A. Yuen, and Y. Altintas, "Pre-compensation of contour errors in five-axis CNC machine tools," *International Journal of Machine Tools and Manufacture*, vol. 74, pp. 1 – 11, 2013.

[156] A. E. Khalick and N. Uchiyama, "Contouring controller design based on iterative contour error estimation for three-dimensional machining," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 4, pp. 802 – 807, 2011.

[157] D. Wang and Y. Ye, "Design and experiments of anticipatory learning control: frequency-domain approach," *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 3, pp. 305 – 313, 2005.

[158] D. Prévost, S. Lavernhe, C. Lartigue, and D. Dumur, "Feed drive modelling for the simulation of tool path tracking in multi-axis high speed machining," *CoRR*, vol. abs/1107.3229, 2011.

[159] K. R. Simba, N. Uchiyama, and S. Sano, "Iterative contouring controller design for biaxial feed drive systems," *20th Conference on Emerging Technologies & Factory Automation*, pp. 1 – 5, 2015.

[160] B. D. Bui, N. Uchiyama, and S. Sano, "Nonlinear friction modeling and compensation for precision control of a mechanical feed-drive system," *Sensors and Materials*, vol. 27, no. 10, pp. 971 – 984, 2015.

[161] F. Al-Bender, V. Lampaert, and J. Swevers, "The generalized maxwell-slip model: a novel model for friction simulation and compensation," *IEEE Transactions on Automatic Control*, vol. 50, no. 11, pp. 1883 – 1887, 2005.

[162] Y. Li, Q. Zheng, and L. Yang, "Design of robust sliding mode control with disturbance observer for multi-axis coordinated traveling system," *Computers & Mathematics with Applications*, vol. 64, no. 5, pp. 759 – 765, 2012.

[163] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Systems*, vol. 26, no. 3, pp. 96 – 114, 2006.

[164] M. Norrlöf and S. Gunnarsson, "Time and frequency domain convergence properties in iterative learning control," *International Journal of Control*, vol. 75, no. 14, pp. 1114 – 1126, 2002.

[165] F. Padieu and R. Su, "An H∞ approach to learning control systems," *International Journal of Adaptive Control and Signal Processing*, vol. 4, no. 6, pp. 465 – 474, 1990.

[166] H. K. Khalil, "Nonlinear systems, 3rd," *New Jewsey, Prentice Hall*, vol. 9, 2002.

[167] D. A. Bristow and A. G. Alleyne, "Monotonic convergence of iterative learning control for uncertain systems using a time-varying q-filter," *Proceedings of the American Control Conference,*, pp. 171 – 177, 2005.

[168] T. Hashikawa and Y. Fujisaki, "Convergence conditions of iterative learning control revisited: A unified viewpoint to continuous-time and discrete-time cases," *IEEE International Symposium on Intelligent Control*, pp. 31 – 34, 2013.

[169] L. Hladowski, K. Galkowski, Z. Cai, E. Rogers, C. T. Freeman, and P. L. Lewin, "Experimentally supported 2d systems based iterative learning control law design for error convergence and performance," *Control Engineering Practice*, vol. 18, no. 4, pp. 339–348, 2010.

[170] K. L. Moore, *Iterative learning control for deterministic systems*. Springer Science & Business Media, 2012.

[171] R. W. Longman, "Iterative learning control and repetitive control for engineering practice," *International journal of control*, vol. 73, no. 10, pp. 930 – 954, 2000.

[172] A. E. K. Mohammad, N. Uchiyama, and S. Sano, "Sliding mode contouring control design using nonlinear sliding surface for three-dimensional machining," *International Journal of Machine Tools and Manufacture*, vol. 65, pp. 8 – 14, 2013.