Model based Feed-rate Optimization for Machine Tool Trajectories

A dissertation submitted to the
ETH ZURICH

for the degree of
Dr. sc. ETH Zürich

presented by
MARKUS STEINLIN
Dipl. Masch.-Ing. ETH
born March 27th 1980
citizen of Switzerland

accepted on the recommendation of
Prof. Dr. K. Wegener, examiner
Prof. Dr. O. Zirn, co-examiner

2013
Acknowledgments

This thesis was realized during my time at inspire AG and the Institute for Machine Tools and Manufacturing (IWF) of the ETH Zurich. The cooperation with my co-workers and with the industrial project partners greatly contributed to my work.

I would like to thank Prof. Dr. Wegener, head of the IWF and supervisor of this thesis for his generous support, constructive suggestions and for his trust in me and my work.

Research in machine tool engineering is not a thing you can do alone. My group leader Sascha Weikert and my co-supervisor Oliver Zirn supported my research and thesis and contributed with valuable suggestions and ideas.

I would also like to thank my friends and colleagues who supported me with creative problem solving, motivated me and sometimes challenged me with critical remarks. I also like to remember the times we spent together outside of work.

Last but not least I thank my parents, my sisters and my wife for their support and encouragement.

Markus Steinlin
Zurich, March 2013
Contents

Symbols and Abbreviations X

Abstract XV

Zusammenfassung XVII

1 Introduction 1
  1.1 Outline of the Thesis 3

2 State of the Art 5
  2.1 Optimization Problem 5
    2.1.1 Nonlinear Programming Problem 6
    2.1.2 Optimal Control Problem as a NLP Problem 6
    2.1.3 Quadratic Programming Problem 7
    2.1.4 Discrete Minimum Time Optimal Control Problem 7
    2.1.5 Integration Schemes 8
  2.2 Mathematical Description of Path Trajectories 8
    2.2.1 Path Dynamics 9
    2.2.2 Path Geometry 10
  2.3 The Quality of Trajectories 11
    2.3.1 Exploitation of the Geometrical Tolerances 12
    2.3.2 Physically Feasible Trajectories 12
    2.3.3 Minimum Excitation of the Machine Tool Structure 12
2.4 Optimal Path Tracking Algorithms ............................................. 14
  2.4.1 Phase Plane Methods (Indirect Methods) ............................... 15
  2.4.2 Direct Transcription Methods (Direct Methods) ......................... 16
2.5 Numerical Control for Machine Tools ........................................... 19
  2.5.1 Closed Loop Control for Machine Tools ................................. 20
  2.5.2 Trajectory Generation for Machine Tools ............................... 22
  2.5.3 Feed-rate Optimization Algorithm for Arbitrarily Connected Joints
      and Tangential Geometries .................................................. 24
  2.5.4 Feed-rate Optimization Algorithms for Geometries with a Continuous
      Curvature ................................................................. 26
  2.5.5 Feed-rate Optimization for Under-determined Kinematic Systems ... 26
    2.5.5.1 Robotics: Inverse Kinematics Problem ................................ 27
    2.5.5.2 Path Separation Strategies ......................................... 28
    2.5.5.3 Trajectory Separation Strategies ................................... 29
    2.5.5.4 Minimum Time Optimal Control Strategies ......................... 29
  2.6 Compensation of Cross-talk .................................................. 29
  2.7 Motivation .................................................................. 31

3 Feed-rate Optimization for Machine Tools ........................................ 33
  3.1 The Feed-rate Optimization Algorithm ...................................... 33
    3.1.1 Parametric Description of the Dynamics ............................... 33
    3.1.2 Constraint Formulation .................................................. 35
    3.1.3 Parametric / Geometric Path Sequence Switching Condition ........ 35
      3.1.3.1 Parametric Switching of Sequences ................................. 35
      3.1.3.2 Geometric Switching Condition ................................... 37
    3.1.4 Conception of the \textit{feedrateOptim} Algorithm .................... 37
      3.1.4.1 Direct Transcription Formulation .................................. 37
      3.1.4.2 Numerical Optimization with \textit{fmincon} ......................... 40
      3.1.4.3 Identification of Optimality ....................................... 43
      3.1.4.4 Computational Accuracy .......................................... 44
3.2 Optimal Trajectories ........................................ 45
  3.2.1 Influence of Path Rounding ............................. 45
    3.2.1.1 Rounding by Low-pass Behavior of the Controller 45
    3.2.1.2 Rounding by Geometrical Preprocessing .......... 46
  3.2.2 Acceleration, Jerk and Jerk-rate Limitations .......... 47
3.3 Achievement in Feed-rate Optimization .................... 48
  3.3.0.1 Reliability ........................................ 48

4 Feed-rate Optimization for an Under-determined Kinematic System 50
  4.1 Characteristics of an Under-determined Kinematic System .... 50
  4.2 Conception of an Under-determined Kinematic System ........ 52
  4.3 Under-determined Systems: Trade-offs .................... 56
  4.4 The axOptim Separation Method for an Under-determined Kinematic System 56
    4.4.1 Definition of the Master Trajectory .................. 57
    4.4.2 Constraint Formulation .............................. 58
    4.4.3 Conception of the axOptim Algorithm ................ 59
      4.4.3.1 Discretization .................................. 59
      4.4.3.2 Numerical Optimization with quadprog .......... 61
      4.4.3.3 Identification of Optimality .................... 62
    4.4.4 Example: Trajectory Separation with axOptim ........ 63
  4.5 General Approach for Trajectory Generation ............... 68
    4.5.1 Path Description of the Main Subsystem ............... 68
    4.5.2 Parametric Description of the Dynamics ............... 68
    4.5.3 Conception of the feedrateOptimUDKS Algorithm ....... 70
      4.5.3.1 Discretization .................................. 70
      4.5.3.2 Objective Function and the Minimum Set of Constraints 72
      4.5.3.3 From the Nonlinear Program to the Numerical Optimization with fmincon 72
      4.5.3.4 Identification of Optimality .................... 73
4.5.4 Application of the Algorithm ........................................... 73
  4.5.4.1 Unaffected Master Trajectory .................................... 73
  4.5.4.2 Dynamic of the Main Subsystem affects the Master Trajectory ......................................................... 74
4.5.5 Application Limitations .................................................. 75
4.6 Region of Application ...................................................... 81
  4.6.1 Large Geometries and Reliability .................................... 81
4.7 Interaction of the Subsystems .......................................... 81
  4.7.1 Controller Response and Bandwidth of the Control System ........................................ 82
  4.7.2 Dynamical Interaction .................................................. 82
5 Open loop Compensation for Cross-talk ............................... 84
  5.1 Measurement - Identification of Cross-talk ......................... 84
  5.2 Modeling of Position Dependency ..................................... 87
  5.3 Compensation Algorithm ............................................... 89
  5.4 Dynamic Benchmark .................................................... 91
  5.5 Prove of Concept ....................................................... 91
    5.5.1 Using an Industrial NC ......................................... 91
    5.5.2 Measurement Results ............................................ 93
6 Conclusion and Outlook .................................................. 94
  6.1 Feed-rate Optimization ................................................ 94
  6.2 Under-determined Kinematic Systems ................................ 96
Appendix ................................................................. 98
A Equations ................................................................. 98
  A.1 Jerk-Limited Positioning Movement ................................. 98
  A.2 Working Area for a Square Geometry .............................. 99
B Optimal Solutions ................................................................ 101
B.1 \textit{feedrateOptim} used with Different Limitations \hfill 101

Bibliography \hfill 104

List of Publications \hfill 114
Symbols, Abbreviations and Glossary

Symbols

\( x^{(i)} \) \( i \text{th derivative of } x \) with respect to the time
\( a \) \( \text{Acceleration variable} \)
\( A \) \( \text{Total working area of the dynamic subsystem} \)
\( A_{eq} \) \( \text{Linear equality constraint function: } A_{eq}x - b_{eq} = 0 \)
\( A_{ineq} \) \( \text{Linear inequality constraint function: } A_{ineq}x - b_{ineq} \leq 0 \)
\( a_{max} \) \( \text{Maximum acceleration} \)
\( b_{eq} \) \( \text{Linear equality constraint function: } A_{eq}x - b_{eq} = 0 \)
\( b_{ineq} \) \( \text{Linear inequality constraint function: } A_{ineq}x - b_{ineq} \leq 0 \)
\( c(x) \) \( \text{Nonlinear constraint function of an optimization problem} \)
\( c_{ct} \) \( \text{Cross-talk proportional factor} \)
\( D \) \( \text{Relative damping} \)
\( \Delta \) \( \pm \Delta \text{ defines the working area of the dynamic subsystem} \)
\( \Delta_x \) \( \text{Axial TCP offset} \)
\( \Delta_y \) \( \text{Lateral TCP offset} \)
\( EY \dot{X} \) \( \text{Dynamic straightness} \)
\( F \) \( \text{Force variable} \)
\( f_0 \) \( \text{Predominant resonant frequency} \)
\( f_{overload} \) \( \text{Overload factor} \)
\( F_x \) \( \text{Actuator force} \)
\( g \) \( \text{Set of inequality constraints} \)
\( G_i \) \( \text{Transfer function of the current control loop} \)
\( h \) \( \text{Set of nonlinear equality constraints} \)
\( h_k \) \( \text{Time interval between two discretized states} \)
\( j \) \( \text{Jerk variable} \)
\( J \) \( \text{Objective function of an optimization problem} \)
\( j_r \) \( \text{Jerk-rate variable} \)
\(\kappa\)  Weight for the jerk reducing term of the objective function

\(k_{C,\text{rot}}\)  Representative rotational stiffness (stiffness of the guideways)

\(k_P\)  Velocity control gain

\(k_V\)  Positioning gain (servo gain)

\(m\)  Mass moved by an axis

\(m(s)\)  Path curve of the main subsystem

\(M\)  Total number of discretization steps

\(\mu\)  Viscous friction coefficient

\(\psi\)  Set of equality constraints

\(q\)  Axes related coordinates

\(r(s)\)  Path curve

\(r_0\)  Jerk limitation

\(s\)  Path parameter (path coordinate)

\(T\)  Linear axes transformation for a under-determined kinematic system

\(t\)  Nonlinear axes transformation for a under-determined kinematic system

\(T\)  Time segment variable

\(t\)  Time variable

\(t_F\)  Final time, process duration

\(\Theta\)  Physical limitation

\(T_{\text{rise}}\)  Time needed to reach \(a_{max}\) with a jerk or jerk-rate limited phase

\(u(t)\)  Control function of an optimal control problem

\(v\)  Velocity variable

\(\bar{v}_{\text{cut}}\)  Mean cutting velocity

\(v_{\text{max,cut}}\)  Maximum cutting velocity

\(x\)  Position variable

\(x\)  TCP related coordinates for a under-determined kinematic system

\(x\)  Vector of unknown states of an optimization problem

\(^jX_i\)  Subsystem i in the coordinate system j

\(^jx_i\)  X-axis of subsystem i in the coordinate system j

\(x^*\)  Optimal solution

\(y_k\)  State vector of the \(k^{th}\) discretization state

\(^jy_i\)  Y-axis of subsystem i in the coordinate system j

\(\zeta\)  Set of equality constraints which solve the state equation
Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAD</td>
<td>Computer aided design</td>
</tr>
<tr>
<td>CAM</td>
<td>Computer aided manufacturing</td>
</tr>
<tr>
<td>CNC</td>
<td>Computerized numerical control</td>
</tr>
<tr>
<td>CTC</td>
<td>Computed torque control</td>
</tr>
<tr>
<td>DCG</td>
<td>Driven at the center of gravity</td>
</tr>
<tr>
<td>DGO</td>
<td>Discrete geometry optimization</td>
</tr>
<tr>
<td>ETH</td>
<td>Eidgenössische Technische Hochschule Zürich</td>
</tr>
<tr>
<td>FEM</td>
<td>Finite element method</td>
</tr>
<tr>
<td>GA</td>
<td>Generic algorithm</td>
</tr>
<tr>
<td>NC</td>
<td>Numerical control</td>
</tr>
<tr>
<td>NLP</td>
<td>Nonlinear program</td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse width modulation</td>
</tr>
<tr>
<td>QP</td>
<td>Quadratic programming</td>
</tr>
<tr>
<td>UDKS</td>
<td>Under-determined kinematic system</td>
</tr>
<tr>
<td>TCP</td>
<td>Tool center point</td>
</tr>
</tbody>
</table>

Glossary

Boundary condition (in this thesis) includes all the constraints that are applied on the problem.

Closed loop control [Regelung]; control with state feedback of the actual position.

CNC (computerized numerical control) also NC, automatic control of a process performed by a device that makes use of numerical data introduced while the operation is in progress [29].

Constraint A constraint is a condition that a solution to an optimization problem must satisfy. There are two types of constraints: Equality constraints and inequality constraints. The set of solutions that satisfy all constraints is called the feasible set [107].

Control system Semiclosed- or closed loop controller of a machine tool.

Decoupled approach The geometry optimization problem is solved prior to the feed-rate optimization problem, in contrast to the direct approach.

Direct (transcription) methods The feed-rate optimization problem is solved with a numerical solver at once. Can handle a general problem formulation.
<table>
<thead>
<tr>
<th><strong>Direct approach</strong></th>
<th>The approach solves the motion planning problem directly, meaning the feed-rate optimization problem is solved together with the geometry optimization problem.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dynamic programming methods</strong></td>
<td>The actual problem is splitted in several easy subproblems</td>
</tr>
<tr>
<td><strong>Equivalent time constant</strong></td>
<td>Time constant representing a transfer function. For example the step response time of the current controller.</td>
</tr>
<tr>
<td><strong>Feed drive</strong></td>
<td>[Antriebsstrang]; drive chain, includes motor, transmission and table</td>
</tr>
<tr>
<td><strong>Feed-forward control</strong></td>
<td>[Vorsteuerung]; velocity and/or acceleration set-point values are preprocessed and added to the control deviation of the position and/or the velocity control loop.</td>
</tr>
<tr>
<td><strong>Feed-rate</strong></td>
<td>[Vorschub]; process speed programmed in the NC-program.</td>
</tr>
<tr>
<td><strong>Feed-rate optimization</strong></td>
<td>Sub-step to the trajectory generation where the feed-rate is optimized.</td>
</tr>
<tr>
<td><strong>Geometry optimization</strong></td>
<td>Sub-step to the trajectory generation where only the geometry is optimized.</td>
</tr>
<tr>
<td><strong>Indirect methods</strong></td>
<td>Feed-rate optimization algorithms that use one-dimensional search algorithms to find switching points.</td>
</tr>
<tr>
<td><strong>Infeasible states</strong></td>
<td>states in a problem description which do not satisfy the constraints.</td>
</tr>
<tr>
<td><strong>Interpolation</strong></td>
<td>determination of points intermediate between known points on a desired path or contour in accordance with a given mathematical function (linear, circular or higher order functions [29]. e.g.)</td>
</tr>
<tr>
<td><strong>Jerk</strong></td>
<td>Third time derivative of position $\frac{d^3x}{dt^3}$ [108]</td>
</tr>
<tr>
<td><strong>Jerk-rate</strong></td>
<td>see snap, fourth time derivative of position $\frac{d^4x}{dt^4}$, used in this thesis due to inconsistent use of snap, jounce and yank in literature.</td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Machine properties limit the behavior of the trajectory. In the context of optimization the limitation is transformed to a constraint equation.</td>
</tr>
<tr>
<td><strong>Motion planning</strong></td>
<td>process by which the robot control program determines how to move the joints of the mechanical structure between the poses programmed by the user, according to the type of interpolation chosen [55]. Expression used in a mathematical context.</td>
</tr>
<tr>
<td><strong>Open loop control</strong></td>
<td>[Steuerung]; prediction of the feed-rate with no feedback loop</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Path coordinate</td>
<td>runs along a path function, usually indicated as $s$</td>
</tr>
<tr>
<td>Path tracking</td>
<td>Equivalent to feed-rate optimization. Expression used in a mathematical context.</td>
</tr>
<tr>
<td>Physically feasible trajectory</td>
<td>A machine tool is feed with a physically feasible trajectory in a way that the closed loop controller only regulates the model inaccuracies. In the context of this thesis the model represents an actuator with low-pass behavior and an one-mass oscillator.</td>
</tr>
<tr>
<td>Regulator</td>
<td>Closed loop controller (former expression)</td>
</tr>
<tr>
<td>Semi closed loop control</td>
<td>Closed loop control with an indirect measurement system.</td>
</tr>
<tr>
<td>Set-point</td>
<td>[Sollwert]; desired value of the actuator/machine (measurement of motor position in case of a ball screw drive e.g.).</td>
</tr>
<tr>
<td>Snap</td>
<td>also jounce, fourth time derivative of position $\frac{d^4x}{dt^4}$ [109]</td>
</tr>
<tr>
<td>Tracking error</td>
<td>[Schleppfehler] Distance between the actual position of an axis and the set-point position</td>
</tr>
<tr>
<td>Trajectory</td>
<td>path in function of time (according to [55])</td>
</tr>
<tr>
<td>Trajectory generation</td>
<td>The path and feed-rate provided by the NC program is optimized by the trajectory generation. The resulting trajectory satisfies the demands to the trajectory generation algorithm. In machine tool context the trajectory generation consists of geometry- and a feed-rate optimization step.</td>
</tr>
<tr>
<td>Trajectory generation (algorithm)</td>
<td>[Führungsgrössengenerator]; Algorithm to define the trajectories for the closed loop controller. Mostly consists of a geometry optimization and a feed-rate optimization step.</td>
</tr>
<tr>
<td>Yank</td>
<td>Derivative of force to time $\frac{dF}{dt}$ [110]</td>
</tr>
</tbody>
</table>
Abstract

Productivity of a machine tool can be significantly influenced by the dynamic of the axes. The dynamic of the axes is maximized under the condition that the product results in the required quality. The properties of the machine tool structure, the drive chain and the process define the boundary conditions for the maximization of the productivity. Due to the dependences of the dynamic properties these boundary conditions cannot be reached permanently.

A feed-rate optimization algorithm minimizes the process time (by maximizing the velocity) for a given geometry (decoupled approach) while respecting the boundary conditions determined by the machine’s properties.

This thesis presents a new feed-rate optimization algorithm which determines velocity, acceleration, force, jerk and jerk-rate limited trajectories. All limitations are realized using a physical description of the limitation at every position along the trajectory. The algorithm uses a direct method and formulates a discrete minimum time optimal control problem which can be solved with a standard solver for optimization problems. Optimal trajectories generated with the feed-rate optimization algorithm are demonstrated.

Dynamic deviations in orthogonal direction to the moving direction (cross-talk) can be compensated by a measurement- and model-based method. It is demonstrated that this compensation method reduces the (lateral) path deviation by a factor two while maintaining the same productivity.

This thesis further presents two trajectory generation algorithms for under-determined kinematic systems. The first approach is a novel optimal algorithm which separates a given trajectory at the tool center point (TCP) into two trajectories for two serially arranged linear subsystems. The separation is such that the excitation of the machine is minimized.

The second approach determines the trajectories for the subsystems from a discrete minimum time optimal control problem. The formulation is based on axis-wise limitations of the under-determined kinematic system.
This thesis shows that feed-rate optimization with limitations based on a physical model of a machine tool is viable at least for limitations up to the 4th derivative of the position with respect to time. Further, it is proven that this approach is also viable for under-determined kinematics systems.
Zusammenfassung

Die Produktivität einer Werkzeugmaschine kann durch die Dynamik der Achsen signifikant beeinflusst werden. Die Dynamik wird unter der Bedingung maximiert, dass das Produkt die erforderliche Qualität aufweist. Die Eigenschaften der Maschinenstruktur, des Antriebsstrangs und des Prozesses definieren die Randbedingungen für die Maximierung der Produktivität. Aufgrund der Abhängigkeiten zwischen den dynamischen Eigenschaften können diese Randbedingungen nicht dauerhaft erreicht werden.

Ein Führungsgrössenalgorithmus minimiert die Prozessdauer (respektive maximiert die Geschwindigkeit) für eine vorgegebene Geometrie (entkoppelter Ansatz) unter Berücksichtigung der Randbedingungen welche anhand der Eigenschaften der Maschine definiert werden.


Dynamische Bahnabweichungen senkrecht zur Bewegungsrichtung (Cross-talk) können mit einer mess- und modellbasierten Methode kompensiert werden. Es wird gezeigt, dass diese Kompensationsmethode die (laterale) Bahnabweichung bei gleichbleibender Produktivität um die Hälfte reduziert.


Der zweite Ansatz bestimmt die Führungsgrössen für die Subsysteme anhand eines diskre-

Die vorliegende Arbeit zeigt, dass Führungsgrössenoptimierung mit Begrenzungen bis mindestens zur vierten zeitlichen Ableitung der Position angewendet werden kann. Ausserdem wird dargelegt dass dieser Ansatz auch für unterbestimmte kinematische Systeme geeignet ist.
Chapter 1

Introduction

The main focus of this thesis is on how machine tools can produce workpieces of a required quality with maximum productivity. The production process from the workpiece design to the final workpiece is illustrated in figure 1.1.

This simplified overview shows the workpiece design which is created with a computer aided design program (CAD). The design is then exported into the workpiece program code (NC-code). In general, the NC-code does not include volume and surface information but only the path of the tool center point (TCP) relative to the workpiece. Additionally, the NC-code can include information about tolerances, dynamic limitations (e.g. maximal feed-rates) and control commands for the process and the automation. This NC-code is processed in the computerized numerical control (CNC). The control converts the programmed input into machine commands. Additionally, it controls the relative movement of the tool and the workpiece as well as the process and the automation. Figure 1.1 indicates an interaction of the next sequences. The control commands the actuators of the machine which interact with the structure of the machine tool and the process. This interaction is observed by a measurement system and fed back to the control which reacts to the interaction. For the productivity of the machine tools the interaction between the CNC, the machine structure and the process is important.
When focusing on the movement of the machine axes including the interaction of the CNC and the machine structure two main function units of the CNC can be distinguished:

First, the trajectory generation transforms the path (NC-code) into a geometry and velocity of a movement (trajectory). The second function unit is a semiclosed or closed loop control system. It is responsible for the trajectory to be followed by the TCP through the actuators. The controller feedback depends on the machine setup: It observes the movement measured directly (close to the TCP) or indirectly by the measurement system of the drive.

![Diagram of trajectory generation](image)

**Figure 1.2**: Trajectory generation consists of a geometry optimization and a feed-rate optimization step.

The trajectory generation itself again consists of two sub-steps as illustrated in figure 1.2: First, the path given by the NC-code is optimized by eliminating the geometrical discontinuities. In a second step, a feed-rate optimization algorithm determines the trajectory along this geometry.

Feed-rate optimization adds the dynamic to the geometry. To maximize the productivity of a machine tool the velocity of the machine tool is maximized under the condition that the workpiece results in the required quality. The properties of the machine tool structure, the drive chain and the process define the boundary conditions for the maximization of the productivity. Due to the dependences of the dynamic properties these boundary conditions cannot be reached permanently.
Various boundary conditions can be defined for the machine tool structure, the drive chain and the process. The most important boundary conditions for the productivity maximization are the maximum velocity, the acceleration and the actuator force of a machine tool axis as well as jerk and jerk-rate limitations to reduce the excitation of the machine tool structure. This thesis demonstrates a feed-rate optimization algorithm which respects these boundary conditions and provides a trajectory with a minimized process time.

The increase of productivity usually goes along with an increase of the excitation of the machine tool structure and elastic tilting, referred to as cross-talk, caused by higher maximal actuator forces. Movements with high actuator forces need set-point filters, jerk or jerk-rate limitations to reduce structural excitation.

**Cross-talk** is a path deviation orthogonal to the direction of motion. This deviation is proportional to the acceleration in the direction of motion. It cannot be approached with a feed-rate optimization algorithm without a reduction of the acceleration but compensating this deviation is a possible remedy. This thesis presents an open loop compensation for cross-talk which allows to reduce the path deviation while increasing the productivity.

An approach to avoid higher maximal actuator forces while maintaining the high productivity is the reduction of the actuated mass. This can be realized by a machine tool design with an under-determined kinematic system. It provides the high dynamic with light and short axes and guarantees a large workspace with a heavier but less dynamic axis system.

Trajectory generation for machine tools with under-determined kinematic systems poses an additional challenge. The path given by the NC-code has less dimensions than the number of available actuators. For a standard feed-rate optimization algorithm the path has to be defined for every axis. A feed-rate optimization algorithm for an under-determined kinematic system separates the path depending on the dynamic properties of the axes. This thesis provides two feed-rate optimization algorithms for under-determined kinematic systems. These optimization algorithms are demonstrated using limitations for the working area, the velocity and the acceleration.

### 1.1 Outline of the Thesis

The state of the art is presented in chapter 2. It presents an overview of known methods for feed-rate optimization. In section 2.1 a general formulation of an optimization problem is introduced and based on this general formulation an optimal control problem formulation is derived. The mathematical description of a path geometry and dynamics is given in
section 2.2. Section 2.3 discusses the demands on the feed-rate optimization by analyzing the quality of trajectories. Different approaches to optimize trajectories can be found in section 2.5. This section starts with an overview of trajectory generation for machine tools in 2.5.2 and in 2.5.2 it focuses on different feed-rate optimization algorithms. An overview of commonly used optimization algorithms is presented in section 2.4. In 2.5.5, the topic of under-determined kinematic systems is introduced. Section 2.6 discusses known methods to reduce cross-talk. The last section (2.7) of this chapter describes the research gap that is approached in this thesis.

Chapter 3 discusses feed-rate optimizaton for machine tools by introducing an optimal feed-rate algorithm to avoid discontinuity in the force (acceleration) profile of the trajectory.

Chapter 4 deals with feed-rate optimization for an under-determined kinematic systems. An optimal separation method which uses a predefined master trajectory is presented in section 4.4. Section 4.5 shows an extended feed-rate optimization approach to define the trajectories of the under-determined kinematic system without a predefined master trajectory. In section 4.6 and 4.7 the two optimization algorithms for under-determined kinematic systems are compared and their challenges for a realization on a machine tool are discussed.

Chapter 5, addresses open-loop compensation of cross-talk and demonstrates a novel identification and quantification method for cross-talk is demonstrated.

Chapter 6 summarizes the results and gives an outlook on required further research work.
Chapter 2

State of the Art

The chapter *State of the Art* presents an overview of known methods for feed-rate optimization. In section 2.1, a general formulation of an optimization problem is introduced and from this general formulation an optimal control problem formulation is derived. The mathematical description of a path geometry and dynamics is given in section 2.2 *Mathematical Description of Path Trajectories*. Section 2.3 *Quality of Trajectories* discusses the demands on the feed-rate optimization. Different approaches to optimize trajectories can be found in section 2.5. This section starts with *Trajectory Generation for Machine Tools* and then focuses on different feed-rate optimization algorithms. Further, an overview of known optimization algorithms is given and recent research on under-determined kinematic systems is introduced. *Compensation of Cross-talk* (2.6) discusses known methods to reduce quasi-static deformations. The last section (2.7) of this chapter describes the research gap that will be approached in this thesis.

2.1 Optimization Problem

An optimization problem can be defined as the task to find the *best solution* from all feasible solutions. The best solution minimizes or maximizes a scalar function, called objective function. A frequent mistake is to try to minimize feature A and B, although an optimization demands an explicit relation between A and B, for example

\[
\text{minimize} \quad A + 4B
\]  

(2.1)

A feasible solution satisfies the limiting conditions of the optimization problem. These limiting conditions, called constraints, define the solution space for the best solution.

In the *optimization toolbox* [68] of the MATLAB software package [67] various solvers
for optimization problems are implemented. The most general form can be solved with \textit{fmincon}.

The next sections introduce some basic optimization problems and their solutions.

\subsection{Nonlinear Programming Problem}

A nonlinear programming (NLP) problem is a general multivariable optimization problem [27]. For a given variable \( x (x \in \mathbb{R}^n) \) a best solution \( x^* \) has to be found which minimizes a scalar objective function

\[ J(x) : \mathbb{R}^n \rightarrow \mathbb{R} \quad (2.2) \]

subject to the constraints

\[ c_L \leq c(x) \leq c_U \quad c : \mathbb{R}^n \rightarrow \mathbb{R}^{n_c} \quad (2.3) \]

and the simple bounds

\[ x_L \leq x \leq x_U \quad (2.4) \]

\textbf{Convex Optimization Problems:} An important property of an optimization problem is convexity. A convex problem set guarantees a global minimum or maximum. More details can be found in [27] [6].

\subsection{Optimal Control Problem as a NLP Problem}

The optimal control problem can be described as an extension of a NLP problem (according to [9]). Using a control function \( u(t) \) the objective function \( J \) is minimized at the final time \( t_F \)

\[ J = \phi(y(t_F), t_F) \quad (2.5) \]

subject to the state equations

\[ \dot{y} = f(y(t), u(t)) \quad (2.6) \]

subject to the boundary conditions

\[ \psi(y(t), u(t), t_F) = 0 \quad (2.7) \]
\[ g(y(t), u(t), t_F) \leq 0 \quad (2.8) \]

where equation (2.7) describes the equality constraints and equation 2.8 the inequality constraints.
2.1.3 Quadratic Programming Problem

The general form of a quadratic programming (QP) problem minimizes the objective function

\[ J = c^T x + \frac{1}{2} x^T H x \]  

subject to the linear constraint functions

\[ Ax - b = 0 \]  
\[ Cx - d \leq 0 \]

If \( H \) is \textit{positive definite} the problem is convex and therefore has a global minimum. A \textit{positive semidefinite} \( H \) matrix is sufficient to define a convex problem if the linear constraint functions guarantee an unique solution. QP problems can be solved efficiently for example with the MATLAB \textit{quadprog} function [27] [68].

2.1.4 Discrete Minimum Time Optimal Control Problem

The optimal control problem from section 2.1.2 is now transformed into a \textit{discretized minimum time optimal control problem}. The discretization method explained in the following is a \textit{direct transcription formulation} [9] with a discretized time vector. The time vector discretization

\[ t_1 \leq t_k \leq t_M = t_F \]  

is transformed into a constant vector \( \tau \) and the final time \( t_F \). For later use the time intervals \( h_k \) is defined.

\[ h_k = (\tau_{k+1} - \tau_k) t_F \quad \text{with} \quad 0 \leq \tau_k \leq 1 \]

The unknown variables are assembled into a vector \( x \) including the final time \( t_F \), the discretized states \( y(t) \) and controls \( u(t) \).

\[ x^T = [t_F, y_1, u_1, ..., y_k, u_k, ..., y_M, u_M] \]  

where

\[ y_k^T = [s_k, \dot{s}_k, \ddot{s}_k, ...] \]

symbolizes a discretized state of the \( k^{th} \) element.
With the definition of the time intervals $h_k$ the state equations (2.6) can be transformed into a set $\zeta$ of equality constraints

$$\zeta_k = y_{k+1} - y_k - h_k f(y_k, u_k) = 0$$  \hspace{1cm} (2.16)

The equality constraints (2.16) can be formulated with any discretization method such as Hermite-Simpson, Runge-Kutta or an Euler method [9].

Using the equations (2.13) and (2.14), the boundary conditions (2.7) and (2.8) and the equality constraints from the state equations (2.16), the optimal control problem can be formulated as a set of NLP constraints depending on $x$.

$$c(x) = [\zeta, \psi, g]^T$$  \hspace{1cm} (2.17)

The objective function of the time minimal optimal control problem can now be defined as

$$J = t_F$$  \hspace{1cm} (2.18)

The formulations (2.17) and (2.18) of the minimum time optimal control problem can be implemented into the MATLAB solver fmincon. The discretization of the state vector $y$ instead of the discretization of the time vector can also be realized with direct transcription formulation. A detailed view on published methods is given in section 2.4.

### 2.1.5 Integration Schemes

Equation (2.16) defines an Euler integration scheme. To satisfy the ordinary differential equation (2.6) various different direct collocation methods could be used [39] [9] [99] [98], e.g. classical Runge-Kutta method or Hermite-Simpson method.

### 2.2 Mathematical Description of Path Trajectories

Path-tracking algorithms need a predefined path which does not change during the optimization. In path-tracking algorithms, most path curves are described by parametric equations [11] [112] of the form

$$x = r(s)$$  \hspace{1cm} (2.19)

where $s$ is the path parameter of the parametric function $r(s)$ which describes a location $x$ for each $s$. The dynamic over the path is given with the time dependence of $s(t)$.

The following sections illustrate the advantages of this description of path trajectories.
2.2 Mathematical Description of Path Trajectories

2.2.1 Path Dynamics

For each axis and any axes combination, the dynamic behavior can be formulated with a chain rule which includes higher derivatives of $s$ with respect to $t$ and $r(s)$ with respect to $s$. An example is given in equations (2.21) to (2.22). The parametric description has the advantage that the path-tracking problem is solved directly for every strictly increasing function $s(t)$ that matches $s(t_{\text{begin}}) = s_{\text{begin}}$ and $s(t_{\text{end}}) = s_{\text{end}}$. No additional terms apply for the synchronization of the movement of the axes, the path parameter describes a specific point in the working area.

\begin{align*}
\text{Position} & \quad \mathbf{x}(t) = r(s(t)) \quad (2.20) \\
\text{Velocity} & \quad \dot{x}(t) = r'(s(t)) \dot{s}(t) \quad (2.21) \\
\text{Acceleration} & \quad \ddot{x}(t) = r''(s(t)) \dot{s}(t)^2 + r'(s(t)) \ddot{s}(t) \quad (2.22)
\end{align*}

Figure 2.1 illustrates the decoupling of geometry and dynamics. $r(s)$ describes the $x$ and $y$ coordinates of the actuator axes. With any given profile for $s(t)$ (left figure) the position and the velocity of the axes (right figure) can be calculated through the equations (2.20) and (2.21). The geometry used to determine $s(t)$ is a L-shape curve.

Figure 2.1: Decoupling of geometry and dynamics. The optimization problem determines a profile for $s(t)$ which defines together with the parametric path function $r(s)$ the dynamics of the axes.

To construct a \textit{minimum time optimal control problem} according to the definition in chapter 2.1.2, the unknown path parameter dynamics $s(t)$ correspond to the state function $y(t)$. The path parameter dynamics are subject to the boundary conditions from equation...
(2.8), the state equation equality constraints (2.6) and the objective function (2.5), which minimizes the process time $t_F$.

### 2.2.2 Path Geometry

The $n^{th}$ parametric derivative of the function $\mathbf{r}(s)$ with respect to $s$ is defined as

$$r^{(n)}(s) = \frac{d^n}{ds^n} \mathbf{r}(s)$$

(2.23)

Thereby, the first derivative with respect to $s$ is called *parametric velocity*, the second derivative *parametric acceleration*. If the function $r^{(n)}(s)$ is continuous, the path is called $C^n$ parametric continuous.

Geometrical continuity is defined by the vector properties of the path. A path is $G^1$-continuous if the tangential vector changes continuously in orientation and length. $G^2$-continuity is given if the path is $G^1$-continuous and its curvature vector changes continuously in orientation and length. A detailed discussion can be found in [45].

Usually, a path is defined by several path sequences of which each is described as a parametric function $\mathbf{r}_i(s)$. Figure 2.2 illustrates a sequence-wise path definition. The demands on the quality of the parametric path functions depend on the optimization algorithm. To illustrate the demands on the description of the geometry, a continuous acceleration profile movement is discussed:

To describe the acceleration profile from equation (2.22) continuously, a continuous behavior of $s$, $\dot{s}$, $\ddot{s}$, $\dot{r}'(s)$ and $r''(s)$ is required. Therefore, a $C^2$-continuity is needed for the parametric description. A similar demand applies to the switching from sequence $i$ to sequence $i + 1$ in case $s$, $\dot{s}$ and $\ddot{s}$ are continuous across the switching. In the following, the switching of the sequence with continuous $\dot{s}$ and $\ddot{s}$ and therefore a continuous path description

$$\mathbf{r}_i^{(0,1,2)}(s_{end}) = \mathbf{r}_{i+1}^{(0,1,2)}(s_{begin})$$

(2.24)
will be called *parametric path switching condition*.

Optimization algorithms which use *geometric path switching condition* require a $G^2$-continuous instead a $C^2$-continuous path description. These algorithms do not force a continuity for $\dot{s}$ and $\ddot{s}$ across the switching points but they need additional switching conditions for the geometrical velocity and the acceleration profile.

A common path pattern is a straight segment followed by a circular one. At the switching point this curve is $G^1$-continuous and has a discontinuous curvature. A continuous velocity profile across the switching point results in a discontinuous force profile due to the instantaneous occurrence of the centripetal force that keeps the object on the circle. To create a continuous force profile either the velocity at the switching point has to be reduced to standstill or the geometry has to be modified such that it is $G^2$-continuous.

### 2.3 The Quality of Trajectories

The actual quality of trajectories can be determined by measurement of the workpiece geometry, the machine tool behavior at the tool center point (TCP) or by simulation of the machine tool. Besides the quality of the part, the trajectory’s quality influences the productivity of the process. To increase the productivity and the quality of the workpiece are two opposing demands. The best trajectory produces a product that fulfills the minimum requirements and is produced in minimum time. The demands on trajectories and therefore on feed-rate optimization are the following:

I Path tracking (following the given path)

II Process time minimization

III Exploitation of the geometrical tolerances

IV Trajectories are physically feasible

V Minimum excitation of the machine tool structure

The path-tracking behavior (demand I) is met by the definition of the problem description discussed in the previous section 2.2. To define a mathematical optimization problem as discussed in section 2.1 an objective function and the constraints are defined. The objective function of the optimal control problem minimizes the process time (demand II) under the condition that the demands III to V are met. The following sections discuss measurement values and limitations for the constraints of the optimization problem.
2.3.1 Exploitation of the Geometrical Tolerances

The designer of the workpiece provides not only a workpiece geometry but inherently also defines tolerances for the final workpiece. The cumulated statical and dynamical deviations [70, 4, 54, 53] as well as e.g. the influence of path rounding have to respect these tolerances.

The dynamic deviations of the machine tool are influenced by the trajectories, which is discussed in section 2.3.3 about machine excitation. In contrast, the geometric deviations are independent of the trajectories but depend on the positions. They could be compensated by changing the path as long as a model reproduces them sufficiently well.

For most feed-rate optimization algorithms the limitations for the rounding are defined based on an estimation of the maximum dynamic deviation. The estimation is a simplification created to avoid an iterative calculation of the following circular dependences. The dynamic deviations reduce the residual tolerance for rounding and the rounding influences the trajectories which again define the dynamic deviations. A discussion on path rounding can be found in [44] [112].

2.3.2 Physically Feasible Trajectories

The closed loop controller of a machine tool generally has a tracking error in the millimeter scale caused by the low-pass behavior of the controller. This leads to a path deviation in case of a curved path [102]. If the trajectories demand an axis movement that cannot be realized by the actuator the tracking error is augmented and therefore the deviation exceeds present values. These control deviations are part of the dynamic deviations discussed in section 2.3.1. Most restrictions for feasibility stem from the actuator. [114] claims a maximum velocity, acceleration and jerk limitation depending on electrical motor properties. The voltage-limiting characteristics illustrated in figure 2.3 link the maximum torque to the engine speed. Related models can be found in [89] [88] [94] [116]. From a feed-rate optimization perspective physically feasible trajectories define limitations for the constraints of the optimization problem.

2.3.3 Minimum Excitation of the Machine Tool Structure

Figure 2.4 lines out the causes of structural excitation (vibrations) and possible approaches to avoid the excitation. In this section the discussion of the structural excitation is focused on the influence of the trajectory generation.

The excitation of the machine structure caused by the trajectory generation is discussed
2.3 The Quality of Trajectories

The torque of a synchronous motor depends on the drive speed. The maximum torque is defined by the converter and motor properties, the nominal torque is mainly defined by the thermal properties of the motor.

Figure 2.3: The torque of a synchronous motor depends on the drive speed. The maximum torque is defined by the converter and motor properties, the nominal torque is mainly defined by the thermal properties of the motor.

The source of excitation (vibration) of the machine structure and their possible approaches for avoidance. In contrast to [3] other axis is assigned to trajectory generation instead of to external.

according to three different phenomena:

Firstly, elastic machine parts such as guides are deformed (quasi static) by eccentric actuator forces (see section 2.6). This dynamic deviation is proportional to the acceleration of the trajectory and could therefore be influenced by the trajectory generation. Instead of a lower acceleration limit a compensation algorithm to keep the productivity high is suggested in [93], section 2.6 and chapter 5.

The machine tool can be excited around its first eigenfrequency by the frequency content
of the acceleration profile. This content is influenced by geometrical characteristics in combination with dynamic limitations such as velocity, acceleration, jerk etc. One phenomenon is an excitation which is directly related to the path geometry and the time used to complete it. In this case an axial movement lasts about as long as the inverse of the first eigenfrequency. For example a circle followed with a constant velocity that directly excites the machine structure. To avoid such an excitation either the geometrical context must be interpreted or the spectrum has to be influenced somehow in the trajectory generation process. In the model based feed-rate optimization none of these steps are attempted. The phenomenon directly influenced by the trajectory generation is the reduction of the high frequency content of the acceleration profile caused by jerk and jerk-rate limitations. 

The influence of jerk and jerk-rate limitations on the excitation of the machine tool structure is discussed in [3] [114] [35]. To define jerk and jerk-rate limitations [114] quantifies the excitation of the structure based on the vibration amplitude of the load mass position of a two-mass-spring model. [114] determines an optimal prolonged rise time $T_{\text{rise}}$ to reach $a_{\text{max}}$ in dependence of the predominant resonant frequency $f_0$. This prolonged rise time reduces the spreading of the spring between the masses and therefore the overshoot. The prolongation of the rise time is put in context with a jerk and jerk-rate limitation. A similar analysis is done for sinusoidal acceleration profiles. [35] states that a reduction of the jerk limit gives no guarantee for a reduction of the excitation. According to [114]

“Jerk limitation is claimed by control system manufacturers to be a method for vibration suppression, although there is no direct physical limitation for jerk in the structure.”

In actual CNCs, jerk limitation is provided by [87] and sinusoidal acceleration profiles by [47] [87]. Set-point filters are used to reduce the excitation of the machine tool structure after the trajectory generation step. The use of set-point filters is described in [103] [115] [116] [91]. The disadvantage of these filters is that they alter the path geometry and may cause overshoot.

2.4 Optimal Path Tracking Algorithms

This section discusses the optimal control problem formulations found in the literature. Similar to the definition in section 2.2, $s$ denotes to the path parameter. Two basic methods [11] [97] are introduced:

The indirect method, is a linear search of switching points of different active constraints, discussed in section 2.4.1.
The second is a direct method in which the path is discretized and an algorithm searches for a state vector for each discretization step (section 2.4.2).

2.4.1 Phase Plane Methods (Indirect Methods)

Figure 2.5 illustrates the path velocity in dependence of the path parameter $s$. The red line shows the solution of the optimal control problem. The blue dashed line shows a velocity limitation constraint. The solution is composed of different active constraints (phases). The optimization searches for phase switching points. At a phase switching point the active constraint switches for example from an acceleration limited phase (A) to a velocity limited phase (B).

The most general system that can be solved with phase plane method is an autonomous planar system of the type

$$x'(t) = f(x(t))$$ \hspace{1cm} (2.25)

Solutions for this kind of systems can be found in [41] [60] [43]. An early proposal of a method using a switching curve approach for a minimum-time optimal control problem as described in section 2.1.2 is given in [11]:

Along a one-dimensional path parameter $x_1$ an algorithm searches the switching points of the time-minimizing solution. Early publications [11][72][84] of this method use a dynamic programming approach, given in [8], to solve the optimal control problem.

The time optimality is realized by choosing that valid constraint which leads to the highest velocity (active constraint). Figure 2.6(a) illustrates the determination of the active constraint with the choice between the two upper constraints $g_1$ and $g_2$. The solution of the constraint equation $g_1(x) = 0$ at $x_1 > \hat{x}_1$ violates the constraint $g_2(x) \leq 0$. Therefore, the algorithm switches to the solution of $g_2(x) = 0$. Similarly to the choice of the upper constraints, a lower constraint has to be chosen to slow down the movement. Difficulties with this algorithm occur if the edge of an infeasible region is reached, meaning that the upper constraint $g_2(x) = 0$ lies below the lower constraint $g_3(x) = 0$ for $x_1 > x_{e,1}$. In this case the switching curve has to be modified earlier. Switching at $x_s$ allows to pass the infeasible region (green) where no solution could be found with the active constraint $g_4(x) = 0$. Strategies to overcome this difficulty are proposed in [11] [81] [96] [51] [45] [44] [85] [84] [86] [92].

[11] and [96] present solutions for robots without jerk constraints. Based on a Bézier path, [95] [94] propose simplifications in the description of the velocity and acceleration limitations and additionally apply limitations from motor properties. The resulting trajectory
2. State of the art

Figure 2.5: The phase plane method searches phase switching points. The solution is composed of active phases.

(a) Constraint \( g_2(x) \) is more restrictive than \( g_1(x) \) therefore the algorithm chooses \( g_2(x) \) to continue.

(b) The upper bound constraint \( g_2(x) \) lies below the lower bound constraint \( g_3(x) \) for \( x_1 > x_v,1 \). No valid active constraint can be found at state \( x_v \). The algorithm has to switch at \( x_s \) to pass the infeasible region.

Figure 2.6: A switching curve \( g_i(x) = 0 \) which does not violate any constraint \( g_j(x) \leq 0 \) has to be found.

is not continuous in terms of acceleration and force. [51] avoids this discontinuity by formulating \( \dot{s} \) using a cubic spline in terms of \( s \). To include a jerk limitation [30] suggests a bidirectional search which is extended to jerk limitation in [31]. The most competitive phase-plane algorithm is published in [31] which can not be used for jerk-rate limited axes.

2.4.2 Direct Transcription Methods (Direct Methods)

This section describes discretized minimum time optimal control methods which use the direct transcription method presented in section 2.1.4. An early approach corresponding with this method can be found in [10].
Figure 2.7 illustrates the path velocity in dependence of the path parameter $s$. The red line shows the solution of the optimal control problem. The blue dashed line shows a velocity limitation constraint. For each discretized state $y_i$ the boundary limitation $c(y_i) \leq 0$ is satisfied. A time-minimizing solution meets at least one constraint for each state $y_i$. This method does not identify the phase switching points exactly because the boundary limitations are only evaluated at the discretized states $y_i$.

**Objective Function to Reduce Machine Excitation:** [36] suggests minimizing the square of the magnitude of the jerk integrated over the complete path to smooth the movement of the feed drive. This definition is also used in [2] where the jerk $\frac{d^4 s}{dt^4}$ is minimized over the segment length. [32] states jerk level continuity using the objective function (2.26) which minimizes the jerk-rate averaged over the segment.

$$J = \int_0^{T_{seq}} \left( \frac{d^4 x}{dt^4} \right)^2 dt \quad (2.26)$$

[80] approximizes $\dot{s}$ with cubic B-spline and then uses the following objective function

$$J = \int_0^{s_{end}} \frac{ds}{\dot{s}(s)} \quad (2.27)$$

to minimize the process time. [80] simplifies acceleration and jerk limitations inside the $(\ddot{s}, \dot{s}^2)$ space. A *sequential quadratic programming* (SQP) algorithm (nonlinear optimization) is used to solve this problem formulation.

**Convex Optimization Approach:** An optimization problem with convex properties can efficiently be solved and guarantees a global minimum solution. The parametric formulation of, for example the acceleration (2.22) is nonlinear and does not allow a convex
2. State of the art

Problem formulation. [97] suggests a transformation

\begin{align}
    a(s) &= \ddot{s} \\
    b(s) &= \dot{s}^2
\end{align}

(2.28) (2.29)

to formulate a convex optimization problem. This method can be applied up to the use of acceleration constraint (jerk limitation is not possible). Any trajectory smoothing has to be realized using the objective function which minimizes beside the process time the integral of the absolute value of the rate of change of the actuator force. Alternatively, [81] and [22] work with acceleration constraints linearized in the state space \( y \) and with simplified jerk constraints (pseudo-acceleration).

**Manipulation of Segment Duration:** The algorithm proposed by [2] reduces the machine excitation caused by force/acceleration discontinuities with an axis-wise jerk limitation and a minimization of the jerk integral [36] for each of the path segments. The algorithm is based on a model with a simplified axis dynamic including inertia, viscous and Coulomb friction only. The input for the feed-rate optimization algorithm is a sequenced spline geometry. The algorithm consists of a time minimization problem and a jerk integral minimization sub-problem. The sub-problem is based on the assumption that the path coordinate \( s \) is defined as a quintic polynomial in time for each sequence \( k \). The time variable \( v \) in equation (2.30) is normalized with a maximum time \( T_{\text{max}} \).

\[
s_k(v) = a_k v^5 + b_k v^4 + c_k v^3 + d_k v^2 + e_k v + f_k \quad v \in [0, 1] \quad (2.30)
\]

with \( \theta_k = [a_k, b_k, c_k, d_k, e_k, f_k]^T \) (2.31)

Each sequence is expected to last \( T_k \) or, in a normalized manner, \( \lambda_k = \frac{T_k}{T_{\text{max}}} \). The objective function of the constrained quadratic minimization sub-problem minimizes the jerk integral in equation (2.32).

\[
J = \int_0^{\lambda_k} \left( \frac{d^3 s(v)}{dt^3} \right)^2 dv \quad (2.32)
\]

The objective function is subject to the continuity constraint conditions \((s, \ dots, \ d\dots, \ d\dots\text{match with next sequence})\) at each node. This guarantees a smooth trajectory. A Lagrange Multipliers method is used to find \( \theta \) for a given \( \lambda_k \).

The main step of the algorithm is to minimize the sequence time \( T_k \) by varying the \( \lambda_k \).

\[
T_\Sigma = \min_\lambda \sum_{k=1}^N T_k = \min_\lambda T_{\text{max}} \sum_{k=1}^N \lambda_k \quad (2.33)
\]
In this step the limitations of the axis are built into boundary conditions $C(t) \leq 0$. These boundary conditions are formulated for discrete moments on each sequence. The path coordinate is formulated in a polynomial form of $\lambda_k$ with the parameters $\theta$ of the sub-problem. This formulation is known as a *linear programming problem* with nonlinear constraints and guarantees the exploitation of the boundary conditions. The optimization needs several runs in order to reach the optimum. [32] uses a similar approach with a snap objective.

### 2.5 Numerical Control for Machine Tools

Manufacturing on a machine tool can be described with a couple of characteristic steps. The design engineer mainly works with a computer aided design (CAD) tool to design his workpiece in 3D. State of the art CAD programs have integrated finite element method (FEM) modules as well as a visualization of the movement of machine parts or the axis movement during manufacturing of a workpiece. Figure 2.8 illustrates the procedure used to turn a digital workpiece into a movement of an actuator. Due to the fact that a machining process is based on movements along trajectories all the plane and surface information of the CAD has to be converted in a (milling, turning, erosion, cutting, e.g.) tool path description, called NC-program or NC-code. This production step is processed by a computer aided manufacturing (CAM) tool [45][112].

An NC-program contains a segment-wise definition of the tool path including a feed-rate which represents the set-point path velocity. A basic set of commands consisting of linear- and circular paths are defined in [56] [28] [111]. In addition, most CNCs support spline definitions and a variety of manufacturer-specific geometric and process commands. The NC moves the actuators of a machine tool. While the NC-program is nonspecific for a

![Figure 2.8: Procedure for manufacturing on machine tools](image-url)
certain machine tool axis configuration, the output of the NC is specific for the properties of the axes and their configuration.

The CNC can be separated into two successive tasks. The first task is the trajectory generation algorithm (also interpolation). Here machine properties, NC-program sequences and tolerances define the trajectories which are provided for the second task. This subsequent task is a closed loop control algorithm that uses at least the position, typically also velocity and current feedback to control the motion. Finally, the CNC commands an electrical current from the amplifier electronics [103] in order to empower the actuator.

Section 2.5.1 introduces the closed loop control for machine tools and then focuses on the trajectory generation in section 2.5.2. Trajectory generation is split into two sub-tasks: First the geometry optimization and secondly the feed-rate optimization. Sections 2.5.3 and 2.5.4 discuss different feed-rate optimization algorithms. The last section 2.5.5 focuses on the feed-rate optimization for under-determined kinematic systems.

### 2.5.1 Closed Loop Control for Machine Tools

The aim of a closed loop control of a feed drive is to reduce the deviation between the actually performed trajectory and the trajectory defined by the preceding trajectory generation algorithm. Ideally, it suppresses the effect of influences acting on the feed drive such as friction or process forces.

Most feed drive controllers have a cascaded architecture [3] [103] [116]. Figure 2.9 illustrates a cascaded controller consisting by principle of a position-, a velocity- and a current feedback loop. The controller inputs are trajectories (time-dependent position values). From the time variant signal the velocity and the actuator force can be determined to be used in the feed-forward function block in order to minimize the tracking error. A cascaded controller allows a stepwise controller design and set-up [115] [116] [58] [102]: The innermost current feedback loop can be designed and tested in a stand-alone setting without the velocity loop; the velocity feedback loop can then be tested and optimized without the position feedback loop. This stepwise design can easily be performed on a machine tool without the simulation-supported control design. In general, the current controller is designed as a PI (proportional gain with integral part) controller [116] with a bandwidth of around 1 kHz, using a pulse width modulation (PWM) converter of 4-20 kHz. The surrounding velocity control loop is usually also designed as PI controller to minimize the steady state error. A characteristic bandwidth of the velocity control loop is below 150 Hz. As the velocity control loop mainly defines the damping of the position movement this bandwidth must include the important structural eigenfrequencies. In a
2.5 Numerical Control for Machine Tools

classical design, the position control loop consists of only a proportional gain and typically has a bandwidth of about 30 to 80 Hz.

On most CNCs, the trajectories fed to the controller are preprocessed. A short impression of a commercially available realization is given in [90] [91] based on the Siemens Sinumerik 840D sl and is shown in figure 2.10. Figure 2.10 outlines the intermediate steps between the trajectory generation output and the cascaded controller input. The closed-loop control function block specifies the position controller only with the positioning gain $k_V$. Its output is processed in the speed set-point processing function block where the input for the velocity controller is loaded with friction compensation data. The precontrol function block determines the feed-forward values for the cascaded controller. Feed-forward algorithms can be found for various applications [58][3].

The tracking error of an axis is mainly influenced by its positioning gain $k_V$. A straight path with an interpolated axis movement gains a straightness error caused by different positioning gains [102]. This can either be corrected by choosing the weakest positioning gain for all interpolated axes or by the dynamic response adaption function block (figure 2.10). The function block delays all faster axes to the equivalent time constant of the slowest axis.

Most trajectory generation algorithms do not a priori provide physically feasible trajectories. For example, an only velocity-limited generation strategy results in a step set-point for the current. This step cannot be realized by a real actuator. Stepwise changes of velocity and acceleration further excite the machine structure (see section 2.3.3 in chapter 2). The Jerk limiting function block (figure 2.10) reduces the excitation of the structure by suppressing higher frequencies. Set-point or jerk filtering against the excitation of the structure could also be done in the trajectory generation algorithm or inside the cascaded controller. Often a higher frequency excitation is reduced by a low-pass filter in contrast to
a set of band pass filters to reduce dominant eigenfrequencies [114] [116]. The application of band pass filters is more difficult because the eigenfrequencies of a machine may vary depending for example on the actual axis positions or the weight of the workpiece.

Trajectory generation algorithms usually do not provide feed-rate data for every position control loop time step. Therefore a fine interpolation (fine interpolation function block) step is inserted. The fine interpolation inserts the additional values needed for the position controller. Fine interpolation algorithms usually use linear or cubic interpolation schemes.

Besides the closed-loop control algorithm used in commercial CNCs the control science provides a wide range of approaches [42] [40]. Especially state controllers are barely implemented in machine tools. A detailed discussion of state controllers for machine tools can be found in [46] [58].

## 2.5.2 Trajectory Generation for Machine Tools

The trajectory generation algorithm of the CNC is handled as a well-protected business secret by the machine tool control manufacturers. Manuals supply the users with a variety of parameters but the actual optimization principle stays hidden in the software and cannot be modified. The CNCs of the control manufacturers Fanuc [34], Heidenhain [47] and Siemens [87], for example, produce reliable trajectories for most of the machine tool applications. Furthermore, these CNCs provide matching solutions for various machine configurations, process types, compensation schemes, measurement systems and actuator mechanics. This thesis in contrast focuses on highly dynamical applications without process forces such as laser sheet metal cutting machines.
The trajectory generation is not only a machine tool-specific research field. *Motion planning*, a minimum time movement without the violation of a set of boundaries, is a long known optimal control problem [26] [17] [98] [9] [41] [7]. Numerous publications on robots discuss point to point movements with obstacle avoidance [65] [105] [86] [84] [83] [73] [82] [59]. This is a different task than tracking the tool-path for every instant with an optimal trajectory like it is done in the machine tool industry. The demands on the quality of trajectories are discussed in chapter 2.3.

For path-tracking applications the trajectory generation algorithm is usually divided into two separated steps (see figure 2.11). This is called a *decoupled approach* [97]. The first step is the geometry optimization, the second the feed-rate optimization.

**Geometry Optimization:** Firstly, the geometry from the NC-program is optimized [45] [103] with respect to the path continuity demands from step two (see section 2.2.2) and the given geometrical tolerances (see section 2.3.1). For sophisticated feed-rate optimization algorithms the path after optimizing the geometry is at least $G^2$-continuous. On the other hand, some machine tool manufacturers still work with $G^1$-continuous geometries and more rudimentary feed-rate optimization algorithms. Figure 2.12 illustrates the rounding of a corner with a circular segment. The radius $r$ is defined in dependence of the tolerance $tol$ and the angle $\alpha$ of the corner [49]. To make the geometry $G^2$-continuous a spline segment can be inserted instead of the circular segment [90].

A promising method for geometry optimization is the *discrete geometry optimization* [79]. *Discrete geometry optimization* is a procedure used to round discretely described geometries within a given tolerance. The resulting path information includes discrete derivatives with respect to the path parameter. The derivatives are parametrically continuous up to high orders.
**Feed-rate Optimization:** The second step defines the trajectory by calculating a velocity profile in dependence of a given tool path (modification of the feed-rate defined in the NC-program). In a robotic context this step is generally called a *time optimal path-tracking algorithm*, in a machine tool context it is referred to as *feed-rate optimization*. The above mentioned separation of geometry and feed-rate optimization is arbitrary but often practiced for simplification of the trajectory generation problem. The heuristic assumption of a rounded geometry description [21] [69] and the dynamical limitations in the feed-rate optimization step define a trajectory generation algorithm which produces reliable trajectories for machine tools and matches the subsequent closed loop control step. If in the trajectory generation algorithm the path and the feed-rate are calculated simultaneously the algorithm is called a *direct approach*.

Figure 2.13 shows a simplified categorization for the wide field of trajectory generation algorithms. With reference to the separation of geometry optimization and the algorithm to calculate the trajectory, figure 2.13 indicates that the different feed-rate optimization algorithms demand a certain quality of geometry. The quality of the geometry is either improved in a precedent geometry optimization step or the programmed geometry already satisfies a specific quality. The complexity of the constraints used to determine the trajectory influences the behavior of the closed loop controller. A trajectory optimization based on a detailed model formulation generates trajectories that can be followed by the closed-loop controller more accurately than only rough trajectories with discontinuities.

### 2.5.3 Feed-rate Optimization Algorithm for Arbitrarily Connected Joints and Tangential Geometries

[103][115][116] propose an algorithm to define velocity profiles over an arbitrarily connected geometry. This algorithm does not respect any machine limitations. A basic approach for simple geometries is to create a trajectory which limits acceleration and velocity sequence-wise. Therefore, some assumptions help to define orthogonal and path accelerations. All
the sequences are then composed and manipulated to satisfy the limitations discussed in section 2.3. [52] [49] briefly describe a similar process. Figure 2.14 explains a procedure to avoid a stop using a so-called overload factor which is documented in [90]. A similar procedure can be found in [33]. Due to the acceleration limitations $a_x$ and $a_y$, a corner can only be passed with a stop in the corner. The overload procedure makes a non-zero velocity profile around the corner by starting the acceleration of the $y$-axis $\Delta t$ before and ending the deceleration of the $x$-axis $\Delta t$ after the corner. For this additional starting and stopping phase the acceleration limitation $a = \min(a_x, a_y)$ is multiplied by the overload factor $f_{OL}$, $\Delta t$ is an NC setting. The resulting geometry will be slightly rounded.

The velocity profile is discontinuous since it is cruising over a corner, and so is the acceleration profile too. For a tangential but not curvature continuous geometry, for example switching from a straight to a circular sequence, a trajectory with continuous velocity but discontinuous acceleration can be found. A discontinuous acceleration profile is smoothened by the limited dynamic of the actuator or additionally by set-point filters. Both effect reduce the machine excitation. The excitation of the machine is additionally damped by structural properties of the drive chain and the machine structure.
2. State of the art

2.5.4 Feed-rate Optimization Algorithms for Geometries with a Continuous Curvature

Geometries with a continuous curvature can be tracked with continuous acceleration profiles. Continuous acceleration profiles lead to continuous force/torque profiles which reduce the excitation of the machine structure significantly. Documentations of algorithms used by the leading machine-tool manufacturers [87][47][34] for this kind of profiles are rarely available.

A widespread approach to generate smooth trajectories concerns spline geometries [21][62]. [62] divides the path into segments with a characteristic behavior of the velocity profile related to velocity, acceleration and jerk limitations. With the use of a flowchart the different segments are combined. A similar approach can be found in [52][69].

2.5.5 Feed-rate Optimization for Under-determined Kinematic Systems

Under-determined kinematic systems have more actuators (unknown control parameter) than mechanical degrees of freedom (equations) that have to be controlled. A machine tool with an under-determined kinematic system increases the productivity by increasing the resulting dynamics at the TCP. The dynamic of large machine tools suffers from the inertia of the moving machine components. On one hand powerful drives are necessary, on the other hand significant excitations of the machine structure are the result. To circumvent this, the conventional machine tool has to manage significantly larger inertial forces if the dynamic of the axis increases while the machine keeps the initial working area.

To work around this, the dynamic could be provided by a smaller, lighter and more dynamic system (dynamic subsystem in figure 2.15) while the larger working area is provided by a less dynamic system (main subsystem in figure 2.15). The main and the dynamic subsystem are arranged in a serial manner, meaning that the TCP position can be reached...
from various positions of the main and the dynamic subsystem (figure 2.15). [5] presents
different concepts for machine tools and gives a detailed overview of the patent situation.
[75] [76] [64] claim patents for basic principles of a machine tool with an under-determined
kinematic.

![Figure 2.15: Principle of a under-determined kinematic system. The TCP position can be reached from various positions of the main (red) and the dynamic (green) subsystem.](image)

The trajectory generation for an under-determined kinematic system has to separate the
tasks of the two axis subsystems. Unlike the feed-rate optimization problem described in
section 2.5.2 the corresponding paths for the two subsystems require additional criteria
for their contribution to the movement of the TCP. The next section discusses the related
inverse kinematics problem from a robotics’ perspective. The following sections describe
three basic approaches classified by [77]:

- **Path separation:** The geometry is separated and then the feed-rate optimization
  problem is solved with the methods presented in section 2.5.2

- **Trajectory separation:** The feed-rate optimization problem is solved with a method
  presented in section 2.5.2 and then the trajectory is separated to the different axis
  systems.

- **Minimum time optimal control problem:** For a given TCP path a minimum time-
  optimal control problem is formulated with a direct method as discussed in 2.4.2,
  including the interactions of all axes.

### 2.5.5.1 Robotics: Inverse Kinematics Problem

In the field of robotics the manipulators usually have nonlinear and under-determined
kinematics. Lots of publications can be found on the transformation from a given TCP
trajectory into the movement of the joints of the manipulator (*inverse kinematics problem*). For under-determined kinematics, an additional objective function must be applied.

A basic formulation of the *inverse kinematics problem* of an under-determined manipulator can be found in [106][18][63].

The cartesian coordinates $x$ (end effector position) are connected to the manipulator or joint coordinates $\Theta$ by the basic kinematic equation

$$x = f(\Theta)$$

If the dimension of $x$ is smaller than the dimension of $\Theta$ the system is under-determined, therefore an additional objective function must be developed to find a well-defined inverse function (2.35)

$$\Theta = f^{-1}(x)$$

A common approach to optimize the *inverse kinematics problem* is the use of *genetic algorithms* [1] [113] [23] [24] [25] [101]. *Genetic algorithms* (GA) are search heuristics that mimic the process of natural evolution (part of the *evolutionary algorithm* class). A frequently referenced example can be found in [71]. [1] solves the *inverse kinematics problem* minimizing the accumulative path deviation using a *continuous genetic algorithm*. Smooth curves are used to represent the required geometric paths in the joint coordinates. For a given point-to-point movement polynomials are used to describe the trajectory in [113]. [113] works with kinematic limitations and minimizes the vibration of a flexible manipulator (robot with many joints) by choosing joint positions that maximize the total stiffness of the machine. [100] presents a nonlinear programming technique to solve the *inverse kinematics problem* for a given trajectory by minimizing the positioning error. The methods presented here do not satisfy the demands formulated in section 2.3.

### 2.5.5.2 Path Separation Strategies

A path separation strategy reduces an under-determined kinematics problem to a path description for multiple axes. The geometric output can be processed with any feed-rate optimization algorithm proposed in 2.5.2. Very few patents and publications can be found in [12] [5].

Any method that determines trajectories for under-determined kinematic systems explicitly defines a separated path, too (e.g. [15] [77]). This geometry could be fed into a commercial CNC.
2.6 Compensation of Cross-talk

2.5.5.3 Trajectory Separation Strategies

The trajectory separation strategies work with a given TCP trajectory (given time vector). Its task is to split the dynamic into the main and dynamic system. [20] proves that the optimally separated trajectory has two active constraints at the same time. In machine tool related publications and patents (overview in [5] and [12]) the separation is based on the difference of the dynamic capacities of the axis systems. Filtering of the trajectory is a common strategy to separate a high from a low dynamic trajectory [13] [14]. [77] proposes a moon landing algorithm as a separation strategy. In [12] a spline approximation of the trajectory is used. [15] demonstrates and compares various different approaches such as filters, PID-methods, dog curve and moon landing.

2.5.5.4 Minimum Time Optimal Control Strategies

For a given TCP path with axis limitations an optimization approach similar to the description in section 2.1.4 is possible. [20] [19] formulate a linear programming problem with path limitations and postulate bang-bang behavior for the parametric acceleration. [37] presents a time-optimal control formulation for a kinematically under-determined manipulator and suggests a phase plane method similar to the one described in section 2.4.1. A control law for a real-time solution can be found in [38] but the basic background behind remains unknown.

2.6 Compensation of Cross-talk

Inertial cross-talk (ISO/TR 230-8:2010 [57]) [104] consists of displacements (EYZ) perpendicular to the intended direction of motion. A lateral offset between the driving force and the center of mass causes an elastic tilting motion during acceleration and deceleration. Frictional cross-talk is a similar effect but it is caused by the lateral offset between the frictional force and the center of mass. The tilting causes an orthogonal deviation EYZ of the tool center point (TCP) depending on the axial TCP offset from the center of mass (see Figure 2.16).

Ordinarily, in the development of machine tool structures inertial cross talk should be minimized as far as possible [74]. Beside an increase of the stiffness of the guide-way system a reduction of the lateral or axial offset reduces the cross-talk effect. A good example for avoidance of any lateral offset by design can be found in machine tools produced by Mori Seiki. Their drive at the center of gravity (DCG) [78] concept is illustrated in figure 2.17.
It consists of the application of two gantry drives Y and Z axis which locate the resulting drive force at the momentary center of the moved mass.

In a large number of cases inertial cross-talk cannot be fully eliminated by conceptual means only due to the application, work-space accessibility and restraints concerning cost [66]. In an industrial context the maximum acceleration is reduced due to the correlation...
of the deviation with the acceleration. [93] shows a measurement-based compensation on an actual machine tool which is discussed in chapter 5. [61] presents a rigid body model based estimation of deviations caused by dynamic effects which could be used for compensation as well.

2.7 Motivation

The dynamic of the movement of a machine tool is often limited by its structural excitation. The acceptable structural excitation is defined by the quality of the produced workpiece. In machines without any process forces the structural excitation is only caused by the trajectory. In this chapter the discontinuity of the actuator force is identified to have a significant influence on the excitation of the machine tool structure.

As a consequence, the trajectory has to be generated in a way that it reduces the excitation of the structure while the productivity of the machine tool remains at least on the same level. A common way to reduce the excitation is to avoid a discontinuous force or acceleration trajectory. In order to achieve this, either the actual trajectory generation strategies filter the trajectory, the optimization algorithm minimizes the integrated jerk value or the continuity is established through a spline approach. Only a few individual axes-wise approaches with jerk or jerk-rate limitations can be found in the literature. A general approach for feed-rate optimization however would have to include axial jerk or jerk-rate limitations. Jerk and jerk-rate limitations are used to reduce the excitation of the machine tool structure.

The aim of the design of a machine tool with an under-determined kinematic system is to reduce the mass that is accelerated by the high dynamics. The challenge is to separate the path for the TCP into the paths for the single axes. The separation method should result in an optimization which defines the maximally smooth trajectories of the heavy axes and respects the physical limitations of the system.

A feed-rate optimization algorithm should not restrict the mathematical form of the boundary conditions and should allow to formulate an optimization criterion. Boundary conditions can then be formulated based on physical models. If the geometry optimization is performed previously to the feed-rate optimization (decoupled approach) then the feed-rate optimization can be formulated as a discrete minimum time optimal control problem. In contrast to [97] this thesis allows the definition of boundary conditions with a nonlinear mathematical form. Therefore, the implementation of jerk and jerk-rate limitations becomes possible.
Conventional approaches to reduce cross-talk are the reduction of the acceleration or the avoidance of cross-talk by conceptional means concerning the design of the machine structure. For a given machine structure a compensation of cross-talk can improve the dynamic of the machine tool. The quantity of cross-talk depends on the position of the axes. Cross-talk has a good repeatability. For a successful compensation the cross-talk has to be measured and then modeled depending on the position of the axes. Based on the model the compensation can be included into the CNC.

The deficiencies described in feed-rate optimization and cross-talk compensation are approached in this thesis and novel solutions are presented.
Chapter 3

Feed-rate Optimization for Machine Tools

Jerk or jerk-rate limited trajectories are one way to reduce the excitation of the machine tool structure as described in section 2.3.3. The algorithms described in section 2.4 limit the jerk or jerk-rate values of the trajectory indirectly by punishing the integrated jerk or jerk-rate values in the objective function. This chapter presents a feed-rate optimization algorithm called feedrateOptim which supports jerk or jerk-rate limitations as boundary conditions. The algorithm and the notation are presented for systems with linear axes.

3.1 The Feed-rate Optimization Algorithm

This section introduces the formulations for the constraints used in a minimum time optimal control problem. Subsequently, the application of these constraints in a discrete minimum time optimal control problem is discussed and the algorithm feedrateOptim is introduced.

3.1.1 Parametric Description of the Dynamics

The physical limitations of trajectories discussed in section 2.3.2 provides constraints for the maximum axis velocity, actuator force and cutting velocity. Jerk and jerk-rate limitations are discussed in section 2.3.3. Because of the parametric problem description (see section 2.2.1) the optimization has only to search for the best course of the path parameter $s(t)$ as a function of time. For any given $s(t)$ its derivatives are defined too. In the following $y(t)$ represents the state of $[s(t) \dot{s}(t) \ldots u(t)]$ and describes the dynamics at a
specific time \( t \). The derivatives of the path vector \( \mathbf{r}(s(t)) \) with respect to \( s \) are denoted as \( \mathbf{r}'(s) \), the derivatives with respect to \( t \) as \( \dot{\mathbf{r}}(t) \).

In the following a selection of limiting constraints is presented in detail:

**Velocity:** The velocity for a single axis in the parametric form is given by

\[
v(y(t)) = \mathbf{r}'(s) \dot{s}
\]  

(3.1)

The cutting velocity on a three dimensional path has the form of:

\[
v_{\text{cut}}(y(t)) = \sqrt{\mathbf{r}'_x(s)^2 + \mathbf{r}'_y(s)^2 + \mathbf{r}'_z(s)^2} \dot{s}(t)
\]

(3.2)

**Actuator Force:** The force limitation for a single axis consists of inertia and friction, for which a viscous model is assumed:

\[
\mathbf{F}(y(t)) = \ddot{\mathbf{r}}(t) \mathbf{m} + \dot{\mathbf{r}}(t) \mathbf{\mu}
\]

(3.3)

\[
\mathbf{F}(y(t)) = (\mathbf{r}''(s) \dot{s}^2 + \mathbf{r}'(s) \ddot{s}) \mathbf{m} + \mathbf{r}'(s) \dot{s} \mathbf{\mu}
\]

(3.4)

where \( \mathbf{\mu} \) is a vector with viscous friction coefficients and \( \mathbf{m} \) with the moved masses for each coordinate of \( \mathbf{r}(s) \).

**Acceleration:** The acceleration for a single axis can be computed by differentiating the parametric velocity with respect to time. The acceleration is usually limited to reduce dynamical effects, such as cross-talk and the excitation of the machine structure, instead of the maximum actuator force.

\[
\mathbf{a}(y(t)) = \mathbf{r}''(s) \dot{s}^2 + \mathbf{r}'(s) \ddot{s}
\]

(3.5)

**Jerk and Jerk-rate:** Jerk- \( j \) and Jerk-rate-limitations \( j_r \) can be implemented to reduce the excitation of the machine structure

\[
j(y(t)) = \mathbf{r}'''(s) \dot{s}^3 + 3 \mathbf{r}''(s) \dot{s} \ddot{s} + \mathbf{r}'(s) \dddot{s}
\]

(3.6)

\[
j_r(y(t)) = \mathbf{r}''''(s) \dot{s}^4 + 4 \mathbf{r}'''(s) \dot{s}^2 + 6 \mathbf{r}''(s) \dot{s}^2 \ddot{s} + 4 \mathbf{r}''(s) \dot{s} \dddot{s} + \mathbf{r}'(s) \frac{\partial^4 \mathbf{s}}{dt^4}
\]

(3.7)
3.1 The Feed-rate Optimization Algorithm

3.1.2 Constraint Formulation

In section 3.1.1 the physical limitations are defined depending on the function $s(t)$ and its derivatives with respect to time (state $y(t)$). To generalize the constraint formulation the function $\Phi(y)$ represents a limitation alike equations (3.1) to (3.7). Now $\Phi_{max}$ defines the physical upper limit and $\Phi_{min}$ the opposing minimum limit ($\Phi_{max} > \Phi_{min}$) of the condition. The inequality constraint can now be written as

$$\Phi_{max} \geq \Phi(y(t))$$

(3.8)

$$\Phi_{min} \leq \Phi(y(t))$$

(3.9)

A lower limit is needed, because the limiting function is defined with respect to the axis and not the path. Thus, if the axis moves in the negative direction its function value $\Phi(y(t))$ is negative.

To speed up the optimization algorithm a normalized form is desirable. The limitation can directly be used to normalize the constraint function as follows

$$\frac{\phi(y(t))}{\Phi_{max}} - 1 \leq 0 \quad \Phi_{max} > 0$$

(3.10)

$$\frac{\phi(y(t))}{\Phi_{min}} - 1 \leq 0 \quad \Phi_{min} < 0$$

(3.11)

The inequality constraints (3.10) and (3.11) of the physical limitations have to be satisfied for all times $t$.

3.1.3 Parametric / Geometric Path Sequence Switching Condition

The sequence-wise definition of NC codes implicates that after the geometry preprocessing step the path definition is still composed of different sequences. To switch from one to the next sequence the feed-rate optimization algorithm demands a certain property of the geometric description or needs an additional set of path switching constraints. This section discusses two different approaches as introduced in section 2.2.2.

3.1.3.1 Parametric Switching of Sequences

The concept of the parametric path switching is a reduction from $n$ path sequences with $s \in [0 .. 1]$ to one single sequence with $s \in [0 .. n]$. The continuity demands for the
parametric path description \( r(s) \) can be defined according to the continuity demands of the trajectory.

A jerk continuous trajectory \( x(t) \) for a single axis uses a jerk-rate control variable. Hence \( \dot{x}(t), \ddot{x}(t) \) and \( \frac{d^4}{dt^4} x(t) \) are continuous and \( \frac{d^3}{dt^3} x(t) \) is limited but not continuous.

The following steps explain the need of a continuous parametric path description for a jerk-limited trajectory. The equations (3.1) to (3.7) can be solved to the path parameter with the highest time derivative:

The parametric velocity

\[
\dot{s}(t) = \frac{\dot{x}(t)}{r_x'(s)} \tag{3.12}
\]

is continuous if \( r_x'(s) \) is continuous.

The parametric acceleration

\[
\ddot{s}(t) = \frac{\ddot{x}(t)}{r_x'(s)} - \dot{s}(t)^2 \frac{r_x''(s)}{r_x'(s)} \tag{3.13}
\]

is continuous if \( r_x'(s) \) and \( r_x''(s) \) are continuous.

The parametric jerk

\[
\dddot{s}(t) = \frac{\dddot{x}(t)}{r_x'(s)} - 3 \dot{s}(t) \ddot{s}(t) \frac{r_x''(s)}{r_x'(s)} - \dot{s}(t)^3 \frac{r_x'''(s)}{r_x'(s)} \tag{3.14}
\]

is continuous if \( r_x'(s), r_x''(s) \) and \( r_x'''(s) \) are continuous.

Because the parametric jerk-rate

\[
\frac{d^4}{dt^4}s(t) = \frac{d^4}{dt^4}x(t) \frac{1}{r_x'(s)} - \dot{s}(t)^4 \frac{d^4 r_x(s)}{dt^4} \frac{1}{r_x'(s)} \frac{d^4}{dt^4} s(t) - 6 \dot{s}(t)^2 \ddot{s}(t) \frac{r_x''(s)}{r_x'(s)} - \left( 3 \dot{s}(t)^2 + 4 \dot{s}(t) \dddot{s}(t) \right) \frac{r_x'''(s)}{r_x'(s)} \tag{3.15}
\]

is the control value it only has to be limited, therefore \( \frac{d^4 r_x(s)}{dt^4} \) does not have to be continuous.

The equations (3.12), (3.13) and (3.14) demand for a continuous description of \( r_x'(s), r_x''(s) \) and \( r_x'''(s) \).

Later the gradient values and the Hessian of the constraint functions are discussed. For the gradient values the constraint has to be differentiated once with respect to \( s \), for the Hessian the constraint has to be differentiated twice with respect to \( s \). Experiments indicate that continuous gradient values are needed for the optimization to converge.
3.1.3.2 Geometric Switching Condition

In contrast to the parametric switching described in the section above the geometric switching has less demands to the geometry description. The geometrical conditions such as continuous curvature (vectorial) are sufficient demands on the path description. Therefore additional path switching constraints have to be defined.

The optimal control problem is solved in every sequence and at the beginning and the end of each sequence the physical properties such as acceleration and velocity must match to the properties of the neighboring sequences. Instead of the state equation (2.6) the path switching constraints

\[ x_i^{(1,2,\cdots)}(s_{\text{end}}) = x_{i+1}^{(1,2,\cdots)}(s_{\text{start}}) \]  

(3.16)

connect these sequences together for each axis with \( i \) denoting the index of the sequence and \( (1,2,\cdots) \) the order of the derivatives of the axis-position with respect to time. For a jerk-limited problem formulation the velocity and the acceleration must match at this point.

3.1.4 Conception of the feedrateOptim Algorithm

The algorithm presented in this chapter is a direct method (introduced in section 2.4.2). For direct methods the movement has to be discretized. The discretization concept applied in feedrateOptim is the direct transcription formulation that solves a discrete minimum time optimal control problem. The formulation is introduced in section 2.1.4.

3.1.4.1 Direct Transcription Formulation

For the numerical optimization the feed-rate optimization problem has to be discretized. Usually one of the following two discretization approaches is applied for direct transcription.

**Discretization of \( s \):** Figure 3.1(a) illustrates the approach discussed in [97] which discretizes the path parameter \( s \) to a set of \( M \) states \( s_k, k \in \{1..M\} \). The algorithm determines the duration \( \Delta t_k \) and the remaining elements of the state vector \( y_k \) for each \( k \). \( \Delta t_k \) is the time used to reach \( s_{k+1} \) from \( s_k \). To define a minimum time optimal control problem either the sum of all unknown time steps (3.17) or of all reciprocal of the parametric
velocity (3.18) is minimized.

\[ J = \sum_{k} \Delta t_k \]  
\[ J = \sum_{k} \frac{1}{\dot{s}_k} \]  

The latter discretization approach has the disadvantage that for \( \dot{s} = 0 \) a special treatment is needed, which calculates the profile around \( \dot{s} = 0 \) algebraically. This avoids the division by zero in the objective function and a long time step \( \Delta t_{k=1} \) to reach the first state after \( \dot{s} = 0 \).

**Discretization of the final time:** The `feedrateOptim` algorithm uses an alternative discretization approach, shown in figure 3.1(b). The problem is discretized into time steps \( h_k \) depending on the final time \( t_F \) according to (2.13). For each time step \( k \), the optimization algorithm determines the corresponding state vector \( y_k \). The objective function \( J \) (3.19) now minimizes the final time \( t_F \).

\[ J = t_F \]  

**Variable Space of the Optimization Algorithm:** The states vector \( y_k \) consists of the path parameter \( s_k \) and its derivatives with respect to time.

\[ y_k^T = [s_k, \dot{s}_k, \ddot{s}_k] \]  

The control variable \( u_k \) which corresponds to the derivative of \( s \) with the highest order (e.g. \( u_k = \ddot{s}_k \) if a jerk limitation is the highest derivative) is not included in the state vector \( y_k \).
The variable vector $x$ of the optimization algorithm is defined in dependency of the final time $t_F$ and all the states $y_k, u_k, k \in \{1..M\}$.

\[ x^T = [t_F, y_1, u_1, \ldots, y_k, u_k, \ldots, y_M, u_M] \]  

(3.21)

**System Dynamic:** The path parameter dynamics $s(t)$ are defined by the state equation (2.6). The individual states $y_k$ are connected as in equation (2.16) which is created from a system of first order differential equations. The differential equation to describe the path parameter dynamics for a trajectory that includes jerk limitation is

\[
\dot{y}(t) = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0
\end{bmatrix} y(t) + \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} u(t)
\]  

(3.22)

The control variable $u(t)$ corresponds to the jerk of the path parameter.

To solve the state equation (3.22) a discretization scheme such as Hermite-Simpson or Runge-Kutta could be selected. Discretization schemes which depend on the derivatives at the discretization steps $k$ and $k+1$ produce more satisfying results due to their easier implementation with start and end conditions. In the `feedrateOptim` algorithm a trapezoidal method [9] is selected for discretization and applied in equation (2.16). This results in the nonlinear equality constraints (3.23) to (3.25).

\[
\zeta_i = s_{k+1} - s_k - \frac{h_k}{2}(\dot{s}_{k+1} + \dot{s}_k) \\
\zeta_{i+1} = \dot{s}_{k+1} - \dot{s}_k - \frac{h_k}{2}(\ddot{s}_{k+1} + \ddot{s}_k) \\
\zeta_{i+2} = \ddot{s}_{k+1} - \ddot{s}_k - \frac{h_k}{2}(u_{k+1} + u_k)
\]  

(3.23) (3.24) (3.25)

For each discretization step $k \in \{1..M\}$ and every first order differential equation an equality constraint $\zeta_i$ is added to the complete set of equality constraints $\zeta$.

**Constraint Function:** The constraints defined in section 3.1.2 are applied for each discretization step $k$. For constraints which depend on geometrical information every state $y_k$ defines its actual path parameter $s_k$ and in doing so the properties of the geometry $r(s)$ (see section 2.2.2) too.

If the gradient values and the Hessian of the constraints are included in the problem definition, then the parametric derivatives of $r(s)$ with respect to $s$ must be available two orders higher than they are in the constraint definition.
It is possible to define limitations (e.g. jerk-rate limitations based on equation (3.15)) that need higher derivatives of $s$ with respect to $t$ than are available in the variable vector $x$ of the optimization algorithm. This is realized with a discrete differentiation scheme using the provided $s^{(k)}$. For better convergence of the optimization it is favorable to instead increase the state vector dimension.

**Nonlinear Programming (NLP) Constraints:** Depending on the solver the NLP constraints have be be brought to the correct mathematical form. For the numerical optimization solver discussed in the next section, linear and nonlinear constraints are handled independently. All nonlinear equality equations including the equality constraint $\zeta$ of the system dynamics, the path switching conditions and the boundary conditions are collected to

$$h(x) = 0$$  \hspace{1cm} (3.26)

All nonlinear inequalities are collected in

$$g(x) \leq 0$$  \hspace{1cm} (3.27)

The linear equality and inequality constraints of the form

$$A_{eq} x = b_{eq}$$  \hspace{1cm} (3.28)

$$A_{ineq} x \leq b_{ineq}$$  \hspace{1cm} (3.29)

can directly be implemented into the optimization solver.

A minimal notation to define all boundary conditions for the optimization problem is used to collect equality and nonequality, linear and nonlinear functions together to

$$c(x) \leq 0$$  \hspace{1cm} (3.30)

Each equality constraint has to be defined with two inequality constraints (upper and lower bound).

### 3.1.4.2 Numerical Optimization with *fmincon*

`feedrateOptim` uses a standard solver for the numerical optimization. To feed this solver, the feed-rate optimization problem must be formulated as a nonlinear programming problem (NLP) as discussed in section 2.1.1. The numerical optimization is realized in Matlab [67] using the `fmincon` [68] function. `Fmincon` includes various optimization algorithms for different types of problems.
The problem formulation discussed afterwards is a nonlinear optimization problem with inequality constraints. It allows to analytically calculate gradient values for the objective function and the nonlinear constraints as well as their Hessian. In these cases an interior-point method is used for optimization. For other environments than Matlab similar algorithms can be found. An interior-point method is not able to find the global minimum of the objective function. Therefore, an initial guess close to the solution improves the quality of the optimized solution.

Algorithm 3.1 Basic conception for a optimization algorithm using fmincon.

```matlab
% BASIC CONCEPTION OF FEEDRATE OPTIM
[J, gradJ] = objectiveFunction(x);
[C, Ceq, gradC, gradCeq] = nonlinearConstraints(x);
[H] = hessianFunction(x);
options = optimset('Algorithm', 'interior-point', ... 'GradObj', 'on', ... 'GradConstr', 'on', ... 'Hessian', 'user-supplied', ... 'HessFcn', @hessianFunction);
x* = fmincon(@objectiveFunction, x0, ... Aineq, bineq, Aeq, beq, ... lowerbound, upperbound, ... @nonlinearConstraints, options);
```

For a given variable vector \( \mathbf{x} \), the interior-point method in fmincon searches for the best solution \( \mathbf{x}^* \) which satisfies a set of constraints \( \mathbf{c}(\mathbf{x}) \) while minimizing an objective function \( J(\mathbf{x}) \). The pseudo-algorithm 3.1 indicates the key functions to use the interior-point algorithm with Matlab. The objective function and the nonlinear constraints (3.26) and (3.27) as well as their gradient functions are defined in the sub functions objectiveFunction and nonlinearConstraints. The function hessianFunction calculates the Hessian matrix depending on the variable \( \mathbf{x} \). optimset configures the fmincon function. Linear constraints can be fed directly into the fmincon function (Aineq, bineq, Aeq, beq).

**Gradient Values and Hessian:** The gradient values and the Hessian are used to determine the optimality conditions depending on the variable vector \( \mathbf{x} \) provided by the algorithm. These optimality conditions define a subproblem to solve the actual optimization problem. The gradient \( \nabla_\mathbf{x} \mathbf{c}(\mathbf{x}) \) is defined for each constraint equation and objective function. The Hessian \( H \) (3.32), is built from the Lagrangian \( \mathcal{L} \) (3.31), in every optimization
step.

\[
L(x, \lambda, \mu) = J(x) + \lambda^T h(x) + \mu^T g(x) \tag{3.31}
\]

\[
H = \nabla_x^2 L(x, \lambda, \mu) \tag{3.32}
\]

\(J(x)\) is the objective function, \(h(x)\) the equality constraints and \(g(x)\) the inequality constraints. \(\lambda\) and \(\mu\) are the Lagrange multipliers, provided by the optimization algorithm for each optimization step. If the gradient values and the Hessian are not supplied to the algorithm, the time to solve increases and the quality of the solution gets worse.

**Interior-point Method**  The feedrateOptim problem formulation is solved in fmincon [68] using the interior-point method. This method is based on a reformulation of the inequality constraints into a sequence of approximate minimization problems with only equality constraints:

\[
\min_{x, b} J_\mu(x, b) = \min_{x, b} J(x) - \mu \sum_i \ln(b_i) \tag{3.33}
\]

subject to

\[h(x) = 0 \quad \text{and} \quad g(x) + b = 0 \tag{3.34}\]

Each solution of the approximate minimization problem approaches the solution of the original problem (the minimum of \(J_\mu\) approaches the minimum of \(J\)) if in each iteration the barrier parameter \(\mu\) is reduced towards zero. Thereby, the solution of the previous approximate minimization problem is used as an initial guess for the next smaller \(\mu\). The logarithmic term in (3.33) is called barrier function. The \(\ln(b_i)\) term grows to negative infinity if the slack variables \(b_i\) come close to zero. This prohibits the solution to become infeasible. If \(b_i < 0\) then the estimation for \(x\) is not valid. For each iteration the approximate minimization problems is either solved with one of the following steps:

- **direct step** by solving the Karush-Kuhn-Tucker (KKT) equations: The KKT equations are derived from the KKT conditions, which are first order necessary conditions for an optimal solution of a nonlinear programming problem.

- **conjugate gradient (CG) step** using a trust region algorithm: CG is an extension of the gradient descent algorithm.
3.1.4.3 Identification of Optimality

An optimal solution of a minimum time optimal control problem satisfies all constraints and consists of a bang-bang solution [41] which requires at least one constraint to be active. At every discretization point \( k \) at least one inequality constraint is at its limit.

![Figure 3.2: Function values of the nonlinear inequality function \( g(x) \) at the optimal solution \( x^* \).](image)

The actual function value of the inequality function \( g(x) \) of an optimal solution \( x^* \) is visualized in figure 3.2. The abscissa indicates the inequality constraints in the order of their definition and the ordinate shows the distance of a constraint toward its activation. If the function value is equal zero then the constraint is active. The function value of \(-2\)
indicates the opposing constraint is active, for example a velocity limitation of the opposing direction. The sections \(a\) to \(c\) represent velocity, acceleration and jerk constraints. The subsections in figure 3.2 divide the X- and Y-axes constraints as well as the upper and lower bounds of the limitation. Inside the subsection the constraints are sorted chronically. An interpretation of the solution is possible: The velocity limitation of each axis is only reached for a short period, the trajectory is mostly jerk-limited. The maximum acceleration is never reached. The optimal solution satisfies the given constraints because there are no positive function values. For a minimum time optimal control problem at least one inequality constraint must be active at each time-step.

Figure 3.3 shows the function values of the nonlinear equality function \(h(x)\) at the optimal solution \(x^*\) in the ordinate. The abscissa indicates the equality constraints in the order of their definition. Section \(A\) contains the values of equation (3.23) which connects \(s\) and \(\dot{s}\), \(B\) represents equation (3.24) and \(C\) represents equation (3.25). If the optimal solution \(x^*\) satisfies the equality constraint then the function values of \(h(x)\) are close to zero. In this case \(10^{-10}\) is sufficiently small.

3.1.4.4 Computational Accuracy

The numerical accuracy of the computation is limited and should be considered when the constraints are defined. This is especially critical if function values equal zero are numerically only close to zero. Subtractions of two large numbers, evaluations of polynomials and trigonometric functions show this behavior. Matlab, for example, calculates for the cosine function for \(\frac{\pi}{2}\) instead of zero.

Looking at a velocity constraint for geometrical path switching: Equation (3.16) can be rewritten for an x- and y-axis

\[
\begin{align*}
 r'_{x,i}(s_{i,\text{end}}) \dot{s}_{i,\text{end}} &= r'_{x,i+1}(s_{i+1,\text{start}}) \dot{s}_{i+1,\text{start}} \quad (3.35) \\
 r'_{y,i}(s_{i,\text{end}}) \dot{s}_{i,\text{end}} &= r'_{y,i+1}(s_{i+1,\text{start}}) \dot{s}_{i+1,\text{start}} \quad (3.36)
\end{align*}
\]

If it is assumed that the path direction at the switching point is parallel to the x-axis, then \(r'_{y,i}(s_{i,\text{end}})\) and \(r'_{y,i+1}(s_{i+1,\text{start}})\) are equal to zero. But if one of these is due to numerical accuracy problems only close to zero, then the optimization algorithm will satisfy the equation (3.36) by reducing the corresponding velocity of the path parameter to zero. This phenomena is avoided in the algorithm by redefining the switching constraint for specific cases. For a geometry with acceleration limitation constraints that is strictly tangential, the acceleration switching constraints can be ignored and the axis wise velocity switching equations (3.35) and (3.36) are substituted by a path-related velocity switching constraint.
like the following equation (3.37):

\[
\zeta = \dot{x}_{\text{end},i}^2 + \dot{y}_{\text{end},i}^2 - \dot{x}_{\text{start},i+1}^2 - \dot{y}_{\text{start},i+1}^2
\]  

(3.37)

3.2 Optimal Trajectories

This section shows optimal trajectories determined with the *feedrateOptim* algorithm and briefly discusses the influence of path rounding and the influence of the low-pass behavior of the controller.

3.2.1 Influence of Path Rounding

The productivity of a machine tool can significantly be influenced by rounding the path (see section 2.3.1). The *feedrateOptim* algorithm produces trajectories based on physical limitations, therefore the properties of the path must match these limitations in order to produce a satisfying trajectory. For example, if the path has no continuous curvature it is not possible to find a jerk-limited trajectory that does not stop at the discontinuous points. Similarly, for a non-tangential path no acceleration-limited trajectory can be found which does not stop at the corners. The reference path (figure 3.4, black line) which is used to compare several path properties is a square in counter-clockwise direction with the side length of 80 mm. The first side is run twice.

3.2.1.1 Rounding by Low-pass Behavior of the Controller

A common way in industry to exploit the geometrical tolerance is not only the rounding of the path prior to the trajectory generation but also the use of the tracking error to smooth the resulting trajectory. Figure 3.4 visualizes this effect. The green curve shows the velocity profile for the unrounded reference path. Caused by the discontinuous properties (angular) of the path the velocity goes to zero in the corners. The blue velocity profile represents the output of a simulation of this trajectory. Here, a low-pass filter with a time constant of 15 ms is used. The simulated profile avoids a complete stop and rounds the discontinuous (angular) path by 1.7 mm. With the similar acceleration-limited optimization algorithm a path with continuous curvature is processed. The resulting trajectory (red) does not have stops in the corners.
3.2.1.2 Rounding by Geometrical Preprocessing

In the following text different rounding strategies are compared qualitatively. The trajectories are generated with an acceleration-limited optimization algorithm that allows geometrical path switching conditions (see section 2.2) for velocity and force. The feedrateOptim algorithm with parametric path switching conditions could not be used because the geometries are not parametrically continuous.

The first geometry denoted as corner represents the reference path without any rounding. This path has neither a tangential path switching nor does it have a continuous curvature. The second set of geometries has a circular rounding in each corner. This circle path has no continuous curvature but is tangential. The third set of geometries (spline) has corners rounded with a Bézier spline. The Bézier spline allows to create a path with continuous curvature.

Figure 3.5 shows the velocity profiles at the second corner ($X = 80 \, \text{mm}, \, Y = 0 \, \text{mm}$). The solid lines represent the values of the X-axis, the dashed lines the ones of the Y-axis. Similar to the trajectories presented in figure 3.4, the velocity profile of the corner path (blue) stops in the corner. The two circle geometries (red, orange) pass the corner with a non-zero path velocity. They do not have to stop because the trajectory is only acceleration-limited.
3.2 Optimal Trajectories

Figure 3.5: The velocity over a corner varies depending on the type of rounding. Solid lines represent X-values, dashed lines Y-values. For rounded geometries, the transition from the X to the Y movement takes place earlier than for the unrounded corner geometry.

Figure 3.6: The acceleration profile over a corner varies depending on the type of rounding. Solid lines represent X-values, dashed lines Y-values. The paths with continuous curvature (spline rounded) have a continuous acceleration profile even though the trajectory is not jerk-limited.

Figure 3.6 shows the corresponding acceleration profiles. As indicated earlier in this chapter, the acceleration of a tangential but not curvature continuous path is discontinuous. The optimization algorithm accepts a discontinuous acceleration because no jerk limitation is applied. In contrast to the circle geometries, the spline geometries (dark green, green) have a continuous curvature. Therefore, the acceleration profiles are continuous (figure 3.6). Thus, a feed-rate optimization with jerk limitations would result in a trajectory without any stopping in the corners. Due to the rounding, the transition from the X- to the Y-movement takes place earlier, at 77.7 mm path length for the 100 µm splined path and at 79.7 mm path length for the 100 µm circle-rounded path.

3.2.2 Acceleration, Jerk and Jerk-rate Limitations

The feedrateOptim algorithm described in the previous section supports constraints up the fourth derivative of the position with respect to time (jerk-rate limitation). This section demonstrates different physical limitations for the Bézier spline path shown in figure 3.7.
The spline represents a corner of a side length of 80 mm. The corner is rounded to a tolerance of 3.5 mm.

Figure 3.7 shows acceleration profiles of differently limited trajectories. They all need $t_{ref} = 0.585$ s for the motion. This time corresponds to the process time of the reference trajectory with an acceleration limitation of $a_{ref} = 3\frac{m}{s^2}$, what is a characteristic value for a laser cutting machine tool. For the same process time the jerk limitation has to be chosen at $60.9\frac{m}{s^3}$ and reaches an acceleration value of $5.2\frac{m}{s^2}$. The jerk-rate limited trajectory with the same process time has a maximum jerk-rate value of $j_{r, max} = 1776\frac{m}{s^4}$ and reaches a jerk value of $92\frac{m}{s^3}$ and a acceleration value of $5.0\frac{m}{s^2}$.

Another example of jerk and jerk-rate limited trajectories can be found in the appendix B.1.

3.3 Achievement in Feed-rate Optimization

The presented feed-rate optimization algorithm `feedrateOptim` finds an optimal trajectory based on a physical constraint definition including acceleration, force, jerk and jerk-rate constraints. Jerk and jerk-rate limitations reduce the excitation of the machine tool structure. The concept of the algorithms is not limited to any type of constraints and can therefore be adapted to any kind of formulation of the limitations. The `feedrateOptim` algorithm is a viable tool to investigate the behavior and effects of limitations. If this flexibility is not used and no higher limitations than acceleration are applied then the algorithm of [97] is more powerful than the `feedrateOptim` algorithm. Its convex formulation of the optimization problem guarantees that the optimal solution to the problem formulation is always found. The algorithm is also faster. This algorithm uses a transformation of the state variables and is therefore not able to respect jerk and jerk-rate limitations.

3.3.0.1 Reliability

Finding a solution in nonlinear optimization can not be guaranteed. Starting the optimization with a close to feasible initial guess is important to find a local minimum which corresponds to the expected solution. A good initial guess for this kind of problems is a constant velocity profile over the geometry. The success also depends on a wise choice of discretization steps, the geometry description and the problem dimension.

The `feedrateOptim` algorithm which uses a parametric switching of the sequences finds optimal solutions for most cases. A jerk-rate limited trajectory with 600 discretization step is calculated within 80 s on a 3.2 GHz Intel Xeon CPU.
3.3 Achievement in Feed-rate Optimization

Figure 3.7: The acceleration (red), jerk (green) and jerk-rate (blue) limited trajectory have a similar process time.

The \texttt{feedrateOptim} algorithm with \textit{geometric switching conditions} is sensitive to the geometry description, the choice of discretization steps for the sequences as well as the \textit{geometrical path switching conditions} described in section 3.1.4.4. This algorithm uses a symbolic computation for the gradient values and the Hessian and is therefore very slow compared to the algorithm with a \textit{parametric switching} of the sequences.

A frequent source for an optimization failure is an inappropriate geometry description. Paths with a higher order parametric continuity (e.g. $C^4$-continuity) are difficult to define in a way that the derivatives do not grow beyond a factor $10^8$ compared to the first derivative.
Chapter 4

Feed-rate Optimization for an Under-determined Kinematic System

This chapter introduces two approaches to generate a trajectory for an under-determined kinematic system. Both approaches are based on the concept of feed-rate optimization presented in chapter 3.

The first three sections discuss the characteristics, the concept and the advantages and disadvantages of a machine tool which is driven with an under-determined kinematic system. In section four, the optimization algorithm axOptim is presented. In section five, the more general approach feedrateOptimUDKS is introduced. The last section highlights the challenges for the control engineering concerning the interaction of the subsystems.

4.1 Characteristics of an Under-determined Kinematic System

From a mathematical point of view, two different kinds of under-determined kinematic systems can be identified. The first kind are linear under-determined systems. Thereby, linear defines the transformation from the \( m \) axis related coordinates \( q \) to the \( n \) tool center point (TCP) related coordinates \( x \).

\[
x = T q \quad x \in \mathbb{R}^n, \quad q \in \mathbb{R}^m, \quad T : \mathbb{R}^n \rightarrow \mathbb{R}^m
\]  

A basic kind of linear system are two cross tables above each other, schematically shown in figure 4.1(a). The TCP position \( ^0x_2 \) can be calculated with a linear function from the axis positions \( ^0x_1 \) and \( ^1x_2 \). The orientations of the axes in one subsystem does not
4.1 Characteristics of an Under-determined Kinematic System

(a) System with \textbf{linear} under-determined kinematics.

(b) System with \textbf{nonlinear} under-determined kinematics.

Figure 4.1: Systems with under-determined kinematics. The master path $0\mathbf{x}_2$ (TCP position) is defined by the axis position $0\mathbf{x}_1$ of the main subsystem and the position $1\mathbf{x}_2$ of the dynamic subsystem. The mathematical properties of the mapping function define whether the total system is linear or nonlinear.

necessarily have to be pairwise orthogonal or pairwise parallel to each other. For the illustrated parallel and orthogonal setup the Cartesian TCP position can be calculated as follows:

\begin{align}
0x_2 &= 0x_1 + 1x_2 \\
0y_2 &= 0y_1 + 1y_2
\end{align}

(4.2)

(4.3)

In contrast, the second kind of systems are \textit{nonlinear under-determined systems}. To describe the transformation from the \(m\) axis related coordinates \(\mathbf{q}\) to the \(n\) TCP related coordinates \(\mathbf{x}\), a nonlinear function \(\mathbf{t}\) has to be defined.

\[ \mathbf{x} = \mathbf{t}(\mathbf{q}) \quad \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{q} \in \mathbb{R}^m, \quad \mathbf{t} : \mathbb{R}^n \rightarrow \mathbb{R}^m \]

(4.4)

Typical nonlinear subsystems for machine tools are parallel kinematics or polar axis systems. In the context of under-determined kinematic systems, some axis may behave in a linear manner. Figure 4.1(b) represents a cross table (for itself a linear system) with a (nonlinear) polar axis subsystem above. The TCP position $0\mathbf{x}_2$ is calculated from the axis positions $0\mathbf{x}_1$ and with a nonlinear function from the axis positions $r$ and $\varphi$. For the illustrated polar system the Cartesian TCP position can be calculated as follows:

\begin{align}
0x_2 &= 0x_1 + r \cos(\varphi) \\
0y_2 &= 0y_1 + r \sin(\varphi)
\end{align}

(4.5)

(4.6)

If \(m > n\), the system is under-determined. In that case, several feasible axis positions \(\mathbf{q}\) can be found for a given TCP position \(\mathbf{x}\). The task in this chapter is to find an inverse transformation of \(\mathbf{T}\) and \(\mathbf{t}\).
In the following, the description of two serial subsystems with different dynamics is presented. Thereby, the large less dynamic system is called main subsystem, denoted as \( ^0X_1 \) in figure 4.1(a). The shorter more dynamic system is called dynamic subsystem and is denoted in figure 4.1(a) as \( ^1X_2 \). Each of the subsystems covers the complete vector space of the TCP. To describe the TCP position \( ^0X_2 \), the two subsystems are superposed.

\[
^0x_2 = ^0x_1 + ^1x_2 \tag{4.7}
\]

The notation \( x \) includes all the coordinates of a subsystem in the base directions of the under-determined kinematic system. The coordinates of the subsystem can be calculated with

\[
x_{\text{subsystem}} = \hat{T} q_{\text{subsystem}} \tag{4.8}
\]

\( \hat{T} \) describes the linear bijective transformation in the subsystem.

### 4.2 Conception of an Under-determined Kinematic System

A system with an under-determined kinematic increases the productivity if the mean cutting velocity \( \bar{v}_{\text{cut}} \) of a characteristic workpiece can be improved. The maximum cutting velocity \( v_{\text{max,cut}} \) of for example a laser cutting machine tool is given by the cutting process, mainly the laser performance, and the thickness of the sheet metal. The mean cutting velocity \( \bar{v} \) for a workpiece is directly influenced by the maximum acceleration.

The trajectory of a geometry with sharp corners consists in a first approximation of single positioning movements. Their characteristic positioning length influences the possible improvement of the mean cutting velocity. Figure 4.2 illustrates the dependency of the potential increase of the mean cutting velocity \( \bar{v}_{\text{cut}} \) and the characteristic positioning length \( l \). In case the maximum cutting velocity is never reached, the maximum relative enhancement of the mean cutting velocity is

\[
\frac{\bar{v}}{\bar{v}_{\text{ref}}} = \sqrt{\frac{a}{a_{\text{ref}}}} \quad \text{as long as} \quad \max v \leq v_{\text{max,cut}} \tag{4.9}
\]

In figure 4.2 the maximum increase of the mean cutting velocity is

\[
\frac{\bar{v}}{\bar{v}_{\text{ref}}} = \sqrt{\frac{5}{1}} = 2.2 \tag{4.10}
\]
4.2 Conception of an Under-determined Kinematic System

The characteristic positioning length $l$ influences the potential to increase the mean cutting velocity $\bar{v}$. The reference acceleration is $a_{ref} = 1 \frac{m}{s^2}$, the enhanced acceleration $a_{enh} = 5 \frac{m}{s^2}$ and the maximum velocity $v_{max} = 0.5 \frac{m}{s}$.

If the acceleration is raised by a factor 5 and the maximum velocity is never reached, which is the case for a characteristic positioning length below

$$l_{v_{max}} = \frac{v_{max}^2}{a_{enh}} = \frac{0.5^2}{5} = 50 \text{ mm} \quad (4.11)$$

For longer positioning lengths, the maximum velocity $v_{max}$ is reached and the enhancement of the mean cutting velocity looses importance.

The enhanced maximum acceleration of the master trajectory has to be realized by the dynamic subsystem introduced in the previous section. A significant increase of the acceleration is technically realized by a reduction of the moved mass and the range of the axis which both decrease the axis inertia.

The range of the dynamic subsystem is essential for the design of an under-determined system. A large range of the dynamic axis reduces its maximum acceleration as well as the maximum acceleration of the main subsystem. A very short subsystem brings the maximum master acceleration close to the maximum acceleration of the main subsystem.

A first estimation of an optimal range for the dynamic subsystem $\mathbf{x}_2$ can be defined with an accelerated movement starting from a standstill (see patent [75]). Figure 4.3 shows the velocity profile of the master trajectory in blue. The trajectory of the main subsystem is drawn in red, the dynamic subsystem is indicated in green. The displacement $\Delta$ of the dynamic subsystem correlates with the area (transparent green) under the green curve. First the master trajectory is accelerated with $a_{master}$ and then goes over to the constraint

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.2}
\caption{The characteristic positioning length $l$ influences the potential to increase the mean cutting velocity $\bar{v}$. The reference acceleration is $a_{ref} = 1 \frac{m}{s^2}$, the enhanced acceleration $a_{enh} = 5 \frac{m}{s^2}$ and the maximum velocity $v_{max} = 0.5 \frac{m}{s}$.}
\end{figure}
of the maximum velocity $v_{\text{master}}$. This acceleration process lasts

$$T_{\text{master}} = \frac{v_{\text{master}}}{a_{\text{master}}}$$

During the acceleration phase the dynamic subsystem compensates the difference between the acceleration of the main subsystem and the master trajectory and reaches a velocity of

$$v_{\text{dyn}} = T_{\text{master}} (a_{\text{master}} - a_{\text{main}})$$

After $T_{\text{master}}$, the velocity of the master trajectory $v_{\text{master}}$ stays constant and therefore the the dynamic subsystem slows down as much as the main subsystem accelerates. Thus, the deceleration of the dynamic subsystem corresponds to the acceleration $a_{\text{main}}$ of the main subsystem. The main subsystem reaches the maximum master velocity after

$$T_{\text{main}} = \frac{v_{\text{master}}}{a_{\text{main}}}$$

The displacement $\Delta$ of the dynamic subsystem is

$$\Delta = \frac{1}{2} v_{\text{dyn}} T_{\text{main}}$$

$$= \frac{v_{\text{master}}^2}{2} \frac{a_{\text{master}} - a_{\text{main}}}{a_{\text{main}} a_{\text{main}}}$$

At the beginning of the accelerated movement, the position of the dynamic subsystem can reasonably be assumed in the center of the total range of the subsystem $^1x_2$. Therefore, the total working area $A$ for the dynamic subsystem is

$$A = 2 \Delta = v_{\text{master}}^2 \left( \frac{1}{a_{\text{main}}} - \frac{1}{a_{\text{master}}} \right)$$
4.2 Conception of an Under-determined Kinematic System

Following (4.17), the total working area \( A \) can be influenced by the maximum acceleration of the main subsystem, the maximum acceleration of the master trajectory or by the square of the velocity of the master trajectory. This velocity is usually defined by the process and a reduction would reduce the productivity.

Figure 4.4 visualizes this correlation: The velocity limitation of the master trajectory is \( 0.5 \ \text{m/s} \). If the subsystem and the master trajectory have equal acceleration limitation (blue curve), then no dynamic subsystem is used (\( A = 0 \)). If the acceleration limitation of the main subsystem is \( a_{\text{main}} = 4 \ \text{m/s}^2 \) and the acceleration of the master trajectory is \( a_{\text{master}} = 20 \ \text{m/s}^2 \) (dotted lines), then the required minimum total travel of the dynamic subsystem is \( 50 \ \text{mm} \) (cyan curve).

From a machine-design point of view, mainly the dimensioning of the working area \( A \) of the dynamic subsystem influences the dynamic properties of the main subsystem. Because the maximum velocity given from the productivity demands and the master acceleration is usually one order higher than the acceleration of the main subsystem and therefore barely reduces the working area.

The consideration of velocity and acceleration neglects the influence of inertial forces between the two subsystems. In the worst case scenario discussed above the actuator force \( F_{\text{main,max}} \) of the main subsystem has to be able to accelerate the mass \( m_{\text{main}} \) of the main
subsystem with $a_{\text{main}}$ and the mass $m_{\text{dyn}}$ of the dynamic subsystem with $a_{\text{master}}$:

$$ F_{\text{main,max}} \geq a_{\text{main}} m_{\text{main}} + a_{\text{master}} m_{\text{dyn}} $$

(4.18)

4.3 Under-determined Systems: Trade-offs

With the additional productivity of an under-determined kinematic system, additional challenges arise:

For the system with a serial cross table the design complexity increases. The number of drive chains, including actuator and power units is doubled. The serial setup additionally reduces the stiffness of the machine tool due to additional coupling elements. To estimate the position uncertainty of the whole system, the uncertainties of the serial axes have to be superposed, but because the Abbee-offset increases also additional errors arise.

From a control perspective, additional challenges arise from the interaction of the subsystems. The synchronization of the tracking errors and the distribution of the position error feedback has to be guaranteed. Additionally, the controller has to take care of the coupling forces of the subsystems. A more detailed discussion of the control aspects can be found in section 4.7.

The following two sections discuss trajectory generation strategies for under-determined kinematic systems. With a focus on the maximization of productivity of the machine tool it is interesting to consider two subsystems with significantly different dynamics and a main working area which is significantly larger than the working area of the more dynamic subsystem.

4.4 The axOptim Separation Method for an Under-determined Kinematic System

This section explains an optimal method, called axOptim, for separating a master trajectory into a trajectory for the main subsystem and a trajectory for the dynamic subsystem. Other methods are presented in section 2.5.5.2. The master trajectory can previously be defined by any trajectory generation algorithm. The method presented here applies for two serial linear subsystems. The definition of the master dynamics is discussed later in this section.

The separation method must guarantee that the limitations of the drive chain are respected. As is shown in the discussion about the quality of trajectories in section 2.3 the
The axOptim Separation Method for an Under-determined Kinematic System

4.4 The axOptim Separation Method for an Under-determined Kinematic System

A thorough definition of the master trajectory (next section) secures the limitations of the dynamic subsystem and the maximum velocity of the main subsystem. The order of the derivatives of the maximum limitation of the dynamic subsystem is given by the order of the limitation of the master trajectory. For the dynamic subsystem, the jerk limitation is neglected due to the high maximum acceleration and the high stiffness of the subsystem ([115] and section 2.3.3).

The axOptim trajectory generation works with an acceleration-limited profile for the main subsystems (demand II). Demand III is addressed with the definition of the objective function of the axOptim algorithm. The objective function minimizes the jerk values of the main axes. In order to ensure a minimization of the jerk values, several authors (section 2.4) suggest an integration of the square of the jerk values similar to equation (2.32). The main subsystem is therefore indirectly jerk-limited. The acceleration limitation of the main subsystem in the axOptim algorithm is sufficient to satisfy the demands II and III. The concept of the axOptim separation method allows an extension of the problem formulation to limitations of higher order derivatives for the main subsystem.

In contrast to the feedrateOptim algorithm presented in section 3.1, the axOptim algorithm does not solve a minimum time optimal control problem but minimizes the integral of the square of the jerk values of the main subsystem. The time vector and the corresponding state of the master system are specified in the previously defined master trajectory.

Similarly to the feedrateOptim algorithm, the axOptim algorithm defines a discretized state equation and constraints for the dynamic of the axes.

4.4.1 Definition of the Master Trajectory

Like most separation methods, axOptim determines an optimal trajectory for the main subsystem from a predefined master trajectory. The trajectory of the dynamic subsystem is calculated by a subtraction of the trajectory of the main subsystem from the master trajectory. For these methods, [5] suggests choosing the dynamic of the master trajectory
so that the dynamic subsystem is always greater than the superposed dynamic of the main axis and the master trajectory. This suggestion assumes a movement of the main subsystem in the opposite direction of the master trajectory.

In order to use the master trajectory with the *axOptim* algorithm, the acceleration $0\dot{x}_{2,\text{max}}$ of the master trajectory should at most correspond to the maximum acceleration $1\ddot{x}_{2,\text{max}}$ of the dynamic subsystem. The maximum velocity $0\dot{x}_{2,\text{max}}$ or the cutting velocity $v_{\text{cut}}$ of the master trajectory have to be defined at or below the maximum velocity $0\dot{x}_{1,\text{max}}$ of the main subsystem. A common impression is that the main axis has to catch up if the maximum velocity of the master trajectory is reached. However, there is no need to catch up because in the next step the master trajectory has to decelerate and therefore the displacement of the dynamic subsystem is anyway reduced due to the lower dynamics of the main subsystem. In practice, this discussion may be irrelevant because the velocity limitations of the axes are usually higher than the limitation that comes from the process.

### 4.4.2 Constraint Formulation

Due to the definition of the master trajectory, several physical limitations are already satisfied. If the master trajectory is defined as discussed in section 4.4.1, the physical constraints of the dynamic subsystem are respected with the exception of the working area of the dynamic subsystem. Table 4.1 shows the predefined constraints of the trajectory generation and the constraint for the working area of the dynamic subsystem.

To stick to the working area of the dynamic subsystem the constraints

\begin{align}
0x_1 & \leq 1x_{2,\text{max}} + 0x_2 \\
0x_1 & \leq 1x_{2,\text{max}} - 0x_2
\end{align}

must hold at all times. The maximum working area of the dynamic subsystem $1x_{2,\text{max}}$ and the actual position of the TCP $0x_2$ define the valid positions of the main subsystem $0x_1$. $0x_1$ is determined with *axOptim* and the TCP position by the master trajectory.

<table>
<thead>
<tr>
<th>System</th>
<th>Constraints trajectory generation</th>
<th>Constraints <em>axOptim</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>Working area</td>
<td>-</td>
<td>$1x_{2,\text{max}}$ -</td>
</tr>
<tr>
<td>Velocity</td>
<td>$v_{\text{cut}}$</td>
<td>- -</td>
</tr>
<tr>
<td>Acceleration</td>
<td>$0\dot{x}_{2,\text{max}}$</td>
<td>- $0\dot{x}_{1,\text{max}}$</td>
</tr>
<tr>
<td>Jerk</td>
<td>-</td>
<td>- minimize</td>
</tr>
</tbody>
</table>
Additionally, the dynamics of the main subsystem have to be limited. Its velocity is limited by the velocity of the master trajectory (see section 4.4.1). Equation 4.17 predicts a strong correlation between the working area of the dynamic subsystem and the acceleration of the main subsystem. Therefore, solutions can be found if the maximum acceleration of the main subsystem is chosen in a way that the working area calculated with equation 4.17 is smaller than the working area of the dynamic subsystem. The acceleration constraint of the main subsystem is defined corresponding to the definition in section 3.1.2. No force limitations are needed because equation (4.18) describes the worst case scenario and has to be satisfied in the conception phase.

The jerk limitation is not defined by a physical limitation but by the objective function which minimizes the integral of the square of the jerk values over the movement. This definition is useful because it allows defining the worst case with the equation (4.17).

In the worst case, the maximum acceleration $0\ddot{x}_{1,\text{max}}$ of the main subsystem is needed to follow the master trajectory and therefore the acceleration constraint is activated with infinite jerk. For all other cases, the jerk is minimized. Once the maximum velocity is reached the master trajectory will never accelerate in the same direction until the velocity is reduce previously.

### 4.4.3 Conception of the axOptim Algorithm

The axOptim algorithm formulates an optimization problem as introduced in section 2.1. In contrast to the feedrateOptim algorithm, the axOptim algorithm receives the time vector from the previously defined master trajectory. Due to the known time steps $h_k$ from equation (2.16), the state equations of the main subsystem and all constraints are linear. The objective function can be defined in a quadratical manner, therefore the defined problem has the form of a quadratic programming (QP) problem which is discussed in section 2.1.3.

#### 4.4.3.1 Discretization

The direct transcription formulation discussed in section 3.1.4.1 is not needed for the axOptim algorithm. The predefined master trajectory provides already discretized master positions and the corresponding time vector. This also defines all higher derivatives with respect to time.

The axOptim algorithm determines the trajectory of the main subsystem. In this section the movement of the main subsystem is denoted as $r(t)$. In a discretized problem
formulation the unknown states \( y_k \) of the main subsystem are

\[
y_k^T = [r_k, \dot{r}_k, \ddot{r}_k]
\]  
(4.21)

where \( k \) defines the discretization step corresponding to the position of the master trajectory \( 0_x_{2,k} \) and the time step \( h_k \).

**Variable Space of the Optimization Algorithm:** The variable vector \( x \) of \textit{axOptim} is defined in dependence of all states \( y_k, k \in \{1 \ldots M\} \). \( M \) corresponds to the number of discretization steps of the master trajectory.

\[
x^T = [y_1, \ldots, y_k, \ldots y_M]
\]  
(4.22)

**Dynamics of the Main Subsystem:** Similarly to the equation (3.22) of the system dynamic \( s(t) \) in section 3.1.4.1, a first order state equation is defined for the dynamics of the main subsystem. This section neglects a state for the jerk but the definition can be done for any order of derivatives. To satisfy the state equation a set of equality constraints is defined analogically to the equations (3.23) to (3.25).

\[
\zeta_i = r_{k+1} - r_k - \frac{h_k}{2}(\dot{r}_{k+1} + \dot{r}_k)
\]  
(4.23)

\[
\zeta_{i+1} = \dot{r}_{k+1} - \dot{r}_k - \frac{h_k}{2}(\ddot{r}_{k+1} + \ddot{r}_k)
\]  
(4.24)

For each discretization step \( k \in \{1..M\} \) and every first order differential equation, an equality constraint \( \zeta_i \) is added to the complete set of equality constraints \( \zeta \). In contrast to the constraint for the \textit{feedrateOptim} algorithm, these equality constraints are linear because the time vector is derived from the master trajectory. For every discretization step \( k \) the equality constraint equations (4.23) and (4.24) are assembled to a system of linear equations

\[
A_{eq}x - b_{eq} = 0
\]  
(4.25)

where the variable state of the optimization problem \( x \), the matrix \( A_{eq} \) and the vector \( b_{eq} \) define a linear system of equality equations.

**Constraint Function:** In section 4.4.2 the constraints for the \textit{axOptim} algorithm are introduced. These constraints hold for each discretization step \( k \). The equations (4.19) and (4.20) as well as the acceleration constraints for the main subsystem can be rewritten for each time step \( k \) in the form

\[
A_k y_k - b_k \leq 0
\]  
(4.26)
In a next step, the constraint equations (4.26) are assembled for all steps $k$ to fulfill the definition

$$A_{ineq}x - b_{ineq} \leq 0 \quad (4.27)$$

Similar to equation (4.25) the variable state of the optimization problem $x$, the matrix $A_{ineq}$ and the vector $b_{ineq}$ define a linear system of inequalities.

**Objective Function:** The objective function minimizes the integral of the square of the jerk values. Because the state $y_k$ does not define $\ddot{r}(t)$, the jerk value for the integration is determined by first order finite differences of the acceleration values $\ddot{r}$.

$$j_k = \frac{\ddot{r}_{k+1} - \ddot{r}_k}{\Delta t_k} \quad (4.28)$$

Instead of a jerk-integrated objective function a jerk-rate integral could be used. The jerk-rate can be determined with 2nd order central finite differences of the acceleration values $\ddot{r}$.

$$j_{rk} = \frac{\ddot{r}_{k-1} - 2 \ddot{r}_k + \ddot{r}_{k+1}}{\Delta t_k \Delta t_{k+1}} \quad (4.29)$$

In consideration of the definition of the objective function in the optimization algorithm, a quadratic form of $x$ is useful to conveniently solve the optimization problem. Therefore, a good formulation for the objective function is the sum of the squares of the jerk values over all discretization steps $k$. This objective function minimizes positive and negative jerk values and penalizes higher jerk values more than small values.

$$J = \sum_k j_k^2 \quad (4.30)$$

If the matrix $H$ specifies the quadratic part of the objective function then with the equations (4.28) and (4.30) the objective function can be written in the form

$$J = \frac{1}{2} x^T H x \quad (4.31)$$

**4.4.3.2 Numerical Optimization with quadprog**

Similarly to the feedrateOptim algorithm, the axOptim algorithm uses a standard solver for the numerical optimization. AxOptim is realized in Matlab [67] using the quadprog [68] function. Quadprog solves the quadratic programming problem introduced in section 2.1.3.
For a given variable vector $\mathbf{x}$ the *interior-point-convex* optimization algorithm of *quadprog* finds an optimal solution $\mathbf{x}^\ast$ that minimizes the objective function defined in equation (4.31). The optimization problem has a global minimum if the matrix $\mathbf{H}$ of the objective function is positive definite. From the quadratic definition of $J$ follows

$$\mathbf{x}^T \mathbf{H} \mathbf{x} > 0 \quad \forall \mathbf{x} \neq 0$$

which is exactly the condition for a *positive definite* matrix.

The pseudo-algorithm 4.1 shows the definitions for the usage of the *quadprog* function in Matlab. *optimset* configures the *quadprog* function. The objective function is defined directly by the matrix $\mathbf{H}$ from equation (4.31). The equality constraints are implemented with $\mathbf{A}_{eq}$ and $\mathbf{b}_{eq}$ from equation (4.25). The inequality constraints from equation (4.27) are implemented identically ($\mathbf{A}_{ineq}$, $\mathbf{b}_{ineq}$).

**Algorithm 4.1** Basic conception for a optimization algorithm using *quadprog*.

```
% BASIC CONCEPTION OF AXOPTIM
options = optimset('Algorithm','interior-point-convex');
\[ \mathbf{x}^\ast = \text{quadprog}(\mathbf{H}, [], . . . \)

\[ \mathbf{A}_{ineq}, \mathbf{b}_{ineq}, \mathbf{A}_{eq}, \mathbf{b}_{eq}, . . . \)

\[ \text{lowerbound, upperbound, . . .} \)

\[ [], \text{options}); \]
```

4.4.3.3 Identification of Optimality

In case an optimal solution $\mathbf{x}^\ast$ of the trajectory separation problem exists, the *axOptim* algorithm finds it. A well defined QP problem always finds its globally optimal solution. If no optimal solution can be found the dynamic of the master trajectory is too fast and the maximum acceleration limit of the main subsystem can not be satisfied.

Compared to most trajectory separation algorithms, *axOptim* finds an optimal trajectory for the main subsystem with respect to the problem definition. Therefore, in non-trivial cases (e.g. short positioning movement) the working area of the dynamic subsystem is completely exploited but never violated. The jerk values of the main subsystem are reduced as far as a solution can be found. In case a high acceleration of the main subsystem is needed to satisfy the constraints, the jerk values become larger (see section 4.2).
4.4.4 Example: Trajectory Separation with axOptim

An important fact for the reliability of the separation method is the handling of trajectories that are difficult to separate, e.g. start from standstill. In axOptim the definition of constraints allows to find a solution in any case except if the maximum acceleration of the main subsystem is exploited. This limitation makes sense because in this case the main axis could not follow the trajectory. Most trajectories will not request the maximum acceleration. Therefore, the smoothness is maximized respectively the jerk is minimized. In contrast to separation methods mentioned in section 2.5.5.4, the axOptim algorithm completely exploits the working area of the dynamic subsystem but never violates it.

If equation 4.17 is satisfied in the definition of the master trajectory, the maximum acceleration of the main subsystem is not violated. A short lead-time in the master trajectory or an optimized starting position of the under-determined kinematic system reduces the maximum acceleration of the main subsystem.

Table 4.2 presents a possible configuration of the axOptim algorithm for the path separation of a given master trajectory. The master trajectory is generated with an acceleration of \(30 \frac{m}{s^2}\). The acceleration of the main subsystem is limited to \(10 \frac{m}{s^2}\) and the working area of the dynamic subsystem is specified with \(\pm 50 mm\). In the worst case the acceleration of the main subsystem following (4.17) comes to

\[
a_{\text{main, lower limit}} = \left( \frac{2 \Delta v}{v_{\text{master}}^2} + \frac{1}{a_{\text{master}}} \right)^{-1} = 8.73 \frac{m}{s^2}
\]  

(4.33)

The axOptim problem definition is demonstrated with the path from figure 4.5. The geometry contains a square with a side length of 80 mm and an inside circle with a diameter of 50 mm. The path starts in the lower left corner and then goes (1) on the circle in the

Table 4.2: axOptim problem definition. For a given master trajectory, the axOptim algorithm is constrained with the maximum working area of the dynamic subsystem and the maximum acceptable acceleration of the main subsystem. The actual properties of the main subsystem show small maximum acceleration and jerk values.

<table>
<thead>
<tr>
<th>System</th>
<th>Constraints trajectory</th>
<th>Constraints axOptim</th>
<th>Resulting trajectory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working area [m]</td>
<td>Master</td>
<td>±0.05</td>
<td>-</td>
</tr>
<tr>
<td>Velocity [m/s]</td>
<td>1.11</td>
<td>-</td>
<td>0.29</td>
</tr>
<tr>
<td>Acceleration [m/s²]</td>
<td>30</td>
<td>-</td>
<td>&lt; 10</td>
</tr>
<tr>
<td>Jerk [m/s³]</td>
<td>-</td>
<td>-</td>
<td>minimize 95</td>
</tr>
</tbody>
</table>
Figure 4.5: Path of a master trajectory (blue) which was separated with the axOptim algorithm into a trajectory for the main subsystem (red) and a displacement for the dynamic subsystem (green). The green lines represent displacements of the dynamic subsystem at equal distant time steps.

first quadrant (2). After the clockwise travel on the circle (3), the path switches (4) on the square at the upper straight (5). The square is also passed clockwise (6). The master trajectory (blue) which is fed to the axOptim algorithm for separation is generated by a commercial trajectory generation software. The process duration is 0.869 s. The resulting trajectory is acceleration-limited and includes in the corners a minor rounding of the path of about $\sim 30 \mu m$. The jerk value calculated from the (discrete) acceleration profile of the master trajectory is $4467 \frac{m}{s^3}$. The axOptim algorithm separates this trajectory into a trajectory for the main subsystem (red) and the dynamic subsystem (green).

As shown in figure 4.6, the working area of the dynamic subsystem ($\pm 50 \ mm$) is completely exploited. The maximum velocity of $0.5 \frac{m}{s}$ is never exploited, neither by the master trajectory nor by the two subsystems (figure 4.7). The maximum velocity of the main subsystem is $0.29 \frac{m}{s}$, which is nearly four times below the maximum velocity of the master trajectory.

The acceleration of the system is illustrated in figure 4.8. All phases of high acceleration are managed by the dynamic subsystem. Therefore, the maximum acceleration of the main
subsystem is only 2.39 m/s^2 which is more than 12 times below the maximum acceleration of
the master trajectory. The maximum acceleration of the main subsystems is not exploited
due to the jerk minimizing objective function and a sufficiently large working area of the
dynamic subsystem. The maximum jerk reached by the main subsystem is 95 m/s^3 (see
figure 4.9)

Most separation methods mentioned in section 2.5.5.4 need a post-processing of the tra-
jectory of the main subsystem because they do not stand still at the beginning and the
end of the movement. The axOptim algorithm provides a solution that starts with zero
velocity and stops at the end. Examples for filter methods can be found in [15] too.
Figure 4.6: Position profiles of a trajectory separated with the axOptim algorithm. Blue represents the master trajectory, red the trajectory of the main system. The displacement of the dynamic subsystem (green) exploits the maximum possible displacement of ± 50 mm several times, indicated with the arrows.

Figure 4.7: Velocity profiles of a trajectory separated with the axOptim algorithm. Blue represents the master trajectory, red the trajectory of the main system and green the trajectory of the dynamic subsystem. The master trajectory never reaches the maximum velocity of 0.5 m/s. Due to the definition of the master trajectory the subsystems never reach the velocity limitation either.
4.4 The axOptim Separation Method for an Under-determined Kinematic System

Figure 4.8: Acceleration profiles of a trajectory separated with the axOptim algorithm. Blue represents the master trajectory, red the trajectory of the main system and green the trajectory of the dynamic subsystem. The master trajectory reaches the acceleration limitation of $30 \, \text{m/s}^2$. The acceleration of the main system reaches only $2.36 \, \text{m/s}^2$.

Figure 4.9: Jerk profile of the main trajectory. The trajectory of the main subsystem reaches a maximum jerk value of $95 \, \text{m/s}^3$. The jerk of the master trajectory reaches a maximum of $4467 \, \text{m/s}^3$. The jerk of the trajectory of the dynamic subsystem corresponds to the difference of the master- and the main values.
4. General Approach for Trajectory Generation of Under-determined Kinematic Systems

The axOptim optimization method presented in section 4.4 focuses on the separation of a given master trajectory. According to chapter 2.4 the more general feed-rate optimization approach feedrateOptimUDKS is presented. The separation of the task for the different axis subsystem and the feed-rate optimization problem are assembled to a single optimization problem. This section presents a formulation of the general optimization problem of under-determined kinematic systems as a discrete minimum time optimal control problem and discusses the generated trajectories.

4.5.1 Path Description of the Main Subsystem

Figure 4.10 shows the master path with the parametric path description $r(s)$ (blue). The curve $s(t)$ of path parameter defines the trajectory through

$$0x_2(t) = r(s(t))$$

similar to the discussion in section 2.2.1. The path of the main subsystem $m(s)$ (red) is described with the same concept as $r(s)$. The paths $r(s)$ and $m(s)$ describe the geometry, $s(t)$ describes the dynamics of the trajectory. The path of the dynamic subsystem can be described as the difference of the two parametric path descriptions.

$$1x_2 = r(s) - m(s)$$

To define constraints for the main subsystem in the next section, the derivatives of $m(s)$ with respect to the path parameter $s$ have to be available. These derivatives are defined by the state equation similar to equation (2.6).

The curve $m(s)$ of the main subsystem is not given by the problem formulation but determined as a part of solution of the optimal control problem.

4.5.2 Parametric Description of the Dynamics

The parametric description of the optimization problem is based on the description introduced in chapter 3. There, the path is defined with a parametric path function $r(s)$ and the path parameter $s$. The trajectory is then defined by $s(t)$ depending on the physical limitations given in the problem formulation. The feedrateOptim algorithm searches for
4.5 General Approach for Trajectory Generation

4.5.1 Definition of the parametric path function used for feed-rate optimization of an under-determined kinematic system.

an optimal curve of $s(t)$. For an under-determined kinematic system the formulation of physical limitations for the trajectories of the master and the two subsystems is similar to the formulation for the feedrateOptim algorithm.

**Maximum Working Area of the Dynamic Subsystem:** The distance between the main subsystem and the master trajectory has to be shorter than the working area of the dynamic subsystem. If

$$\Delta(s) = r(s) - m(s)$$

(4.36)

defines the distance between the two subsystems an upper $\Delta_{max}$ and lower $\Delta_{min}$ bound can be used to define the working area of the dynamic subsystem.

**Velocity:** The velocity of the main subsystem is defined similar to equation (3.1) using the parametric velocity $m'(s)$ instead of $r'(s)$. The formulation of the velocity of a single axis of the master trajectory stays the same as for the feedrateOptim problem formulation. So does the parametric form of the cutting velocity

**Acceleration of the Dynamic Subsystem:** Similar to the acceleration limitation in the feedrateOptim problem formulation the parametric form is derived by differentiation of equation (4.35) with respect to the time.

$$\ddot{x}_2(s) = (r''(s) - m''(s)) \dot{s}^2 + (r'(s) - m'(s)) \ddot{s}$$

(4.37)

**Other Limitations of the Main Subsystem and the Master Trajectory:** The physical limitations discussed in section 3.1.1 can be applied either for the main subsystem or the master trajectory accordingly.

The limitation of the acceleration of the master trajectory is not needed if the acceleration or the actuator force of the dynamic subsystem as well as the main subsystem is limited.
Force Coupling between the Subsystems: Under-determined kinematic systems with a serial design might influence each other due to inertial forces. The dynamic subsystem is designed for high accelerations and affects the main subsystem already if only a small mass is accelerated. In these cases it might be useful to include a coupled actuator force limitation for both subsystems. The coupling describes the forces that occur from the acceleration of the other subsystem.

\[ F_i = \ddot{x}_i (m_i + m_j) + \dot{x}_j m_j + \mu_i \dot{x}_i \]  
\[ F_j = (\ddot{x}_i + \ddot{x}_j) m_j + \mu_j \dot{x}_j \]  

with the indexes \( i \) for the main and \( j \) for the dynamic subsystem, \( m \) is the mass of the subsystems and \( \mu \) are the viscous friction factors. The values of the dynamic subsystem refer to the underlying main subsystem and not to the base.

Constraint Formulation: The constraints discussed above are formulated similar to the inequality constraints (3.10) and (3.11).

4.5.3 Conception of the feedrateOptimUDKS Algorithm

The feedrateOptimUDKS algorithm uses, as well as the feedrateOptim algorithm from section 3.1.4, the direct method with a direct transcription formulation to solve the discrete minimum time optimal control problem introduced in section 2.1.4.

4.5.3.1 Discretization

The direct transcription formulation is set up similarly to the feedrateOptim algorithm. The unknown final time \( t_F \) is used to describe the discrete time steps \( h_k \) similar to (2.13). For each time step a discretization state \( y_k \) is defined. In contrast to the (3.20) the state for the feedrateOptimUDKS algorithm is extended with the unknown path variable \( m(s) \) of the main subsystem. If the highest order constraint for the master trajectory and the main subsystem is an acceleration limitation, then the state variable \( y_k \) is defined as

\[ y_k^T = [y_{s_k}^T, y_{m_k}^T] \]  
\[ y_{s_k}^T = [s_k, \dot{s}_k] \]  
\[ y_{m_k}^T = [m_k, m_k] \]  

The highest order derivatives \( \ddot{s}_k \) and \( m_k'' \) are the control variables \( u_{s_k} \) and \( u_{m_k} \) for the two state equations of \( s(t) \) and \( m(t) \). If the highest order of the constraints is a jerk or
**4.5 General Approach for Trajectory Generation**

**Jerk-rate limitations** then higher derivatives of $s(t)$ with respect to the time have to be considered.

**Variable Space of the Optimization Algorithm:** The variable vector $x$ of the optimization problem from equation (3.21) is extended with the path description of the main subsystem. This description is included to the state $y_k$ therefore the variable space can again be written as

$$x^T = \left[ t_F, y_1, u_1, \ldots, y_k, u_k, \ldots, y_M, u_M \right]$$

where $M$ denotes the last discretization element.

**State Equation:** Beside the state equation of the path parameter $s(t)$ the state equation for $m(s)$ (see section 4.5.1) has to be included in the formulation of the optimization problem. The state equation (3.22) of the feedrateOptim algorithm defines the curve of $s(t)$, a similar state equation defines the curve of $m(s)$. The new ordinary differential equations (ODE) are

$$\dot{y}_s = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} y_s + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_s$$  
(4.44)

$$\dot{y}_m' = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} y_m + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_m$$  
(4.45)

To solve these state equations a discretization scheme has to be defined. For the feedrateOptimUDKS algorithm similar to section 3.1.4.1 a trapezoidal method is selected. This results in the nonlinear equality constraints (4.46) to (4.49).

$$\zeta_i = s_{k+1} - s_k - \frac{h_k}{2} (\dot{s}_{k+1} + \dot{s}_k)$$  
(4.46)

$$\zeta_{i+1} = \dot{s}_{k+1} - \dot{s}_k - \frac{h_k}{2} (u_{s,k+1} + u_{s,k})$$  
(4.47)

$$\zeta_{i+2} = m_{k+1} - m_k - \frac{s_{k+1} - s_k}{2} (m'_{k+1} + m'_k)$$  
(4.48)

$$\zeta_{i+3} = m'_{k+1} - m'_k - \frac{s_{k+1} - s_k}{2} (u_{m,k+1} + u_{m,k})$$  
(4.49)

For each discretization step $k \in \{1..M\}$ and every first order differential equation an equality constraint $\zeta_i$ is added to the complete set of equality constraints $\zeta$. $m(s)$ is defined for every coordinate of the main subsystem. In a two-dimensional context $m(s) = [m_x(s), m_y(s)]$. 
4.5.3.2 Objective Function and the Minimum Set of Constraints

To define the *minimum time optimal control problem* properly not all the constraints presented have to be implemented. For example it is not necessary to define an acceleration and an actuator force constraint. A basic set could be the maximum velocity and acceleration of the master trajectory and the maximum acceleration of the main subsystem as well as the limitation of the working area of the dynamic subsystem. For the master trajectory either the cutting velocity or a velocity for each axis could be defined.

If the objective function is defined similar to the objective function of the *feedrateOptim* algorithm, minimizing the final time $t_F$, more than one solution could be found. For example the special case, where the master trajectory for an optimally minimized final time $t_f$ is never delayed by the dynamic of the main subsystem. In this case the trajectory of the main subsystem is not defined properly. To avoid multiple solutions an additional term is included in the objective function:

Similar to the objective function of the *axOptim* algorithm an integral over the complete movement of the square of the jerk values of the main subsystem is added. Consequently, the objective function is formulated as follows:

$$ J = t_F + \kappa \sum_k j_k^2 $$

(4.50)

where $\kappa$ is the weight for the additional jerk reducing term. $\kappa$ should be chosen in order that the weighted jerk reducing term is at least one order smaller than the final time $t_F$.

Alternatively, the jerk values of the dynamic subsystem could also be included, for example by adding the jerk values weighted with the proportions of the moved masses of the axes. This guarantees that the process time minimization is not influenced.

4.5.3.3 From the Nonlinear Program to the Numerical Optimization with *fmincon*

The formulation of the nonlinear program is composed analog to the formulation in the *feedrateOptim* algorithm (see section 3.1.4.1). All nonlinear equality equations, including the system dynamic equations $\zeta$ and the boundary conditions are assembled in

$$ h(x) = 0 $$

(4.51)

All nonlinear inequalities are collected in

$$ g(x) \leq 0 $$

(4.52)
The linear equality and inequality constraints are represented similar to equation (3.29) with $A_{eq}$, $b_{eq}$, $A_{ineq}$ and $b_{ineq}$.

In section 3.1.4.2 the fmincon function of the Matlab Optimization Toolbox was introduced. The feedrateOptimUDKS algorithm uses the same solver of the fmincon function as the feedrateOptim algorithm. This includes the need for an algebraic definition of the gradient values and the Hessian matrix.

### 4.5.3.4 Identification of Optimality

In contrast to the optimal solution discussed in section 3.1.4.3, the optimal control problem of the under-determined kinematic system has two constraints that have to be maximized or minimized simultaneously [20].

### 4.5.4 Application of the Algorithm

This section discusses two types of solutions of the feedrateOptimUDKS algorithm. In the first solution, the dynamic of the main subsystem does not slow down the master trajectory. In the second solution the optimized trajectory slows down the master trajectory due to a low acceleration limit of the main subsystem. A detailed determination of the required working area of a square geometry can be found in appendix A.2.

The example geometry is a square with a side length of 80 mm. This geometry is preprocessed with the discrete geometry optimization algorithm (section 2.5.2) and a rounding tolerance of 80 µm. The geometry is shown in figure 4.11. It starts in the lower left corner and goes counterclockwise around the square to the lower right corner. The working area of the dynamic subsystem for both solutions is ±50 mm. For the optimization the discretization is set to 2000 steps.

#### 4.5.4.1 Unaffected Master Trajectory

In the first case the master trajectory is limited to a maximum acceleration of $10 \text{ m/s}^2$ and a maximum velocity of $0.8 \text{ m/s}$. The main subsystem has for both axes an acceleration limitation of $3 \text{ m/s}^2$. For these settings, equation (A.11) requests a minimum range of ±42.6 mm for the dynamic subsystem. The required working area is smaller than the working area that is available for optimization. The dynamic of the main subsystem does not affect the master trajectory.

Figure 4.11 shows the path of the master trajectory (blue) as well as the path of the main
subsystem (red). The green lines indicate the displacement of the dynamic subsystem.

At every time interval one constraint of the master trajectory is active. Either the velocity limitation (figure 4.13) or the acceleration limitation (figure 4.14) is reached. In this case the process time can not be minimized further. Because the constraints of the main subsystem are not reached at any time, the second term of the objective function (4.50) becomes important. It minimizes the jerk of the main subsystem. Figure 4.14 confirms a smooth behavior of the acceleration profile. The working area of the dynamic subsystem is not exhausted (figure 4.12).

4.5.4.2 Dynamic of the Main Subsystem affects the Master Trajectory

To obtain a master trajectory that is affected by the limitations of the main subsystem the maximum acceleration of the main subsystem is lowered to $2 \frac{m}{s^2}$ for both axes compared to the example before. The master trajectory is still limited to a maximum acceleration of $10 \frac{m}{s^2}$. For these settings, equation (4.16) requests a minimum range of $\pm 128 \ mm$ for the dynamic subsystem. This is more than is available for the optimization ($\Delta = \pm 50 \ mm$).

Figure 4.16 shows the path of the master trajectory (blue) as well as the path of the main subsystem (red). The green lines indicate the displacement of the dynamic subsystem. At the beginning of the movement, the acceleration of the main subsystem is limiting the acceleration of the master trajectory (figure 4.19). In contrast to the trajectory in section 4.5.4.1, the maximum velocity of the master trajectory is not reached. From theory, a second maximized or minimized constraint is expected but the acceleration of the dynamic subsystem is not at its limitation.

The active constraint is given by the maximum working area of the dynamic subsystem. In figure 4.18, the integrated area under the velocity profile of the dynamic subsystem can never exceed the working area of the dynamic subsystem (figure 4.17). If the acceleration of the master trajectory would be increased during the first $0.15s$ (until $\Delta$ is reached) then the working area would exceed its limitations due to the velocity difference between the main and the master trajectory. Therefore, the maximum working area is the second active constraint during the first $0.15s$.

After $0.15 \ s$, the optimization choses a solution which is time optimal without the working area limitation. Any two constraints of the acceleration of the main subsystem or of the master acceleration or of the velocity are active. The end of the movement is again governed by the limited working area of the dynamic subsystem.
4.5 General Approach for Trajectory Generation

4.5.5 Application Limitations

The feedrateOptimUDKS optimization algorithm is sensitive to the set of constraints of the problem formulation. With the minimum set of constraints discussed in section 4.5.3.2 the algorithm can find a local minimum that satisfies the constraints. Even though a minimum is found, for some problem formulations minor numerical oscillations in the acceleration profiles could not be avoided completely. The oscillations in figure 4.14 are small but obvious in the corresponding jerk profile (figure 4.15). This effect can be seen more clearly in the acceleration profiles of figure 4.20. A wide range of opportunities might help to resolve this issue. Most distortions can be resolved by increasing the number of discretization steps. From figure 4.20(a) to 4.20(b) the number of discretization steps is raised from 600 to 2000 steps. In this case this modification did not resolve the distortion.

The following possible improvements have not been tested for the feedrateOptimUDKS algorithm. Oscillations may occur due to the choice of the integration scheme. The profiles
Figure 4.12: Position profiles of a solution with an unaffected master trajectory (blue). The displacement of the dynamic subsystem (green) does not exploit the working area completely (only ±48 mm, arrows) but more than equation (4.16) demands for a minimum working area. This is because the optimization reduces the jerk values of the main subsystem (red).

Figure 4.13: Velocity profiles of a solution with an unaffected master trajectory (blue). The main subsystem is printed in red. The velocity of the dynamic subsystem (green) has no constant maximum value due to the accelerating main subsystem. The phases where the maximum acceleration of the master trajectory reaches its limitations are indicated (arrows).

which oscillate do not violate any constraints, therefore the algorithm might find suboptimal solutions for the equality constraints. A Runge-Kutta integration scheme might resolve this issue. To determine smoother acceleration profiles for the master trajectory an extension of the control variable \( u_k \) (4.44) from \( \ddot{s} \) to \( \dddot{s} \) might help.

The trajectory described in section 4.5.4.1 is calculated with the \textit{feedrateOptimUDKS} algorithm within 12 min on a 3.2 GHz Intel Xeon CPU.
4.5 General Approach for Trajectory Generation

Figure 4.14: Acceleration profiles of a solution with an unaffected master trajectory. The acceleration of the master trajectory (blue) is maximized whenever the velocity of the master trajectory is not limiting the movement. The acceleration of the main axis (red) never reaches its limitation, the maximum value is $2.8 \text{ m/s}^2$ at the beginning and the end of the movement. The arrows mark the phases of the velocity limitation of the master trajectory, where the acceleration profile of the master trajectory and the trajectory of the dynamic system (green) show convergence problems. The solver could not find a proper solution at these points.

Figure 4.15: Jerk profile of a solution with an unaffected master trajectory. The jerk profile of the main subsystem (red) is not limited at the beginning and the end of the movement (reaches $3063 \text{ m/s}^3$). In between, the jerk values are minimized by the objective function. Due to numerical issues in the acceleration profile (green, scaling by 100) the jerk values are unsteady (partly marked with arrows).
4. Feed-rate Optimization for an Under-determined Kinematic System

Figure 4.16: Path and trajectories of the subsystems which were determined with the feedrateOptimUDKS algorithm. The master trajectory (blue) is affected by the limitations of the two subsystems. The trajectory of the main subsystem is printed in red and the trajectory of the dynamic subsystem in green.

Figure 4.17: Position profiles of a solution with an affected master trajectory. The master trajectory is printed in blue and the trajectory of the main subsystem in red. The displacement of the dynamic subsystem (green) completely exploits the working area (±50 mm, see arrows).
4.5 General Approach for Trajectory Generation

Figure 4.18: Velocity profiles of a solution with an affected master trajectory. The velocity of the main subsystem (red) rises maximally due to the acceleration limitation of the main subsystem. The velocity profile of the dynamic subsystem (green) is restricted due to the limitation $\Delta$ of its working area in combination with the acceleration limitation of the main subsystem. The arrow indicates the time span during which the dynamic subsystem reaches the maximum working area. Therefore, the master trajectory (blue) can not reach the maximum velocity.

Figure 4.19: Acceleration profiles of a solution with an affected master trajectory. The trajectory of the main subsystem (red) completely exploits its limitation of $2 \frac{m}{s^2}$. The dynamic subsystem (green) cannot compensate this loss of dynamic because it reaches its maximum working area $\Delta$. Therefore, the acceleration of the master trajectory (blue) can not be maximized to its limitation.
Figure 4.20: Influence of the time discretization on the optimal trajectory. Here, the raising of the number of discretization steps does not help to reduce the distortion in the acceleration profiles.
4.6 Region of Application

The axOptim and the feedrateOptimUDKS algorithm determine trajectories for an underdetermined kinematic system. The input for the axOptim algorithm is a master trajectory which has to satisfy equation (4.17) on page 54 in order to guarantee that the axOptim algorithm finds the optimal solution. The feedrateOptimUDKS algorithm determines an optimal trajectory for the path geometry and is not limited to equation (4.17). For a specific machine tool the trajectory of the feedrateOptimUDKS algorithm is more productive if equation (4.17) limits the productivity of the axOptim algorithm. The disadvantage of the feedrateOptimUDKS algorithm is its non convex formulation and therefore the lack of a guaranteed solution for the optimization problem.

If possible, both algorithms minimize the excitation of the machine tool by minimizing a summarized jerk or jerk-rate value. The influence of each axis can be weighted in the objective function.

4.6.1 Large Geometries and Reliability

For the feedrateOptimUDKS algorithm the task of finding the optimal solution is similar to the one discussed for the feedrateOptim algorithm in section 3.3.0.1 on page 48. The axOptim algorithm is reliable and efficient in solving the optimization problem up to a problem dimension of 20’000 discretization steps. This value varies depending on the numerical quality of the input trajectory concerning the differentiability of the acceleration profile.

This dimensional limitation can be avoided if a windowing algorithm is applied. Using appropriate boundary conditions, optimal solutions do not vary depending on their scope. The overlapping time of two windows can be estimated based on the range of the influence of the ending condition in a window. The state at the beginning of the overlapping defines the starting condition for the subsequent optimization of the window.

4.7 Interaction of the Subsystems

This section presents a collection of critical issues for the controller design of an underdetermined kinematic system. The suggested concepts have not yet been implemented on a machine tool.
4.7.1 Controller Response and Bandwidth of the Control System

In a common serial design the main subsystem is arranged between the dynamic subsystem and the machine base. The main subsystem is less agile than the dynamic subsystem. Therefore, usually the bandwidth of the dynamic subsystem is designed about three times larger than the bandwidth of the main subsystem.

**Position Control Gain:** The maximum possible position control gain $k_V$ is different for every single axis. For machine tools with a well-determined kinematic system the position control gain $k_V$ has to be chosen identically for all interpolating axes to avoid a contouring error caused by the different tracking errors (section 2.5.1). In the case of a serial under-determined kinematic system, the tracking error of the main subsystem could be compensated by the dynamic subsystem due to the larger bandwidth and a higher $k_V$. A first approach for an implementation could be to feed back the positioning error of the main subsystem to the positioning set-point of the dynamic subsystems. Such an implementation is difficult due to phase loss and delay times. For a successful compensation a model predicted approach could be promising.

**Velocity Feed-Forward Control:** Different tracking errors of the main and the dynamic subsystem cause systematic contouring errors. In the case of a cascaded controller (section 2.5.1) the tracking error corresponds to the input of the velocity controller. Thus a tracking error is needed to move the actuator. To reduce the tracking errors of the different subsystems, a velocity feed-forward control as described in section 2.5.1 can be implemented.

4.7.2 Dynamical Interaction

The two serial subsystems are dynamically coupled. Equations (4.38) and (4.39) describe the interaction between the two subsystems if no dedicated design means have been used (counter balance masses e.g.).

Regular laser cutting machine tools and their controllers are designed to manage the response to set-point changes. For machine tools with under-determined kinematics the design has to consider the disturbance reactions of the subsystems to each other.

In dependence of the control response of the subsystems, the interaction forces influence the contouring error. For example if the dynamic subsystem is accelerated the main system gets this inertial force as load. Because the acceleration profiles of the single subsystems
are predictable and the force can be determined from this acceleration, a force feed-forward control gain can reduce the contouring error. The idea of the force feed-forward control is to avoid the disturbance caused by the interacting force by adding the predicted force to the force set-point of the actual trajectory. In literature this control method is called \textit{computed torque control} CTC \cite{58} \cite{50}. 


Chapter 5

Open loop Compensation for Cross-talk

Section 2.6 describes the occurrence of inertial cross-talk in detail. For a large number of machine designs inertial cross-talk cannot fully be eliminated by conceptual means only due to the application, the work-space accessibility and the restraints concerning cost. For these numerous cases the procedure described in this chapter shall be applied. This procedure consists of measuring the systematic cross-talk values and then using the derived proportional factors for the displacement compensation. Due to the linear nature of the inertial cross-talk phenomenon a position and movement dependent generation of additional superposed axes movements can be applied.

This chapter starts with the discussion of the measurement-based identification of cross-talk. Secondly, a simplified model for compensation is illustrated and measurement data is shown.

5.1 Measurement - Identification of Cross-talk

Figure 5.1(a) indicates the variables that influence the cross-talk behavior of a machine tool. $\Delta_x$ indicates the axial TCP offset (X-offset), $\Delta_y$ the lateral offset (Y-offset) and $k_{C,rot}$ the representative rotational stiffness of the structure given by the individual stiffnesses in Y-direction (e.g. guideways). For most machine tools the major portion of the stiffness comes from the guideways. The machine tool bodies is comparatively stiff. [57] defines
5.1 Measurement - Identification of Cross-talk

(a) The axial TCP offset $\Delta x$ (X-offset) and the lateral offset $\Delta y$ (Y-offset) together with the representative rotational stiffness $k_{C,\text{rot}}$ of the guideways cause an orthogonal deviation $e_{EYX}$ if an actuator force $F_x$ is applied.

(b) Grid encoder for measurement of the perpendicular deviation: Cross-talk in Y direction while accelerating in X direction or cross-talk in X while accelerating in Y direction.

Figure 5.1: The origin of cross-talk: An orthogonal deviation caused by an accelerated axis. E.g. acceleration in X-direction and deviation in Y-direction

the correlation of inertial cross-talk $e_{EYX}$ and the actuator force $F_x$ as follows:

$$e_{EYX} = \frac{F_x \Delta x \Delta y}{k_{C,\text{rot}}}$$  \hspace{1cm} (5.1)

$$c_{ct} = \frac{e_{EYX}}{F_x} = \frac{\Delta x \Delta y}{k_{C,\text{rot}}}$$  \hspace{1cm} (5.2)

Equation (5.1) makes obvious that for a predefined position in the workspace the orthogonal deviation (dynamic straightness $e_{EYX}$) is proportional to the actuator force. In (5.2) this proportion is denoted as the proportional factor $c_{ct}$. If $c_{ct}$ is calculated from machine data a large error band arises, caused by the two offsets $\Delta x$, $\Delta y$ and the stiffness $k_{C,\text{rot}}$ that have to be estimated. The following measurement approach allows to determine the proportional factor $c_{ct}$ directly from the measurement without any estimation of the stiffness or of the offsets. The measurement approach is appropriate if the cross-talk deviation has a good repeatability. $c_{ct}$ only depends on the position in the workspace.

The example given here is a positioning movement in X-direction with an observed orthogonal deviation in Y-direction ($e_{EYX}$). For a complete compensation of cross-talk on a machine tool every axis has to be accelerated and every orthogonal direction has to be
measured individually. How to identify the proportional factor $c_{ct}$ at every position in the workspace is discussed in the next section.

The identification procedure for a given position in the workspace is the following: Firstly, the perpendicular displacements caused by acceleration (or deceleration) for a predefined position in the workspace have to be measured. An appropriate measurement system is the grid encoder from Heidenhain [48] shown in figure 5.1(b). Here, the axial (X-direction) and the lateral (Y-direction) components of motion can be captured simultaneously with moderate effort. To determine the proportional factor $c_{ct}$ for a specific deviation direction (example $e_{EYX}$ for X-axis acceleration) and machine configuration in the working volume, several different set-point accelerations have to be defined. Exemplarily, in figure 5.2(a) the suggested maximum acceleration $\ddot{X}_{ref}$ for a displacement over 100mm is varied by factors of 0.5, 1 and 2.

Figure 5.2(b) shows the grid encoder measurements and the corresponding set-point accelerations over the positioning movement. When comparing the acceleration profiles with the orthogonal deviation profile in figure 5.2(a) some correlations become obvious [16]. Cross-talk appears to be proportional to the acceleration as it is predicted in equation (5.2). This proportional factor can be determined through a linear fit (trend) in an acceleration to cross-talk plot as shown in figure 5.3. By plotting the acceleration versus the orthogonal deviation the linear correlation becomes even more obvious.

For the linear fit, all the measurements of different accelerations at the same position in the workspace are put together; thereby some measurement points are neglected. A measured acceleration never goes to zero, thus measurement points with low accelerations do not represent the cross-talk effect and therefore should not affect the linear fitting process. Therefore, these points are neglected as described in figure 5.3 by defining a lower boundary for the acceleration. A lower boundary for the orthogonal deviation is defined to ignore non-cross-talk effects in the fitting process. Additionally, all the measurement points from 25 to 75mm are neglected too because here the movement is not accelerated or decelerated.

Figure 5.3 contains additional information about the possibilities of the cross-talk modeling. The span of the orthogonal deviation on the ordinate direction describes the amount of orthogonal errors that cannot be modeled. Taking a short look at figure 5.2(b) it becomes obvious that excited free oscillations (vibration span) and the oscillation due to the rate of change of the acceleration cannot be compensated. In contrast to that, the cross-talk which appears to be proportional to the acceleration in figure 5.2(b) can be compensated. This is a quasi-static description of the situation similar to ISO/TR 230-8:2010 [57].
5.2 Modeling of Position Dependency

The proportional factor $c_{ct}$ defined in the previous section depends on the measurement position in the workspace. To describe a proper cross-talk compensation model the pro-
orthogonal deviation \( \text{e}_{xy} \)

(a) The position of the center of mass depends on the TCP position. The proportional factor increases with the increasing Y-position.

(b) The Y offset of the center of mass can be calculated from the lateral positions of every single mass.

Figure 5.4: Position dependence of cross-talk

The position dependency of the proportional factor is due to the change in the position of the center of mass (figure 5.4(a)) and the axial offset of the tool center point (TCP) from the center of mass. A larger lever arm from the force application point to the center of mass causes more orthogonal deviation for a similar actuator force. Similarly, the deviation increases with a larger axial offset.

A simple position-dependent cross-talk model can be applied if the proportional factor only depends on the position of the Y axis (figure 5.4(b)) and the axial offset remains constant. In this case, equation (5.2) can be written as:

\[
c_{ct} = k_{c,rot} \Delta_x Y_c \frac{m_c + (Y_{TCP} - Y_0) m}{m_c + m} \quad (5.3)
\]

By converting all the unknown parameters into two new parameters \( H \) and \( H_0 \) the proportional factor \( c_{ct} \) can be described with

\[
c_{ct} = H_0 + H Y_{TCP} \quad (5.4)
\]

With a set of measurement data \( c_{ct} \) and the corresponding positions in the workspace \( Y_{TCP} \) the two unknowns \( H_0 \) and \( H \) can be determined.
A more complex machine configuration including a variable axial offset at the TCP demands for an model extension. Using a similar assumption as for equation (5.4) the main equation (5.2) can be rewritten as

\[ c_{ct} = (H_0 + H Y_{TCP}) (h_0 + h X_{TCP}) \]  

(5.5)

Table 5.1 illustrates this position-dependent cross-talk model exemplarily. For a set of positions in the workspace \((Y_{TCP}, X_{TCP})\) the proportional factors were identified. Minimizing the square of the difference \(\Delta\) between the cross-talk model from equation 5.5 and the measured proportional factors the cross-talk model variables \(H_0, H, h_0\) and \(h\) were determined. The identification of the proportional factor and the position-dependent cross-talk model only use measured properties of the machine. There is no further machine model needed. The derived position-dependent cross-talk model can be evaluated efficiently for a given position in the workspace.

### 5.3 Compensation Algorithm

The section above describes a model to estimate the proportional factor for all positions in the workspace. In the previous chapters the trajectory generation and the closed loop control step were introduced. To compensate the cross-talk, a position compensation scheme is proposed in this section. Figure 5.5 shows the implementation of the compensation into the NC. Thereby, the acceleration and position set-point values from the trajectory generation are used to estimate the orthogonal cross-talk deviation which is superposed to the original position set-points. By using multiple cross-talk models at a simultaneous axis movement, the compensation values can be accumulated axis by axis.

Table 5.1: Comparison of the position-dependent proportional factor \((c_{ct})\). Measurement indicates the calculation by a set of position-dependent measurements. The model values are calculated with a cross-talk model similar to equation (5.5).

<table>
<thead>
<tr>
<th>Offset Position</th>
<th>Cross-talk Proportional Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Y_{TCP})</td>
<td>(X_{TCP})</td>
</tr>
<tr>
<td>Measurement</td>
<td>Model</td>
</tr>
<tr>
<td>(\Delta)</td>
<td></td>
</tr>
<tr>
<td>0.25 -0.18</td>
<td>0.00 0.13 0.13</td>
</tr>
<tr>
<td>0.25 0.00</td>
<td>0.30 0.38 0.08</td>
</tr>
<tr>
<td>0.25 0.18</td>
<td>0.70 0.63 -0.08</td>
</tr>
<tr>
<td>1 -0.18</td>
<td>0.31 0.21 -0.11</td>
</tr>
<tr>
<td>1 0</td>
<td>0.60 0.61 0.01</td>
</tr>
<tr>
<td>1 0.18</td>
<td>1.00 1.02 0.02</td>
</tr>
</tbody>
</table>
5. Open loop Compensation for Cross-talk

Figure 5.5: The cross-talk position compensation is situated in between the trajectory generation and the (semi-) closed loop control block. The compensation values are added to the original position set-point of the trajectory generation.

Figure 5.6: The influence of the actuator dynamic on the set-point at the measurement system. The set-points from the cross-talk model (red) or with a scaling $f_s$ (green) are simulated (corresponding dashed lines) using the basic plant model. The measured signal (black) of the indirect measurement system matches the simulated movement of the compensation set-point signal (green dotted). The peak in the measurement system signal is due to a backlash compensation of the CNC.

Figure 5.6 illustrates the implementation of the compensation. The compensation approach as mentioned above estimates the cross-talk deviation at the TCP (red line). Due to the motor dynamics the compensation set-point will not be followed accurately by the TCP. The basic plant model with low-pass behavior (figure 5.7) is used to investigate the transfer function behavior from the set-point (red line) to the internal measurement
system (black line). As shown in figure 5.6, the plant model applied to the compensating movement inhibits the compensation due to its limited dynamics (dotted red line). To resolve this loss a model inversion is substituted by a scaling factor $f_s$ which is applied to the compensation movement of the position-dependent cross-talk model. Its aim is to reduce the loss of compensation on an actual system which is caused by the motor dynamic. The scaling factor for the compensation set-point (green line) is determined by comparing the plant model simulation (dotted green line) of the compensation set-point with the cross-talk predicted by the model (red line).

### 5.4 Dynamic Benchmark

For a successful set-point compensation the demanded acceleration and jerk have to correspond to the dynamic capabilities of the compensating axes. A characteristic acceleration shall be assumed to be $3 \text{ m/s}^2$ which causes a cross-talk deviation of $x_0 = 30 \mu m$ (proportional factor $c_{ct} = 10 \mu m/m/s^2$). To analyze the dynamics necessary for the positioning movement a bang-bang jerk trajectory is assumed (see appendix section A.1). For a given jerk limitation $r_0$ the maximum acceleration is

$$a_{max} = \sqrt{\frac{x_0 r_0^2}{2}}$$

(5.6)

With a jerk limitation of $r_0 = 150 \text{ m/s}^3$, the acceleration of the compensating axis is $a_{max} = 0.70 \text{ m/s}^2$ for the positioning $x_0$ of $30 \mu m$. The duration of the positioning movement is $19 \text{ ms}$. For the evaluated machine tool the dynamic capabilities are sufficient. Figure 5.6 shows the feedback of the measurement system after the compensation set-point trajectory was tested on the machine tool. Cross-talk caused by the compensational movement itself might diminish the advantages of the compensation. The deviation can be estimated based on the product of the equation (5.6) and the proportional factor for the corresponding directions. The new cross-talk movement is again orthogonal to the direction of the compensational movement and may therefore not affect the accuracy of the movement.

### 5.5 Prove of Concept

#### 5.5.1 Using an Industrial NC

The position-dependent cross-talk model described above allows a machine configuration-dependent implementation of the compensation on an industrial numerical control. To
prove the concept the calculation of the compensation is not integrated into the NC. This could be realized with some programming effort. For the measurements shown below firstly the original set-point values for the 100mm positioning movement were captured on a Siemens NC control (Sinumerik 840D). Afterwards the cross-talk compensation indicated in figure 5.5 is performed outside the NC and then the modified compensated set-point values were fed back directly as input for the position feed-back controller of the compensating axis. These set-point values are no more treated by the feed-rate optimization.

(a) The predicted cross-talk (red) is reduced with a simulated compensational movement (according to figure 5.7). The simulated residual deviation (green dashed) can be further reduced if a scaling factor $f_s = 1.1$ is applied to the compensational movement (green).

(b) Measured cross-talk deviation for the non-compensated system (red), for the compensated system (green). The programmed compensation set-point is indicated in blue.

Figure 5.8: Simulation and measurement of the cross-talk compensation approach
5.5.2 Measurement Results

For a corresponding machine configuration, a compensated set-point trajectory was applied. All the settings for the positioning movement, including jerk, acceleration and velocity stayed the same except for the small lateral compensational movements. Figure 5.8(a) shows a simulation of the compensation approach using the actuator model from figure 5.7. The deviation predicted by the cross-talk model is reduced by the compensational movement. In figure 5.8(b) the original measurement is compared with the resulting compensated measurement. The compensation is calculated with the proportional factor determined from a set of grid encoder measurements. The results show a reduction of the maximal cross-talk deviation by about 50% without limiting the dynamics of the machine tool.

The remaining 50% consist of a mix of not further specified effects such as unmodeled dynamic effects and an incomplete compensational movement caused by a limited bandwidth and stiffness of the controller as well as limited positioning accuracy.
Chapter 6

Conclusion and Outlook

This thesis proposes solutions for two basic problems in trajectory generation for machine tools. Firstly, a trajectory generation algorithm with physical limitations for higher order derivatives with respect to time was presented. Secondly, this thesis introduced two different trajectory generation algorithms for a machine tool with an under-determined kinematic systems. The presented trajectory generation algorithms find optimal solutions for their problem formulation.

6.1 Feed-rate Optimization

A key issue in trajectory generation is the definition of the limitations of the machine tool. Various useful algorithms can be found if the limitations do not exceed the second order derivative of the position with respect to time. These limitations (acceleration or actuator force) are however not a satisfactory instrument to control the machine excitation. Therefore, state of the art CNCs use set-point filters in a post-processing step to compensate this deficit.

Trajectories determined including a jerk or jerk-rate limitation cause less excitation of the structure than trajectories with force and acceleration limitations only. The disadvantage of jerk and jerk-rate limited trajectories is a loss of productivity. If jerk and jerk-rate limitations are activated and the increase of the process time is to be avoided, a higher limitation of maximum acceleration or force has to be defined than in a purely acceleration-limited trajectory. For short geometry elements on which the acceleration of the jerk or jerk-rate limited trajectory does not exceed the acceleration values of the acceleration-limited trajectory, the process time increases.

Another effect that reduces the productivity of machine tools are dynamic deviations
caused by the acceleration and deceleration of an axis. Unless the design of the machine is altered, these effects can only be reduced through reduction of the acceleration of the trajectory, which always leads to a loss in productivity. Therefore, instead of reducing the acceleration, these effects should be compensated through post-processing based on a cross-talk compensation model. This thesis suggested building a model to predict cross-talk based on a set of measurements. Experiments showed that the cross-talk effect could either be reduced by a factor two or the acceleration could be doubled for the same deviation.

The trajectory generation procedure suggested in this thesis requires a geometry rounding step before determining a trajectory with jerk or jerk-rate limitations. The physical definition of the limitations requires a specific geometrical description for satisfactory trajectories as well as a certain level of parametric or geometric continuity.

To determine the trajectory, the feed-rate optimization algorithm `feedrateOptim` was presented. This algorithm determines trajectories with combinations of axial velocity, acceleration, actuator force, jerk and jerk-rate and cutting velocity limitations. This method formulates the feed-rate optimization problem as a `discrete minimum time optimal control problem`. The `minimum time optimal control problem` can then be solved with a standard solver for optimization problems. Due to the nonlinear character of the optimization problem a solution found by the solver is not necessarily a global optimum. For most but not all feed-rate optimization problems solutions can be found. The geometry description, the number of discretization steps, the problem dimension and the initial guess are possible reasons to explain why the solver does not always find an optimal solution.

This thesis concludes that the path description is important and should be investigated further. The properties and the mathematical description of the input geometry could be attuned to the algorithm. For long geometries the number of discretization steps increases and with them the problem dimension. Therefore, a windowing procedure has to be developed to determine the trajectories of long paths.

From a conceptional point of view, the `feedrateOptim` algorithm could be applied for a machine tool with nonlinear kinematics as well as for more than two axes. In this thesis, the algorithm was neither tested with a nonlinear machine description nor with more than two axes.

This thesis provides an algorithm to generate optimal trajectories. To parametrize the limitations for the optimization more precisely, further research has to be done. The best choice of jerk and jerk-rate limitations in dependence of machine and control properties was not investigated far enough as to provide a meaningful rule of thumb. For the next generation of trajectory generation algorithms the discussion should not concern jerk or
jerk-rate limitations but how to include the structural behavior of the machine tool in the optimization. As a consequence the objective function of the optimization would not only reduce the process time but also the machine excitation. Productivity and machine excitation are contrary demands and have to be balanced.

6.2 Under-determined Kinematic Systems

In the previous section 6.1, a trajectory generation approach for a system with a bijective transformation between the coordinate system of the tool center point (TCP) and the coordinate system of the actuators was described. This section outlines the trajectory generation for a system with additional actuators for the same amount of degrees of freedom at the TCP. Therefore, the transformation defines an under-determined kinematic system.

This thesis identified two basic strategies for trajectory generation. A first strategy is to use a regular feed-rate optimization algorithm for the well-determined problem at the TCP. This trajectory is then separated into the subsystems. The second strategy is a direct formulation of a \textit{discretized minimum optimal control problem} with the under-determined kinematic system. Both strategies provide optimal solutions with respect to their particular problem formulation.

Separating a given trajectory into two trajectories for two different subsystems is a frequently used strategy to find trajectories for an under-determined system. This work introduced an optimal strategy for this kind of separation. Thereby, it is important to define the optimization problem with regard on the properties of the subsystems. For subsystems with different dynamic properties the main subsystem (low dynamics) should move with maximum smoothness and the more dynamic subsystem should never exceed its working area. The corresponding dynamic relations were defined in this thesis. The separation strategy \textit{axOptim} minimizes the excitation of the main subsystem with the guarantee that the dynamic system respects its working area.

Compared to the optimization in the \textit{feedrateOptim} algorithm, the optimization problem formulated in the \textit{axOptim} algorithm is a quadratic optimization problem with linear constraints. The solver used to solve this kind of problems always finds a global minimum. To reduce the maximum number of unknowns in the optimization problem a windowing strategy can be applied over the given trajectory.

The separation algorithm \textit{axOptim} can only be used if the working area of the dynamic subsystem is large enough compared to the dynamics of the master trajectory and the
main subsystem. A more general strategy of feed-rate optimization which does not have this limitation was provided with the \textit{feedrateOptimUDKS} algorithm.

In this thesis, the problem formulation discussed in the previous section was applied to under-determined kinematic systems. The algorithm presented can be used if it is possible to define two serially arranged subsystems. For one of these subsystems the unknown path description was added to the optimization formulation. The path of the other subsystem was defined as the difference between the given path and the unknown path description. This thesis presented several trajectories for an under-determined kinematic system which were determined with the \textit{feedrateOptimUDKS} algorithm.

Due to the under-determinedness of the mathematical system the minimization of the process time does not lead to a unique solution of the optimization problem, depending on the definition of the limitations. An additional optimization criterion can be formulated to specify a solution out of the time minimizing solutions. Because the excitation of the structure is critical, the jerk values of the axes are minimized additionally to the prioritized minimization of the process time.

In the present work, only velocity and acceleration limitations were realized and the constraints were formulated for linear subsystems. The concept of the formulation of the optimization problem should allow higher order derivatives as well as the optimization of nonlinear systems. For this, further investigation is needed. The \textit{feedrateOptimUDKS} algorithm showed to have similar disadvantages as identified for the \textit{feedrateOptim} algorithm.

The separation algorithm \textit{axOptim} works with a linear problem formulation based on linear subsystems. Further work has to be done to expand this separation approach for a nonlinear subsystem which could consist of a polar system kinematic. An important issue in running a machine tool with under-determined kinematics is the control strategy. The two subsystems with different dynamics have different time constants as they move. To satisfy the geometrical accuracy demands without having to accept a reduction of the dynamics of the dynamic subsystem innovative control strategies are needed. The problem formulation for the optimization is based on a mechanical model of the machine tool. A further step could be to also include the mechatronic behavior into the optimization and to blur the present separation of open and closed loop control towards optimal closed loop control for under-determined kinematic systems.
Appendix A

Equations

A.1 Jerk-Limited Positioning Movement

A time optimal positioning movement of the length $x_0$ with a jerk limitation $r_0$ shall be defined. The limitations of the acceleration and velocity are not reached. From the optimality condition for minimum time problems the jerk $r$ is inside its limits $r_0$ (see figure A.1).

The rise-time $t_a$ to reach the acceleration peak $a_{max}$ is

$$t_a = \frac{a_{max}}{r_0}$$  \hspace{1cm} (A.1)

Half of the velocity $v_{max}$ is reached after $t_a$, therefore the maximum velocity reached can be calculated from

$$v(t_a) = \frac{v_{max}}{2} = \frac{1}{2} r_0 t_a^2 = \frac{1}{2} \frac{a_{max}^2}{r_0}$$  \hspace{1cm} (A.2)

The integration of the velocity profile comes to

$$x(2 t_a) = \frac{x_0}{2} = \frac{v_{max}}{2} \cdot 2 t_a = \frac{a_{max}}{r_0} a_{max} = \frac{a_{max}^3}{r_0^2}$$  \hspace{1cm} (A.3)

Using equation (A.3) the maximal acceleration can be expressed as

$$a = \sqrt[3]{\frac{x_0 r_0^2}{2}}$$  \hspace{1cm} (A.4)
A.2 Working Area for a Square Geometry

This section determines the required working area for a dynamic subsystem during a movement along a square geometry to ensure that the master trajectory does not slow down by the main subsystem. If the side length of the square is shorter than the length the main axis uses to reach the maximum velocity then the working area used by the dynamic subsystems is \( d \) smaller than the working area \( \Delta \) defined with equation (4.16). The following equations determine the reduction \( d \) of the working area for the case, that the dynamic subsystem reaches the maximum velocity before the end of the side of the square is reached.
The time used by the master trajectory to reach the maximum velocity $v_{\text{master}}$ while acceleration with $a_{\text{master}}$ is

$$T_{\text{master}} = \frac{v_{\text{master}}}{a_{\text{master}}} \quad (A.5)$$

and the used by the main trajectory to reach $v_{\text{master}}$ is

$$T_{\text{main}} = \frac{v_{\text{master}}}{a_{\text{main}}} \quad (A.6)$$

The time which the master trajectory travels with constant velocity depends on the side length $l$ of the square

$$T_{v,\text{const}} = \frac{l - v_{\text{master}} T_{\text{master}}}{v_{\text{master}}} \quad (A.7)$$

At the time $t_{\text{intersection}}$ the main trajectory has the same velocity as the master trajectory

$$a_{\text{master}} t_{\text{intersection}} = v_{\text{master}} - a_{\text{master}} (t_{\text{intersection}} - T_{\text{master}} - T_{v,\text{const}}) \quad (A.8)$$

$$t_{\text{intersection}} = \frac{v_{\text{master}} + a_{\text{master}} (T_{\text{master}} + T_{v,\text{const}})}{a_{\text{main}} + a_{\text{master}}} \quad (A.9)$$

The area $d$ is calculated by adding the not passed path of the master trajectory up to the time $t_{\text{intersection}}$ and the not passed path of the main trajectory after $t_{\text{intersection}}$:

$$d = \frac{1}{2} a_{\text{master}} (t_{\text{intersection}} - T_{\text{master}} - T_{v,\text{const}})^2$$

$$+ \frac{1}{2} a_{\text{main}} (T_{\text{main}} - t_{\text{intersection}})^2 \quad (A.10)$$

The needed working area for a square geometry is $\pm \Delta_{\text{square}}$. $\Delta_{\text{square}}$ can be determined by reducing the, for a general geometry, minimally needed working area from equation (4.16) by $d$.

$$\Delta_{\text{square}} = \Delta - d \quad (A.11)$$

Figure A.2 illustrated the concept of the determination of the equations (A.5) to (4.15). Similar to figure 4.3 red represents the accelerated main axis and blue the trajectory of the TCP.

Section 4.5.4 discusses two different kinds of behavior of an optimized trajectory for a square geometry with a side length of $l = 80mm$. For the master trajectory the maximum velocity is $v_{\text{master}} = 0.8 \frac{m}{s}$ and the maximum acceleration $a_{\text{master}} = 10 \frac{m}{s^2}$. For one kind the maximum acceleration for the main subsystem is chosen $a_{\text{main}} = 2 \frac{m}{s^2}$. This results in a working area of $\pm 53mm$ which slows down the master trajectory if the available working area is only $\pm 50mm$. For the other kind the maximum acceleration of the main trajectory is limited to $a_{\text{main}} = 3 \frac{m}{s^2}$ which results in a working area of $\pm 43mm$. 
Appendix B

Optimal Solutions

B.1 feedrateOptim used with Different Limitations

Table B.1 shows a collection of different limitations used for the demonstration of the feedrateOptim algorithm. Only a velocity limitation of $0.4 \frac{m}{s}$ and either an acceleration, jerk or jerk-rate limitation are active at one time. The values on the right side of the table indicate the maximum values reached. The jerk and jerk-rate limitations are chosen according to [114] with $a_{\text{max}} = 3 \frac{m}{s^2}$ which is a characteristic value for a laser cutting machine tool. The first resonant frequency is chosen as $f_0 = 30 \text{ Hz}$.

\begin{equation}
\text{Jerk limitation} \quad j_{\text{opt}} = a_{\text{max}} f_0 \tag{B.1}
\end{equation}

\begin{equation}
\text{Jerk-rate limitation} \quad j_{r,\text{opt}} = a_{\text{max}} f_0^2 \tag{B.2}
\end{equation}

The geometry for which the trajectories are optimized is a single L-shape Bézier spline with 80mm side length. The corner is rounded to a tolerance of 3.5mm.

The figures B.2 to B.5 show the velocity, jerk and jerk-rate profile of the trajectories. The blue line represents the acceleration-limited trajectory, the green line represents the jerk-limited trajectory and the red line the jerk-rate-limited trajectory. The trajectories are determined with the feedrateOptim algorithm and a 2000 step discretization.

Table B.1: Trajectories with different physical limitations. The right side of the table indicated the maximally reached values.

<table>
<thead>
<tr>
<th>Trajectory</th>
<th>Limitation</th>
<th>Acceleration</th>
<th>Jerk</th>
<th>Jerk-rate</th>
<th>$t_F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>blue</td>
<td>Acceleration</td>
<td>$3 \frac{m}{s^2}$</td>
<td>$3.0 \frac{m}{s^2}$</td>
<td>$2.4 \times 10^4 \frac{m}{s^3}$</td>
<td>$3.2 \times 10^6 \frac{m}{s^4}$</td>
</tr>
<tr>
<td>green</td>
<td>Jerk</td>
<td>$90 \frac{m}{s^2}$</td>
<td>$5.9 \frac{m}{s^2}$</td>
<td>$90.0 \frac{m}{s^3}$</td>
<td>$1.9 \times 10^5 \frac{m}{s^4}$</td>
</tr>
<tr>
<td>red</td>
<td>Jerk-rate</td>
<td>$2700 \frac{m}{s^2}$</td>
<td>$4.8 \frac{m}{s^2}$</td>
<td>$112.2 \frac{m}{s^3}$</td>
<td>$2.7 \times 10^4 \frac{m}{s^4}$</td>
</tr>
</tbody>
</table>
Figure B.1: Position profiles of an acceleration (blue), jerk (green) and jerk-rate (red) limited trajectory.

Figure B.2: Velocity profiles of an acceleration (blue), jerk (green) and jerk-rate (red) limited trajectory.

Figure B.3: Acceleration profiles of an acceleration (blue), jerk (green) and jerk-rate (red) limited trajectory.
Figure B.4: Jerk profiles of a jerk (green) and jerk-rate (red) limited trajectory.

Figure B.5: Jerk-rate profile of jerk-rate limited trajectory.


**Bibliography**


List of Publications
