TIME AND FREQUENCY OPTIMAL MOTION CONTROL OF CNC MACHINE TOOLS

Vom Promotionsausschuss der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation

by
Loay Alkafafi

from
Palestine

2013

1. Gutachter: Prof. Dr. Norbert Hoffmann
Institut für Mechanik und Meerestechnik
Technische Universität Hamburg-Harburg

2. Gutachter: Prof. Dr. Tomas Sauer
Numerische Mathematik und Analysis
Universität Passau

3. Gutachter: Prof. Dr.-Ing. Wolfgang Papiernik
Institutes für Werkzeugmaschinen, Roboter und Montageanlagen
Technische Universität Hamburg-Harburg

Tag der mündlichen Prüfung: 27. März 2013

Acknowledgments

This work would not have been possible without the guidance and help of many to whom I am grateful.
First, I would like to deeply thank Prof. Wolfgang Papiernik for giving me this opportunity to pursue my Ph.D. degree and for his support and valuable discussions.
I would like to express my sincerest gratitude to my advisors, Prof. Norbert Hoffmann, Prof. Tomas Sauer and Dr. Carsten Hamm for their advice, support and patience throughout the course of this research. In many ways, without their intellectual support and guidance, this work would not have been possible.
My thanks also go to the many friends and colleagues who, during my research time, contributed to the development of my research and provided me with valuable discussions, suggestions and friendship.
In addition, I would like to thank the Siemens AG, Motion and Control department for funding my work and providing the needed tools for the project.
My heartfelt gratitude goes to my beloved family for their love, patience and unconditional support and motivation.

لِ أَخِي، أَغْمَضَ عَيْنَيْه مُتَبَسِّمَاً

لِ أُمِّي، تَعُدُّ أَصَابِعِي العِشْرِينَ عَنْ بُعْد

لِ أَبِي، يَلْعَنُ سَجَّانِيه أَلْفَ مَرَّة

Abstract

The increasing demand in terms of accuracy and speed of CNC machines forces the machine tool industry to reduce the inertia of the machine parts in order to increase their dynamic capabilities. On the other hand, this reduction in the parts' inertia reduces their dynamic stiffness and thus increases the overall elastic behavior of the machine. The effects of the dominating elasticities in the machine influence their dynamic response in the form of undesirable mechanical oscillations (vibrations) at the cutting tool. This, in turn, leads to significant deterioration in the productivity and quality of the machining process. In this thesis, a strategy is developed for attenuating the mechanical oscillations in high dynamics multi-axis CNC machine tools while preserving their high-speed characteristics. The thesis is divided into three main parts.
In the first part, the process chain of CNC machine tools is studied thoroughly to identify the possible locations in the chain where measures can be taken to suppress the machine oscillations.
The second part reviews the existing state of the art for solutions of the oscillation problem. The existing solutions are classified according to their point of action in the machine process chain. Each method is discussed in detail with its advantages and disadvantages in terms of machining accuracy and speed. The existing techniques suffer from either decreasing the machine speed and thus the overall machine productivity, or from introducing undesirable changes to the reference contours especially in multi-axis machining applications.
In the third part, a new method for attenuating the machine oscillations based on a time-scaling technique is developed. The method uses a localized shaping of the machine reference trajectories via time-scaling to manipulate their frequency spectrum such that they are free of the machine critical frequencies. The localization is achieved by targeting only designated regions in the trajectories, namely those critical regions where the frequency content of the reference trajectories matches the resonance frequencies of the machine axes and the oscillations exceed a given limit. Due to its localized nature this method suppresses the machine oscillations while increasing the overall machining time only by as little as possible, thus maintaining the high-speed characteristics of the machine. In contrast to the existing approaches, the shaping process in this method is done on the path trajectories rather than the axis trajectories. Shaping the path trajectories guarantees that the modifications are applied to all moving axes *simultaneously* and prevents undesirable changes in the reference contours. To characterize the machine oscillations, a new error signal called the *vibrational contour error* was developed. This signal provides a global measure for the oscillatory behavior of the machine on the contour level. The identification process of the critical regions within the vibrational contour error signal uses a specifically designed wavelet-based approach. The scaling function which is used to shape the reference path trajectories is then locally tuned via optimization routines as required by the individual regions.

The methods developed in this thesis were tested using MATLAB simulations and an actual test rig setup. The simulation and real test results proved the effectivity and efficiency of the proposed approach in attenuating the mechanical oscillations in high dynamics multi-axis CNC machine tools while preserving their high-speed capability.

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

$\hat{a}$	Axis maximum acceleration.	$\mathrm{m/s^2}$
a	Wavelet dilation factor.	
a_{i}	Axis acceleration $(\mathrm{i} = 1, 2, 3, \cdots)$.	$\mathrm{m/s^2}$
$\hat{\vec{b}}$	Normalized binormal vector.	
$\vec{B}_{\mathrm{i}}$	Bisector vector $(\mathrm{i} = 1, 2, 3, \cdots)$.	
b	Wavelet translation factor.	s
$c\,(t)$	Scaling function.	
c	Constant scaling function.	
C_{FA}	Acceleration feed forward controller.	$-$
C_{FV}	Velocity feed forward controller.	$-$
$\mathcal{D}_c$	Coordinate set of search directions.	
$\mathcal{D}_p$	Pattern set of Search directions.	
$\mathcal{D}_v$	Vectors set of search directions.	
D	Torsional damping coefficient.	$\mathrm{Nm\,s/rad}$
D_{i}	Double ratio $(\mathrm{i} = 1, 2, 3, \cdots)$.	$-$
$e_c\,(t)$	Contour error.	m
$e_e\,(t)$	Elasticity error.	m
$e_t\,(t)$	Tracking error.	m
$e_{c,vib}\,(t)$	Vibrational contour error.	m
$e_{e,vib}\,(t)$	Oscillatory periodical terms in the elasticity error.	m
$e_{t,\upsilon}\,(t)$	Tracking error in $\upsilon-$ direction with $\upsilon = x, y$ or z.	m
$e_{t,ap}\,(t)$	Aperiodical terms in the tracking error.	m
$e_{t,vib}\,(t)$	Oscillatory periodical terms in the tracking error.	m
$e\,(t)$	Error signal.	$-$
$\tilde{\vec{e}}_c$	Estimate contour error vector.	
$\vec{e}_t$	Tracking error vector.	
$\mathcal{F}\,(\cdot)$	Optimization objective function.	
f	frequency variable.	Hz
F_c	System critical frequency.	Hz
F_n	Natural frequency.	Hz
F_p	Pole frequency.	Hz
F_z	Zero frequency.	Hz
$g\,(t)$	Scaling function template.	
$G\,(s)$	General transfer function.	
$\mathcal{H}\,(\cdot)$	Heaviside unit step function.	
i	Ordinal numbers.	$(\mathrm{i} = 1, 2, \cdots)$
i	Complex number $i = \sqrt{-1}$.	
$\hat{j}$	Axis maximum jerk.	$\mathrm{m/s^3}$
J_L	Load moment of inertia.	$\mathrm{kg\,m^2}$
J_M	Motor moment of inertia.	$\mathrm{kg\,m^2}$
j_{i}	Axis jerk $(\mathrm{i} = 1, 2, 3, \cdots)$.	$\mathrm{m/s^3}$
J_{tot}	Total moment of inertia.	$\mathrm{kg\,m^2}$
k	Ordinal numbers.	$(\mathrm{k} = 1, 2, \cdots)$

K	Torsional spring coefficient.	$\mathrm{Nm/rad}$
K_E	Voltage constant.	$\mathrm{Vs/rad}$
K_v	Position loop proportional controller gain.	$\mathrm{s}^{-1}, \frac{\mathrm{m/min}}{\mathrm{mm}}$
K_{pn}	Velocity loop proportional controller gain.	$\mathrm{Nm\,s/rad}$
K_p	General proportional controller gain.	$-$
m	Ordinal numbers.	$(\mathrm{m} = 1, 2, \cdots)$
M_L	Torque to load.	Nm
M_M	Motor torque.	Nm
$M_M\,(s)$	Motor torque in Laplace domain.	Nm
$\hat{\vec{n}}$	Normalized normal vector.	
$\hat{\vec{n}}'$	Normalized average normal vector.	
$\vec{N}_\mathrm{i}$	Normal vector $(\mathrm{i} = 1, 2, 3, \cdots)$.	
O	Center of curvature.	
p	Transfer function poles.	
P_{ref}	Reference position.	
P_{tcp}	Machine TCP actual position.	
$\dddot{s}\,(t)$	Path jerk.	$\mathrm{m/s^3}$
$\ddot{s}\,(t)$	Path acceleration.	$\mathrm{m/s^2}$
$\dot{s}\,(t)$	Path velocity.	$\mathrm{m/s}$
$\tilde{s}\,(t)$	Path trajectory after time scaling.	m
$s\,(t)$	Path trajectory.	m
$\hat{\dddot{s}}$	Maximum path jerk.	$\mathrm{m/s^3}$
$\hat{\ddot{s}}$	Maximum path acceleration.	$\mathrm{m/s^2}$
$\hat{\dot{s}}$	Maximum path velocity.	$\mathrm{m/s}$
s	Path length.	m
$\hat{\vec{t}}$	Normalized tangential vector.	
$\hat{\vec{t}}'$	Normalized average tangential vector.	
τ	Scaled time.	s
$\vec{T}$	Unit tangent vector.	$-$
T	General time delay.	s
$T_\phi x\,(b, f)$	Time–frequency transform of $x\,(t)$ by means of a time–frequency atom $\phi_{b,f}\,(t)$.	
T_{Ei}	Equivalent delay time of the current control loop.	s
T_{FF}	Feed forward position balancing time delay.	s
T_{nn}	Velocity loop integration time constant.	s
T_n	General integration time constant.	s
$T_{s\theta}$	Sampling time of the position loop.	s
T_{sFIPO}	Sampling time of the fine interpolator.	s
T_{sIPO}	Sampling time of the interpolator.	s
T_{sn}	Sampling time of the velocity loop.	s
$u\,(t)$	Control command signal.	
u	Arc length parameter.	$-$
$\hat{v}$	Axis maximum velocity.	$\mathrm{m/s}$
v_V	Programmed feed rate.	$\mathrm{m/min}$
v_i	Axis velocity $(\mathrm{i} = 1, 2, 3, \cdots)$.	$\mathrm{m/s}$
$w\,(t)$	Time window for the Windowed Fourier transform.	
ψ^*	Wavelet transform coefficients of $x\,(t)$ at time b and scale a.	
$W_\psi x\,(b, \mathrm{a})$	Wavelet transform coefficients of $x\,(t)$ at time b and scale a.	

$x(t)$	General input signal.	
$\hat{x}$	Step height.	m
$\vec{x}$	Vector of optimization parameters.	
$\vec{x}_l, \vec{x}_u$	Lower and upper limits of the optimization parameters vector.	
$X(b, f)$	Windowed Fourier transform of $x(t)$ with translation factor b.	
$y(t)$	General output signal.	
z	Transfer function zeros.	
α	Optimal step length to be taken along the current search direction in the pattern search method.	
α_l, α_u	Lower and upper limits of the Optimal step length.	
$\alpha_{u,0}$	Initial guess for the step size upper bound in the bracketing algorithm.	
β	Wavelet function damping or control parameter.	
$\Delta\alpha_{final}$	Interval tolerance in the golden section method.	
$\delta(\cdot)$	Impulse function.	
δ_α	Interval reduction factor in the golden section method.	
$\dot{\theta}_L = \omega_L$	Load angular velocity.	rad/s
$\dot{\Theta}_L(s)$	Load angular position in Laplace domain.	rad
$\dot{\theta}_M = \omega_M$	Motor angular velocity.	rad/s
$\dot{\Theta}_M(s)$	Motor angular position in Laplace domain.	rad
ϵ_1	Interval tolerance in the golden section method.	
η_s, η_f	Success and failure step length factors in the bracketing algorithm.	
$\phi_{b,f}(t)$	Time–frequency atom with time index b and a frequency index f.	
$\psi(t)$	Wavelet function.	
λ	Control parameter for the exponential and sigmoidal type scaling function template.	
μ_f	Frequency localization of a time–frequency atom.	
μ_t	Time localization of a time–frequency atom.	
ω_c	Wavelet function center frequency.	
ω_d	Damped angular frequency.	rad/s
ω_n	Natural angular frequency.	rad/s
ρ	Radius of curvature.	
σ_f	Frequency spread of a time–frequency atom around μ_f.	
σ_t	Time spread of a time–frequency atom around μ_t.	
θ_L	Load angular position.	rad
θ_M	Motor angular position.	rad
φ	Golden ratio $\varphi = \frac{1+\sqrt{5}}{2}$.	
$\vec{\kappa}$	Curvature vector.	$-$
ζ	Damping ratio.	$-$
C_ψ	Admissibility constant.	
SF	Success/fail indicator in the bracketing algorithm: $0 =$ success, $1 =$ fail, $-1 =$ start.	

Acronyms

CNC	Computer Numerical Controlled
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
NC	Numerical Control
BCS	Basic Coordinate System
IPO	Interpolator
MCS	Machine Coordinate System
FIPO	Fine Interpolator
TOC	Time Optimal Control
SISO	Single-Input Single-Output
PI	Proportional-Integral
ZV	Zero Vibration
ZVD	Zero Vibration Derivative
ZVDD	Zero Vibration Derivative Derivative
EI	Extra Insensitive
TCP	Tool Center Point
FT	Fourier Transform
WFT	Windowed Fourier Transform
STFT	Short Time Fourier Transform
WT	Wavelet Transform
CWT	Continuous Wavelet Transform
s	Second
mm	Milli meter, i.e. $1\,\mathrm{mm} = 10^{-3}$ meter
$\mu\mathrm{m}$	Micro meter, i.e. $1\,\mu\mathrm{m} = 10^{-6}$ meter
rad	Radian
Hz	Hertz, $1\,\mathrm{Hz} = 1$ cycle per second

1 Introduction

1.1 Motivation

The technological advances in production machinery have created an entirely new way to perform machining processes. The introduction of computers to the control of machine tools has made multi-axis Computer Numerical Controlled (CNC) machine tools the default choice for most machining applications. In response to the increasing demand in the complexity and accuracy of the machined shapes, CNC machine tools as well as the manufacturing processes have dramatically evolved focusing the attention of the machine tool community on high-speed and high-precision machining. In these processes, multi-axis machine tools are required to be fast, i.e. to have high feedrate and acceleration capabilities, while still maintaining a high degree of positioning accuracy. The two requirements are intended to reduce the production costs by realizing high feedrates and to provide products that satisfy the customers' needs in terms of machining quality.
The high feedrate and acceleration capabilities of a machine tool require lightweight machines, time optimal motion profiles which utilize the maximum capabilities of the axes of the machine and fast controllers to drive the axes. On the other hand, the high precision necessitates rigid machines, slow motion profiles and well-damped controllers to regulate the axes of the machine. Indeed, the speed and precision requirements are highly contradicting and require a good compromise to achieve acceptable levels in both. The growing demand for an optimal solution for such a compromise between speed and precision in CNC machine tools has resulted in a growing attention towards the topic considered in this thesis.

1.2 Problem Statement

A common practice in the field of CNC machining is to tune the machine for high speed performance, i.e. to use time optimal motion profiles and fast controllers to regulate the axes of the machine, and to introduce additional measures to enhance the accuracy of the machining process. The accuracy of any machining process is, to a large extent, measured by the amount of vibrations caused by the machine tool itself. The primary reasons for the appearance of vibrations in the response of the machine are the existence of elasticities in the machine's structure

and the highly dynamic motion profiles which contain a very wide range of frequencies that can excite the resonance frequencies of the machine. The elasticities in the structure of the machine are often unavoidable and mainly exist due to the lightweight elements in the machine and the existence of transmission elements such as ball-screws, belt-pulleys and gears.
In recent years special attention has been paid to the vibration problem in CNC machine tools and intensive efforts have been made and still are made in several directions to suppress them to significantly low levels. The state of the art solutions suffer from either decreasing the machine speed and thus the overall machine productivity, or from introducing undesirable changes to the reference contours, especially in multi-axis machining applications. In this thesis, a new solution of the vibration problem in multi-axis CNC machine tools via proper design of the reference motion profiles will be presented and investigated.

The general motion control problem in this thesis is to devise motion profiles, i.e. position, velocity, acceleration and jerk, that drive a flexible multi-axis CNC machine tool to machine a given contour with high speed and high precision. More precisely, the problem under study is the design of fast motion profiles which will result in no or insignificant vibrations when used as reference values for a contour following motion control of flexible multi-axis CNC machine tools.
The system under study is a multi-axis CNC machine tool with an independent joint control scheme. By independent joint control we mean that the control inputs of each axis in the machine will depend on the measurements of the position and velocity of that axis only. The dynamics of each axis can be described by a set of rigid body modes and a set of flexible modes which define the resonance frequencies of the corresponding axis. Each axis will be regulated by a cascade controller which contains a feedback compensator and a feed forward controller as shown in Figure 1.1.

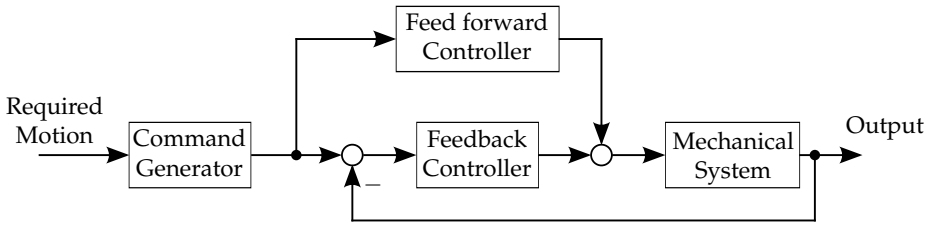


Fig. 1.1: Illustrative block diagram for a closed loop control system

Each axis will have some physical constraints which are caused by one or more mechanical or electronic components. These constraints include the maximal velocity, the maximal acceleration and the maximal jerk which can be attained by the corresponding axis of the machine and a resonance frequency which must be avoided by the command generator when designing the motion profiles.

Definition. *Motion profiles are said to be time and frequency optimal when they result in the best compromise between machining time and amount of mechanical vibrations when used as reference values for CNC machine tools.*

The objective of this thesis is to design time and frequency optimal motion profiles for a flexible multi-axis CNC machine tool to machine a given contour in a minimal machining time and a limited amount of vibrations in the machine response.

1.3 Organization of Chapters

The dynamics of the machine, the different stages of the Numerical Control (NC) kernel and the servo controllers used to regulate the axes of the machine will be recalled in Chapter 2. In Chapter 3, the state of the art methods which address the problem of this thesis are discussed. These methods are classified according to their point of action in the machine process chain. First methods working on the mechanical system of the machine axis are considered, followed by those which work on the control loops of the axes and then the ones which work on the reference trajectories fed to the machine.
In Chapter 4, different types of error signals for describing the machine performance are presented and a new and appropriate type of error signal is developed to characterize the machine oscillatory behavior on the contour level for free-form machining applications.
Chapter 5 presents the techniques to generate the time and frequency optimal motion profiles. A general flow digram for the method is first given, followed by the development of a time-frequency analysis approach for identifying the critical oscillation regions in the reference trajectories and a discussion of the basic principle of the time-scaling concept and its effects on the frequency content of the reference trajectories. At the end of this chapter, the design approach of the scaling function and the optimization routines for tuning the function parameters are discussed.
In Chapter 6, the theoretical findings from the previous chapters are validated through tests using MATLAB simulations and a physical test rig setup. Results from both test methods will be presented.
The thesis will be finished by Chapter 7 which presents the conclusions following this research and gives an outlook for future research directions to build upon this work.

2

CNC Machine Tools: Architecture and Control

Machining a certain geometry using a CNC machine tool passes through several stages before the actual machining takes place. The workflow of the machining process from the Computer Aided Design (CAD) system right through to the actual machining is typically characterized by the so called machining process chain. A general functional representation of such a chain is shown in Figure 2.1.

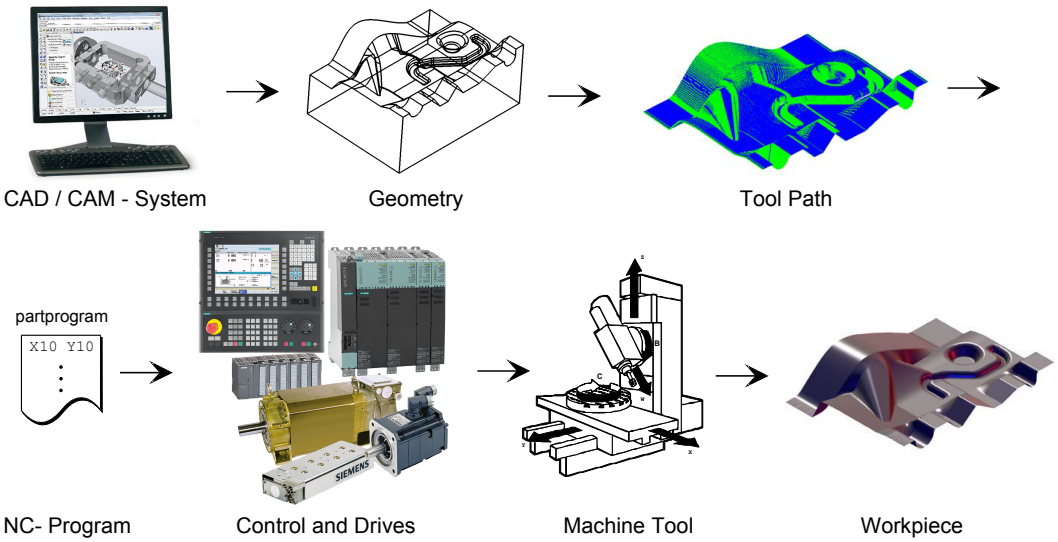


Fig. 2.1: Process chain from CAD/CAM-system to workpiece [1]

The machining process chain starts with the workpiece design using a CAD system. The data generated at this stage is a purely geometrical data describing the workpiece to be machined. As a second step, the Computer Aided Manufacturing (CAM) system converts the geometric data into a tool path trajectory in the form of a machine understandable code called the part program. The control and drives unit, the brain of the CNC machine tool, analyzes the part program code and designs proper time optimal trajectories (position, velocity, acceleration and jerk trajectories) for the machine axes based on their capabilities and provides the machine's

actuators with the required driving signals in order to make sure that the geometry described by the part program is machined as fast as possible and with the requested contouring accuracy.
In this thesis, the tool path trajectory will be assumed to be given in the form of a part program code, normally written in the form of G-Code. The focus will be mainly on the control and drives unit of the process chain. Figure 2.2 shows a typical structure of a control and drives unit of a single axis in a multi-axis CNC machine tool. The structure is, roughly, divided into three main parts: the NC kernel, the servo control and the mechanical system.

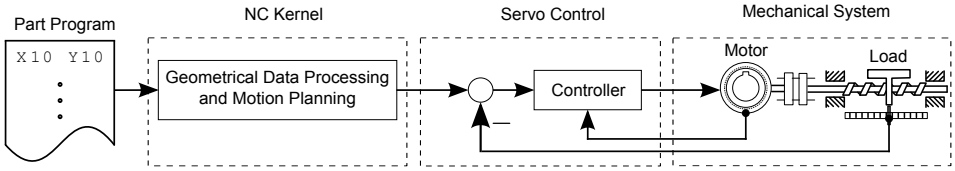


Fig. 2.2: Control structure of one axis in a CNC machine tool

2.1 Numerical Control Kernel

The NC kernel is the unit responsible for the geometrical data processing and motion planning in the control unit. It accepts inputs like the required contour defined by the part program in the form of linear, circular, helical, polynomial or spline blocks and has access to machine data such as the maximal axis velocity, acceleration and jerk. The output of the NC kernel are a time optimal trajectories serving as reference setpoints for the servo controllers. An abstract overview of the basic functional blocks incorporated in the NC kernel is shown in Figure 2.3.

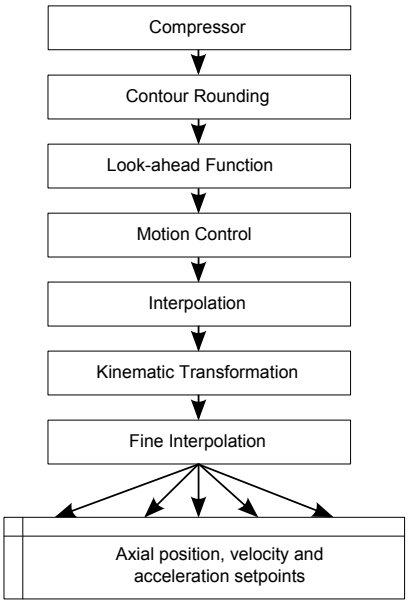


Fig. 2.3: Components of the NC kernel (without position control) [2]

Compressor: It is the first process in the NC kernel. It has the task of approximating the various part program blocks using smooth spline elements within a predefined tolerance. The compressor step substantially reduces the number of blocks used to describe the contour and thus reduces the stored program volume as well as the computing time required by the NC kernel. The compressor step results in a smooth and continuous spline representation of the given contour. The splines are stored in the form of polynomials sets $\vec{P}_x$, $\vec{P}_y$, $\vec{P}_z$, etc. Each set contains multiple polynomials, each of which is parametrized according to the parameter $u \in [0, 1]$.

Contour Rounding: It has the objective of removing discontinuous transitions between neighboring contour blocks. It transforms the discontinuous blocks into tangential (C^1) or curvature (C^2) continuous blocks (C^1 continuity implies equal 1st derivative, whereas C^2 continuity implies equal 2nd derivative at the joining points). The contour rounding function modifies the programmed contour locally by introducing extra "contour rounding blocks" to overcome the discontinuity of the original blocks. Figure 2.4 shows an illustrating example of two linear blocks with the additional rounding block. Whether or not to add the rounding as well as the type and size of that rounding are normally selected by the user in the form of part program commands or machine data.

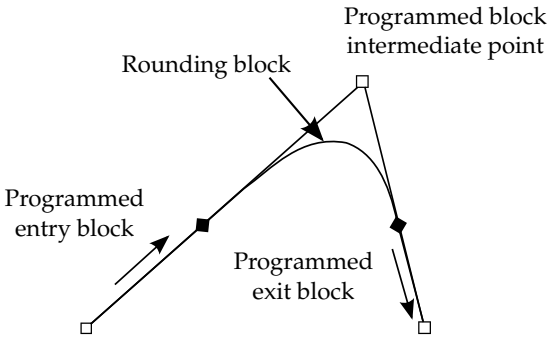


Fig. 2.4: Contour rounding example

Look-Ahead Function: It is the part responsible for calculating the axial restrictions in the NC kernel. It identifies the "critical points" in the contour, where the programmed path feedrate or the velocity, acceleration and jerk limits of the axes may not be maintained. The "critical points" which can influence the axial restrictions include, for example, the singular or simi-singular points and areas with extreme curvatures. The output of this function are the path velocity, acceleration and jerk limitation curves defined for each block in the programed contour. The limitation curves represent a road map for designing the path velocity, acceleration and jerk profiles, which will be discussed in the trajectory planning step later on.

Motion Control Function: Its basic task is the planning of path trajectories. It transposes a motion state vector, which is composed of the scalar path variables (jerk, acceleration, velocity and position) from an existing initial state to a desired final state with minimal time and under axial restrictions. The inputs to the motion control function are the contour description given by the compressor, the contour rounding function and the axial restrictions defined by the

look-ahead function, whereas the output are time optimal path jerk, acceleration, velocity and position trajectories. A more detailed discussion of the path trajectory planning process will be given in Subsection 2.1.1.

Interpolation: The interpolation function has the task of sampling the trajectories and projecting the generated path trajectories into the Basic Coordinate System (BCS). The sampling is done according to a predefined interpolation type and machine sampling rate called the Interpolator (IPO) sampling cycle. The type of interpolation can be linear, quadratic or cubic interpolation. The projection process splits or distributes the path trajectories into Cartesian trajectories such that their combined motion reconstitutes the initially defined contour. The splitting process is defined according to the relationship between the path variables and the BCS as will be shown in Subsection 2.1.1.

Kinematic Transformation: This function has the task of transforming or projecting the time optimal setpoints generated by the interpolator from the BCS to the Machine Coordinate System (MCS). The existence of the transformation step and the kind of transformation to be used depends on the machine in use. For example, Cartesian machines do not need a kinematic transformation step since the MCS is the same as the BCS.

Fine Interpolation: As a final stage in the NC kernel comes the fine interpolation function. It is responsible for mapping the MCS setpoints from the IPO sampling cycle to servo control sampling cycle which is also known as the Fine Interpolator (FIPO) sampling cycle. The type of interpolation can be linear, quadratic or cubic interpolation which is defined via machine data.

2.1.1 Path Trajectory Planning

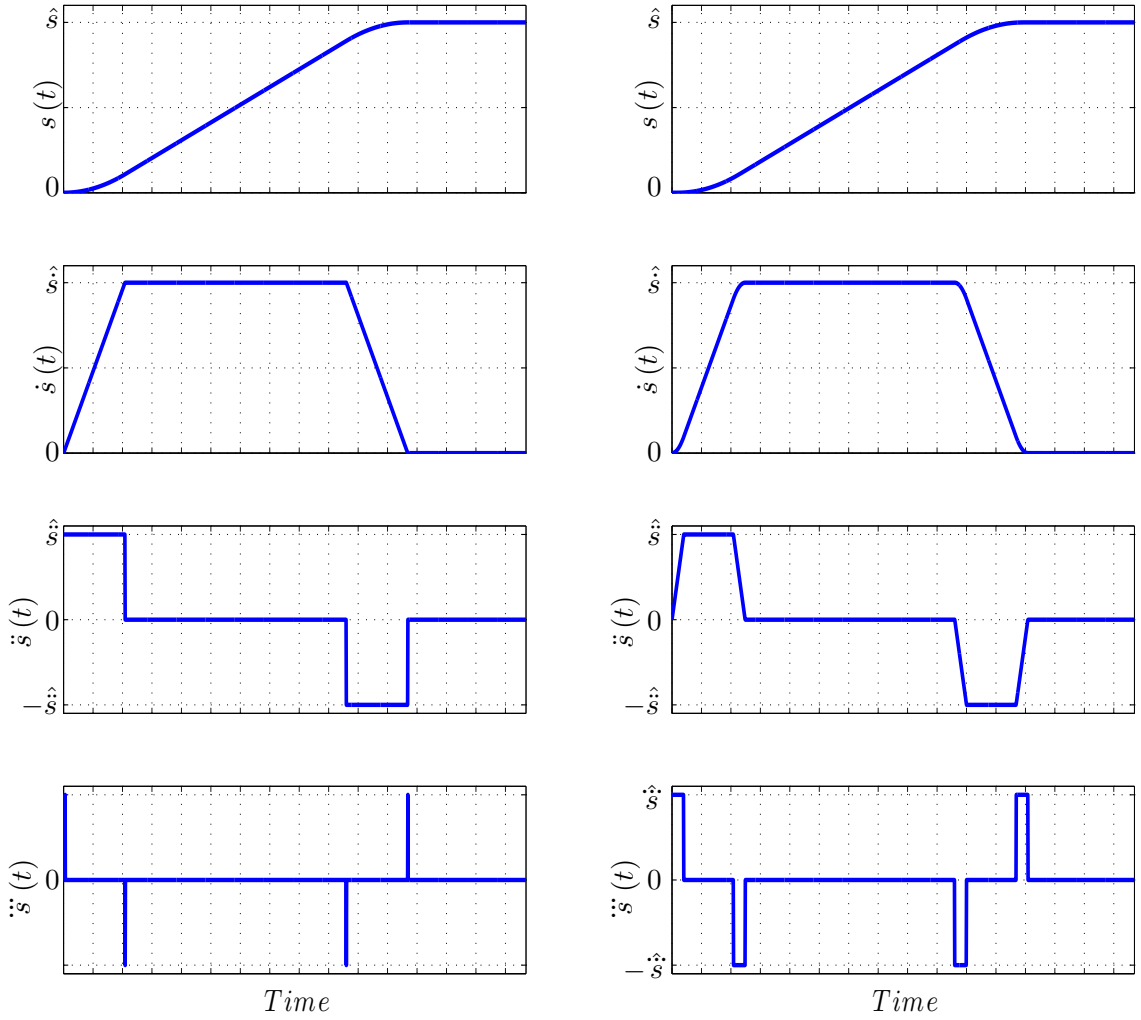
In the context of motion control, a trajectory is defined as a curve that describes the machine evolution in time. It incorporates dynamical information about the machine movement. A path is defined as a curve in the Cartesian space that reflects purely geometrical information about the task to be performed and describes the spatial evolution of the machine cutting tool. In this sense, the path trajectory planning process refers to the process of designing a path trajectory $s\left(t\right)$ with velocity $\dot{s}\left(t\right)$, acceleration $\ddot{s}\left(t\right)$ and jerk $\dddot{s}\left(t\right)$ profiles for a given path.

Several objectives have to be fulfilled in the path trajectory planning process for a CNC machine tool, the most important of which are the generation of trajectories which are smooth, physically realizable by the machine axes and time optimal.

The **smoothness** of the generated trajectories is measured according to their degree of continuity. In this sense, we distinguish between two main motion profiles, namely, the Brisk and the Soft motion profiles, see Figure 2.5.

In the Brisk motion profiles, the path acceleration is allowed to exhibit stepwise changes between its maximum and minimum values $[\hat{\ddot{s}}, -\hat{\ddot{s}}]$. The resulting path velocity is a trapezoidal shaped profile and the trajectory is said to be velocity continuous, whereas in the Soft motion profiles, the path jerk is allowed to exhibit stepwise changes between its maximum and minimum values $[\hat{\dddot{s}}, -\hat{\dddot{s}}]$, leading to a trapezoidal shaped path acceleration and the trajectory is said to be acceleration continuous. It is mathematically proven that the time optimal motion

profiles are Brisk motion profiles. Or according to LaSalle [3] *"if there is an optimal control, then there is always a 'bang-bang' control that is optimal"*. However, due to the lack of smoothness in the acceleration profiles and the induced vibrations from the Brisk motion profiles, the Soft motion profiles are the default choice for most of the industrial applications.



a) Brisk motion profile b) Soft motion profile

Fig. 2.5: Normalized Brisk (left) and Soft (right) motion profiles

To be **physically realizable**, the designed trajectories must maintain some constraints imposed by the machine axes capabilities (system constraints) and others imposed by the geometry of the given path (geometric constraints) [4].
The system constraints reflect the physical limitations of the motors driving the machine axes, such as the maximum motor speed and current. They are represented in the form of maximum axis velocity $\hat{v}$, maximum axis acceleration $\hat{a}$ and maximum axis jerk $\hat{j}$. The individual machine

axis velocity v_{i}, acceleration a_{i} and jerk j_{i} are not to violate these limits.

$$-\hat{v}_{\mathrm{i}} \leq v_{\mathrm{i}} \leq \hat{v}_{\mathrm{i}}, \tag{2.1a}$$

$$-\hat{a}_{\mathrm{i}} \leq a_{\mathrm{i}} \leq \hat{a}_{\mathrm{i}}, \tag{2.1b}$$

$$-\hat{j}_{\mathrm{i}} \leq j_{\mathrm{i}} \leq \hat{j}_{\mathrm{i}}, \tag{2.1c}$$

where $\mathrm{i} = 1, \ldots, n$ and n is the number of axes.

The geometric constraints reflect the dynamics of the path to be machined. In the context of CNC machine tools, this type of constraints appears mainly in the form of high curvature areas in the given path. Such areas generate centrifugal forces that extremely restrict the path velocity, acceleration and jerk.

Given a general geometry described in the Cartesian space by a vector of positions $\vec{P} = [P_x \quad P_y \quad P_z]^T$ and parameterized with respect to the path length parameter s as $\vec{P}(s)$, the relationship between the geometry dynamics and the axis and path velocities, accelerations and jerks for the case of Cartesian machines can be written as

$$\vec{v} = \frac{d\vec{P}}{dt} = \frac{d\vec{P}}{ds}\frac{ds}{dt} = \frac{d\vec{P}}{ds}\dot{s}, \tag{2.2a}$$

$$\vec{a} = \frac{d^2\vec{P}}{dt^2} = \frac{d\vec{P}}{ds}\frac{d^2s}{dt^2} + \frac{d^2\vec{P}}{ds^2}\left(\frac{ds}{dt}\right)^2 = \frac{d\vec{P}}{ds}\ddot{s} + \frac{d^2\vec{P}}{ds^2}\dot{s}^2, \tag{2.2b}$$

$$\vec{j} = \frac{d^3\vec{P}}{dt^3} = \frac{d\vec{P}}{ds}\frac{d^3s}{dt^3} + 3\frac{d^2\vec{P}}{ds^2}\frac{ds}{dt}\frac{d^2s}{dt^2} + \frac{d^3\vec{P}}{ds^3}\left(\frac{ds}{dt}\right)^3 = \frac{d\vec{P}}{ds}\dddot{s} + 3\frac{d^2\vec{P}}{ds^2}\dot{s}\ddot{s} + \frac{d^3\vec{P}}{ds^3}\dot{s}^3, \tag{2.2c}$$

where

$\vec{v}, \vec{a}$ and $\vec{j}$ are respectively the axes velocity, acceleration and jerk vectors,

$\dot{s}$, $\ddot{s}$ and $\dddot{s}$ are the path velocity, acceleration and jerk respectively,

$\frac{d\vec{P}}{ds}$ is the unit tangent vector $\vec{T}$,

$\frac{d^2\vec{P}}{ds^2}$ is the curvature vector $\vec{\kappa}$.

It is worth noting that the machine velocity $\vec{v}$, acceleration $\vec{a}$, and jerk $\vec{j}$ in Equation (2.2) are described in the BCS. The actual velocity, acceleration and jerk of the machine axes are obtained via kinematic transformation from the BCS to the MCS as described in Section 2.1. For the sake of simplicity we limit ourselves in this thesis to the case of Cartesian machines where BCS $\equiv$ MCS.

The unit tangent vector $\vec{T}$ in Equation (2.2a) works as a transformation that maps the path velocity into an axis velocity vector. The magnitude of each element in $\vec{T}$ represents the contribution of the corresponding machine axis to the path velocity. The effects of the path dynamic constraints are more obvious in the axis acceleration, Equation (2.2b). The existence of high curvature areas in the path to be machined implies that most of the axis acceleration capabilities will be used to compensate the path dynamics and only a small part is used for the path velocity and acceleration. This in turn will impose a strong limitation on both path velocity and acceleration.

Combining the system constraints in Equation (2.1) and the geometric constraints in Equation (2.2), we formulate the necessary limitations for the path velocity, acceleration and jerk, which have to be maintained during the trajectory planning process in order to generate a physically realizable trajectories, as follows

$$-\hat{v}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\dot{s} \leq \hat{v}_{\mathrm{i}}, \tag{2.3a}$$

$$-\hat{a}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\ddot{s} + \frac{d^2P_{\mathrm{i}}}{ds^2}\dot{s}^2 \leq \hat{a}_{\mathrm{i}}, \tag{2.3b}$$

$$-\hat{j}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\dddot{s} + 3\frac{d^2P_{\mathrm{i}}}{ds^2}\dot{s}\ddot{s} + \frac{d^3P_{\mathrm{i}}}{ds^3}\dot{s}^3 \leq \hat{j}_{\mathrm{i}}, \tag{2.3c}$$

where $\mathrm{i} = 1, \ldots, n$ and n is the number of axes. For the case of Cartesian machines $n \leq 3$ where the machine axes are related to the x, y and z coordinates.
An additional constraint on the path velocity to be maintained is the programmed feed rate v_V. It is a user defined parameter which controls the maximum feed rate of the machine and it is directly related to some technological characteristics such as the surface finishing quality and the material to be machined:

$$|\dot{s}| \leq v_V. \tag{2.4}$$

Finally, the **time optimality** of the generated trajectories is achieved via minimizing the total traveling time required for realizing such trajectories. This corresponds to designing trajectories with the maximally possible path velocity, acceleration and jerk.
Based on the above discussion, the process of trajectory planing can be formulated as a constrained Time Optimal Control (TOC) problem as follows:

Given a geometric path defined by its Cartesian position vector $\vec{P}$ and parameterized with respect to its path length s as $\vec{P}(s)$ with $s \in [0, s_f]$.
Design a path trajectory $s(t)$ with path velocity $\dot{s}(t)$, path acceleration $\ddot{s}(t)$ and path jerk $\dddot{s}(t)$ with

- *Objective function:*

$$\min J = \int_0^{t_f} 1 \, \mathrm{d}t, \tag{2.5}$$

  where t_f is the trajectories total traveling time.

- *Boundary conditions:*

$$\begin{aligned} \dot{s}\,|_{s=0} &= 0, \qquad \dot{s}\,|_{s=s_f} = 0, \\ \ddot{s}\,|_{s=0} &= 0, \qquad \ddot{s}\,|_{s=s_f} = 0, \\ \dddot{s}\,|_{s=0} &= 0, \qquad \dddot{s}\,|_{s=s_f} = 0. \end{aligned} \tag{2.6}$$

- *System and geometric constraints:*

$$-\hat{v}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\dot{s} \leq \hat{v}_{\mathrm{i}}, \tag{2.7}$$

$$-\hat{a}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\ddot{s} + \frac{d^2P_{\mathrm{i}}}{ds^2}\dot{s}^2 \leq \hat{a}_{\mathrm{i}}, \tag{2.8}$$

$$-\hat{j}_{\mathrm{i}} \leq \frac{dP_{\mathrm{i}}}{ds}\dddot{s} + 3\frac{d^2P_{\mathrm{i}}}{ds^2}\dot{s}\ddot{s} + \frac{d^3P_{\mathrm{i}}}{ds^3}\dot{s}^3 \leq \hat{j}_{\mathrm{i}}. \tag{2.9}$$

- *Programed feed rate constraints:*

$$|\dot{s}| \leq v_V. \tag{2.10}$$

The problem as formulated above is a highly nonlinear constrained optimization problem. An explicit closed-form solution for such a problem is impossible to achieve and an algorithmic solution is also difficult to realize, especially for multi axis machines. However, for industrial applications, there are several well–developed online or near online approaches for solving this problem, see for example [5–8].
The detailed design and solution of the trajectory planning process is not in the scope of this thesis, a more detailed description for the state of the art of Siemens solution for the trajectory planning process can be found in [9, 10].

2.2 Mechanical System

The mechanics of a CNC machine tool consists of a group of axes, also known as feed drives, each of which consists of a motor to provide the driving forces or torques and a train of mechanical transmission elements and links that perform the actual movement. A schematic configuration of a typical machine tool feed drive is shown in Figure 2.6.

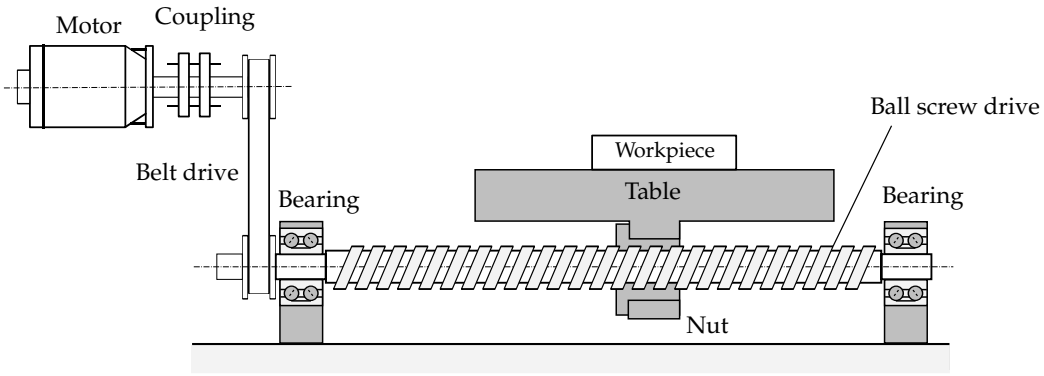


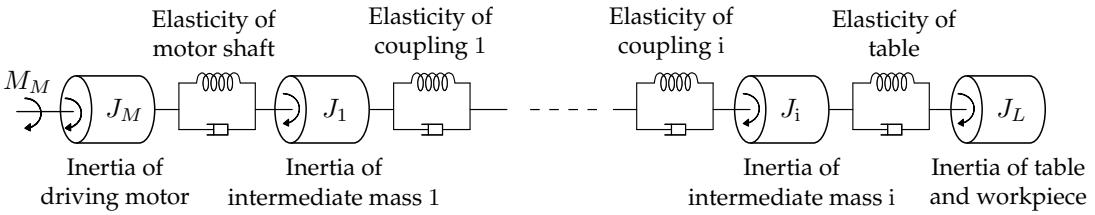
Fig. 2.6: Schematic configuration of a typical machine tool ball screw feed drive

In the above feed drive system, the servomotor torque is transmitted to the ball screw shaft through a belt drive system. The Screw-Nut mechanism converts the servomotor rotational motion into a linear motion and moves the table which holds the workpiece and attached to the nut. The different shafts in the system are coupled together through elastic couplings. Bearings are used to support the shafts and allow the smooth rotational movement.

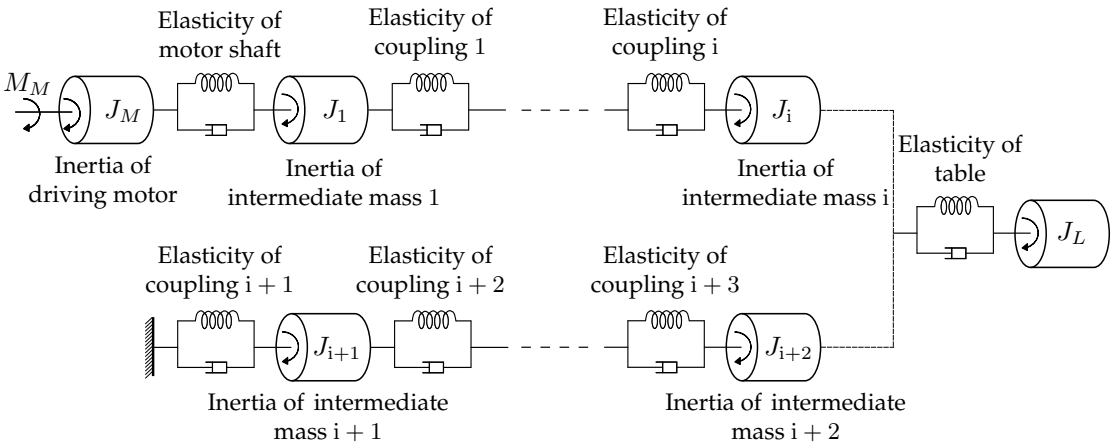
The dynamical behavior of the feed drive system is greatly influenced by the mechanical properties, i.e. mechanical stiffness and damping, and dimensions of the transmission and connection elements. Although the system identification theory or more precisely the black-box system identification theory is a well known and developed theory, industries prefer the use of direct modeling procedures based on the dynamical analysis of the system to be modeled in order to derive more robust and inside models. The dynamical modeling of the feed drive systems in industrial applications is normally done by using the lumped masses method [2, 11–14].

In such a method, the different elements of the feed drive system are approximated by two modes: a rigid-body mode represented by a single mass with an inertia depending on the dimension and material of the modeled element and a flexible mode represented by a spring and/or damper to model the elastic behavior of the modeled element [15].

Figure 2.7 shows schematic representations of two possible feed drive models, a and b, using the lumped masses method. The shape and structure of the resulting multi mass-spring-damper system depends on the mechanical structure and complexity of the modeled feed drive. The different spring and damper coefficients and mass inertias depend on the mechanical properties, material and dimensions of the elements composing the feed drive.



a) First modeling configuration



b) Second modeling configuration

Fig. 2.7: Schematic representations of two possible feed drive models using the lumped masses method

Creating detailed and accurate models for the machine tool feed drives is an essential require-

ment for achieving high performance CNC machine tools. However, in practice it happens often that practical limitations constrain the level of modeled details. Instead, industries intend to simplify the feed drive models up to a minimal realization where only the basic dynamical behavior of the feed drive is modeled. Intensive investigations in this direction showed that a simple two mass-spring-damper system is usually enough to model each elastic mode in the feed drive system and the resulting feed drive model is a serial train of two mass-spring-damper systems as depicted in Figure 2.7.a. [16].
For the sake of simplicity and lucidness of ideas, but without loss of generality, we will concentrate on the main building block of such models in this thesis, namely the two mass-spring-damper system shown in Figure 2.8. Hereinafter we will consider the two mass-spring-damper system as a representative model for each axis in the CNC machine tool.

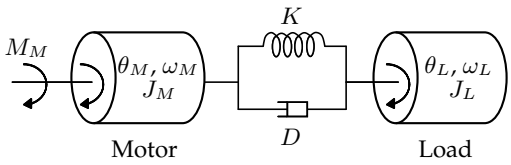


Fig. 2.8: CNC machine tool axis model as a two mass-spring-damper system

Using Newton's laws of motion, the equations of motion governing the two mass-spring-damper system are written as

$$J_M\ddot{\theta}_M(t) + D\left(\dot{\theta}_M(t) - \dot{\theta}_L(t)\right) + K\left(\theta_M(t) - \theta_L(t)\right) = M_M(t), \tag{2.11}$$

$$J_L\ddot{\theta}_L(t) + D\left(\dot{\theta}_L(t) - \dot{\theta}_M(t)\right) + K\left(\theta_L(t) - \theta_M(t)\right) = 0, \tag{2.12}$$

where

M_M is the driving motor torque,

J_M and J_L are the motor and load inertias respectivly,

$\dot{\theta}_M$ and θ_L are the motor velocity and position,

K is the torsion spring constant,

D is the damping coefficient,

$\dot{\theta}_L$ and θ_L are the load velocity and position.

From Equations (2.11) and (2.12), the transfer function describing the relationship between the input torque M_M and the motor angular position $\theta_M(t)$ can be written as

$$\frac{\Theta_M(s)}{M_M(s)} = \frac{J_L s^2 + Ds + K}{J_M J_L s^4 + D\left(J_M + J_L\right)s^3 + K\left(J_M + J_L\right)s^2}, \tag{2.13}$$

and the transfer function describing the relationship between the motor angular position $\theta_M(t)$ and load angular position $\theta_L(t)$, also known as the mechanics of the system, is written as

$$\frac{\Theta_L(s)}{\Theta_M(s)} = \frac{Ds + K}{J_L s^2 + Ds + K}. \tag{2.14}$$

The block digram describing the two mass-spring-damper system, following the convention in [15], is shown in Figure 2.9

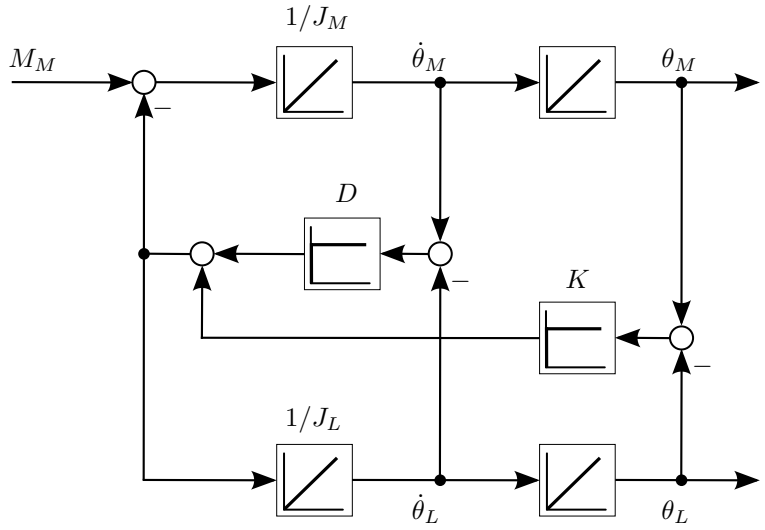


Fig. 2.9: Two mass-spring-damper system block diagram

2.3 Servo Control

CNC Machine tools in industrial applications are generally controlled using independent joint control schemes [17]. In this type of control, each axis in the machine tool is modeled and controlled independently as a Single-Input Single-Output (SISO) system. Coupling effects among axes due to varying configurations during motion are treated as disturbance inputs. The controllers in use to control each axis are servo controllers [18]. The basic reasons for preferring servo controllers over open loop controllers include the need to improve transient response times, reduce the steady state errors, reduce the sensitivity to changes in load and system parameters and a better handling of disturbances.

2.3.1 Cascade Control

The predominant control structure in the field of CNC machine tools is the cascade control structure [15]. The simple structure and high disturbance rejection properties the cascade control has, favors it over the other control schemes. In addition, the cascade configuration offers the following advantages in comparison to other methods [15]:

- Step-by-step start-up from the innermost to the outermost control loop. Each control loop can be adjusted efficiently and independently, ensuring a safe start-up of the entire system.
- The internal control variables can be easily limited via the command variable of the corresponding control loop.
- Effects of non-linearities are controlled and limited, i.e. the higher-level loop operates with improved non-linearities.

- If multiple delay times occur in the controlled system, they can be reduced or canceled for the higher level controller through compensation in a lower level control loop.
- Disturbance variables in lower level control loops will be cancelled there. They do not have to pass through the entire controlled system, and can therefore be more quickly compensated.

An example for a cascade controller structure for a feed drive system is shown in Figure 2.10. The actuating device in Figure 2.10 is a simple transistor power converter which acts as the actuating device for the axis actuator. It supplies the motor with the necessary power to drive the system with the required acceleration.

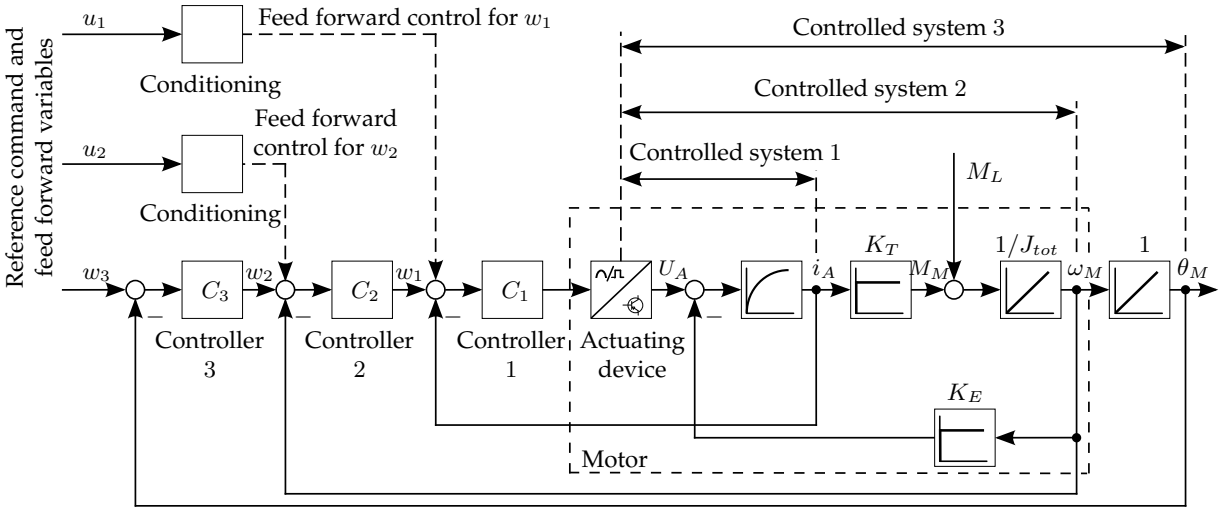


Fig. 2.10: Block diagram of cascade control structure for a CNC machine tool feed drive [15]

The motor in Figure 2.10 is characterized by the torque constant K_T and the voltage constant K_E. $1/J_{tot}$ represents the mechanical time constant of the motor with J_{tot} as the motor total moment of inertia. The coupling effects among the machine axes are described by the disturbance torque M_L. The motor current is controlled by control loop 1 via controller C_1, the motor angular velocity ω_M is controlled by control loop 2 via controller C_2 and the motor angular position θ_M is controlled by control loop 3 via controller C_3.
The current controller is usually realized with Proportional-Integral (PI) controllers integrated in the motor drive system and tuned to its optimal gains before it is placed in use [2, 15]. Therefore the current control loop and the motor in Figure 2.10 are usually approximated by a first order lag element with a time constant T_{Ei} dependent on the motor data as

$$G_{Ei}(s) = \frac{1}{T_{Ei}s + 1}. \tag{2.15}$$

The simplified cascade control structure with the equivalent time constant for the current control loop is shown in Figure 2.11

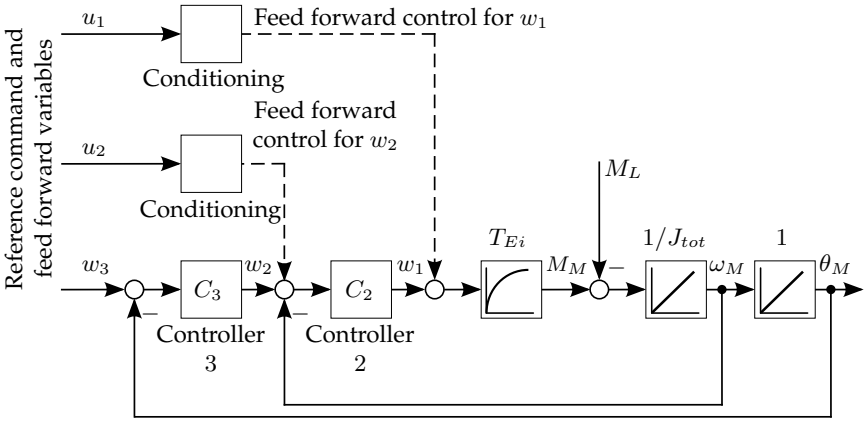


Fig. 2.11: Block diagram of a simplified cascade control structure for a CNC machine tool feed drive

The second controller C_2 in Figure 2.11 is the velocity loop controller. This loop is one of the most important control loops in the feed drive system. It includes, besides the high noise, the most crucial mechanical disturbances M_L in the system, the ones modeling the axes interaction. Therefore, keeping a high performance in the velocity control loop is a priority for any successful feed drive servo controller design process. In practical applications, this loop is normally controlled via PI controller, due to its immediate response to input signals and its ability to eliminate the steady state system errors.The a PI controller algorithm is described by

$$u\left(t\right) = K_p \left[e\left(t\right) + \frac{1}{T_n} \int_0^t e\left(t\right)\, \mathrm{d}t \right], \tag{2.16}$$

where

$u\left(t\right)$ is the control command signal,

$e\left(t\right)$ is the control error signal,

K_p is the proportional gain,

T_n is the integration time constant,

and a block diagram as shown in Figure 2.12

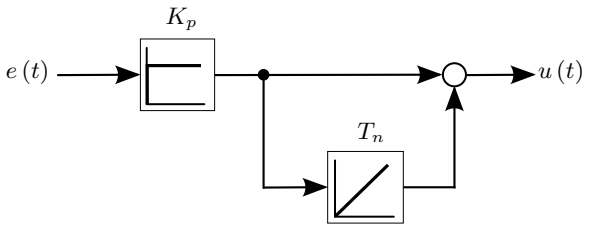


Fig. 2.12: Proportional-Integral controller

The third controller C_3 in Figure 2.11 is the position loop controller. It is common in machine tools industry to use only proportional controllers in this loop, since integration elements can

cause overshoot and this is not allowed at the position level. Moreover, as a result of the cascade control configuration, all system disturbances are well eliminated in the current and velocity loops leaving the position loop to react only according to changes in the commanded positions and a proportional controller is sufficient in this case. The most important variable, which describes the behavior of a position control loop is the position loop gain or the K_v-factor. It gives a measure for machine's tracking or following error and gives an indication for the machine speed of response [19].
In addition to the cascade configuration, Figure 2.11 shows multiple feed forward control paths. The feed forward control is a common practice for minimizing the following errors of the servo controllers in CNC machine tools. The basic idea of the feed forward control is to bypass the servo controllers and pass the reference values, i.e. position, velocity and/or acceleration, directly to the corresponding control loops. The values are fed to the loops via matching elements, called conditioning in Figure 2.11, directly as additive command variables to the corresponding controllers. A correct conditioning of the feed forward variables will lead to the result that the bypassed controller must only correct the disturbance variables of its own control loop, and the end result is an over all speeding up of system response and so better tracking behavior. In this thesis we will consider two feed forward paths, namely a velocity and acceleration feed forward. The different feed forward control strategies are not in the scope of this thesis, a detailed discussion of this topic can be found in [15].

2.3.2 Servo Controller for Elastic System

Combining the different elements discussed in the previous sections, Figure 2.13 shows the servo controller with feed forward for the elastic CNC machine tool axis shown in Figure 2.9 which will be used in this thesis.

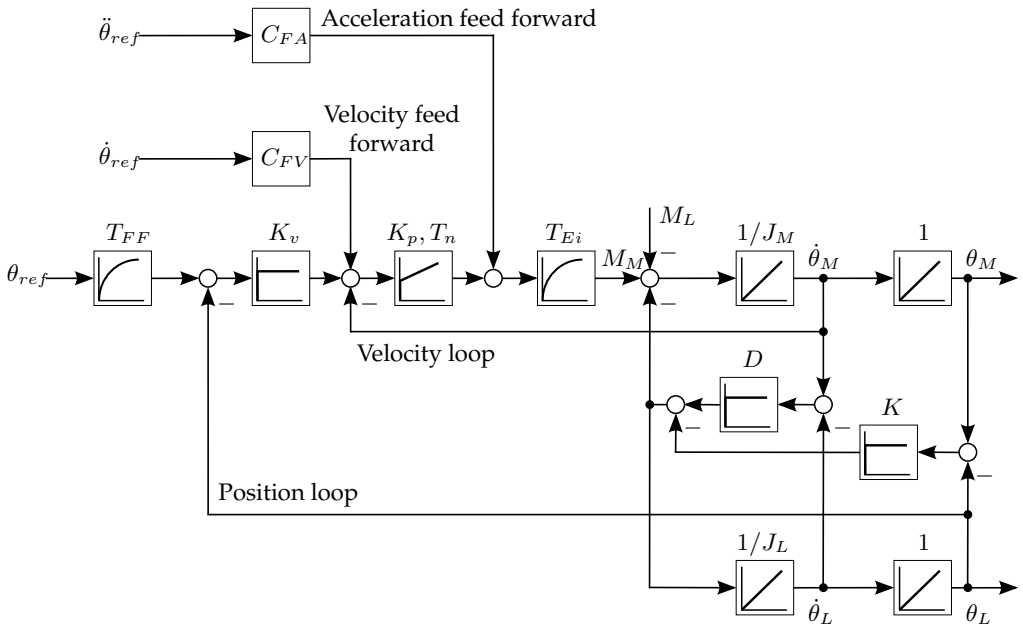


Fig. 2.13: Elastic CNC machine tool axis with servo controller and feed forward

T_{FF} in front of the position loop in Figure 2.13 is a first order lag element for balancing the effects of the additional velocity and acceleration feed forward paths. C_{FV} and C_{FA} are the velocity and acceleration feed forward controllers, respectively. They are proportional controllers with, sometimes, additional first order lag elements as needed for loops balancing. The velocity control loop is closed with a feedback from the motor side for stability reasons. The position control loop is closed either with a feedback from the motor side or from the load side. Depending on the location at the machine tool where the measured data acquisition takes place, we differentiate between direct (from load side) and indirect (from motor side) feedback. In this thesis, we consider the direct feedback only.

The parameters of the velocity and position controllers K_{pn}, T_{nn} and K_v are normally tuned experimentally on the already assembled machine tool with the help of some rules of thumb [19]. In this thesis the tuning process is done following the steps of [15]. The process is carried out in both frequency and time domain. In the frequency domain the parameters are tuned to achieve maximal stability, whereas in time domain characteristics such as oscillation and speed of response are considered. Numerical values for the parameters of the different controllers will be given in Chapter 6.

3
State of the Art

Suppressing residual vibrations in motion control of flexible structures can be tracked back to the early 70's where the motion control of flexible manipulator arms was first studied. Several methods and techniques have been intensively studied in this field, from feedback controllers to the shaped function synthesis (form functions), passive damping techniques, impulse shaping filters, system inversion based motion planning and many others. A detailed survey on the different available approaches can be found in [20]. In this chapter a quick summary for the available state of the art techniques is provided. The different vibration attenuation techniques are classified according to their point of action in the control structure of the machine tool axes in Figure 3.1.

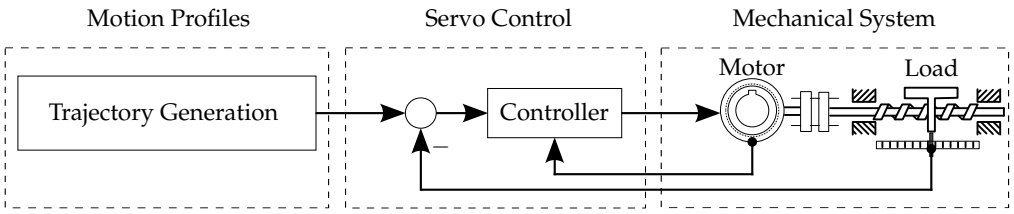


Fig. 3.1: Control structure of CNC machine tool axis

Based on Figure 3.1, the available attenuation techniques are classified into three main categories. The first involves methods acting on the system's mechanical structure to improve its damping characteristics, a category known as passive damping techniques. The second involves methods acting on the control structure and parameters used to control the machine axes to increase the axis damping. The third considers methods acting directly on the reference motion profiles fed to the machine tool. They modify the shape of the reference motion profiles to avoid the excitation of the machine resonance frequencies. This category involves methods based on shaping filters and others based on form functions.

3.1 Passive Damping Techniques

In the context of vibration damping, one distinguishes between two different types of damping techniques, active and passive damping. Active damping refers to energy dissipation from the

system by external means. In this approach, the vibrational behavior of the mechanical system is actively controlled via special control schemes. Passive damping refers to energy dissipation within the mechanical structure itself by physically adding mechanical damping to the system. Passive damping techniques include increasing the overall machine inertia, the use of vibration isolators, the application of thin layers of visco-elastic material to absorb energy and the use of dynamic vibration absorbers. A very good survey on the available mechanical damping methods is found in [20–24].
The extra structural rigidity in the passive damping techniques is obtained at great expense. Increasing the machine inertia leads to a tremendous rise in the machine construction costs and lowers the machine's dynamic behavior. This, in turn, limits the maximum attainable acceleration by the machine and leads to slow machining. Furthermore, passive damping techniques do not always lead to satisfactory results since they are considered at early stages of the machine design process where it is hard to accurately predict the machine vibrational modes.

3.2 Controller Tuning Techniques

Another approach to attenuate the residual vibrations in machine tools is based on fine tuning the servo control loops parameters. It compensates the flexibilities of the machine axes by increasing the overall damping of the closed control loops by optimally tuning the controller parameters. This can be achieved via empirical methods, using either time domain analysis for the step response, or in the frequency domain based on the resonant rise as an evaluation criteria [15]. Although empirical methods can deliver "good" results, they are hard to automate and require intensive experience to handle. On the other hand, the literature offers a number of different well–developed strategies to determine the optimal values of the controller parameters theoretically. Among those, the double ratios method is the most efficient and most often used tuning method for optimal damping of the closed control loops of industrial CNC machine tools.

The double ratios method belongs to the damping optimal type methods for tuning the controllers. Assuming a linear closed control loop transfer function with m as the highest numerator order and n as the highest denominator order, $n \geq m$, given by

$$G(s) = \frac{b_0 + b_1\, s + b_2\, s^2 + \cdots + b_m s^m}{a_0 + a_1\, s + a_2\, s^2 + \cdots + a_n s^n}. \tag{3.1}$$

The numerator represents the non-homogeneous part of the system's differential equations. The coefficients b_m determine the system response characteristics to input signals. The denominator represents the homogeneous part of the system's differential equation, i.e. the feedback coupling between the system states. Its coefficients, $a_0, \ldots, a_n$, determine the damping ratio and the stability of the system. Thus, introducing any damping to the control system can only be achieved by manipulating the denominator coefficients of the transfer function.

In the double ratios method, an optimal response is achieved by constraining the relationship between the denominator coefficients $a_0, \ldots, a_n$. This constraint is defined in the form of dimensionless ratios D_{i} as [25]

$$D_{\mathrm{i}} = \frac{\frac{a_{\mathrm{i}}}{a_{\mathrm{i}-1}}}{\frac{a_{\mathrm{i}-1}}{a_{\mathrm{i}-2}}} = \frac{a_{\mathrm{i}} a_{\mathrm{i}-2}}{a_{\mathrm{i}-1}^2} \leq \frac{1}{2} \qquad \text{with} \quad \mathrm{i} = 2, \ldots n. \tag{3.2}$$

With the help of this relation, the control parameters contained in the coefficients a_0 through a_n can be systematically calculated in such a way that the overall system damping behavior is optimal. The double ratios method leads to a good reference tracking and disturbance rejection as well as to a well–damped control response with a large degree of insensitivity to parameter fluctuations and approximations in the system structure.
The strength of the double ratios method can be better illustrated with the following example. Consider a closed control system described by a second order transfer function as

$$G(s) = \frac{y(t)}{x(t)} = \frac{1}{1 + a_1 s + a_2 s^2}. \tag{3.3}$$

According to the double ratios rule in Equation (3.2), the relationship between the denominator coefficients a_1 and a_2 is given by

$$D_2 = \frac{a_2 a_0}{a_1^2} = \frac{1}{2} \qquad \Rightarrow \qquad a_2 = \frac{a_1^2}{2}. \tag{3.4}$$

Using Equation (3.4), the system transfer function in Equation (3.3) is rewritten as

$$G(s) = \frac{1}{1 + a_1 s + \frac{a_1^2}{2} s^2} \tag{3.5}$$

$$= \frac{1}{(s + a_1 + i\, a_1)(s + a_1 - i\, a_1)}. \tag{3.6}$$

The above transfer function has a pair of complex conjugate poles with equal real and complex parts at

$$p_{1,2} = -\frac{a_1}{2} \pm i\, \frac{a_1}{2}. \tag{3.7}$$

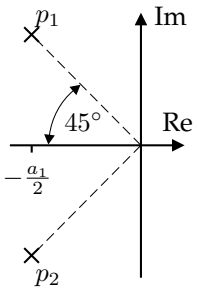


Fig. 3.2: Second order control system poles according to the double ratios method

with system damping ratio defined as

$$\zeta = \cos(45^\circ) = \sqrt{\frac{1}{2}} = 0.707, \tag{3.8}$$

which is the optimal damping ratio for linear second order systems. This is not a coincidence, but rather a property of the double ratios optimization method. Apart from the system order, the method selects the denominator coefficients such that the system damping ratio is always optimal.

For testing the double ratios method, the control parameters of each axis in the machine tool were tuned such that the motor side of the axis is practically enforced and it behaves approximately in the same manner as the reference trajectories, i.e. $\theta_M(t) \approx \theta_{ref}(t)$. As a result, the load side of the machine axis will behave according to the transfer function of the mechanics of the axis which is defined by the second order system in Equation (2.14), a more detailed discussion on this point will be given in Section 4.2. With the above assumption, a valid approximation is to test the double ratios method using a second order underdamped system.

Figure 3.3 demonstrates the response of a second order system for a unit step input after optimizing its parameters using the double ratios method. Although the response demonstrates a well damped behavior, it shows some undesirable overshooting and the system response time is relatively high. The overshooting behavior can be easily removed using filters. Unfortunately, the system speed of response cannot be improved by any means, hence, the slow speed of response is the most crucial drawback of the damping optimum methods.

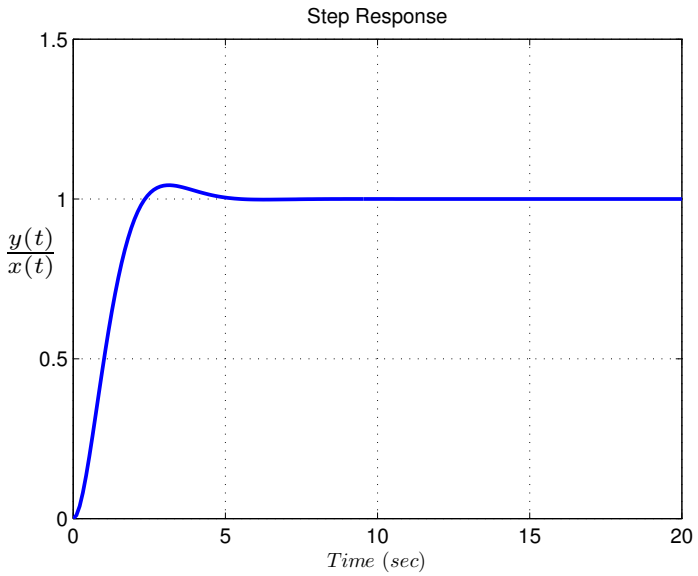


Fig. 3.3: Second order control system step response after double ratios

The double ratios method provides a great solution for attenuating the residual vibrations in CNC machine tools. It reduces the machine residual vibrations by limiting the bandwidth of the machine axes' control loops. This, in turn, decreases the system speed of response and thus the machine productivity. Since the main aim of this study is to remove the machine residual

vibrations while maintaining high speed characteristics, using the double ratios method is not a satisfying solution for the problem at hand.

3.3 Motion Profiles Techniques

Yet another category of methods to attenuate the residual vibrations in CNC machine tools is the one acting directly on the motion profiles fed to the machine. With these methods, the mechanical system is designed for high dynamics and the control parameters are tuned to achieve maximum speed of response. The existing flexibilities in the machine are compensated through modifying the motion profiles to be fed to the machine as reference trajectories. This category is further subdivided into two classes: the first contains methods based on shaping filters where trajectories are designed in the classical way and then modified using shaping filters, the second considers methods based on form functions where special functions are used in the trajectory design process to compensate the machine flexibilities.

3.3.1 Shaping Filters

The main reason for the oscillations in the machine response is the existence of frequencies in the input trajectories which are matching or are close to the machine resonance frequencies. The purpose of shaping filters, also known as shapers, is to remove such frequencies from the reference trajectories. The shaping techniques are based on the filter theory developed for signal processing. They are not intended to prescribe the reference trajectories, but just modify their frequency content to reduce the residual vibration. The reference trajectories have to be carefully designed by the trajectory planner of the CNC machine tool to satisfy the different geometrical and system constraints. Shapers are generally not meant to be used within closed loop systems. This strategy will not be effective since cascading a filter with the system would essentially be equivalent to adding a compensator, and the freedom of manipulating the root locus of the system to avoid the oscillations will be limited [26]. For this reason, shapers are normally located between the reference input and the closed loop system as illustrated in Figure 3.4.

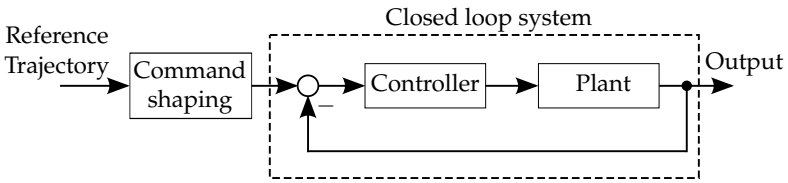


Fig. 3.4: Closed loop input shaping techniques

In general, all existing methods for reference trajectory shaping share the same concept of altering the shape of the reference trajectories so that the system oscillations are reduced or eliminated. As examples, in this thesis only two of such methods will be discussed namely the *Posicast* method for being the origin of the command shaping techniques and the *Input Shaper* method for being the most used method.

- **Posicast Method**

It is one of the earliest forms of command shaping techniques introduced by Tallman and Smith [27]. The method is based on splitting the reference input into several segments such that all transient terms sum to zero after the last excitation. Figure 3.5 illustrates the principle of a Posicast controller applied to an undamped second order system. The reference command to the system is the heaviside unit step function $\mathcal{H}(t)$. The step is broken down into two smaller steps each of a certain magnitude, in this case 0.5. The first step is fed directly to the system, whereas the second is delayed by a time T. The system response to each of the two steps is harmonic with high amplitude and frequency matching the system natural frequency ω_n as shown in Figure 3.5 at the right side top and bottom graphs. The delay time T is selected to be half the period of the system harmonic output $\frac{\pi}{\omega_n}$. In effect, the system responses will have oscillations with the same magnitude but out of phase and the superposition of the two responses leads to vibration cancellation. Summing the outputs of the system to the two step inputs results in a non-oscillatory response as shown by the middle right graph of Figure 3.5.

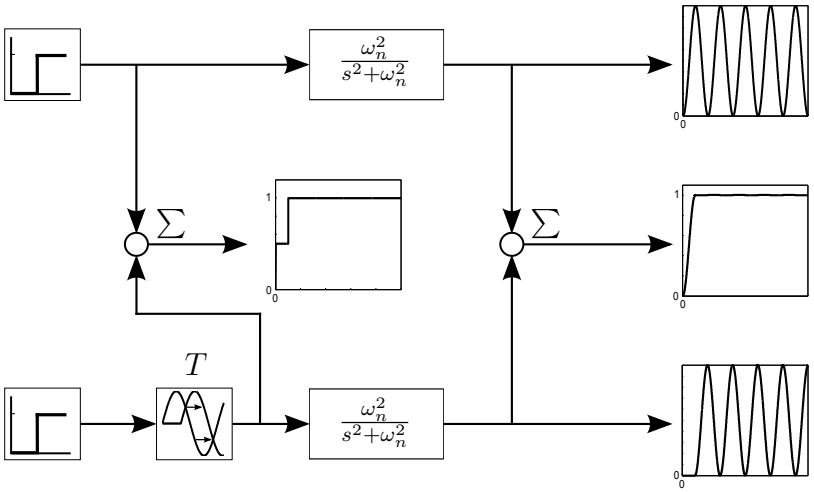


Fig. 3.5: Posicast Control [28]

The Posicast controller can also be interpreted as a time delay filter. The concept is straightforward to see if the relationship between the original unit step function and the actual input signals to the second order system in Figure 3.5 is considered. The actual input signal to the second order system is defined by

$$u(t) = 0.5\,\mathcal{H}(t) + 0.5\,\mathcal{H}\left(t - \frac{\pi}{\omega_n}\right). \tag{3.9}$$

This is written in the Laplace domain as

$$\begin{aligned} U(s) &= \frac{0.5}{s} + \frac{0.5}{s}\,e^{-s\frac{\pi}{\omega_n}} \\ &= \frac{1}{s}\left(0.5 + 0.5e^{-s\frac{\pi}{\omega_n}}\right), \end{aligned} \tag{3.10}$$

which can be seen as the output of a time delay filter subject to a unit step input $R(s)$ and has a transfer function as:

$$\frac{U(s)}{R(s)} = 0.5 + 0.5e^{-s\frac{\pi}{\omega_n}}. \tag{3.11}$$

This transfer function has complex conjugate zeros at:

$$z = \pm i\,(2\mathrm{k}+1)\,\omega_n, \tag{3.12}$$

where $\mathrm{k} = 0, 1, 2, \ldots$.
One pair of zeros in Equation (3.12) lies at $\pm i\omega_n$ which cancels the poles of the undamped second order system of Figure 3.5. The main concept here is to generate a time delay filter with zeros "exactly" the same as the controlled system poles so they can be canceled out and no oscillations are generated. This concept of zero-pole cancellation has attracted many researchers and led to the development of many powerful time delay filters for suppressing residual vibrations in elastic systems.

- **Input shaper Method**

Since the introduction of the Posicast controller, various types of filters have been introduced such as the time delay filters, notch filters and the casual and non-casual shaping filters, among which input shapers, introduced by Singer and Seering [29], are the most often used filters in the field of elastic systems control. The method uses the system's natural tendency to vibrate to cancel the residual vibrations. The basic principle is to excite the elastic system by two impulses with suitably designed amplitudes and time locations such that the resulting oscillations from both impulses are canceled out. Figure 3.6 illustrates the working principle of the method. A_1 is the first impulse applied at time $t_1 = 0$ and A_2 is the second impulse applied at time $t_2 = T$.

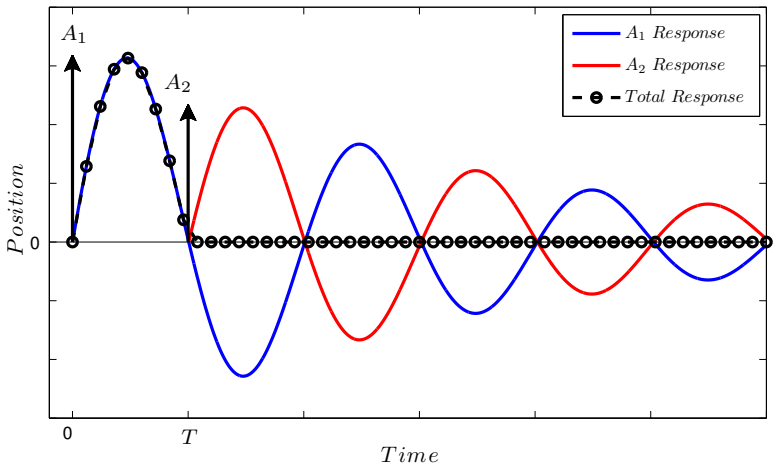


Fig. 3.6: The two impulses vibration cancellation concept [29]

The impulses' magnitude and their appliance times are derived based on linear system theory. Without loss of generality, the derivation will be done using a second order underdamped

system with transfer function

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n\, s + \omega_n^2}, \tag{3.13}$$

where

ω_n is the system natural frequency,

ζ is the system damping ratio.

The system response to an impulse input applied at a time instance t_0 with a magnitude A is given by

$$y(t) = \left[A\frac{\omega_n}{\sqrt{1-\zeta^2}}e^{-\zeta\omega_n(t-t_0)}\right]\sin\left(\omega_n\sqrt{1-\zeta^2}\,(t-t_0)\right). \tag{3.14}$$

The system response to a multi-impulse input, each with magnitude A_i and applied at time instance t_i, is given by

$$y(t) = B_1\sin(\omega_d\,(t-t_1)) + B_2\sin(\omega_d\,(t-t_2)) + \cdots + B_N\sin(\omega_d\,(t-t_N)), \tag{3.15}$$

where

$\omega_d = \omega_n\sqrt{1-\zeta^2}$ is the system damped angular frequency,

$B_\mathrm{i} = \frac{A_\mathrm{i}\omega_n e^{-\zeta\omega_n(t-t_\mathrm{i})}}{\sqrt{1-\zeta^2}}, \quad \mathrm{i} = 1,\ldots,N,$

N is the number of impulses.

Using trigonometric identities, Equation (3.15) is rewritten as:

$$y(t) = A_{amp}\sin(\omega_d t + \beta), \tag{3.16}$$

where

$$A_{amp} = \sqrt{\left(\sum_{\mathrm{i}=1}^{N} B_\mathrm{i}\cos(\phi_\mathrm{i})\right)^2 + \left(\sum_{\mathrm{i}=1}^{N} B_\mathrm{i}\sin(\phi_\mathrm{i})\right)^2},$$

$$\beta = \tan^{-1}\left(\frac{\sum_{\mathrm{i}=1}^{N} B_\mathrm{i}\cos(\phi_\mathrm{i})}{\sum_{\mathrm{i}=1}^{N} B_\mathrm{i}\sin(\phi_\mathrm{i})}\right),$$

$$\phi_\mathrm{i} = \omega_d\, t_\mathrm{i}$$

To eliminate the residual vibrations after the last impulse, A_{amp} must be zero at the time where the last impulse is applied t_N. This is achieved when both squared terms in the A_{amp} equation equal zero, which can be written as:

$$B_1\cos(\phi_1) + B_2\cos(\phi_2) + \cdots + B_N\cos(\phi_N) = 0, \tag{3.17}$$

$$B_1\sin(\phi_1) + B_2\sin(\phi_2) + \cdots + B_N\sin(\phi_N) = 0. \tag{3.18}$$

Using trigonometric identities again and eliminating the common factors, Equations (3.17) and (3.18) can be further simplified into

$$\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} e^{-\zeta\omega_n(t_N - t_{\mathrm{i}})} \sin(\omega_d t_{\mathrm{i}}) = 0, \tag{3.19}$$

$$\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} e^{-\zeta\omega_n(t_N - t_{\mathrm{i}})} \cos(\omega_d t_{\mathrm{i}}) = 0. \tag{3.20}$$

Solving the above two equations, the impulse amplitudes and time locations that would lead to zero residual vibrations can be obtained. To avoid the trivial solution of all zero-valued impulses and to obtain a normalized and bounded solutions for A_{i}, an additional one of the following two constraints on the impulse amplitudes have to be imposed [30]:

$$\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} = 1, \tag{3.21}$$

$$A_{\mathrm{i}} > 0 \quad , \qquad \mathrm{i} = 1, \ldots, N. \tag{3.22}$$

For the case of two-impulse input, shown in Figure 3.6, there are four unknowns to be found, the two impulse amplitudes (A_1, A_2) and the two impulse time locations (t_1, t_2). To minimize the time delay introduced by the shaping process, the first impulse is normally placed at time zero, i.e. $t_1 = 0$. This reduces Equations (3.19) and (3.20) to

$$A_2 \sin(\omega_d t_2) = 0, \tag{3.23}$$

$$A_1 e^{-\zeta\omega_n t_2} + A_2 \cos(\omega_d t_2) = 0. \tag{3.24}$$

Solving the above equations together with the constraints in Equations (3.21) and (3.22) results in the following impulse magnitudes and time locations:

$$A_1 = \frac{1}{1+Q}, \qquad A_2 = \frac{Q}{1+Q}, \tag{3.25}$$

$$t_1 = 0, \qquad t_2 = \frac{\pi}{\omega_d}, \tag{3.26}$$

with $\quad Q = e^{-\zeta\omega_n \frac{\pi}{\omega_d}}$.

In order to make the input shaper approach suitable for real applications, the properties of the impulse sequence given in Equations (3.25) and (3.26) have to be transferable to any real input command. Based on the linear system theory, convolving any sequence with zero vibration property with any other command will also result in a zero vibration command. This can be further interpreted in terms of filter theory. The sequence of impulses acts like a filter which conditions any input command so that the filtered command will result in a zero vibration when applied to the system, see Figure 3.7. Such a filter is called Zero Vibration (ZV) input shaper.

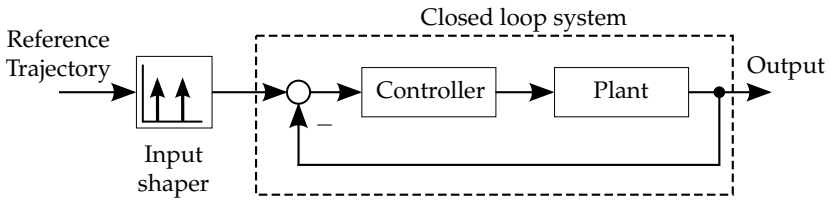


Fig. 3.7: Input shaper as a filtering technique

Figure 3.8 shows a second order underdamped system response to a jerk limited step command with and without input shaping. The system has a natural frequency of $F_n = 25\,\mathrm{Hz}$ and a damping ratio of $\zeta = 0.1$. The time delay effects of the ZV shaper are clearly seen between the shaped and unshaped reference commands. The residual vibrations existing in the unshaped response of the system are perfectly damped after the ZV shaper.

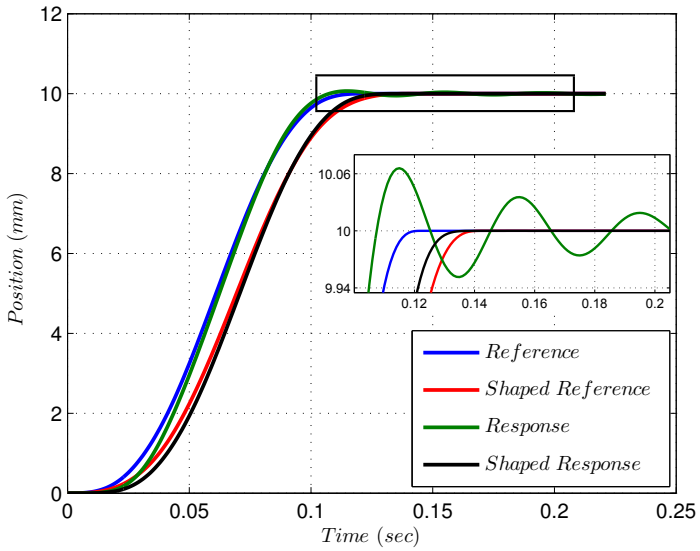


Fig. 3.8: response of a second order underdamped system to a jerk limited step input with and without ZV shaping

The effectivity of the ZV shapers depends on how accurate the system parameters (ω_n, ζ) are estimated. If there are errors in estimating these values, which is always the case in real applications, then the impulse sequence will not result in a zero vibration. In fact, for the two-impulse case, a small modeling error can result in a lot of vibrations. This lack of robustness is managed by adding extra constraints to the solution of the impulse sequence amplitudes and time locations. To overcome the lack of robustness in the system frequency ω_n, the first derivative of Equations (3.19) and (3.20) with respect to the system's natural frequency ω_n as

an additional constraints [29] are included.

$$\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} t_{\mathrm{i}} e^{-\zeta\omega_n(t_N - t_{\mathrm{i}})} \sin\left(\omega_d t_{\mathrm{i}}\right) = 0, \tag{3.27}$$

$$\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} t_{\mathrm{i}} e^{-\zeta\omega_n(t_N - t_{\mathrm{i}})} \cos\left(\omega_d t_{\mathrm{i}}\right) = 0. \tag{3.28}$$

In the same way, robustness conditions with respect to the system damping ratio ζ can be achieved by including the first derivatives of Equations (3.19) and (3.20) with respect to the system damping ratio ζ as additional constraints. Singer has proved in [29] that achieving high robustness with respect to system natural frequency and damping ratio can be done by the same constraints in Equations (3.27) and (3.28).
To compensate for the additional two equations that insure the robustness, two more unknowns must be added to the system by increasing the input from two to three impulses. An input shaper designed in this way is called Zero Vibration Derivative (ZVD) input shaper. Figure 3.9 shows the sensitivity curves, i.e. the percentage of residual vibration as a function of the system natural frequency variation, for both ZV and ZVD shapers. It is noticeable that the ZVD shaper has a wider sensitivity curve around the true system natural frequency $(\omega_a/\omega_n = 1)$ compared to that of the ZV shaper. This corresponds to higher robustness to errors in modeling the system natural frequency.

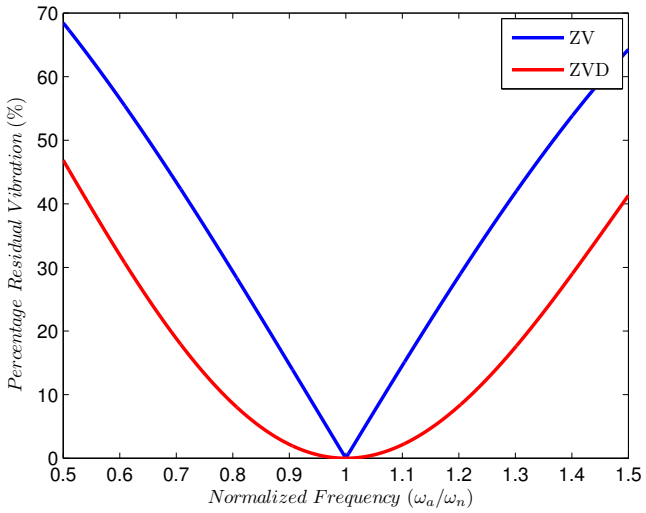


Fig. 3.9: Sensitivity curves of input shapers to system frequency

Since the development of the ZVD shaper, several other robust shapers have been introduced, such as Zero Vibration Derivative Derivative (ZVDD), Extra Insensitive (EI) sparers and many others. In fact, shapers can now be designed to have any degree of robustness to modeling errors in system parameters [30]. Table 3.1 summarizes the impulse amplitudes A_{i} and their appliance times T_{i} for the ZV, ZVD and ZVDD shapers.

Shaper	A_1	T_1	A_2	T_2	A_3	T_3	A_4	T_4
ZV	$\frac{1}{1+Q}$	0	$\frac{Q}{1+Q}$	$\frac{\pi}{\omega_d}$	–	–	–	–
ZVD	$\frac{1}{1+2Q+Q^2}$	0	$\frac{2Q}{1+2Q+Q^2}$	$\frac{\pi}{\omega_d}$	$\frac{Q^2}{1+2Q+Q^2}$	$\frac{2\pi}{\omega_d}$	–	–
ZVDD	$\frac{1}{1+3Q+3Q^2+Q^3}$	0	$\frac{3Q}{1+3Q+3Q^2+Q^3}$	$\frac{\pi}{\omega_d}$	$\frac{3Q^2}{1+3Q+3Q^2+Q^3}$	$\frac{2\pi}{\omega_d}$	$\frac{Q^3}{1+3Q+3Q^2+Q^3}$	$\frac{3\pi}{\omega_d}$

Table 3.1: Summary of different input shapers parameters

The robustness of the ZVD shapers comes with the cost of longer shapers, which, in turn, corresponds to longer time delays in the shaped commands. The period length of the ZV shaper, for example, is one half of the system's oscillation period (π/ω_d), whereas it is a complete oscillation period $(2\pi/\omega_d)$ in the case of ZVD.
Because of their simplicity and ease of implementation, input shapers have been applied to many real-world applications with great success. They proved noticeable capabilities in suppressing residual vibrations in point-to-point motion applications. However, little has been done in the field of contour following applications where several axes are moving simultaneously and time coordination between the axes' movements is important. The time delay introduced by the input shapers alters the given contours and degrades the contour following process. Figure 3.10 shows an example of input shaping effects on 2D contours. The example shows a square contour in the x-y plane. The zoom view illustrates the time delay effects of a ZV input shaper used to shape the x and y trajectories. The lack of time coordination between both axes results in rounding the contour corners, for example at $(x, y) = (10, 10)$.

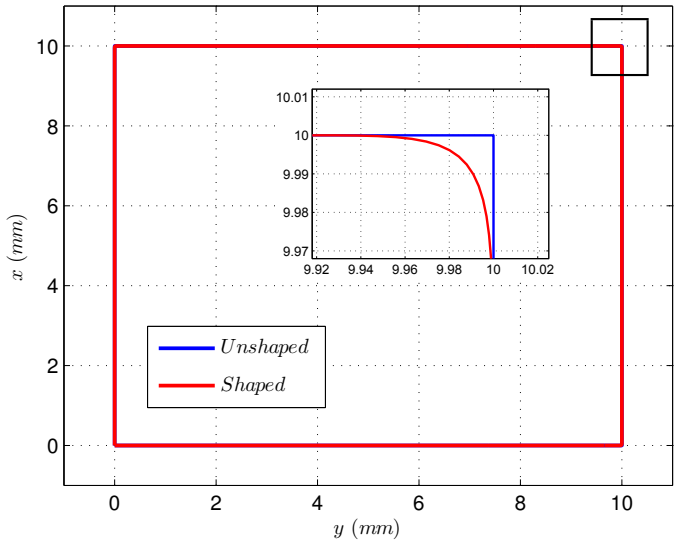


Fig. 3.10: Input shaping effects on 2D square contour

Singhose [31–34] and many other publications have addressed the effects of input shaping on

contour following control. Special solutions have been found for rather simple cases such as the circular and square contours. For such cases the original contours have been made bigger or smaller in order to compensate the shaper's time delay effects. However, a general criterion for compensating the effects of input shaping in free-form contour following applications is not available. Since CNC machine tools are often used for free-form contours machining applications, shaping filters are not a practical solution for the problem of this thesis.

3.3.2 Form Functions

Another approach to tackle the problem of residual vibrations in CNC machine tools within the motion profiles is through form functions. In this method, the elimination of residual vibrations is done via proper design for the jerk profile. Investigations have proved that the maximum jerk value (per axis) can limit the oscillatory behavior of CNC machine tool axes [35]. By altering the jerk limit and/or shape, it is possible to act on the smoothness degree of the motion profiles and to minimize the residual vibrations. On the other hand, altering the jerk profile can also dramatically increase the theoretical movement time. Thus, a compromise must be reached to attain the short movement time while attenuating the residual vibrations. We can distinguish between two main branches of the existing solutions for such a compromise.

The **first** branch fixes the shape of the jerk profile to a rectangular profile as in Figure 2.5 and modifies its maximum value $\hat{j}$. To estimate a good value for the maximum axis jerk, Groß has proposed in [15] an approximation formula which depends on the response maximum allowable overshoot and the system mechanical properties:

$$\hat{j} = \frac{4\pi^2}{0.15} \cdot e^{\frac{\pi\zeta_{\min}}{\sqrt{1-\zeta_{\min}^2}}-0.316} \cdot F_{d\min}^3 \cdot \Delta s_m, \tag{3.29}$$

where

$\zeta_{\min}$ is the system minimum damping ratio,

$F_{d\min}$ is the system minimum damped frequency,

Δs_m is the maximum allowable overshoot.

The derivation of the above formula is based on many heuristics and special conditions, its detailed discussion is provided in [15].

Figure 3.11 shows the response of a second order underdamped system to a jerk limited step input. The system has a natural frequency of $F_n = 25\,\mathrm{Hz}$ and a damping ratio of $\zeta = 0.1$. According to Equation (3.29) the maximum allowable jerk value, which satisfies a maximum overshoot of $\Delta s_m = 1\,\mu\mathrm{m}$, is $\hat{j} = 4.1\,\mathrm{m/s^3}$. As shown in the figure, the calculated jerk value is not exceeding the promised maximum overshoot.

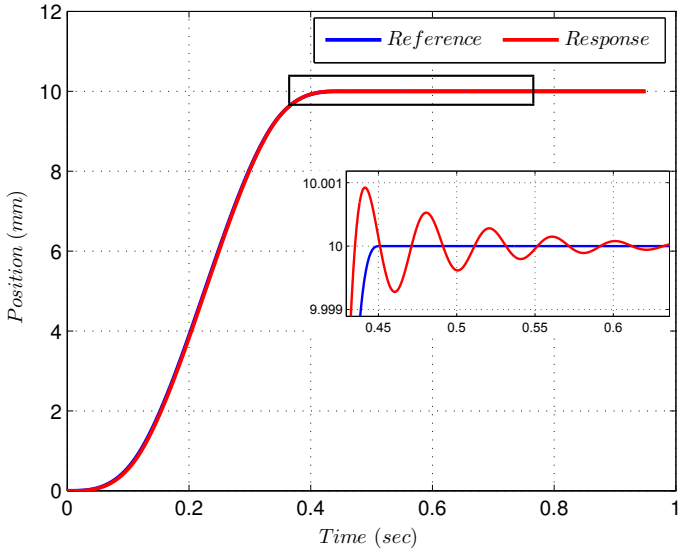


Fig. 3.11: response of a second order underdamped system to a jerk limited step input with optimized jerk limit

The method guarantees the promised maximal overshoot as long as the required assumptions are fulfilled, which unfortunately, are hard to meet in reality. Moreover, the resulting jerk values from Equation (3.29) are very conservative. Since the jerk limit is defined for the complete machining process, this will lead to severe degradation in the machining speed and hence to low productivity. Thus, in practical applications it is often the case that the maximal axis jerk values are tuned on the site based on personal experience using measurements for the oscillation amplitude resulting from simple step response tests as evaluation criteria.

The **second** branch of the form function methods focuses on the type of functions used to build the jerk profile rather than its maximum value. Different functions for building jerk profiles with minimum residual vibrations can be found in literature. Other than the conventional rectangular function, harmonic functions $(\sin$ and $\cos)$ and in particular the sine-squared function $\sin^2$, cycloidal functions and functions built based on Fourier series expansion are just some examples of the existing forms of jerk profiles [36, 37]. The basic principle of this method is the generation of jerk profiles which can provide filtering action at the system's critical frequency. Considering the classical jerk limited step function with rectangular jerk profile shown in Figure 3.12, it can be understood how jerk profiles can provide filtering actions.

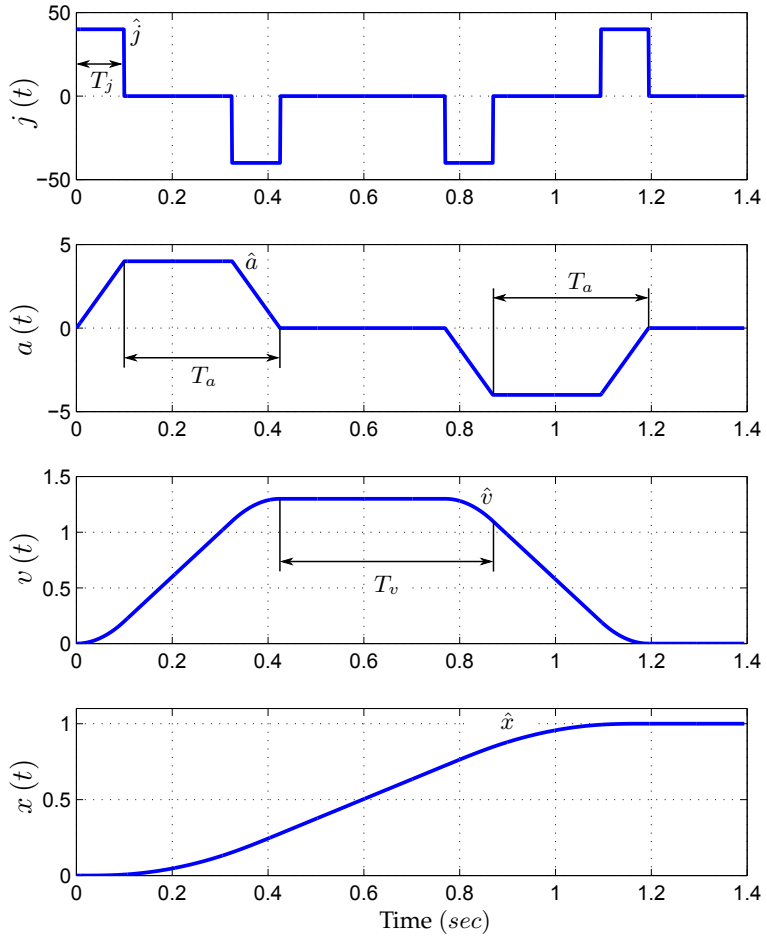


Fig. 3.12: Classical jerk limited step function

$\hat{x}$ is the step hight, $\hat{v}$, $\hat{a}$ and $\hat{j}$ are the maximum velocity acceleration and jerk values respectively. T_v, T_a and T_j are the velocity, acceleration and jerk time constants, which are calculated according to the respective limits. The Fourier transform of such a profile is given by

$$X(f) = \hat{j}T_jT_a(T_a+T_v)\left(\frac{\operatorname{sinc}(\pi T_j f)\cdot\operatorname{sinc}(\pi T_a f)\cdot\operatorname{sinc}(\pi(T_a+T_v)f)}{i2\pi f}e^{-i\pi f(2T_a+T_v+T_j)} + \frac{1}{2}\delta(f)\right), \tag{3.30}$$

with

$\operatorname{sinc}(x) = \frac{\sin(x)}{x}$,

$\delta(\cdot)$ is the impulse function,

f is the frequency variable in Hz.

According to the final value theorem, Equation (3.30) leads to a step hight $\hat{x}$ as

$$\hat{x} = \hat{j}T_jT_a(T_a+T_v). \tag{3.31}$$

Since the Fourier transform of a heaviside unit step function is given by [38]

$$\mathcal{F}\left\{\mathcal{H}\left(t\right)\right\} = \left(\frac{1}{i2\pi f} + \frac{1}{2}\delta\left(f\right)\right), \tag{3.32}$$

Equation (3.30) can be rewritten as

$$X\left(f\right) = \mathcal{F}\left\{\hat{x}\mathcal{H}\left(t\right)\right\}\left(\operatorname{sinc}\left(\pi T_j f\right) \cdot \operatorname{sinc}\left(\pi T_a f\right) \cdot \operatorname{sinc}\left(\pi\left(T_a + T_v\right) f\right)\right) e^{-i\pi f\left(2T_a + T_v + T_j\right)}. \tag{3.33}$$

The above equation implies that the jerk limited step function is simply a heaviside step function with amplitude $\hat{x}$ convolved with a characteristic function. Knowing that the Fourier transform of a rectangular function with a width T and a height $1/T$ is given by

$$\mathcal{F}\left\{g\left(t\right) = \left\{\begin{array}{ll} \frac{1}{T}, & 0 \leq t \leq T \\ 0, & t > T \end{array}\right\}\right\} = \operatorname{sinc}\left(\pi T f\right) e^{-i\pi f T}, \tag{3.34}$$

the characteristic function mentioned above represents the frequency function of three successive rectangular filters with widths T_j, T_a and $T_v + T_a$. Thus, the jerk limited step function generation process can be thought of as a filtering process with three successive filters each of which represent one of the motion limits (velocity, acceleration and jerk). Figure 3.13 shows a schematic representation of a step trajectory generation process using the filter approach.

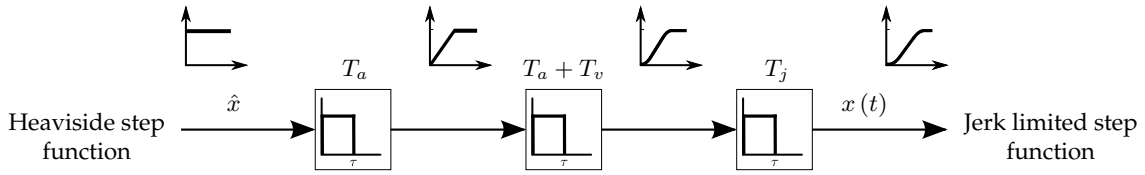


Fig. 3.13: Jerk limited step function using filter concept

To calculate the classical rectangular jerk profile shown in Figure 3.12, the three filters time constants (T_j, T_a and $T_a + T_v$) are defined according to the maximum jerk $\hat{j}$, acceleration $\hat{a}$ and velocity $\hat{v}$ values and the distance $\hat{x}$ to be traveled. The result is the classical Soft motion profile which maintains the selected limits but does not consider the machine residual vibrations.
The form function method uses this filter methodology of the trajectory generation process to compensate for the machine residual vibrations. From the linear system theory, we recall that if at least one filter from the T_a, $T_v + T_a$ and T_j filters has a filtering action at the machine critical frequencies, the generated trajectory will have zero residual vibrations. Since the velocity and acceleration limits have their physical meaning and effects on the machine and the jerk limit is nothing more than the rate of change in the acceleration signal, the method considers only the jerk limit filter T_j to force the filtering action. In this sense, instead of building the jerk limit filter T_j based on the jerk limit value, the method selects the filter to have a zero at the machine critical frequency.

From the different types of functions that can be used to build such a filter, the sine squared

function is considered as an example to build the jerk profile in the following. The function is defined in the time domain by

$$h(t) = \begin{cases} \sin^2\left(\frac{\pi t}{T}\right), & 0 \leq t \leq T, \\ 0, & t > T, \end{cases} \tag{3.35}$$

where T is the period of the sine squared function. The Fourier transform of this function is defined by

$$H(f) = \frac{1}{i4\pi f\left(1 - T^2 f^2\right)}\left[1 - e^{i2\pi T f}\right]. \tag{3.36}$$

Figure 3.14 shows the sine squared function in time and frequency domains.

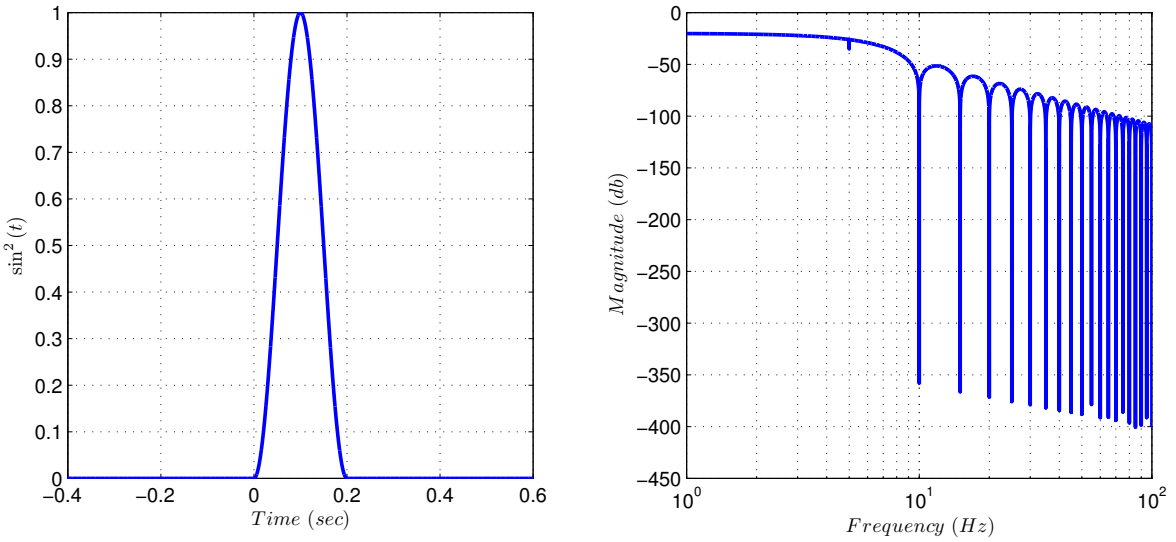


Fig. 3.14: Sine squared function in time (left) and frequency (right) domains

To act as a filter for the machine critical frequency F_c, the Fourier transform function of the sine squared function must have a zero at that frequency. This is achieved by setting Equation (3.36) to zero at the machine critical frequency and solve for the sine square period T. Mathematically, this corresponds to

$$H(F_c) = \frac{1}{i4\pi F_c\left(1 - T^2 F_c^2\right)}\left[1 - e^{i2\pi T F_c}\right] = 0, \tag{3.37}$$

resulting in a sine square period of

$$T = \frac{2}{F_c}. \tag{3.38}$$

As an example, a step input is designed with a sine squared jerk profile, the step height was set to $\hat{x} = 10\,\mathrm{mm}$. The resulting jerk profile is shown in Figure 3.15, the main building block in such a profile is the sine squared function. The width of each jerk interval is defined according to Equation (3.38) and the height is designed to fulfill the required step height, maximum velocity and acceleration values.

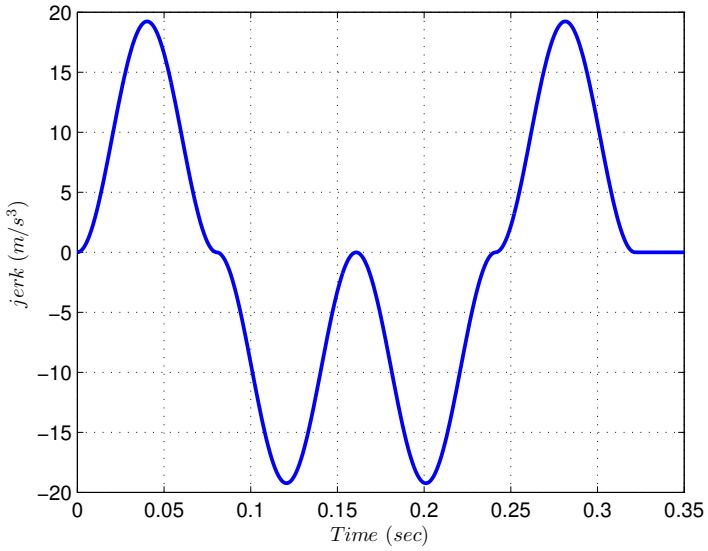


Fig. 3.15: Sine squared jerk profile for a step input

A second order underdamped system response to the step input is shown in Figure 3.16. The system has a natural frequency of $F_n = 25\,\mathrm{Hz}$ and a damping ratio of $\zeta = 0.1$. The zoomed view shows the ability of the sine squared profile in damping the system residual vibrations very clearly. Compared to the classical jerk profile with comparable maximum jerk, the sine squared jerk profile reduced the system residual vibrations from $15\,\mu\mathrm{m}$ to $0.2\,\mu\mathrm{m}$.

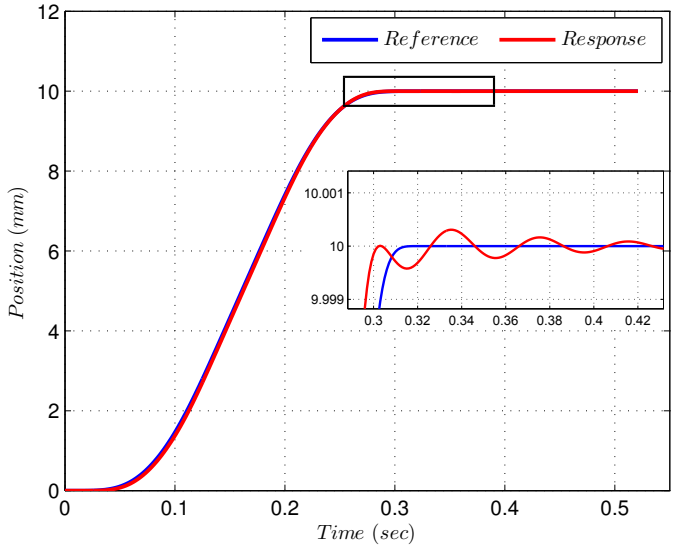


Fig. 3.16: Response of a second order underdamped system to a step input with sine squared jerk profile

Although the form function method can noticeably damp the residual vibrations, a major drawback of such a method is that it works only for single-axis machines where both the path and axis trajectories are exactly the same. For the multi-axis case, the form function method still can be used to design path trajectories with zero residual vibrations property. However, to generate the axis trajectories, the path trajectories have to be distributed along the machine

axes as discussed in the interpolation step in Section 2.1. This additional step of interpolation introduces additional frequencies from the contour geometry which are not considered by the form function. Thus, the use of form functions in the path trajectories will not guarantee the elimination of residual vibrations on the axis level. Since this thesis targets multi-axis CNC machine tools, this major drawback of the form function method also makes it not useful solving the problem of this thesis.

3.4 Summary

In this chapter a quick summary for the most important state of the art residual vibrations removal methods was given. The existing solutions were categorized into three main categories according to their place of act on the machine tool axis control structure.

1. Passive damping techniques: They involve methods acting on the system's mechanical structure. This type of solution is considered at early stages of the machine design process. The machine vibrations are attenuated by increasing the mechanical rigidity of the machine axes. Their main drawback is that they increase the machine inertia and thus lower the machine dynamic behavior and limits the machine maximum attainable acceleration. In addition, they result in a tremendous increase in the machine construction costs and do not always lead to satisfactory results.

2. Controller tuning techniques: They involve methods acting on the control structure and parameters used to control the machine axes. They compensate the machine's elasticities by increasing the overall damping of the closed control loops through an optimal tuning of the controller parameters. The main drawback of such methods is that they drastically limit the bandwidth of the machine axis control loops. This, in turn, decreases the machine speed of response and thus deteriorates the overall machine productivity.

3. Motion profile techniques: They involve methods acting directly on the reference motion profiles fed to the machine tool. With these methods, the mechanical system is designed for highly dynamical behavior and the control parameters are tuned to achieve a maximal speed of response. The machine flexibilities are compensated by modifying the reference motion profiles. Based on their type of action, the motion profile methods are classified into two main classes. The first include methods depending on filter action for compensating the machine flexibilities, known as shaping filters. The second one uses special functions in the trajectory generation step to compensate the machine flexibilities, known as form functions. Both classes show outstanding results in attenuating the residual vibrations for single-axis machine tools. However, their performance in the multi-axis case was adverse. The contour degradation resulting from the time delays introduced by the filter class and the inability of the form functions to compensate for the additional frequencies resulting from the contour geometry make such methods practically not useful for multi-axis machines with free form contours.

Yet another area to be addressed in evaluating the performance of such methods is their locality in attenuating the residual vibrations. Regardless of the method being used, attenuating residual vibrations entails slower machining, thus "loss" in the time optimality of the machining process. Since time optimality implies higher productivity and lower costs, it is convenient to address the effects of any attenuation technique on the time optimality of the machining process. Unfortunately, all techniques discussed above employ global measures in handling the residual vibrations. That is, they compensate the machine elasticities by measures that permanently affect the machine behavior as the case in the passive damping and controller tuning techniques, or measures that influence the entire motion profile regardless if they produce vibrations or not as the case in the filter and form functions techniques. Such global measures highly increase the machining time and thus deteriorate the machine productivity. On the other hand, local measures which apply modifications only in places where they are needed are unfortunately not available yet. In this thesis, a local attenuation technique is developed in order to maintain maximum possible machining quality and time optimality.

4
Error Signals

The machining quality of a CNC machine tool is determined by several factors such as the dynamic response of the machine axes, the performance of the servo control loops used to regulate the machine axes, the shape and structure of the contours to be machined, etc. In contour following applications, the performance of CNC machine tools is judged according to two main performance indices: the *tracking error* and the *contour error*.
A primary requirement for any vibration suppression technique is a measure that quantifies the oscillations in the machine response. Existing measures are often limited to performance measures either for the techniques in use or measures for the residual vibrations on the axis level, i.e. for single-axis machines. A comprehensive survey for the available key residual vibration measures is found in [39]. However, little has been done to define a general vibration measure that can characterize the machine vibrations on the contour level for a multi-axis machine tool in free-form machining applications.
In this chapter, a comprehensive discussion for CNC machine tools performance measures is given. Different methods for approximating the contour error in free-form machining applications are presented and compared and a new general vibration measure for multi-axis machine tools with free-form machining applications is developed.

4.1 Tracking Error

The tracking error $e_t(t)$ is the classical performance measure for control systems. It is defined as the instantaneous difference between the actual system output and the desired reference input. In the context of CNC machining, this corresponds to the instantaneous difference between the actual positions of the machine axes and the reference positions. Using the terminology of the machine axis control structure in Figure 2.13, the tracking error $e_t(t)$ can be written as

$$e_t(t) = \theta_L(t) - \theta_{ref}(t). \tag{4.1}$$

The overall tracking error of a multi-axis CNC machine tool is the vector sum of the tracking errors resulting from each individual axis. It represents the instantaneous deviation of the machine Tool Center Point (TCP) from the desired position. For a Cartesian three axis machine,

the TCP tracking error is defined as

$$\vec{e}_t\,(t) = \vec{e}_{t,x} + \vec{e}_{t,y} + \vec{e}_{t,z}, \tag{4.2}$$

where $\vec{e}_{t,x}, \vec{e}_{t,y}$ and $\vec{e}_{t,z}$ are the tracking errors in x, y and z directions, respectively. Figure 4.1 shows an example of a tracking error signal for a single-axis CNC machine tool with control structure given by Figure 2.13 and a jerk limited step command as a reference input.

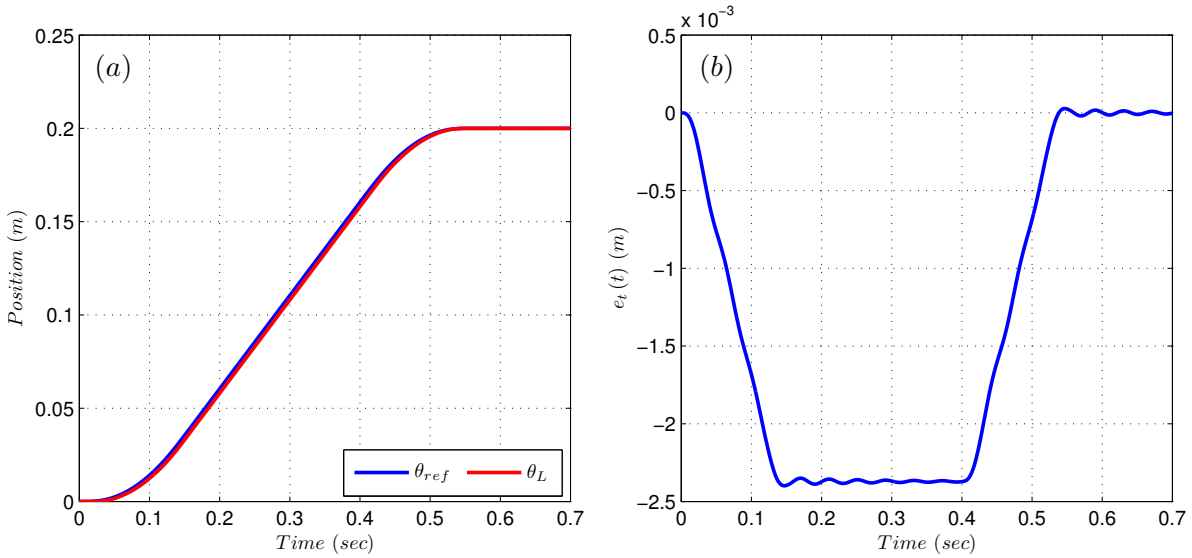


Fig. 4.1: (a) Machine axis reference and actual positions, (b) Tracking error signal

In addition to the vibratory behavior of the axis, the tracking error signal contains static deviations from the input command as a result of the different time delays in the servo control loops. In general two types of errors can be distinguished in any tracking error signal [35]:

1. Aperiodical terms representing errors related to the tracking characteristics of the system, noted as $e_{t,ap}\,(t)$,

2. Oscillatory periodical terms $e_{t,vib}\,(t)$ related to the vibratory behavior of the system,

$$e_t\,(t) = e_{t,ap}\,(t) + e_{t,vib}\,(t)\,. \tag{4.3}$$

In this thesis, only the vibratory behavior of the system is of interest, i.e. the oscillating terms of the tracking error signal. Extracting these terms from the tracking error signal is not straightforward. To better describe the oscillating behavior of the machine axis, a new error signal named *elasticity error* is proposed. The next section discusses in detail the construction and properties of such an error signal.

4.2 Elasticity Error

As discussed in previous chapters, the dynamic behavior of an elastic system consists of two types of motion modes: a rigid-body mode and a flexible mode. The elasticity error signal $e_e\,(t)$

describes the relative deviation between these two modes. For a single-axis CNC machine tool with a control structure given by Figure 2.13, the motor side $\theta_M(t)$ represents the rigid-body behavior of the axis while the load side $\theta_L(t)$ represents its flexible behavior and the elasticity error is defined as

$$e_e(t) = \theta_L(t) - \theta_M(t). \tag{4.4}$$

In the literature the elasticity error has been considered in the input shaper design field as move-vibration error. It first appeared in the work of Pao and Singhose [40], where standard input shaper designs are compared in terms of their move-vibrations. Recent work by Dhanda et al. [41] proposed a measure for the overall move-vibration defined as the square magnitude deviation of the flexible structure position from the position of the rigid body. However, the move-vibration measure was always considered in terms of integrals, i.e. as an energy measure. No consideration was given for the instantaneous vibration of the flexible mode. In this thesis, the elasticity error signal is conditioned such that it gives an instantaneous measure for the machine axis vibratory behavior.

Figure 4.2 illustrates the elasticity error signal for a single-axis CNC machine tool with control structure given by Figure 2.13 and a jerk limited step command as a reference input. It is straightforward to see that the elasticity error signal represents the axis vibrations far better than the tracking error signal shown in Figure 4.1. For instance, it is easy to identify the four oscillation regions in the elasticity error signal whereas some of these regions were completely hidden in the tracking error signal. However, the elasticity error signal still contains some aperiodical terms mixed with the oscillatory periodical terms. Thus, to describe only the vibratory behavior of the machine, the elasticity error signal must be modified.

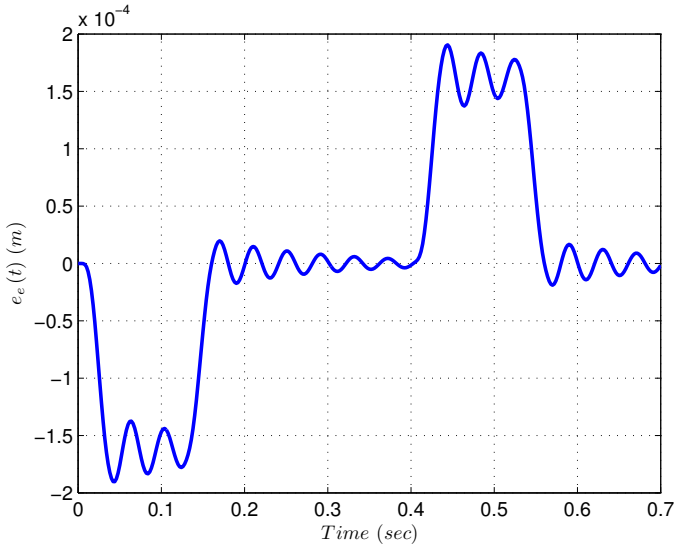


Fig. 4.2: Elasticity error signal for jerk limited step input

By analyzing the elasticity error signal, a relationship between the reference command acceleration $\ddot{\theta}_{ref}(t)$ and the aperiodical terms of the elasticity error signal can be found. Figure 4.3 shows a normalized version of the elasticity error signal $\bar{e}_e(t)$ together with the normalized

acceleration reference command $\ddot{\bar{\theta}}_{ref}(t)$.

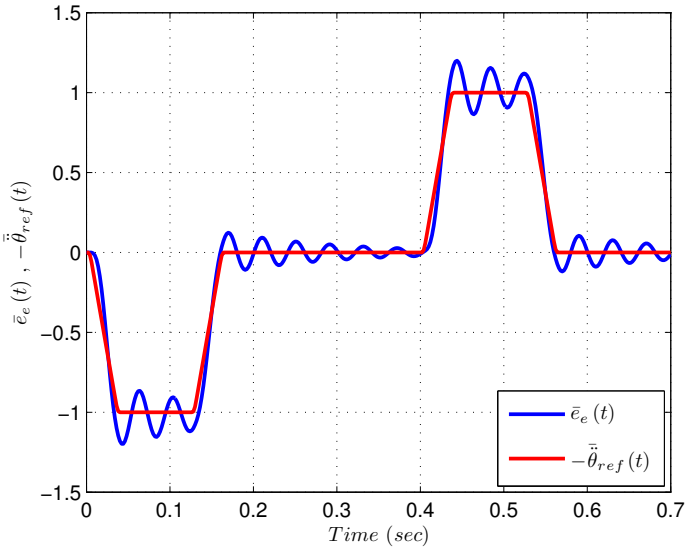


Fig. 4.3: Normalized elasticity error signal (blue) and reference command acceleration signal (red)

For a single-axis CNC machine tool with control structure given by Figure 2.13, the relationship between the system's rigid-body mode and flexible mode was given in Equation (2.14) as:

$$\frac{\Theta_L(s)}{\Theta_M(s)} = \frac{Ds+K}{J_L s^2 + Ds + K}.$$

Together with the elasticity error definition in Equation (4.4), the above equation leads to a transfer function describing the relationship between the elasticity error signal and the reference command as

$$\frac{E_e(s)}{\Theta_{ref}(s)} = \frac{\Theta_M(s)}{\Theta_{ref}(s)} \frac{-J_L s^2}{J_L s^2 + Ds + K}, \tag{4.5}$$

the term $\Theta_M(s)/\Theta_{ref}(s)$ represents the rigid-body behavior of the position closed loop transfer function of the system. With velocity and acceleration feed forward controllers and a good tuning of the machine axis control loops, this term is approximately unity in the respective frequency range, i.e. $\Theta_M(s)/\Theta_{ref}(s) \approx 1$. This implies that the rigid-body mode of the machine axis behaves approximately in the same manner as the reference signals, i.e. $\theta_{ref}(t) \approx \theta_M(t)$. With this observation, the elasticity error transfer function is reduced to

$$\frac{E_e(s)}{\Theta_{ref}(s)} \approx \frac{E_e(s)}{\Theta_m(s)} = \frac{-J_L s^2}{J_L s^2 + Ds + K}. \tag{4.6}$$

Rewriting the above equation, it is straightforward to see the relationship between the elasticity error signal and the acceleration reference command.

$$\frac{E_e(s)}{\ddot{\Theta}_{ref}(s)} \approx \frac{E_e(s)}{\ddot{\Theta}_M(s)} = \frac{-J_L}{J_L s^2 + Ds + K}. \tag{4.7}$$

Figure 4.4 shows the elasticity error signal $e_e(t)$ calculated using the original definition given by Equation (4.4) and the one calculated according to the approximation in Equation (4.7), $\tilde{e}_e(t)$. In the figure, it can be recognized that a small phase shift between both signals exists. Such a time lag results from the different control elements in the control loops of the machine axis. However, the signal amplitudes are almost identical and the differences are hardly seen.

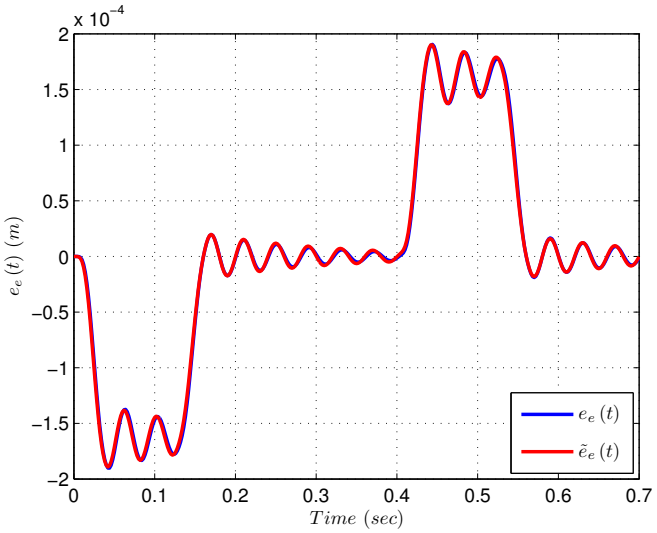


Fig. 4.4: Exact (blue) and approximated (red) elasticity error signal

The approximation transfer function above represents the characteristics of a single-mass-spring-damper system. The poles of such a system are the same as those of the machine axis mechanics transfer function, Equation (2.14), which agrees with the fact that the mechanics of the machine axis represents its elastic behavior.
The aperiodical terms in the elasticity error signal result from the nonzero DC gain of the transfer function, which is calculated as

$$\frac{E_e(0)}{\ddot{\Theta}_{ref}(0)} = \frac{-J_L}{K}. \tag{4.8}$$

Thus, forcing the elasticity error transfer function to have a zero DC gain will eliminate the aperiodical terms from the elasticity error signal. In the Laplace domain, this leads to a vibrational error transfer function

$$\frac{E_{e,vib}(s)}{\ddot{\Theta}_{ref}(s)} = \frac{-J_L}{J_L s^2 + Ds + K} - \frac{-J_L}{K} = \frac{J_L^2 s^2 + J_L D s}{K\left(J_L s^2 + Ds + K\right)}. \tag{4.9}$$

In the time domain, this corresponds to scaling the reference command acceleration by the DC gain factor and subtracting it from the elasticity error as

$$e_{e,vib}(t) = \theta_L(t) - \theta_M(t) - \frac{-J_L}{K} \cdot \ddot{\theta}_{ref}(t). \tag{4.10}$$

Figure 4.5 shows the elasticity error signal together with the vibrational part extracted accord-

ing to Equation (4.10). In the figure, the aperiodical terms of the elasticity error signal are completely eliminated and the signal captures the oscillatory periodical terms perfectly. The vibratory elasticity error signal in Figure 4.5 shows some discontinuities mainly at the transition points from one oscillating region to another. These discontinuities are expected since we are extracting the oscillating part of the elasticity error signal by subtracting a C^0 continuous signal, i.e. the acceleration signal, from the elasticity error signal which is a C^∞ continuous.

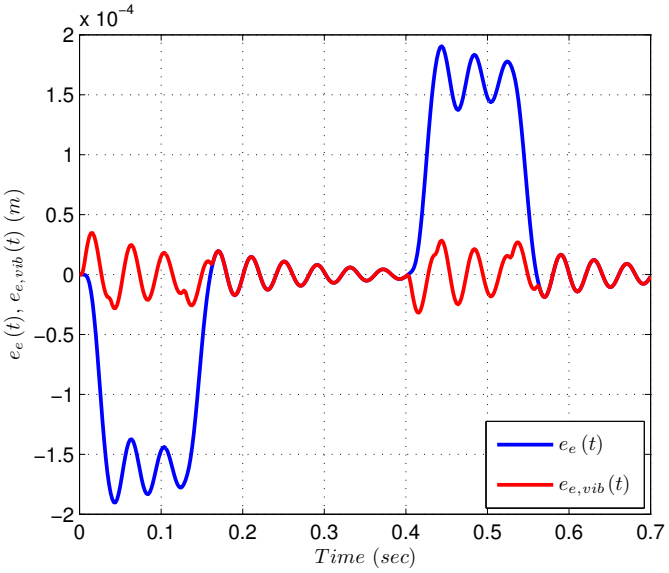


Fig. 4.5: Elasticity error signal (blue) and vibratory elasticity error signal (red)

4.3 Contour Error

The tracking and elasticity errors discussed above describe the machine error on the axis level, i.e. they measure the error resulting from each axis independently. Such error signals do not represent the actual deviation of the machine TCP from the desired contour. This deviation is normally represented by the contour error $e_c(t)$. For continuous machining contours, the contour error is defined as the shortest distance between the desired and actual contours at a given time instant [42]. This definition is schematically illustrated in Figure 4.6.

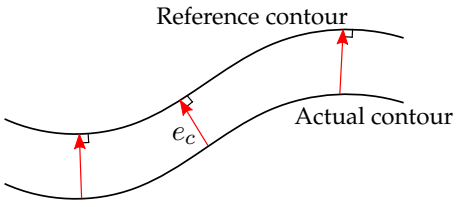


Fig. 4.6: Schematic illustration of contour error

Although industrial CNC machine tools are normally controlled according to the independent axis control concept, minimizing the contour error is the main target in any high-speed high-precision machining process. The classical approach to minimize the contour error is by

eliminating or reducing the tracking error of each individual axis which indirectly reduces the contour error. However, tracking errors cannot give a precise measure for the existing contour error and it is quite possible that even when tracking errors are present in the individual axes, they happen to cancel each other out such that no contour error exists any more. Figure 4.7 shows an example of a two-axis machine tool contour following case with P_{tcp} as the machine TCP actual position and P_{ref} as the corresponding position at the reference contour. Although there exist tracking errors in the machine axes, the machine TCP has perfectly followed the reference contour with no or very small contour error. This emphasizes that tracking errors do not represent the actual deviation of the machine TCP from the desired contour.

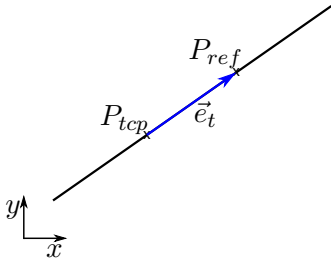


Fig. 4.7: $x - y$ linear contour with zero contour error and nonzero tracking error

Computing the contour error along simple contours, such as linear or circular contours, is simple and straightforward. In these cases, the contour error can be expressed in a closed form based on the geometrical relations in these contours [43]. On the other hand, the contour error for free-form contours cannot be expressed in closed form. Instead, approximation methods are often used to estimate the contour error. In the literature there exist several methods for approximating the contour error for the free-form contour case, the following subsections discuss the working principle of the most commonly used methods.

4.3.1 Linear Reference Path Method

Motivated by the contour error calculation method for linear contours, Yeh [44] has developed a contour error approximation method for free-form contours. The method is based on approximating the contouring error by a vector from the actual machine TCP position to the nearest point on the line that passes through the reference position tangentially. Figure 4.8 illustrates Yeh's method in estimating the contour error for a 2D motion case.

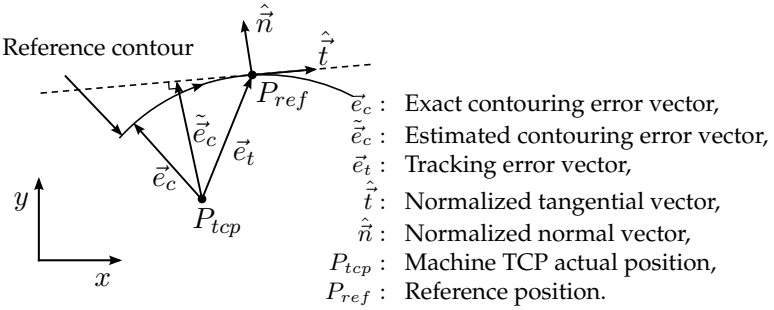


Fig. 4.8: Contour error estimation for 2D motion using linear path method [44]

Yeh's definition for the contouring error estimate is equivalent to the projection of the tracking error vector $\vec{e}_t$ onto the normalized normal vector $\hat{\vec{n}}$ at the reference position P_{ref}. Mathematically this is written as:

$$\hat{\vec{e}}_c = \left\langle \vec{e}_t, \hat{\vec{n}} \right\rangle \cdot \hat{\vec{n}}. \tag{4.11}$$

Since the tracking error vector is easy to compute, $\vec{e}_t = P_{ref} - P_{ac}$, the problem of estimating the contouring error is transformed into finding the normalized normal vector $\hat{\vec{n}}$. One way to find such a vector is by computing the normalized tangential vector $\hat{\vec{t}}$ and then computing its normal. Since the reference velocity profile is always known in the numeric control unit, the normalized tangential vector can be computed from the reference velocity vector at the reference position as:

$$\hat{\vec{t}} = \frac{\vec{v}_{ref}}{\|\vec{v}_{ref}\|}, \tag{4.12}$$

where $\vec{v}_{ref} \in \mathbb{R}^n$ is the reference velocity vector at the reference position $P_{ref} \in \mathbb{R}^n$. Given the normalized tangential vector, the normalized normal vector for the above 2D case is computed as:

$$\hat{\vec{n}} = \mathbf{R}\,(90^\circ) \cdot \hat{\vec{t}}, \tag{4.13}$$

where $\mathbf{R}\,(\cdot)$ is an 2×2 rotation matrix. The extension of Yeh's approximation into the 3D case is straightforward. The tracking error is now projected onto a plane orthogonal to the contour and expanded by the normalized normal vector $\hat{\vec{n}}$ and the normalized binormal vector $\hat{\vec{b}}$.

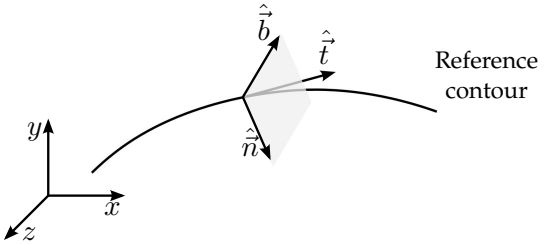


Fig. 4.9: 3D Reference contour with a plane of normal vectors

The contour error estimate in the 3D case is then defined as the vector sum of the tracking error projections onto both the normalized normal vector $\hat{\vec{n}}$ and the normalized binormal vector $\hat{\vec{b}}$.

$$\hat{\vec{e}}_c = \left\langle \vec{e}_t, \hat{\vec{n}} \right\rangle \cdot \hat{\vec{n}} + \left\langle \vec{e}_t, \hat{\vec{b}} \right\rangle \cdot \hat{\vec{b}}. \tag{4.14}$$

By writing the tracking error in terms of the tangential, normal and binormal coordinate system,

$$\vec{e}_t = \left\langle \vec{e}_t, \hat{\vec{t}} \right\rangle \cdot \hat{\vec{t}} + \left\langle \vec{e}_t, \hat{\vec{n}} \right\rangle \cdot \hat{\vec{n}} + \left\langle \vec{e}_t, \hat{\vec{b}} \right\rangle \cdot \hat{\vec{b}}, \tag{4.15}$$

the contour error estimate can be rewritten as

$$\hat{\vec{e}}_c = \vec{e}_t - \left\langle \vec{e}_t, \hat{\vec{t}} \right\rangle \cdot \hat{\vec{t}}. \tag{4.16}$$

The linear path method is easy and straightforward to implement, but it delivers good estimates only in cases where the tracking error is relatively small.

4.3.2 Average Velocity Method

An enhancement to the linear reference path method was introduced by Chuang in [45]. The method follows the same concept as the linear reference path method with the only difference that it calculates the normalized tangential vector. Instead of using the reference velocity alone to calculate the normalized tangential vector $\hat{\vec{t}}$, Chuang used a new normalized average tangential vector $\hat{\vec{t}}'$ which was defined based on the direction of the average velocity resulting from the reference and actual tool velocities. Figure 4.10 illustrates the concept of the average velocity method for the 2D case.

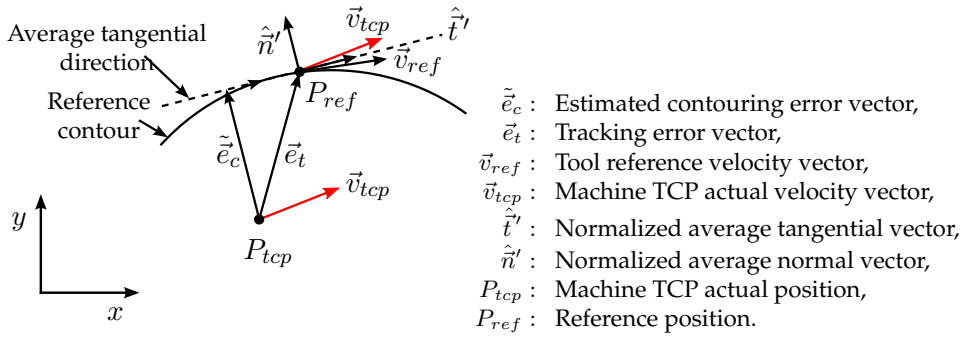


Fig. 4.10: Contour error estimation for 2D motion using average velocity method

If the machine TCP actual and reference velocity vectors are denoted by $\vec{v}_{tcp}$ and $\vec{v}_{ref}$ respectively, the normalized average tangential vector is given by

$$\hat{\vec{t}}' = \frac{\vec{v}_{tcp} + \vec{v}_{ref}}{\|\vec{v}_{tcp} + \vec{v}_{ref}\|}, \tag{4.17}$$

and the normalized average normal vector is defined as:

$$\hat{\vec{n}}' = \mathbf{R}\,(90^\circ) \cdot \hat{\vec{t}}', \tag{4.18}$$

Using the definition of the contour error estimate in the linear path method in Equation (4.11), and the normalized average normal vector of Equation (4.17), an estimate for the contour error can be written as:

$$\tilde{\vec{e}}_c = \left\langle \vec{e}_t, \hat{\vec{n}}' \right\rangle \cdot \hat{\vec{n}}'. \tag{4.19}$$

The extension of the average velocity method from the 2D case into the 3D case follows the same concept as the one discussed in the linear path method.
Using the average velocity to determine the normalized tangential vector **can** enhance the estimation quality of the contour error depending on the dynamic behavior of the machine axes. However, this requires measurements of the actual tool velocity. Knowing that most of the CNC machine tools have no feedback signals from their TCP, the actual tool velocity

requirement in this method hinders its use in practical applications.

4.3.3 Circular Reference Path Method

Inspired by the simplicity of calculating the contour error for circular paths and the fact that all curved contours can be approximated by circular contours with different centers and radii, Koren [46] has developed his approximation method to estimate the contour error for free-form contours. The method is based on approximating the free-form contours with multiple circles each with radius and center equal to the radius and center of curvature at the approximation position. Figure 4.11 illustrates the concept for the 2D case.

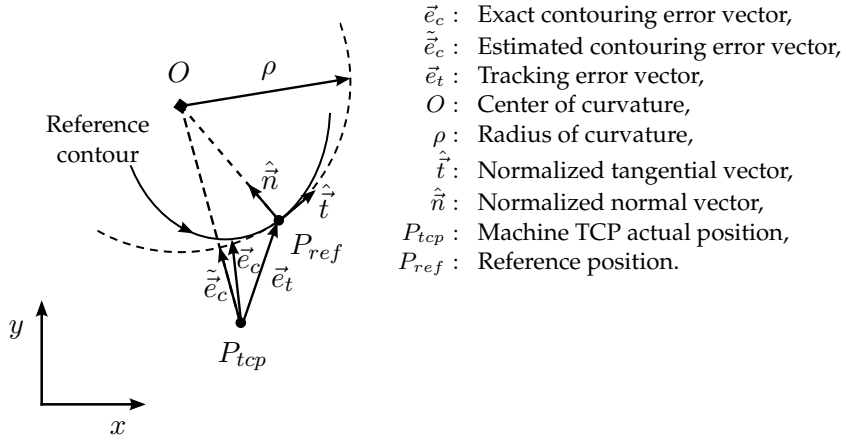


Fig. 4.11: Contour error estimation for 2D motion using circular path method [46]

According to the terminology of Figure 4.11, the center of the approximation circle, i.e. center of curvature O, is given by

$$O = P_{ref} - \rho \cdot \hat{\vec{n}}, \tag{4.20}$$

and with the radius and the center of the approximation circle in hand, Koren's approximate for the contour error is defined as

$$\tilde{\vec{e}}_c = \left(\left\| \overrightarrow{OP_{tcp}} \right\| - \rho \right) \cdot \frac{\overrightarrow{OP_{tcp}}}{\left\| \overrightarrow{OP_{tcp}} \right\|}, \tag{4.21}$$

where $\overrightarrow{OP_{tcp}}$ is the vector pointing toward the shortest distance direction and connecting the machine TCP actual position P_{tcp} and the center of the approximation circle O:

$$\overrightarrow{OP_{tcp}} = P_{tcp} - O. \tag{4.22}$$

Koren's method is simple and easy to implement for contours living in the 2D plane. Unfortunately, its extension to the 3D case is relatively complex and does not always lead to good results. The circle approximation concept in the 2D case is extended to a sphere approximation in the 3D case. The radius and center of each such sphere equals to the radius and center of curvature at the approximation positions. The contour error is then approximated by the

distance between the actual position of the machine TCP and the surface of the approximation sphere. Yet another pitfall of such a method is that it is limited to reference contours with only curved paths. For a free-form contours with linear segments, the method fails to estimate the contour error since the radius of curvature for such segments is infinity, i.e. $\rho = \infty$. Moreover, calculating the radius and center of curvature is not an easy task especially when the contours are given as discrete data points.

4.3.4 Closest Path Segment Method

Another contour error estimation method was defined by Erkorkmaz [47]. He developed a method called the closest reference path segment method to estimate the contour error in free-form paths described by a set of discrete points. Erkorkmaz's idea depends on reconstructing the reference contour using linear segments to connect each two successive points in the data set. Among the contour points Erkorkmaz identified the nearest point to the actual position of the machine TCP and defined two closest reference path segments that contain this point. The contour error is then estimated by the smallest distance between the actual position of the machine TCP and one of the two defined segments. In cases where the contour error cannot be orthogonal to the reference path, i.e. there is no perpendicular line from the actual position of the machine TCP to either of the two segments, Erkorkmaz defined the contour error to be the distance between the nearest reference point and the actual position of the machine TCP. Figure 4.12 illustrates the idea for the 2D motion case. P_{tcp} is the actual position of the machine TCP, $P_{ref,\mathrm{i}}$ is the nearest reference point, $\vec{e}_t$ is the tracking error between the nearest reference point and the actual position of the machine TCP, $P_{ref,\mathrm{i}-1}$ and $P_{ref,\mathrm{i}+1}$ are, respectively, the points before and after the nearest reference point and $\tilde{\vec{e}}_c$ is the estimated contour error.

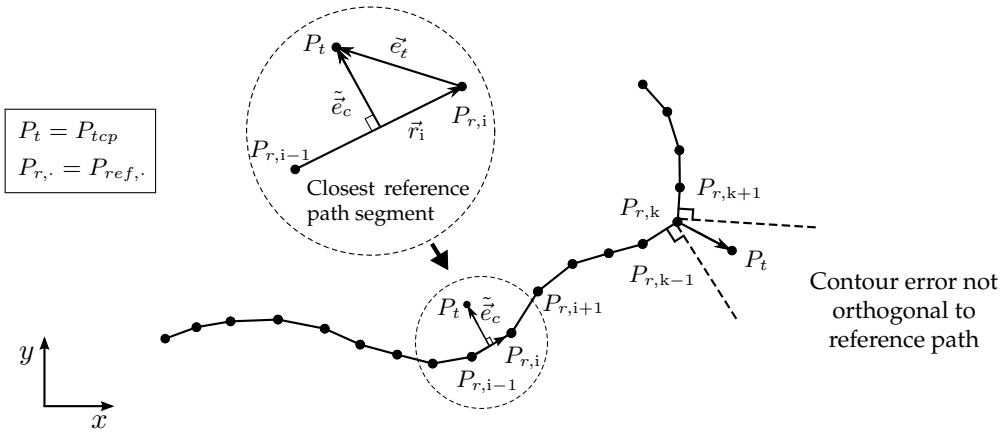


Fig. 4.12: Contour error estimation for 2D motion using closest path segment method [47]

With the nearest reference point $P_{ref,\mathrm{i}}$ identified, the two closest path segments to the actual position of the TCP are define as $\overline{P_{ref,\mathrm{i}-1}P_{ref,\mathrm{i}}}$ and $\overline{P_{ref,\mathrm{i}}P_{ref,\mathrm{i}+1}}$. Based on these segments the region around the current position of TCP is divided into three regions according to the orientation of the segments with respect to each other, Figure 4.13a illustrates these regions.

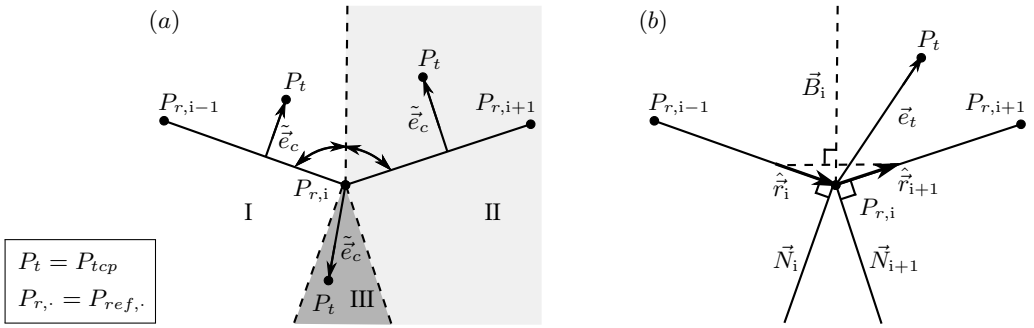


Fig. 4.13: (a) Regions division for contour error estimation, (b) Normals that define the different treatment regions [47]

The boundaries of the three regions are determined by the normals to the closest path segments, $\vec{N}_\mathrm{i}$ and $\vec{N}_{\mathrm{i}+1}$, and the bisector $\vec{B}_\mathrm{i}$ of the angle $\angle P_{ref,\mathrm{i}-1} P_{ref,\mathrm{i}} P_{ref,\mathrm{i}+1}$, Figure 4.13b. Mathematically, the boundaries of the three regions are defined using inner product relationships. Considering Figure 4.13b, two path vectors $\vec{r}_\mathrm{i}$ and $\vec{r}_{\mathrm{i}+1}$ are defined using the reference points $P_{ref,\mathrm{i}-1}$, $P_{ref,\mathrm{i}}$ and $P_{ref,\mathrm{i}+1}$ as:

$$\vec{r}_\mathrm{i} = P_{ref,\mathrm{i}} - P_{ref,\mathrm{i}-1}, \tag{4.23}$$

$$\vec{r}_{\mathrm{i}+1} = P_{ref,\mathrm{i}+1} - P_{ref,\mathrm{i}}. \tag{4.24}$$

Based on the two path vectors, the bisector $\vec{B}_\mathrm{i}$ of the angle $\angle P_{ref,\mathrm{i}-1} P_{ref,\mathrm{i}} P_{ref,\mathrm{i}+1}$ is defined as the normal to the vector $\hat{\vec{r}}_\mathrm{i} + \hat{\vec{r}}_{\mathrm{i}+1}$, where $\hat{\vec{r}}_\mathrm{i}$ and $\hat{\vec{r}}_{\mathrm{i}+1}$ are the normalized path vectors:

$$\left\langle \vec{B}_\mathrm{i}, \hat{\vec{r}}_\mathrm{i} + \hat{\vec{r}}_{\mathrm{i}+1} \right\rangle = 0. \tag{4.25}$$

The normal vectors $\vec{N}_\mathrm{i}$ and $\vec{N}_{\mathrm{i}+1}$ are defined in terms of the two normalized path vectors as:

$$\left\langle \vec{N}_\mathrm{i}, \hat{\vec{r}}_\mathrm{i} \right\rangle = 0, \tag{4.26}$$

$$\left\langle \vec{N}_{\mathrm{i}+1}, \hat{\vec{r}}_{\mathrm{i}+1} \right\rangle = 0. \tag{4.27}$$

Since for any two vectors $\vec{x}$ and $\vec{y}$ the following holds:

1. $\langle \vec{x}, \vec{y} \rangle = 0$ implies that $\vec{x} \perp \vec{y}$,
2. $\langle \vec{x}, \vec{y} \rangle > 0$ define the region that is off the normal in the direction of $\vec{y}$,
3. $\langle \vec{x}, \vec{y} \rangle < 0$ define the region that is off the normal in the direction of $-\vec{y}$,

the three regions in Figure 4.13a can be defined in terms of the inner product of the tracking

error vector $\vec{e}_t$ and the two normalized path vectors $\hat{\vec{r}}_i$ and $\hat{\vec{r}}_{i+1}$ as follows:

$$\left.\begin{array}{rlll} \text{Region I}: & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i}} + \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle < 0 & \wedge & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i}} \right\rangle \leq 0 \\ \text{Region II}: & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i}} + \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \geq 0 & \wedge & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \geq 0 \\ \text{Region III}: & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i}} \right\rangle > 0 & \wedge & \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle < 0 \end{array}\right\}. \tag{4.28}$$

Depending on which region the machine TCP actual position lies in, the contour error estimate $\tilde{\vec{e}}_c$ is defined. For instance, if the machine TCP actual position P_{tcp} falls into region I or II, the contour error estimate $\tilde{\vec{e}}_c$ is defined as the perpendicular distance between the machine TCP actual position and the corresponding closest segment $\overline{P_{ref,\mathrm{i-1}}P_{ref,\mathrm{i}}}$ or $\overline{P_{ref,\mathrm{i}}P_{ref,\mathrm{i+1}}}$. On the other hand, if the machine TCP actual position P_{tcp} falls into region III, where the contour error can not be orthogonal to either of the two segments, the distance between the machine TCP actual position and the nearest reference point $\overline{P_{ref,\mathrm{i}}P_{tcp}}$ is taken as the contour error. Mathematically this can be written in term of the inner product between the tracking error vector $\vec{e}_t = P_{ref,\mathrm{i}} - P_{tcp}$ and the two normalized path vectors $\hat{\vec{r}}_{\mathrm{i}}$ and $\hat{\vec{r}}_{\mathrm{i+1}}$ as:

$$\left.\begin{array}{ll} \text{Region I}: & \tilde{\vec{e}}_c = \vec{e}_t - \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i}} \right\rangle \cdot \hat{\vec{r}}_{\mathrm{i}} \\ \text{Region II}: & \tilde{\vec{e}}_c = \vec{e}_t - \left\langle \vec{e}_t, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \cdot \hat{\vec{r}}_{\mathrm{i+1}} \\ \text{Region III}: & \tilde{\vec{e}}_c = \vec{e}_t \end{array}\right\}. \tag{4.29}$$

Extending the closest path segment method into the 3D case requires no additional work since the method is derived based on the general vector notation. The method's performance, however, is highly dependent on finding the closest reference point to the actual position of the machine TCP and the search algorithm used to find such a point.
To find the closest reference point to the actual position of the machine TCP, an optimization problem is formulated as follows:
Given the reference position vector in discrete form as $\vec{P}_{ref}\left(kT_s\right)$ and the actual position of the machine TCP vector as $\vec{P}_{tcp}\left(kT_s\right)$, where $k = 0, 1, \ldots, n$ and T_s is the sampling time. For a given actual position of the machine TCP $\vec{P}_{tcp}\left(k_0T_s\right)$, the closest reference position is found via minimizing the following cost function

$$\min_{k\in\mathbb{N}} \left\| \vec{P}_{tcp}\left(k_0T_s\right) - \vec{P}_{ref}\left(kT_s\right) \right\|. \tag{4.30}$$

The above minimization problem can be viewed as a line search optimization process with search directions $p_k = \pm 1$ and a step length $\alpha_k \in \mathbb{N} : k_0 + \alpha_k p_k >= 0$. The cost function of Equation (4.30) can be then formulated as a line search problem as:

$$\min_{\alpha_k\in\mathbb{N}} \left\| \vec{P}_{tcp}\left(k_0T_s\right) - \vec{P}_{ref}\left(\left(k_0 + \alpha_k p_k\right)T_s\right) \right\|. \tag{4.31}$$

To solve the above search problem, a series of integer search steps in both search directions, $p_k = \pm 1$, was employed. The search process terminates at points where the cost function in

Equation (4.31) is no longer decreasing in the corresponding search direction. The nearest reference point is then chosen to be the minimizer of Equation (4.31) between the two terminating points. Usually such a method involves a prohibitively large number of cost function evaluations per search direction. Therefore, it is considered as an inefficient solution in the context of line search optimization processes. However, in this case, since the closest reference point is normally very close to the actual position of the machine TCP, the number of cost function evaluations is very small which makes it suitable for the problem at hand. Simulation results show that a maximum of 5 search steps are normally required to find the closest reference point.

Using the closest reference point in the calculation of the contour error estimate has significantly enhanced the estimation quality of the closest path segment method. Inspired with this fact, all the other approximation methods were extended to employ the closest reference point in their calculations. The simulation results showed a remarkable enhancement in the estimation quality of the various methods after using the closest reference point.

4.3.5 Evaluation and Comparison

In the previous subsections, four methods for approximating the contour error for free-form contours were presented. To compare their performance, the four methods are evaluated based on their approximation quality, computational cost and the needed information for calculating the contour error estimates. A test contour is designed for the evaluation process that contains the different contour elements which can be practiced is real applications, those like linear elements with single and multiple axis motion, circular elements as well as parabolic elements. Figure 4.14 shows the test contour in the x-y plane. For the testing purpose, a two-axis CNC machine is considered. Each axis is approximated as a second order underdamped system with natural frequency $F_n = 25\,\mathrm{Hz}$ and damping ratio $\zeta = 0.04$.

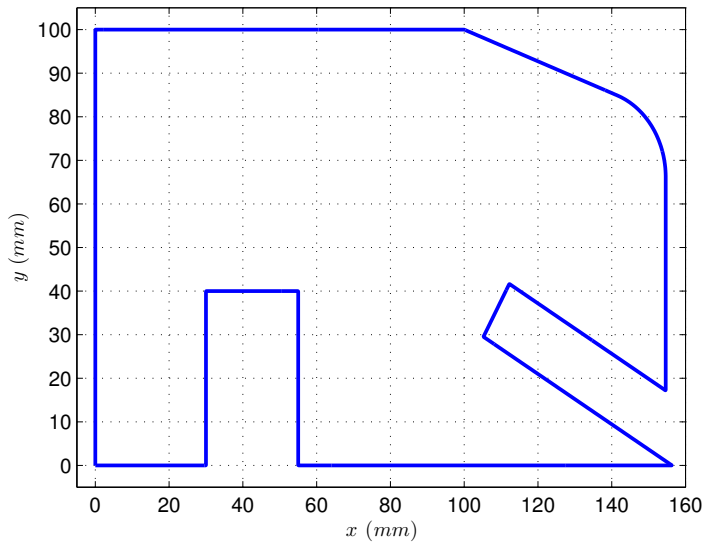


Fig. 4.14: Test contour for evaluating the contour error approximation methods

As reference values for the evaluation process, a quite accurate, but computationally very intensive method for calculating the contour error is developed. The method is based on approximating the discrete positions of the reference contour using splines. The spline equations are then used to calculate the "exact" contour error at each of the machine positions. The computation follows the contour error definition as the shortest distance between the actual machine position and the corresponding reference spline. Figure 4.15 depicts the contour error $x-$ and $y-$ components calculated using the splines method for the test contour in Figure 4.14.

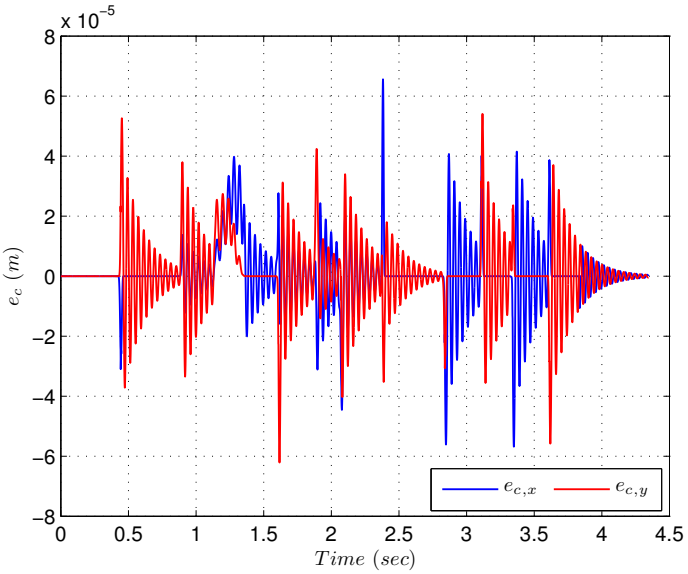


Fig. 4.15: Exact contour error using the spline method

To compare the estimation quality of the different approximation methods, the difference in the magnitude between the contour error calculated according to the spline method $\vec{e}_c$ and the contour error estimates calculated by the approximation methods $\tilde{\vec{e}}_c$ is used as an evaluation measure:

$$\Delta \left\| \vec{e}_c \right\| = \left\| \vec{e}_c \right\| - \left\| \tilde{\vec{e}}_c \right\|. \tag{4.32}$$

Figure 4.16 shows the resulting evaluation measure for the four approximation methods using the test contour of Figure 4.14. The results clearly demonstrate the superiority of the closest path segment method over the other methods in approximating the contour error. The approximation quality of the linear path method and the circular path method are slightly lower than, but comparable to, the one obtained by the closest path segment. On the other hand, the estimation quality of the average velocity method is the worst among the approximation methods. This, in fact, is expected since using a highly oscillating signal such as the actual machine velocity in the estimation process will degrade its performance.

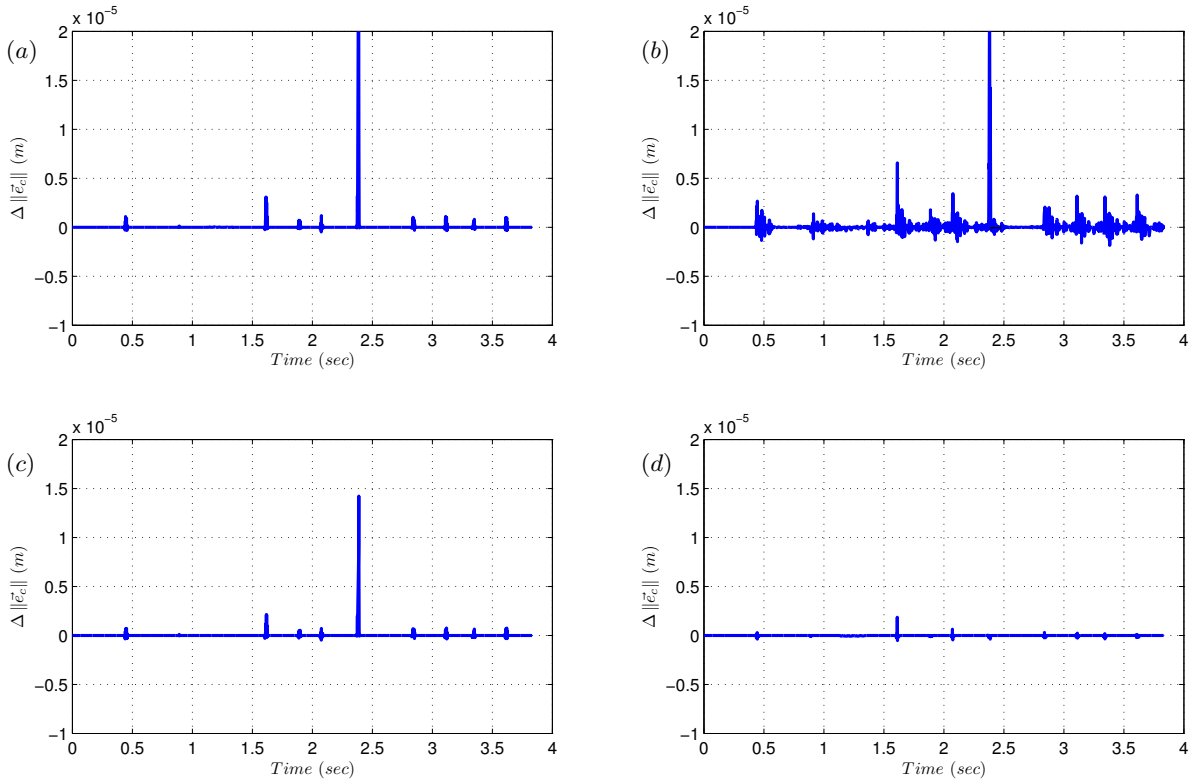


Fig. 4.16: Contour error magnitude difference for: (a) Linear method, (b) Average velocity method, (c) Circular method, (d) Closest path segment method

In comparing the different approximation methods, it is not only the approximation quality that is crucial but also the computational time associated with each method. Again the shortest segment method proofs to outperform the other methods in terms of the computational time. Table 4.1 summarizes the computational times required by each of the approximation methods for computing the contour error of the test contour of Figure 4.14.

	Linear	Average velocity	Circular	Closest path segment
Time (s)	0.719	0.939	0.815	0.085

Table 4.1: Computation time required by the approximation methods

Yet another aspect of comparing the different approximation methods is the required information to calculate the contour error estimate. Such information includes the reference and actual positions, velocities and accelerations. The amount of information required by the method is highly connected to its computational time. The more information necessary to compute the contour error estimate, the higher is the computational time needed by the method. Table 4.2 summarizes the required signals by each of the four approximation methods. Among the four methods, the closest path segment method followed by the linear path method demand the lowest amount of information to estimate the contour error.

Method	$\vec{P}_{ref}$	$\vec{P}_{tcp}$	$\dot{\vec{P}}_{ref}$	$\dot{\vec{P}}_{tcp}$	$\ddot{\vec{P}}_{ref}$	$\ddot{\vec{P}}_{tcp}$
Linear	✓	✓	✓			
Average velocity	✓	✓	✓	✓		
Circular	✓	✓	✓		✓	
Closest path segment	✓	✓				

Table 4.2: Necessary signals to compute the contour error estimate

Considering the above discussions, the closest path segment method delivers the best contour error estimate for the free-form contours with the lowest amount of required information and computational time. Thus, for the rest of this thesis, the closest path segment method will be used for computing the contour error in free-form machining applications.
However, the contour error estimate calculated by this method is still inheriting some undesirable behaviors from the tracking error signal since it is based on projecting the tracking error onto some normal vector. For instance, the resulting contour error estimate contains oscillatory periodical terms as well as aperiodical terms especially in places where the contour has a nonzero curvature, for example the contour error in Figure 4.15 between $t = 1\,\mathrm{s}$ and $t = 1.5\,\mathrm{s}$. Since the main focus of this thesis is to attenuate the oscillatory periodical terms, an extension of the closest path segment method is developed in Section 4.4 in order to generate a general measure for the machine oscillations in the contour level for free-form contours.

4.4 Vibrational Contour Error

To attenuate the machine vibrations, a measure that quantifies the existing vibrations in the machine response is necessary. As mentioned before, the contour error signal contains two types of error: The first is the aperiodical error which results from the time delay of servo control loops regulating the machine axes. The second type is an oscillatory periodical error which results from the elastic behavior of the machine axes. In this work, only the second type of error, i.e. the oscillatory periodical error, is of interest. The reason is that the aperiodical errors can be neither avoided nor compensated. Unfortunately, extracting the oscillatory periodical terms from the contour error signal is not as straightforward as the tracking error signal. In the contour error case the signal is computed numerically where in the tracking error case the calculations are done in a closed form formulas and the two terms are easily identified and separated.

The computation of the contour error signal, according to the closest path segment method, is basically the projection of an error signal, in this case the tracking error signal, onto some normal vector, i.e. computing the contour error can be understood as a mapping operation of an

error signal from the axis level to the contour level. Thus, the characteristics of the computed contour error signal are strongly dependent on the error signal used in the calculation process. If this error signal is chosen such that it only represents the oscillating behavior of the machine axis, the resulting contour error will contain only the oscillatory periodical terms. Motivated by this observation, a method for calculating a contour error signal which reflects only the oscillatory behavior of the machine, called hereafter the *vibrational contour error* $e_{c,vib}$, is developed. The method extends the closest path segment approach by employing the vibratory part of the elasticity error signal $e_{e,vib}$, defined in Section 4.2, in calculating the contour error estimate instead of the tracking error signal. The basic principle of the closest path method remains the same, only the regions boundaries defined by Equation (4.28) are redefined as

$$\left.\begin{array}{rlcl} \text{Region I}: & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i}} + \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle < 0 & \wedge & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i}} \right\rangle \leq 0 \\ \text{Region II}: & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i}} + \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \geq 0 & \wedge & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \geq 0 \\ \text{Region III}: & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i}} \right\rangle > 0 & \wedge & \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle < 0 \end{array}\right\}, \tag{4.33}$$

and the contour error estimate calculation of Equation (4.29) is replaced by

$$\left.\begin{array}{rl} \text{Region I}: & \tilde{\vec{e}}_c = \vec{e}_{e,vib} - \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i}} \right\rangle \cdot \hat{\vec{r}}_{\mathrm{i}} \\ \text{Region II}: & \tilde{\vec{e}}_c = \vec{e}_{e,vib} - \left\langle \vec{e}_{e,vib}, \hat{\vec{r}}_{\mathrm{i+1}} \right\rangle \cdot \hat{\vec{r}}_{\mathrm{i+1}} \cdot \\ \text{Region III}: & \tilde{\vec{e}}_c = \vec{e}_{e,vib} \end{array}\right\} \tag{4.34}$$

Figures 4.17 and 4.18 illustrate, respectively, the $x-$ and $y-$ components of the complete contour error signal calculated using the closest path segment method and the vibrational contour error signal for the test contour of Figure 4.14. As the figures show, the method has perfectly extracted the oscillatory periodical terms from the contour error signal. Comparing the results in Figure 4.18 to those shown in Figure 4.17, especially in regions where the contour has nonzero curvature, e.g. between $t = 1\,\mathrm{s}$ and $t = 1.5\,\mathrm{s}$, the aperiodical terms of the contour error are completely removed and the pure system oscillations are represented in the vibrational contour error signal.

Since the vibrational contour error signal is computed based on the closest path segment method, this approach inherits all the advantages of the method. That is, it delivers the best vibrational contour error estimate for free-form contours, it requires the lowest amount of information and it has a relatively short computing time.

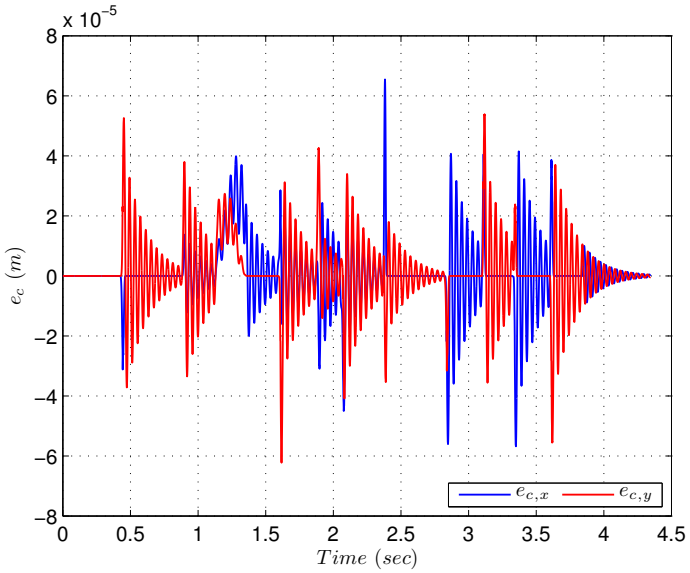


Fig. 4.17: Contour error signal using the closest path segment method

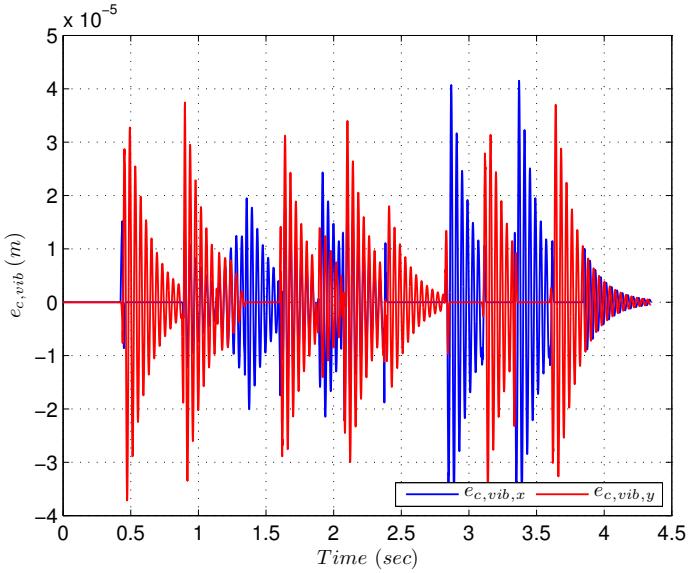


Fig. 4.18: Vibrational contour error signal

5

Time Scaling as a Vibration Removal Strategy

As explained in the preceding chapters, the existing vibration removal strategies suffer either from being tremendously conservative, leading to severe reduction in the machine speed, or from being uncoordinated, leading to significant deterioration in the machined contours and thus poor machining quality. In this chapter a new strategy is developed to attenuate the machine vibrations while maintaining its high-speed and high-precision characteristics. The method focuses on modifying the motion profile along the path trajectories, thus keeping the different motion profiles of the individual axes coupled rather than modifying them separately. The modification of the path trajectories will not only suppress the machine vibrations but will also guarantee a perfect contour machining.

To modify the path trajectories, the method uses a time-scaling concept which has been employed in a variety of fields. In motion planning, for example, it has been used to modify the time distribution along the reference paths in order to keep some machine restrictions or to overcome obstacles. The motivation to introduce time-scaling is often to control some system parameters and/or to gain useful system properties which evolve according to the scaled time. In literature, the time-scaling concept is used to solve different problems: Sampei & Furuta [48], for example, addressed the problem of system linearization by feedback through time-scaling, Kardoss and Kiss [49] employed time-scaling to eliminate the reference tracking error in closed loops by modifying the time of the reference path. In robotics, Hollerbach [50] used time-scaling to determine optimal trajectories for robot manipulators with torque restrictions.

In this chapter, the application range of time-scaling is extended to modifying the frequency content of CNC machine tool trajectories in order to overcome the undesirable vibrations in the machine response. The proposed method starts from time optimal motion profiles and modifies them such that the vibrations are reduced below a certain prescribed limit while increasing the machining time only by as little as possible. Figure 5.1 shows a block diagram of this method. In step 1, time optimal jerk limited trajectories are generated by the NC unit of the machine tool as described in Subsection 2.1.1. In step 2, the machine response to these trajectories is simulated and the vibrational contour error signal is computed as discussed in Chapter 4. In order to maintain the time optimality of the original trajectories, only designated

regions in the trajectories will be targeted, namely those *critical regions* where the frequency content of the reference trajectories matches one of the resonance frequencies of the machine axes and the oscillations appear to exceed a given limit. To identify these regions, the contour error signal is processed in step 3 by a wavelet-based time-frequency approach. This step decomposes the complete trajectory into several local regions for which the time-scaling has to be employed separately and for each such region, the method customizes a specially designed scaling function template by tuning its parameters via an optimization process with the goal of eliminating the machine vibrations in that specific region.

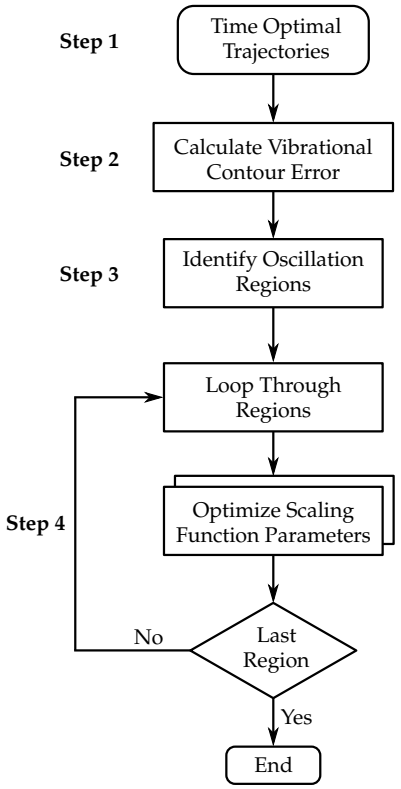


Fig. 5.1: Block diagram of vibration removal strategy

The following sections discuss in detail each step in the vibration removal strategy shown in Figure 5.1 as well as the theory behind these steps.

5.1 Time Frequency Analysis

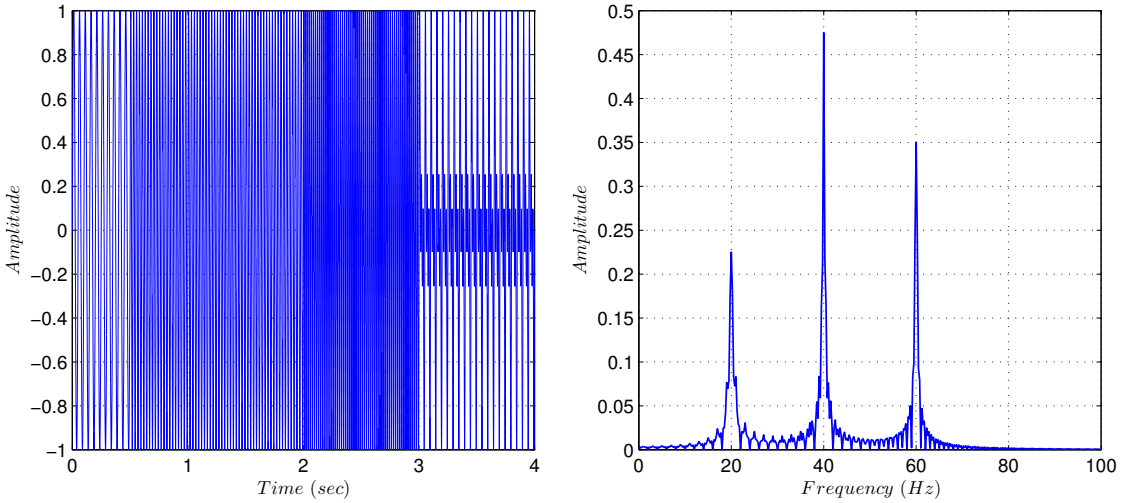
The well–known Fourier Transform (FT) decomposes a signal into its frequency components. To fix notations, for a finite energy signal $x(t) \in L^2(\mathbb{R})$ the Fourier transform is given by

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-i2\pi ft}\, \mathrm{d}t, \tag{5.1}$$

where $X(f)$ is the spectrum of $x(t)$, cf. [51]. The inverse Fourier transform is given by

$$x(t) = \int_{-\infty}^{\infty} X(f)\, e^{i2\pi ft}\, \mathrm{d}f. \tag{5.2}$$

The existence of the Fourier transform and its inverse formula is subject to various conditions, a detailed discussion of such conditions can be found in [38, 51]. The Fourier transform spectrum provides information about the kind of frequencies existing in the signal as well as their amplitudes, but it lacks any information about their duration or their time locations within the signal. As an example, Figure 5.2.a shows a signal constructed from three sine waves with frequencies $20, 40$ and $60\,\mathrm{Hz}$, each with different duration and time of occurrence. The Fourier spectrum of such a signal is shown in Figure 5.2.b.



(a) Time signal

(b) Fourier transform spectrum

Fig. 5.2: Fourier transform example

From the Fourier spectrum, the three frequencies that contributed to the construction of the signal can be easily identified as well as their amount of contribution. However, no information about their duration or their time location within the signal can be extracted.
To obtain both time and frequency information from a signal, a time–frequency analysis must be performed. The most common time–frequency transforms are the Windowed Fourier Transform (WFT), also known as Short Time Fourier Transform (STFT), and the Continuous Wavelet Transform (CWT). In general, the time–frequency analysis decomposes a signal over a dictionary of elementary waveforms, called the *time–frequency atoms* $\phi_{b,f}(t)$, which are indexed by a time parameter b and a frequency parameter f. Any time–frequency transform can be generally written as an integral transform by means of these atoms as [52]

$$T_\phi x(b, f) = \int_{\mathbb{R}} x(t)\, \phi_{b,f}(t) \mathrm{d}t. \tag{5.3}$$

The ability of any time–frequency transform to localize the information in time and frequency

depends mainly on the spread of its time-frequency atoms in the time–frequency plane. Unfortunately, a perfect localization in both time and frequency is not possible, the time–frequency spread of the atoms is always bounded from below by the Heisenberg uncertainty principle.

5.1.1 Heisenberg Uncertainty

For a time–frequency atom $\phi_{b,f}(t)$ with unit norm $\|\phi_{b,f}(t)\| = 1$, the time localization μ_t and its spread σ_t are defined by

$$\mu_t := \int_{\mathbb{R}} t\,|\phi_{b,f}(t)|^2 \mathrm{d}t \quad \text{and} \quad \sigma_t^2 := \int_{\mathbb{R}} |t-\mu_t|^2\,|\phi_{b,f}(t)|^2 \mathrm{d}t, \tag{5.4}$$

cf. [53]. Similarly, the frequency localization μ_f and spread σ_f of the atom are defined by

$$\mu_f := \int_{\mathbb{R}} f\,|\Phi_{b,f}(f)|^2 \mathrm{d}f \quad \text{and} \quad \sigma_f^2 := \int_{\mathbb{R}} |f-\mu_f|^2\,|\Phi_{b,f}(f)|^2 \mathrm{d}t, \tag{5.5}$$

where $\Phi_{b,f}$ is the Fourier transform of the atom $\phi_{b,f}$. The time–frequency localization of an atom $\phi_{b,f}$ is best visualized by the *Heisenberg boxes* which are rectangles living in the time–frequency plane, centered at (μ_t, μ_f) and have sizes as $\sigma_t \times \sigma_f$. Figure 5.3 illustrates an example of a Heisenberg box representing an atom $\phi_{b,f}$.

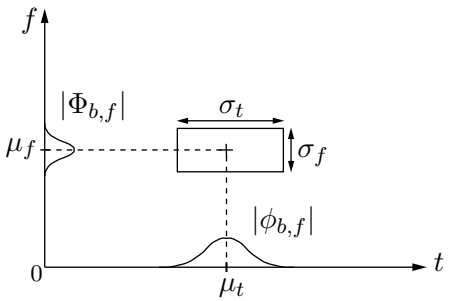


Fig. 5.3: Typical Heisenberg box of a time–frequency atom $\phi_{b,f}$

The Heisenberg uncertainty principle states that the Heisenberg boxes have a minimum area of

$$\sigma_t \sigma_f \geq \frac{1}{2}, \tag{5.6}$$

which limits the joint time–frequency resolution of the atoms and implies that a simultaneous localization in time and frequency is impossible. However, the different time–frequency transforms are distinguished according to their ability to handle such a trade–off between the time and the frequency localization.

5.1.2 Windowed Fourier Transform

The windowed Fourier transform is the oldest and most straightforward approach to pass from the Fourier transform to the time–frequency analysis [52]. In the WFT, instead of Fourier transforming the entire signal, the signal is partitioned into several segments by means of

windowing and the spectrum is computed for each segment independently. For a finite energy signal $x(t) \in L^2(\mathbb{R})$ the windowed Fourier transform is given by [53]

$$X(b,f) = \int_{-\infty}^{\infty} x(t)\, w(t-b)\, e^{-i2\pi ft}\, \mathrm{d}t, \tag{5.7}$$

where $w(t)$ is the time window function centered at $t = 0$ and of a unit norm $\|w(t)\| = 1$, b is the time translation factor and $X(b,f)$ is the windowed Fourier transform of $x(t)$. The time–frequency atoms of the WFT are the modulated window functions

$$\phi_{b,f}(t) = e^{-i2\pi ft} w(t-b). \tag{5.8}$$

The time and frequency spread of the WFT atoms are independent of b and f. This means that each atom $\phi_{b,f}$ corresponds to a Heisenberg box with fixed size $\sigma_t \times \sigma_f$ independent of its position in the time–frequency plane (μ_t, μ_f). Figure 5.4 shows an example of two WFT atoms centered at (μ_t, μ_f) and $\left(\mu_t', \mu_f'\right)$.

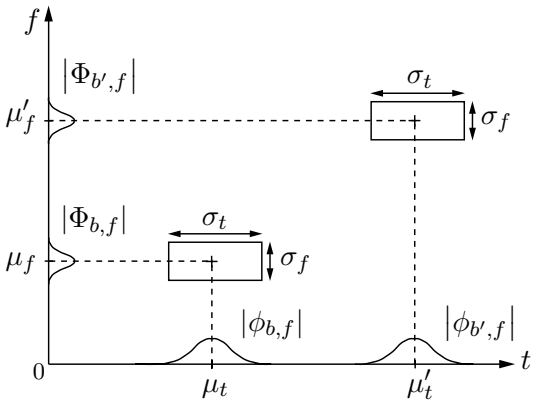
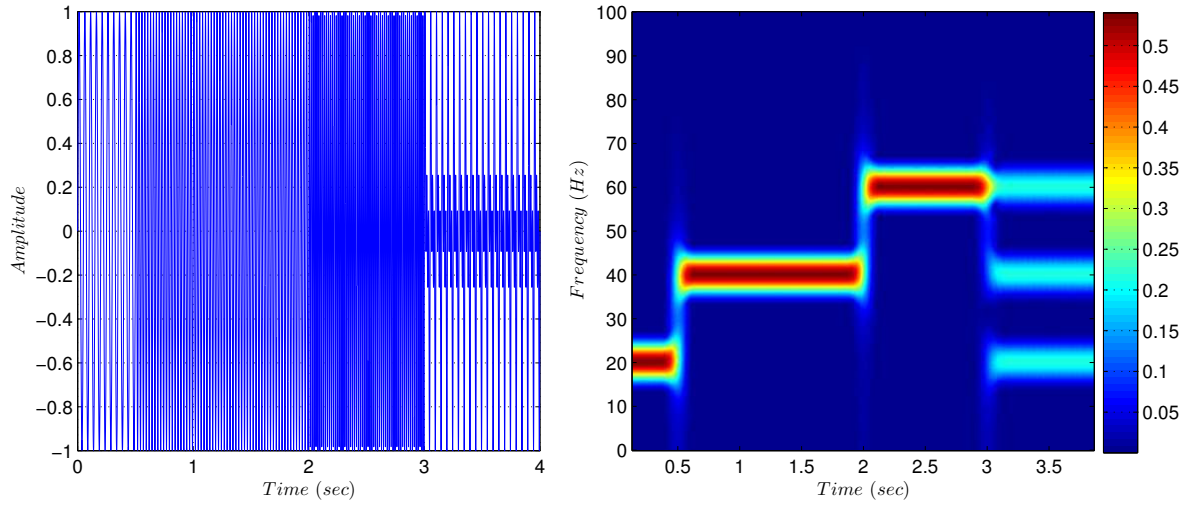


Fig. 5.4: Heisenberg boxes representing the energy spread of two WFT atoms

As the figure illustrates, the two Heisenberg boxes of the WFT atoms are exactly the same in the two time–frequency positions. This in turn, means that the time–frequency resolution of the WFT is constant during the complete analysis process. This, indeed, works fine as long as the signal to be analyzed does not include components with different time–frequency resolutions, some being well-localized in time and others well-localized in frequency. For such signals, the constant time–frequency resolution of the WFT can be a serious pitfall in the WFT analysis.
As an example on WFT, Figure 5.5 shows the WFT spectrogram of the three frequencies sine wave signal of Figure 5.2.



(a) Time signal

(b) Windowed Fourier transform spectrogram

Fig. 5.5: Windowed Fourier transform example

In contrast to the FT spectrum, from the WFT spectrogram in Figure 5.5.b not only the kind of frequencies existing in the signal and their amplitude but also the duration and the starting time of each frequency can be nicely extracted. For example the $40\,\mathrm{Hz}$ frequency exists in the time periods between $0.5-2\,\mathrm{s}$ and $3-4\,\mathrm{s}$. The effects of the constant time–frequency resolution of the WFT can be clearly seen in the high and low frequency ranges where the WFT fails to optimize the compromise between the time and frequency resolutions.
In addition to the resolution problem, to perform a successful WFT analysis it is essential to carefully define several parameters prior to the analysis. Examples of such parameters are the type of the window function to be used in the analysis, the window length and the amount of overlapping between two successive windows during the analysis. Unfortunately, there is no systematic way for choosing proper values for these parameters in practical application, therefore it is customary to adjust these parameters using trial and error.

5.1.3 Continuous Wavelet Transform

Another important family of time–frequency transforms which overcomes many of the WFT pitfalls is the *wavelet transform*. The time–frequency atoms of the wavelet transform are called *wavelets* and defined by

$$\psi_{b,\mathrm{a}}(t) = \frac{1}{\sqrt{\mathrm{a}}}\psi\left(\frac{t-b}{\mathrm{a}}\right), \tag{5.9}$$

where ψ is called the *mother wavelet*, a is the dilation (scale) parameter and b is the translation parameter, cf. [53].

In contrast to the WFT atoms, wavelets have a varying time–frequency resolution which depends on the wavelet dilation parameter a. In the time–frequency plane, the Heisenberg box of a wavelet atom $\psi_{b,\mathrm{a}}$ is a rectangle centered at $(b, \mu_f/\mathrm{a})$, with time and frequency spreads, respectively, proportional to a and $1/\mathrm{a}$. As the wavelet dilation parameter a varies, the time and

frequency widths of the wavelet Heisenberg box changes, but its area remains constant achieving the best compromise between the time and frequency resolutions. Figure 5.6 illustrates the energy spread of two wavelets centered at $(b, \mu_f/\mathrm{a})$ and $(b_0, \mu_f/\mathrm{a}_0)$.

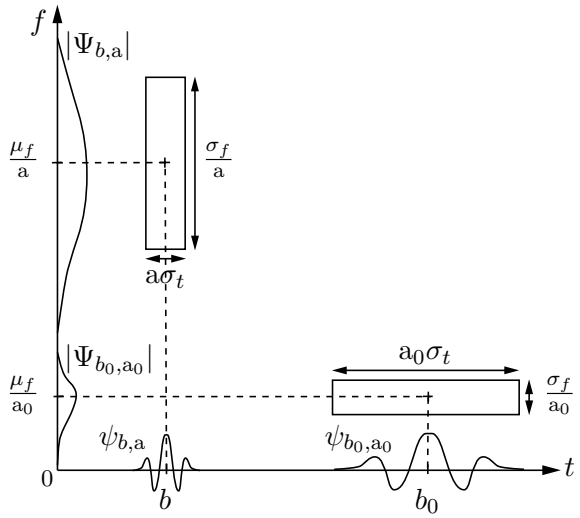


Fig. 5.6: Heisenberg boxes representing the energy spread of two wavelets

The wavelet transform exists in both continuous and discrete forms. In this thesis, only the continuous wavelet transform is of interest. The CWT is the correlation of a time signal with a dictionary of dilated and translated versions of the analyzing mother wavelet ψ. It decomposes a signal into a time–scale representation that elucidates the transient characteristics of that signal. For a finite energy signal $x(t) \in L^2(\mathbb{R})$, the continuous wavelet transform is defined by

$$W_\psi x(b, \mathrm{a}) = \langle x, \psi_{b,\mathrm{a}} \rangle = \int_{-\infty}^{\infty} x(t) \frac{1}{\sqrt{\mathrm{a}}} \psi^* \left(\frac{t-b}{\mathrm{a}} \right) \mathrm{d}t, \tag{5.10}$$

where $(\cdot)^*$ denotes the complex conjugate operator and $W_\psi x(b, \mathrm{a})$ is the wavelet transform of $x(t)$. Since the continuous wavelet transform is complete and maintains an energy conservation, an inverse continuous wavelet transform exists and is given by

$$x(t) = \frac{1}{C_\psi} \int_0^{\infty} \int_{-\infty}^{\infty} W_\psi x(b, \mathrm{a}) \frac{1}{\sqrt{\mathrm{a}}} \psi \left(\frac{t-b}{\mathrm{a}} \right) \mathrm{d}b \frac{\mathrm{da}}{\mathrm{a}^2}, \tag{5.11}$$

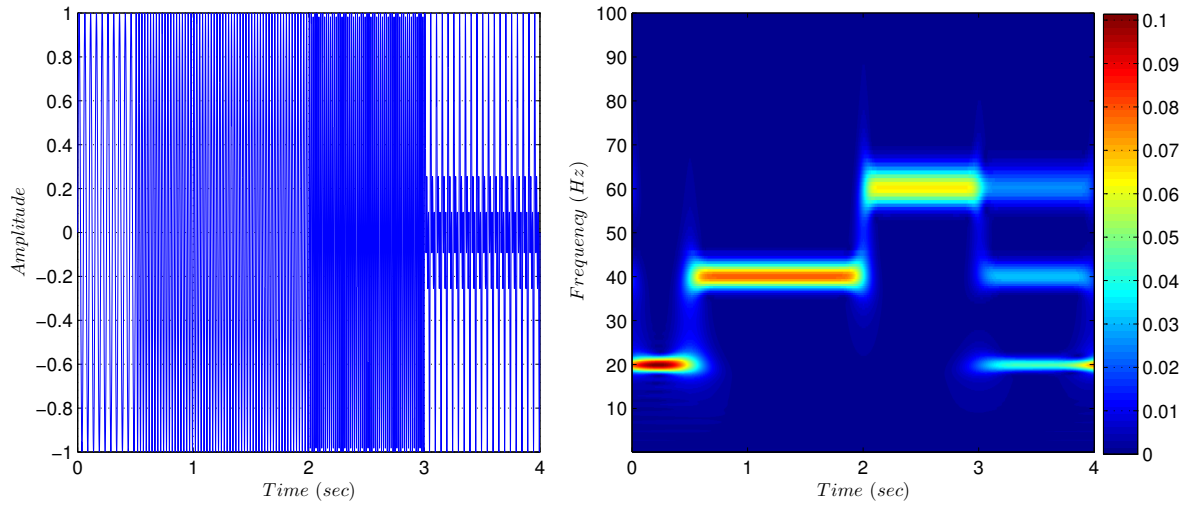
as long as ψ satisfies the *admissibility condition*

$$C_\psi = \int_0^{\infty} \frac{\left| \hat{\psi}(\omega) \right|^2}{\omega} \mathrm{d}\omega < \infty, \tag{5.12}$$

which implies that

$$\Psi(0) = \int_{-\infty}^{\infty} \psi(t) \, \mathrm{d}t = 0. \tag{5.13}$$

As an example on CWT, Figure 5.7 shows the CWT scalogram of the three frequencies sine wave signal of Figure 5.2.



(a) Time signal

(b) Continuous wavelet transform scalogram

Fig. 5.7: Continuous wavelet transform example

As has been discussed in the Heisenberg boxes of the wavelets, the scalogram in Figure 5.7.b illustrates the effectiveness of the CWT in optimally adjusting the time and frequency resolution according to the tested frequencies.
Another feature that makes the wavelet transform attractive for practical use is the possibility to select the wavelet function ψ such that it fits the application at hand. Indeed, in the literature a number of well–developed wavelet functions can be found which cover a wide range of applications. The most commonly used wavelets are:

1. The Haar wavelet: is the simplest existing wavelet and the first function that relates to the wavelet transform. The Haar wavelet is given by

$$\psi(t) = \begin{cases} 1, & 0 \leq t < 1/2, \\ -1, & 1/2 \leq t < 1, \\ 0, & \text{otherwise}, \end{cases} \tag{5.14}$$

cf. [53]. The Haar wavelet provides an excellent time localization, but a poor frequency localization [52].

2. The Mexican hat wavelet: it offers a good time and frequency localization since by construction it is obtained from the second time derivative of the time–frequency optimal Gaussian atom as

$$\psi(t) = \frac{1}{\pi^{\frac{1}{4}}\sqrt{3F_b}} \left(\frac{t^2}{F_b^2} - 1 \right) e^{-\frac{t^2}{F_b^2}}, \tag{5.15}$$

where F_b is the wavelet bandwidth parameter.

3. The Morlet wavelet: is the complex version of the Mexican hat wavelet. It is built based on a complex modulation of the second time derivative of the time–frequency optimal Gaussian atom as

$$\psi(t) = \frac{1}{\sqrt{\pi F_b}}\, e^{i2\pi F_c t}\, e^{\frac{-t^2}{F_b}}, \tag{5.16}$$

where F_c is the wavelet center frequency and F_b is the wavelet bandwidth parameter. The Morlet wavelet as defined above dose not satisfy the admissibility condition. To overcome this, it is customary to add a correction term to the original definition as, cf. [53]

$$\psi(t) = \frac{1}{\sqrt{\pi F_b}}\, e^{-\pi^2 F_b F_c^2}\, e^{\frac{-t^2}{F_b}} \left[e^{i2\pi F_c t}\, e^{\pi^2 F_b F_c^2} - 1 \right]. \tag{5.17}$$

The Morlet wavelet is by construction an analytic wavelet and thus is best suited for applications where the phase and amplitude information of a signal are to be separated.
Besides the above three classical wavelets, the wavelet literature offers a variety of wavelets customized for application since by definition any function can be a wavelet as long as it satisfies the very mild admissibility condition of Equation (5.12).

To identify the critical oscillation regions, several wavelet candidates were tested, e.g. Mexican hat wavelet, Morlet wavelet and impulse response wavelet. All tested wavelets suffer from two main pitfalls: first, an additional optimization process is always required to optimize the wavelet shape parameters in order to achieve satisfactory results. Second, most of the available wavelets are *relatively* symmetric and two-sided wavelets. Thus, whenever such wavelets are used to capture some anti-symmetric single-sided system characteristics such as the system oscillations, additional spurious side effects will show up. Therefore, a new wavelet will be designed to overcome the existing pitfalls and to be more tailored for identifying the oscillation regions.
Since the oscillatory behavior of an elastic system is characterized by its impulse response, the new wavelet function ψ will be constructed based on the system impulse response. This specific wavelet will be called hereafter as the *balanced impulse response wavelet*. The impulse response wavelet itself is not new, different forms of such a wavelet are available in the literature. The starting point for building such a wavelet is the impulse response of an underdamped second order system

$$h(t) = \frac{\omega_n}{\sqrt{1-\zeta^2}}\, e^{-\zeta\omega_n t} \sin(\omega_d t), \tag{5.18}$$

where $\omega_d = \omega_n\sqrt{1-\zeta^2}$ is the damped natural frequency of the system. Since the system impulse response usually does not satisfy the admissibility condition, modifications have to be applied. Junsheng [54], for example, modified the impulse response via direct mirroring to

achieve the admissible impulse response wavelet

$$\psi(t) = \begin{cases} e^{-\frac{\beta\omega_c t}{\sqrt{1-\beta^2}}} \sin(\omega_c t), & t \geq 0, \\ e^{\frac{\beta\omega_c t}{\sqrt{1-\beta^2}}} \sin(\omega_c t), & t < 0, \end{cases} \tag{5.19}$$

where ω_c is the wavelet center frequency and β is a damping or control parameter. The Fourier transform of the impulse response wavelet is given by

$$\Psi(f) = \frac{\beta\omega_c}{i\sqrt{1-\beta^2}} \left[\frac{1}{\frac{\beta^2\omega_c^2}{1-\beta^2} + i(2\pi f - \omega_c)^2} - \frac{1}{\frac{\beta^2\omega_c^2}{1-\beta^2} + i(2\pi f + \omega_c)^2} \right]. \tag{5.20}$$

The resulting impulse response wavelet in time and frequency domains is shown in Figure 5.8.

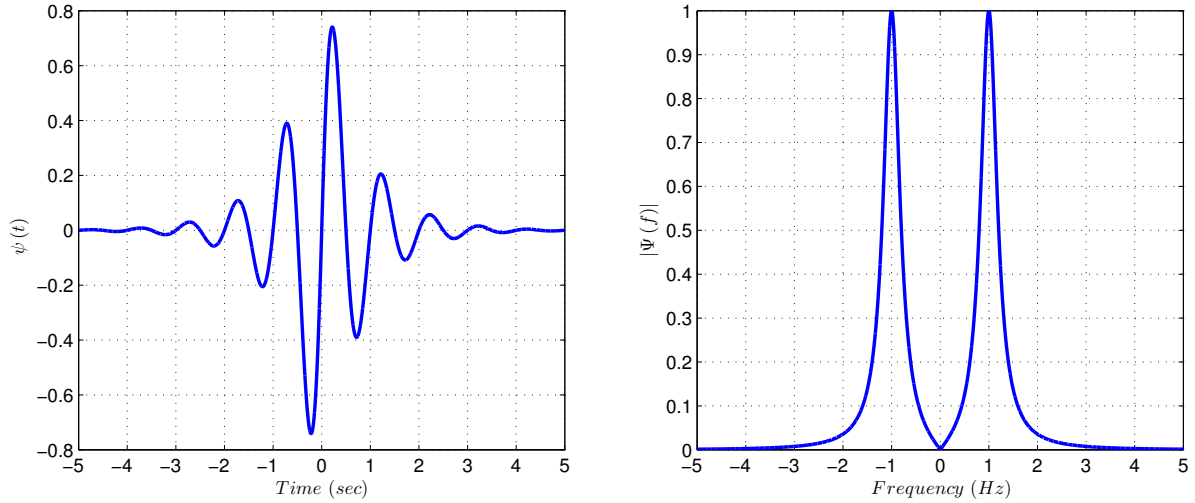


Fig. 5.8: Impulse response wavelet in time domain (left) and frequency domain (right) with $\omega_c = 1\,\text{Hz}$, $\beta = 0.2$

By construction, the impulse response wavelet is an odd function and thus has zero mean which is the essential part of the admissibility condition. On the other hand, the reflection clearly results in a two-sided wavelet that will not reproduce the original system response which, for example, had no symmetry in the beginning.

Our wavelet construction will complete a function with a damped oscillation behavior in $\mathbb{R}_+$ by adding a function with controllable support in $\mathbb{R}_-$ such that the resulting function satisfies not only the admissibility condition but also provides a certain amount of smoothness, so that ψ and Ψ both decay sufficiently fast. Consequently, with a generalized form of the impulse response $h(t)$ of an underdamped second order system where the damping ratio ζ is replaced with a general control parameter β as

$$g(t) = e^{-\frac{\beta\omega_c t}{\sqrt{1-\beta^2}}} \sin(\omega_c t), \tag{5.21}$$

the new wavelet function is given by

$$\psi(t) = \begin{cases} g(t), & t \geq 0, \\ q(t), & T \leq t \leq 0, \\ 0, & t < T, \end{cases} \tag{5.22}$$

where $T < 0$ is a freely chosen parameter that defines the support extension to the negative axis and controls the time localization properties of the resulting wavelet. Moreover, q is a function from a finite dimensional space that has to satisfy the balancing condition

$$\int_T^0 q(t)\ \mathrm{d}t = -\int_0^\infty g(t)\ \mathrm{d}t = \frac{-1}{\omega_c\left(\frac{\beta^2}{1-\beta^2}+1\right)}, \tag{5.23}$$

as well as, for $\mathrm{k} = 0, \ldots, n$, the smoothness conditions

$$q^{(\mathrm{k})}(T) = 0, \tag{5.24}$$

and

$$q^{(\mathrm{k})}(0) = g^{(\mathrm{k})}(0) = \omega_c^{\mathrm{k}} \sum_{\mathrm{i} \leq (\mathrm{k}-1)/2} (-1)^{\mathrm{k}-\mathrm{i}-1} \binom{\mathrm{k}}{2\mathrm{i}+1} \left(\frac{\beta}{\sqrt{1-\beta^2}}\right)^{\mathrm{k}-2\mathrm{i}-1}, \tag{5.25}$$

One way to build q is to use polynomial completions. The $2n+2$ conditions in Equations (5.24) and (5.25) form the well-known Hermite conditions which always have a unique solution in Π_{2n+1}. Thus, the complete problem defined by Equations (5.23) to (5.25) can be solved in Π_{2n+1} if the solution p of Equations (5.24) and (5.25) happens to satisfy the balancing condition of Equation (5.23). Otherwise, the solution belongs to Π_{2n+2} and is given by

$$q = p - \frac{r}{\int_T^0 r(t)\ \mathrm{d}t}\left(\int_T^0 p(t)\ \mathrm{d}t + \int_0^\infty g(t)\ \mathrm{d}t\right), \qquad r(t) = t^{n+1}(t-T)^{n+1}, \tag{5.26}$$

where $r > 0$ on $(T, 0)$, hence, $\int r(t)\ \mathrm{d}t > 0$.

For example, for the case $n = 1$, q can be only constructed using Equation (5.26) and the coefficients $a_0, \ldots, a_4$ of the polynomial completion are the solutions of the system

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & T & T^2 & T^3 & T^4 \\ 0 & 1 & 2T & 3T^2 & 4T^3 \\ -T & \frac{-T^2}{2} & \frac{-T^3}{3} & \frac{-T^4}{4} & \frac{-T^5}{5} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} 0 \\ \omega_c \\ 0 \\ 0 \\ \frac{-1}{\omega_c\left(\frac{\beta^2}{1-\beta^2}+1\right)} \end{bmatrix}. \tag{5.27}$$

The Fourier transform of the resulting wavelet is

$$\begin{aligned}\Psi(f) = &\frac{1}{2i}\left(\frac{1}{\frac{\beta\omega_c}{\sqrt{1-\beta^2}}+i(2\pi f-\omega_c)}-\frac{1}{\frac{\beta\omega_c}{\sqrt{1-\beta^2}}+i(2\pi f+\omega_c)}\right)\\ &+ a_0\frac{i\left(1-e^{-i2\pi fT}\right)}{2\pi f}+a_1\frac{1-e^{-i2\pi fT}(1-2i\pi fT)}{(2\pi f)^2}\\ &- a_2\frac{2i+e^{-i2\pi fT}\left(4\pi Tf+i\left(4T^2\pi^2f^2-2\right)\right)}{(2\pi f)^3}\\ &- a_3\frac{6+e^{-i2\pi fT}\left(12T^2\pi^2f^2-6+i\left(8T^3\pi^3f^3-12T\pi f\right)\right)}{(2\pi f)^4}\\ &+ a_4\frac{24i-e^{i2\pi fT}\left(32T^3\pi^3f^3-48T\pi f+i\left(16T^4\pi^4f^4-48T^2\pi^2f^2+24\right)\right)}{(2\pi f)^5}.\end{aligned}$$

For a wavelet function constructed according to the above method and with an arbitrary polynomial order of $2n+2$, the Fourier transform can be explicitly computed as

$$\begin{aligned}\Psi(f) &= \left(\chi_{[T,0]}\sum_{\mathrm{k}=0}^{2n+2}a_{\mathrm{k}}(\cdot)^{\mathrm{k}}\right)^{\wedge}(f)+\left(\chi_{[0,\infty]}e^{-\beta\omega_c\left(1-\beta^2\right)^{-1/2}\cdot}\sin(\omega_c\cdot)\right)^{\wedge}(f)\\ &= \sum_{\mathrm{k}=0}^{2n+2}a_{\mathrm{k}}\left(\frac{-\mathrm{k}!}{(i2\pi f)^{\mathrm{k}+1}}+\frac{e^{-i2\pi fT}}{(i2\pi f)^{\mathrm{k}+1}}\sum_{\mathrm{l}=0}^{\mathrm{k}}\frac{\mathrm{k}!\,(i2\pi fT)^{\mathrm{l}}}{\mathrm{l}!}\right)+\\ &\quad\frac{1}{2i}\left(\frac{1}{\frac{\beta\omega_c}{\sqrt{1-\beta^2}}+i(2\pi f-\omega_c)}-\frac{1}{\frac{\beta\omega_c}{\sqrt{1-\beta^2}}+i(2\pi f+\omega_c)}\right),\end{aligned}\tag{5.28}$$

where $(\cdot)^{\wedge}$ denotes the Fourier transform operator. The first part of the above expression is singular at $f=0$ and therefore hard to sample in the neighborhood of the origin. The singularity is only removable due to the choice of the coefficients a_{k} which guarantees that Ψ is uniformly continuous. This dependency of the coefficients which requires that the numerator is *precisely* zero in order to apply the l'Hôpital rule cannot be maintained in floating point computations, hence this formula is numerically very unstable in the neighborhood of the origin. Fortunately, there is a series expansion of the truncated polynomial which can be used close to the origin.

Lemma. *For the truncated polynomial function*

$$q = \chi_{[T,0]}\sum_{\mathrm{m}=0}^{2n+2}a_{\mathrm{m}}(\cdot)^{\mathrm{m}},$$

A convergent series representation is defined as

$$Q(f) = -\sum_{\mathrm{k}=0}^{\infty}\frac{(iTf)^{\mathrm{k}}}{\mathrm{k}!}\sum_{\mathrm{m}=0}^{2n+2}a_{\mathrm{m}}\frac{T^{\mathrm{m}+1}}{\mathrm{m}+\mathrm{k}+1}.\tag{5.29}$$

Proof. First,

$$Q^{(\mathrm{k})}(0) = \int_{\mathbb{R}} (it)^{\mathrm{k}} g(t)\, \mathrm{d}t = i^{\mathrm{k}} \int_{T}^{0} \sum_{\mathrm{m}=0}^{2n+2} a_{\mathrm{m}}\, t^{\mathrm{m}+\mathrm{k}}\, \mathrm{d}t = -(iT)^{\mathrm{k}} \sum_{\mathrm{m}=0}^{2n+2} a_{\mathrm{m}} \frac{T^{\mathrm{m}+1}}{\mathrm{m}+\mathrm{k}+1}.$$

Substituting this into the Taylor series

$$Q(f) = \sum_{\mathrm{k}=0}^{\infty} \frac{G^{(\mathrm{k})}(0)}{\mathrm{k}!} f^{\mathrm{k}},$$

which exists since q is compactly supported, hence $Q \in C^{\infty}(\mathbb{R})$, gives Equation (5.29). The sum

$$\sum_{\mathrm{m}=0}^{2n+2} |a_{\mathrm{m}}| \frac{|T|^{\mathrm{m}+1}}{\mathrm{m}+\mathrm{k}+1},$$

is bounded independently of k and the remainder of the series is the series expansion of e^{ifT}, hence the series converges absolutely. □

For small values of $|Tf|$, the series in Equation (5.29) converges very fast and so Equation (5.29) is suitable and a very stable way for sampling Ψ close to the origin, while for large values of $|Tf|$, Equation (5.28) is the more appropriate expression to evaluate. This observation suggests the use of small values of $|T|$ which is in accordance with our application of completing a one–sided wavelet without adding too much support on the negative side.

The resulting balanced impulse response wavelet for the case $n = 1$ in time and frequency domain is shown in Figure 5.9. By construction, the wavelet is real single-sided one that satisfies the admissibility condition.

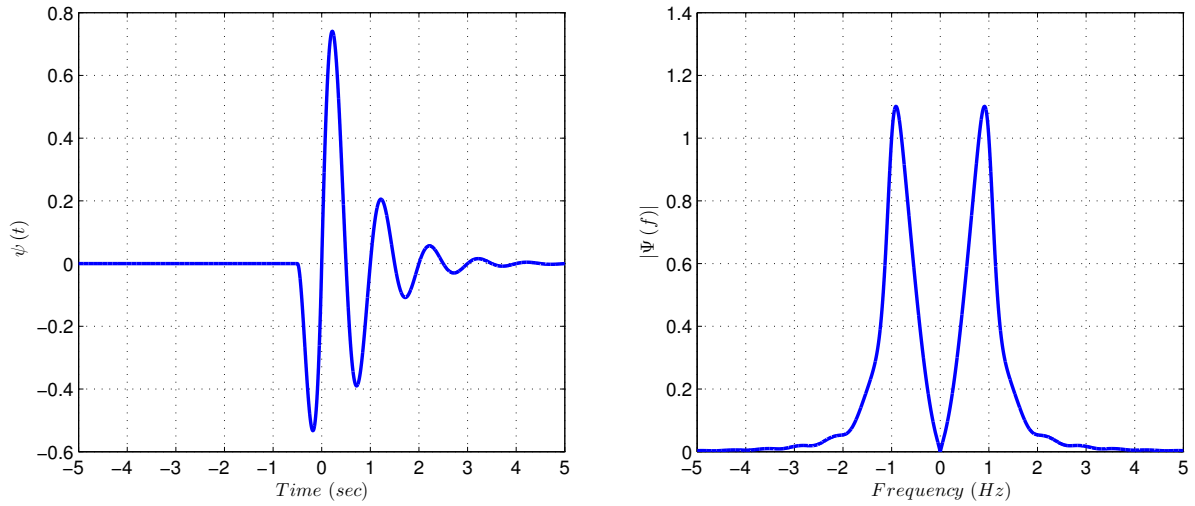


Fig. 5.9: Balanced impulse response wavelet in time domain (left) and frequency domain (right) with $\omega_c = 1\,\mathrm{Hz}$, $\beta = 0.2$, $T = -0.5\,\mathrm{s}$

To compare this wavelet to the behavior of the conventional impulse response wavelet given

by Equation (5.19), a test signal is used which contains two impulse responses of a second order underdamped system with damped natural frequency $F_d = 25\,\mathrm{Hz}$ and damping ratio $\zeta = 0.1$, defined as

$$\begin{aligned} g(t) =& \mathcal{H}(t-0.2)\, e^{-\frac{\zeta 2\pi F_d}{\sqrt{1-\zeta^2}}(t-0.2)} \sin\left(2\pi F_d (t-0.2)\right) \\ &+ \mathcal{H}(t-1)\, e^{-\frac{\zeta 2\pi F_d}{\sqrt{1-\zeta^2}}(t-1)} \sin\left(2\pi F_d (t-1)\right), \end{aligned} \tag{5.30}$$

where $\mathcal{H}(\cdot)$ is the Heaviside function. For this signal, two wavelet transforms were considered with an analyzing frequency of $25\,\mathrm{Hz}$ and identical center frequency and damping parameter, $\omega_c = 25\,\mathrm{Hz}$ and $\beta = 0.1$. A normalized version of the test signal and the modulus of the wavelet coefficients is shown in Figure 5.10. As the results demonstrate, the wavelet constructed according to the above method outperforms the conventional one in catching the impulse amplitude envelopes and their time locations. Thus, it provides a much better alternative for applications where accurate detection of impulses' amplitude and time location are needed such as the identification of the oscillation regions. Our paper in [55] reviews the construction of the balanced impulse response wavelet and its application for extracting the vibrational error in control systems.

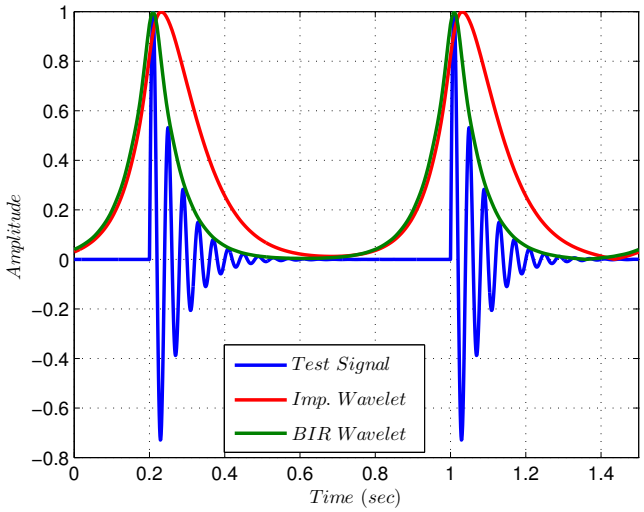


Fig. 5.10: Time domain comparison between conventional impulse response wavelet (red) and the balanced impulse response wavelet (green) with $\omega_c = 25\,\mathrm{Hz}$, $\beta = 0.1$, $T = -0.5\,\mathrm{s}$

To compare the behavior of both wavelets in the frequency domain, Figure 5.11 shows a normalized version of the frequency response of a second order underdamped system, the conventional impulse response wavelet and the balanced impulse response wavelet. Both wavelets satisfy the admissibility condition $\Psi(0) = 0$, however, the the effects of mirroring in the conventional impulse response wavelet can be clearly seen by the smaller bandwidth of the wavelet around its center frequency in comparison to the system response. On the other hand, the balanced impulse response wavelet fits the system response perfectly.

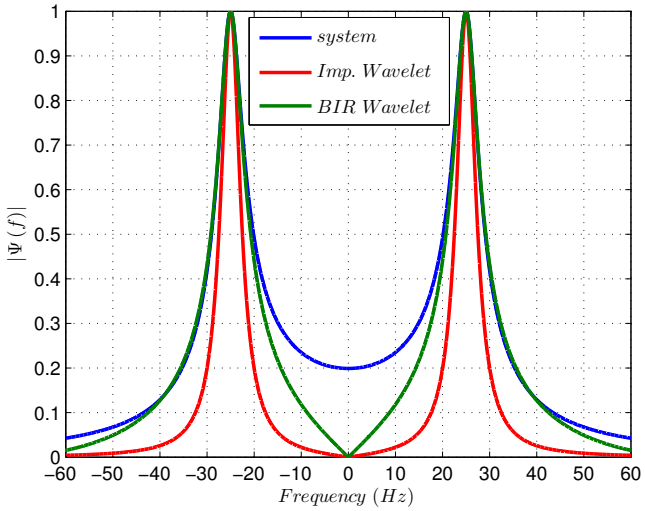


Fig. 5.11: Normalized frequency response of a second order underdamped system (blue), the conventional impulse response wavelet (red) and the balanced impulse response wavelet (green) with $\omega_c = 25\,\mathrm{Hz}$, $\beta = 0.1$, $T = -0.5\,\mathrm{s}$

5.2 Identification of Oscillation Regions

Identifying the critical oscillation regions along the contour is a crucial requirement for the performance of our strategy. The complete oscillating behavior of the machine is described by the vibrational contour error signal as explained in Chapter 4. This signal is composed of many damped oscillatory components with different frequencies and is distributed along the entire signal. Thus, identifying the critical oscillation regions in such a signal requires a simultaneous time and frequency identification. This, in turn, imposes the need for a well localized time-frequency approach in order to accurately identify the critical oscillation regions. According to the previous section, the CWT is the most suitable time–frequency analysis tool to be used for this task. To perform a successful wavelet analysis, two main choices have to be made: the type of the mother wavelet function that best fits the application at hand and the optimal values of the shape parameters describing such a function. Thanks to the balanced impulse response wavelet, both choices are optimally done since the wavelet is carefully adapted to represent the system behavior and its shape parameters (ω_c and β) are directly related to the damped natural frequency ω_d and the damping ratio ζ of the system.
To identify the critical oscillation regions in the vibrational contour error signal, a method is developed based on a wavelet analysis, which makes use of the redundant representation of the wavelet transform and its ability to localize the signal information on the time-scale grid. The method identifies the relevant error regions by performing a forward CWT on the vibrational contour error signal at a small number of selected scales only and consists of the following steps:

1. Define the wavelet shape parameters (ω_c and β) as the system damped natural frequency ω_d and damping ratio ζ, respectively.

2. For each component of the vibrational contour error signal $e_{c,vib,x}$, $e_{c,vib,y}$ and $e_{c,vib,z}$ perform steps $3-6$.

3. Perform a forward wavelet transform on the vibrational contour error signal component. The analyzing scales should cover a small band around the wavelet center frequency, i.e. $\omega \in [\omega_c \pm \Delta\omega]$ where ω is the analyzing frequencies and $\Delta\omega$ is a small percentage of the wavelet center frequency, typically in the range of $5\,\%$.

4. If necessary, a simple soft thresholding can be applied to the resulting wavelet coefficients for reducing the noise and highlighting interesting error features.

5. From the modulus of the thresholded wavelet coefficients identify the maximal points which represent the starting points of the local oscillating behavior of the machine.

6. Based on a predefined vibration amplitude threshold, refine the maximal points.

7. Combine the maximal points from each error component and eliminate the redundant ones.

As an example, Figure 5.12 shows the magnitude profile of the vibrational contour error of a two-axis CNC machine in response to test contour III in Appendix A. The dark regions in the figure were identified as critical oscillation regions using the wavelet approach discussed above. In this example, each machine axis is modeled as a second order underdamped system with natural frequency $F_n = 25\,\text{Hz}$ and damping ratio $\zeta = 0.04$.

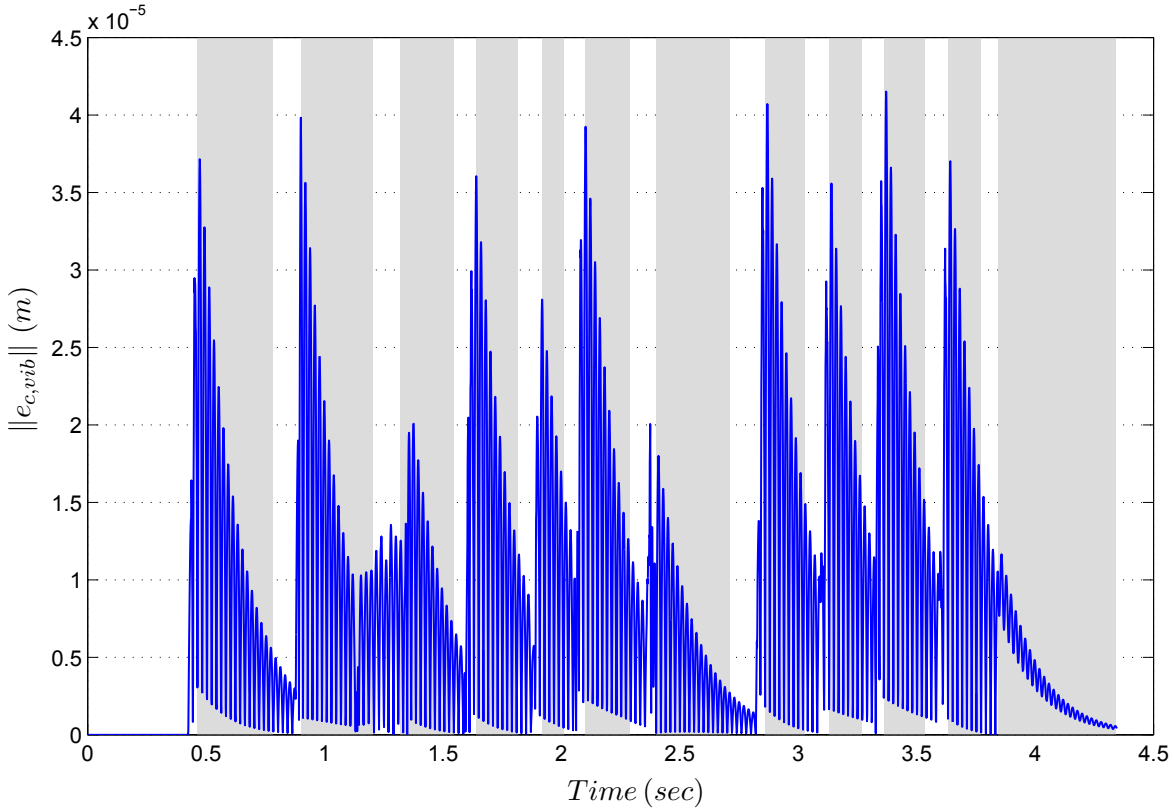


Fig. 5.12: Critical oscillation regions

The method is also valid for cases where the machine axes have different resonance frequencies. As an example, Figure 5.13 shows the resulting critical oscillation regions for the same reference contour with x-axis and y-axis natural frequencies of $25\,\mathrm{Hz}$ and $35\,\mathrm{Hz}$, respectively. The results show the capability of this method to precisely identify the critical vibration regions from the contour vibrational error signal regardless of the axes' frequencies.

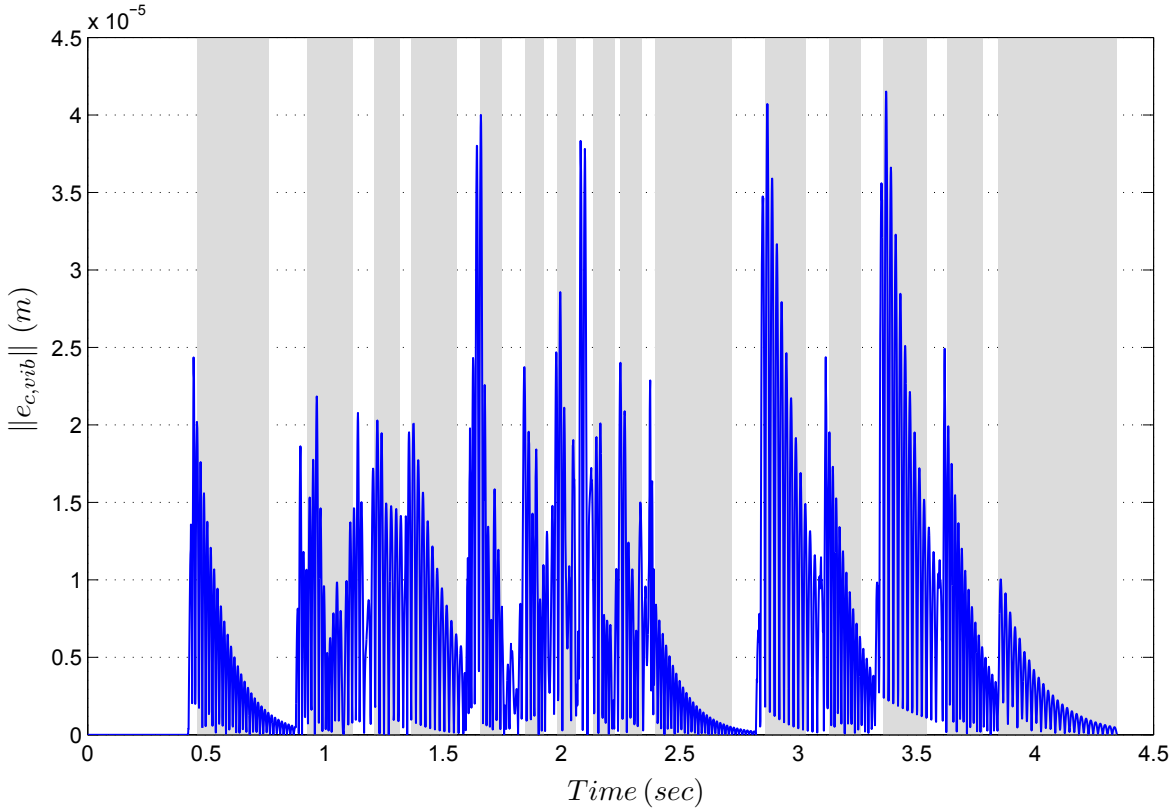


Fig. 5.13: Critical oscillation regions for axes with different frequencies

5.3 Principle of Time-Scaling

Time-scaling is the concept of changing the clock by which a system evolves. Mathematically, this corresponds to a re-parameterization process of the driving functions of the system. In the context of time–scaling it is common to distinguish between two types of time parametrization of a function. The first one parametrizes the function with respect to the real time t while the second parametrizes it with respect to a fictitious scaled time τ which acts as a re-parameterization of the real time. For instance $s\,(t)$ in Figure 5.14 is a general path function for a CNC machine tool parametrized with respect to the real time. Machines evolving with the real time bases, at a time instant t_1 will output a path position of $s\,(t_1)$. On the other hand, machines evolving with the scaled time bases will output $\tilde{s}\,(t_1) = s\,(\tau_1)$ which is basically the original position evaluated at a different time instant τ_1.

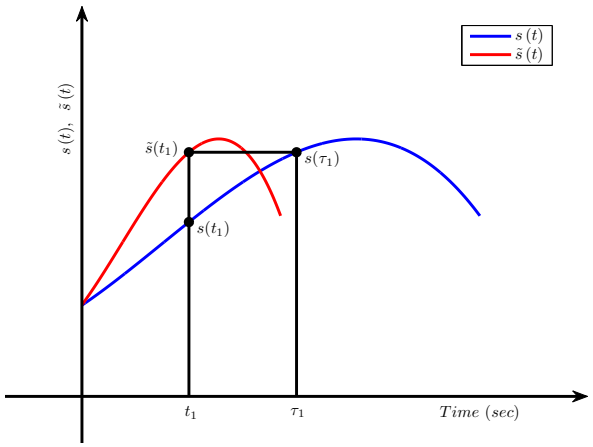


Fig. 5.14: A general path function for CNC machine tool with time–scaling

This can be considered as the time t, describing the path function, being either shortened or elongated based on the type of the scaling process. The scaled time τ is a fictitious time that depends on how fast/slow the path function is sampled inside the machine. The relationship between the scaled time τ and the real time t is defined through an invertible mapping known as the time–scaling law. A typical time–scaling law in the continuous time domain is defined by

$$\tau = \int c\,(t)\ \mathrm{d}t \quad \leftrightarrow \quad \frac{d\tau}{dt} = c\,(t)\,, \tag{5.31}$$

where $c\,(t)$ is called the *scaling function*. In the discrete time domain this corresponds to

$$\tau\,[k\,T_s] = \sum c\,[k\,T_s]\ T_s, \tag{5.32}$$

where $k = 0, 1, \ldots, n$ is the indexing parameter and T_s is the sampling time.
Performing a time–scaling operation on a jerk limited path trajectory of a CNC machine tool modifies the dynamic behavior of that path, i.e. it changes its velocity, acceleration and jerk profiles simultaneously. However, the original contour described by such a path trajectory remains unaffected. With the time–scaling law in Equation (5.31), the machine tools' path trajectories, position, velocity, acceleration and jerk after time–scaling are obtained as

$$\begin{aligned}
\tilde{s}\,(t) &= s\,(\tau) \\
\dot{\tilde{s}}\,(t) &= \frac{d\tilde{s}\,(t)}{dt} = \frac{ds\,(\tau)}{d\tau}\frac{d\tau}{dt} = s'\,(\tau)\,\dot{\tau} \\
\ddot{\tilde{s}}\,(t) &= \frac{d^2\tilde{s}\,(t)}{dt^2} = s''\,(\tau)\,\dot{\tau}^2 + s'\,(\tau)\,\ddot{\tau} \\
\dddot{\tilde{s}}\,(t) &= \frac{d^3\tilde{s}\,(t)}{dt^3} = s'''\,(\tau)\,\dot{\tau}^3 + 3s''\,(\tau)\,\dot{\tau}\ddot{\tau} + s'\,(\tau)\,\dddot{\tau},
\end{aligned} \tag{5.33}$$

with derivatives defined as $s' = \frac{ds}{d\tau}$, $\dot{s} = \frac{ds}{dt}$ and $\dot{\tau} = c\,(t)$.
By Equation (5.33), the scaled version of the velocity profile is directly proportional to the original velocity profile via the scaling function, whereas the acceleration profiles are related

by the scaling function $c(t)$ **and** its first time derivative $\dot{c}(t)$. The existence of the first and even the second time derivative of the scaling function in the jerk profile necessitates a smooth scaling function to maintain the smoothness of the original profiles.

More interesting, however, is the effect of time–scaling on the frequency content of these profiles which is given by the Fourier transform time scaling property [51]. If the Fourier transform of a general signal $x(t)$ is given by $X(f)$,

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-i2\pi ft}\, \mathrm{d}t, \tag{5.34}$$

then the Fourier transform of a time scaled version of that signal $\tilde{x}(t) = x(ct)$, where c is a real constant greater than zero, is determined by substituting $t^* = ct$ in the Fourier integral equation, giving

$$\tilde{X}(f) = \int_{-\infty}^{\infty} x(ct)\, e^{-i2\pi ft}\, \mathrm{d}t = \frac{1}{c} \int_{-\infty}^{\infty} x(t^*)\, e^{-i2\pi \frac{f}{c} t^*}\, \mathrm{d}t^* = \frac{1}{c} X\left(\frac{f}{c}\right). \tag{5.35}$$

The Fourier transform version of the time scaling property implies that the time scaling process shifts and scales the frequency content of the original signal by a factor inversely proportional to the scaling function. Figure 5.15 shows a rectangular signal under different constant time scaling. The left-hand side of the figure shows the signal changes in the time domain, whereas the right-hand side shows the corresponding frequency content changes. As the time scale expands, the frequency scale contracts showing a shift to the left in the frequency content of the original signal and the amplitude increases vertically such that the area under the frequency spectrum remains constant.

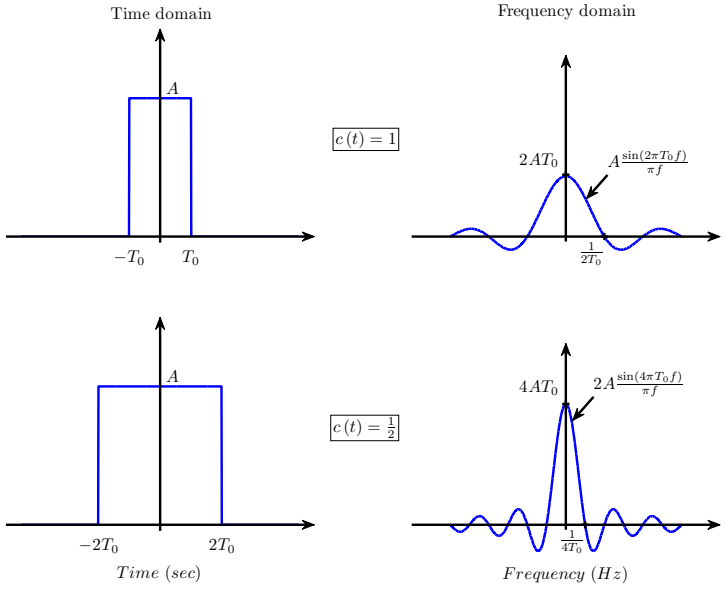


Fig. 5.15: Time scaling property in frequency domain

In this thesis, the property of time–scaling will be used to shift the frequency content of the machine reference trajectories away from the axes resonance frequencies with a suitably designed scaling function.

5.4 The Scaling Function

A time scaling function $c(t)$ can be any function that satisfies the following conditions [50]:

- $0 < c(t) \leq 1$ is a nonnegative function since time cannot be reversed and it is bounded to one since the original profiles to be scaled are time optimal.
- $\tau(0) = 0$: the scaled and original time start from the same point, since the movement must start at the same point.
- $\tau(t_1) = \int_0^{t_1} c(t)\, \mathrm{d}t = t_f$ for some $t_1 > 0$: the movements must finish at some point.

In its simplest form, the scaling function $c(t)$ is a constant value with $\dot{c}(t) = 0$. Despite the linearization effects which the constant function imposes in Equation (5.33), scaling the trajectories of the machine by a constant factor will slow down the trajectories extremely. In addition, using a constant scaling function will only shift the frequency content of the scaled trajectories by a constant factor. Such an operation is helpful when the signal to be scaled contains a single frequency that persists during the entire signal. Unfortunately, practical trajectories are transient signals, where frequencies are widely distributed throughout the entire signal. Figure 5.16 shows an example of a Wavelet transform scalogram of a practical trajectory.

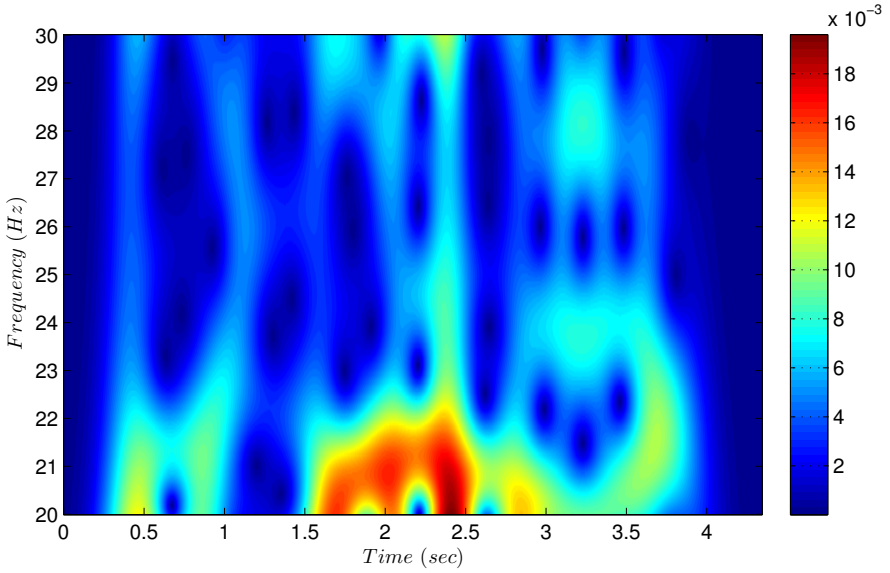


Fig. 5.16: Wavelet transform scalogram of a typical trajectory for a CNC machine tool

This distribution in the frequency content of the trajectories and the need for preserving the time optimality of the original trajectories impose the need for a well localized dynamic scaling function that can locally modify the frequency content of the trajectories.

In this thesis, the methodology in designing the scaling function depends on defining a sufficiently general *scaling function template* $g(t)$. The template function is parameterized by a finite number of parameters which will be tuned via optimization routines such that the frequency content of the targeted region in the trajectory is properly shifted. The complete scaling function $c(t)$ is then built stepwise by suitably shifting tuned copies of the template function $g(t)$ to cover all the required regions in the trajectory.
The template function can consist of any function as long as it is smooth, continuously differentiable and can be parameterized by a finite number of parameters. The general shape of the scaling function template consists of the five main regions shown in Figure 5.17. The first region is a start (idle) region where no scaling is taking place $g(t) = 1$. The second is a transition region where the template function descends from unity to the constant scale factor c. The third region is the constant scaling region where $g(t) = c$. The fourth is another transition region where the template function ascends from the constant scale c to unity. The final region is an idle region where no scaling is taking place $g(t) = 1$ and it lasts until the end of the trajectory or the start of another scaling function template.

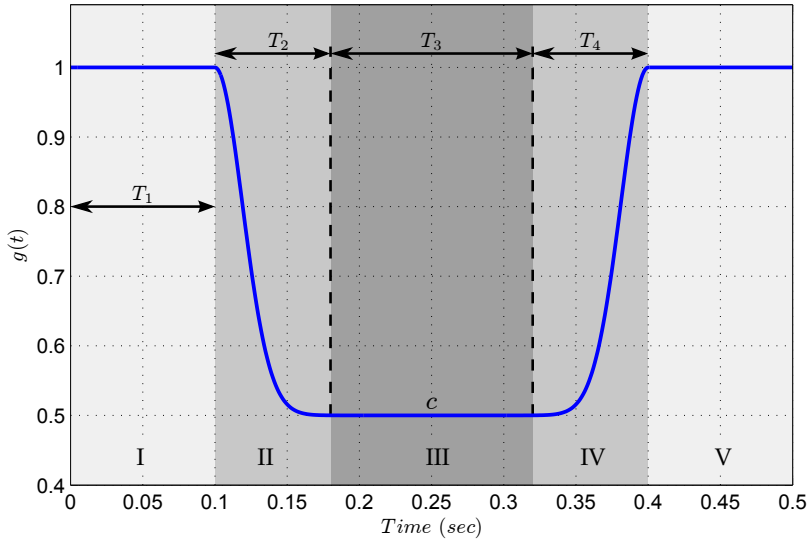


Fig. 5.17: General scaling function template

The characteristics of the template function and the partition of its regions are controlled by five tunable parameters:
1. Scaling Factor c: It defines the factor by which the trajectories are time scaled during the constant scaling region of the template function. It is close in meaning to the constant time scaling factor. However, its effect on the frequency shifting is slightly different due to the falling and rising parts of the template function.
2. Time to Start Scaling T_1: It defines the time instance at which the scaling operation starts to affect the trajectories. This factor depends mainly on the kind of trajectory under scaling and its frequency content.
3. Time to Reach the Scaling Factor T_2: It defines the time required by the scaling function template to descend from unit scale to the constant scaling factor c. Since the scaled acceler-

ation profiles depend on the first time derivative of the scaling function, this factor plays a major role in the smoothness and magnitude of the resulting acceleration profile. A high value for this factor should be always considered to achieve smooth acceleration profiles and to not violate the acceleration limits.

4. Time for Constant Scaling T_3: It defines the duration of the constant scaling region of the scaling function template.

5. Time to Rise to the Idle State T_4: It defines the time required by the scaling function template to ascend from the constant scaling factor c to unity scale. In this thesis, this parameter will be always defined to be equal to T_2. The reason for this is mainly to generate a symmetric scaling function which, in turn, maintains the symmetry of the original unscaled trajectories.

The smoothness of the scaling function is determined by the kind of function used to descend from unity scale to the required scaling factor c and ascending back to unity scale in the template function. Since the time derivatives of the scaled trajectories, i.e. velocity, acceleration and jerk, contain the time derivatives of the scaling function, the choice of the descending and ascending functions is limited to those which are at least twice continuously differentiable in order to maintain the smoothness of the original trajectories. The following subsections discuss possible choices for the template function $g(t)$.

5.4.1 Polynomial Function

One way to build the template function $g(t)$ from polynomial functions is to use Hermite interpolation. The template function consists of five parts marked as regions I,...,V in Figure 5.17. The boundaries of these regions are defined by the time instants $0, T_1, T_1+T_2, T_1+T_2+T_3$ and $T_1+2T_2+T_3$. Using these time instants as interpolation points $t_0, t_1, \ldots, t_m$, the template function can be built as a piecewise Hermite interpolant $S(t)$ such that

(i) $S(t) \in C^n[t_0, t_m]$.

(ii) $S^{(\mathrm{k})}(t_\mathrm{i}) = g^{(\mathrm{k})}(t_\mathrm{i})$ for $\mathrm{k} = 0, 1, \ldots, n$ and $\mathrm{i} = 0, 1, \ldots, m$.

(iii) On each interval $[t_\nu, t_{\nu+1}]$ for $\nu = 0, 1, \ldots, m-1$, $S(t)$ is a polynomial of degree $2n+1$ defined by

$$p_\nu(t) = a_0 + a_1(t-t_\nu) + \cdots + a_{2n+1}(t-t_\nu)^{2n+1}. \tag{5.36}$$

For each polynomial there are $2n+2$ Hermite conditions given by

$$p^{(\mathrm{k})}(t_\nu) = g^{(\mathrm{k})}(t_\nu), \tag{5.37}$$

$$p^{(\mathrm{k})}(t_{\nu+1}) = g^{(\mathrm{k})}(t_{\nu+1}), \tag{5.38}$$

with $\mathrm{k} = 0, \ldots, n$, which always have a unique solution in Π_{2n+1}. To solve for the polynomial coefficients $a_0, \ldots, a_{2n+1}$, a linear system from the $2n+2$ Hermite conditions is formulated in a matrix form as

$$\mathbf{A}\vec{a} = \vec{g}, \tag{5.39}$$

with $\Delta_\nu = t_{\nu+1} - t_\nu$

$$\mathbf{A} = \begin{bmatrix} 0! & 0 & 0 & & & \cdots & & 0 \\ 0 & 1! & 0 & & & \cdots & & 0 \\ \vdots & & \ddots & & & & & \vdots \\ 0 & 0 & \cdots & n! & 0 & & \cdots & 0 \\ \Delta_\nu^0 & \Delta_\nu & \cdots & \Delta_\nu^n & \Delta_\nu^{n+1} & \Delta_\nu^{n+2} & \cdots & \Delta_\nu^{2n+1} \\ 0 & \frac{1!}{0!}\Delta_\nu^0 & \cdots & \frac{n!}{(n-1)!}\Delta_\nu^{n-1} & \frac{(n+1)!}{n!}\Delta_\nu^n & \frac{(n+2)!}{(n+1)!}\Delta_\nu^{n+1} & \cdots & \frac{(2n+1)!}{2n!}\Delta_\nu^{2n} \\ \vdots & & \ddots & & & & & \\ 0 & 0 & \cdots & n! & \frac{(n+1)!}{1!}\Delta_\nu & \frac{(n+2)!}{2!}\Delta_\nu^2 & \cdots & \frac{(2n+1)!}{(n+1)!}\Delta_\nu^{n+1} \end{bmatrix},$$

$$\vec{a} = \begin{bmatrix} a_0, & a_1, & \cdots, & a_n, & a_{n+1}, & a_{n+2}, & \cdots, & a_{2n+1} \end{bmatrix}^T,$$
$$\vec{g} = \begin{bmatrix} g(t_\nu), & g^{(1)}(t_\nu), & \cdots, & g^{(n)}(t_\nu), & g(t_{\nu+1}), & g^{(1)}(t_{\nu+1}), & \cdots, & g^{(n)}(t_{\nu+1}) \end{bmatrix}^T.$$

The template function is then given by

$$g(t) = \sum_{\nu=0}^{m-1} \chi_{[t_\nu, t_{\nu+1}]} \sum_{l=0}^{2n+1} a_l (t - t_\nu)^l. \tag{5.40}$$

For example, in the case $n = 2$ the coefficients $a_0, \ldots, a_5$ of each polynomial are the solutions of the system

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & \Delta_\nu & \Delta_\nu^2 & \Delta_\nu^3 & \Delta_\nu^4 & \Delta_\nu^5 \\ 0 & 1 & 2\Delta_\nu & 3\Delta_\nu^2 & 4\Delta_\nu^3 & 5\Delta_\nu^4 \\ 0 & 0 & 2 & 6\Delta_\nu & 12\Delta_\nu^2 & 20\Delta_\nu^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} g(t_\nu) \\ 0 \\ 0 \\ g(t_{\nu+1}) \\ 0 \\ 0 \end{bmatrix}, \tag{5.41}$$

where $t_\nu \in \{0, T_1, T_1 + T_2, T_1 + T_2 + T_3, T_1 + 2T_2 + T_3\}$. Figure 5.18 shows the resulting template function $g(t)$ and its first time derivative $\dot{g}(t)$ with parameters $T_1 = T_2 = T_3 = 0.1\,sec$ and $c = 0.5$.

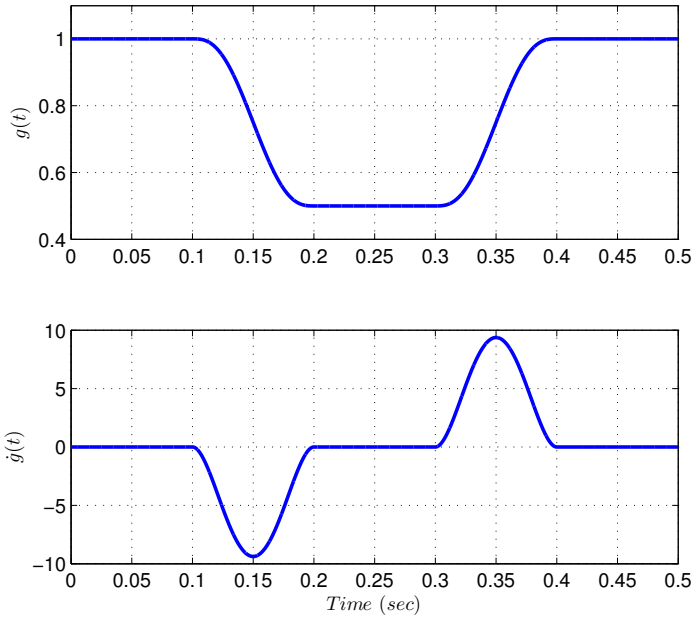


Fig. 5.18: Polynomial based template function (top) and its first time derivative (bottom)

5.4.2 Exponential Function

Another way to build the scaling function template $g(t)$ is by using exponential functions which are smooth and infinitely differentiable functions. The exponential function in this thesis is defined with a variable power n, i.e. $e^{-(\lambda t)^n}$. The variable λ is introduced to control the decay rate of the exponential function which will later be defined as a function of the time to reach the constant scaling factor T_2.
The template function $g(t)$ is constructed as a combination of several well conditioned exponential functions such that the final template function is continuous, smooth and symmetric. More precisely, the template function is defined as

$$g(t) = \begin{cases} 1, & 0 \leq t \leq T_1, \\ c + (1-c)\, e^{-(\lambda(t-T_1))^n}, & T_1 < t \leq T_1 + T_2, \\ c, & T_1 + T_2 < t \leq T_1 + T_2 + T_3, \\ c + (1-c)\, e^{\left((-1)^{(n+1)}(\lambda(t-T_1-2T_2-T_3))^n\right)}, & T_1 + T_2 + T_3 < t \leq T_1 + 2T_2 + T_3, \\ 1, & t > T_1 + 2T_2 + T_3. \end{cases} \tag{5.42}$$

The factor λ is defined so that at a time period of T_2, the exponential function reaches 99.99% of its final value,

$$\lambda = \frac{\sqrt[n]{-\ln(1e^{-4})}}{T_2}. \tag{5.43}$$

An example for a template function $g(t)$ built using exponential functions is shown in Figure 5.19 with parameters $T_1 = T_2 = T_3 = 0.1\ sec, c = 0.5$ and $n = 2$.

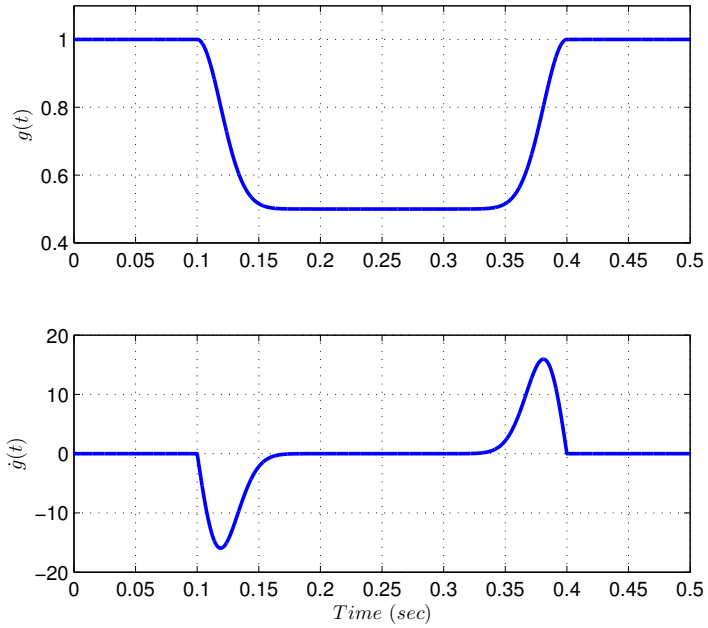


Fig. 5.19: Exponential based template function (top) and its first time derivative (bottom)

5.4.3 Sigmoidal Function

In addition to the exponential function, the sigmoidal function, also known as the logistic function, is another powerful and smooth function. Sigmoidal functions look like a stretched S-shape curves increasing or decreasing between two horizontal lines and have a single change in their curvature. The general equation of a sigmoidal function is given by

$$f\left(t\right)=\frac{L}{1+Qe^{-\lambda t}}, \tag{5.44}$$

where L represents the limiting value of the function beyond which the function cannot grow, Q is the number of time samples that the function needs to reach L and λ is a control parameter that affects the steepness of the function.
The template function $g\left(t\right)$ is constructed as a combination of multiple sigmoidal functions conditioned such that the final template function is continuous, smooth and symmetric. More precisely,

$$g\left(t\right)=\begin{cases}1, & 0\leq t\leq T_1,\\ 1+\frac{c-1}{1+e^{-\lambda\left(t-T_1-\frac{T_2}{2}\right)}}, & T_1<t\leq T_1+T_2,\\ c, & T_1+T_2<t\leq T_1+T_2+T_3,\\ c+\frac{1-c}{1+e^{-\lambda\left(t-T_1-\frac{3T_2}{2}-T_3\right)}}, & T_1+T_2+T_3<t\leq T_1+2T_2+T_3,\\ 1, & t>T_1+2T_2+T_3.\end{cases} \tag{5.45}$$

The factor λ is defined such that at a time period of T_2, the logistic function reaches 99.99% of

its final value,

$$\lambda = \frac{-2\ln\left(\frac{1}{1-1e^{-4}} - 1\right)}{T_2} \tag{5.46}$$

An example for a template function $g\,(t)$ built using sigmoidal functions is shown in Figure 5.20 with parameters $T_1 = T_2 = T_3 = 0.1\,sec$ and $c = 0.5$.

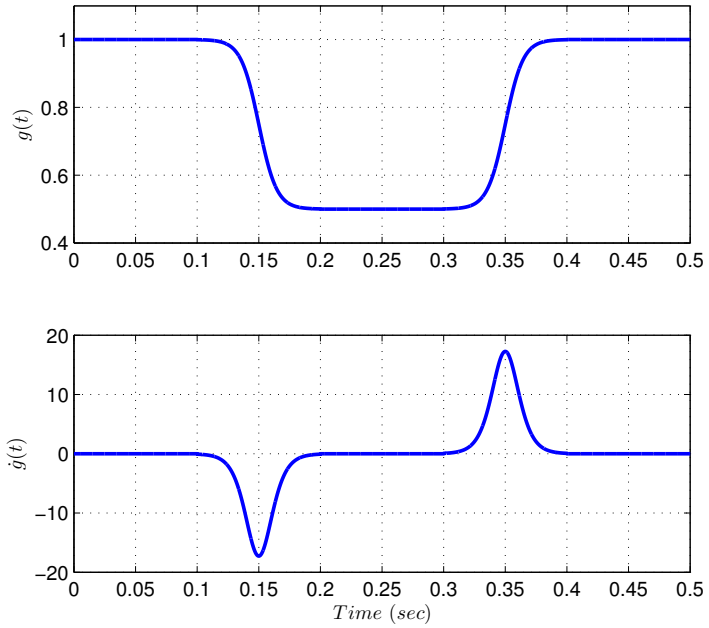


Fig. 5.20: Sigmoidal based template function (top) and its first time derivative (bottom)

To modify the frequency content of the reference trajectories, the scaling function $c\,(t)$ will be built using a scaling function template $g\,(t)$ constructed by a combination of exponential function with power $n = 2$ as defined by Equation (5.42). Figure 5.21 shows an example of a scaling function built from 12 template functions which are used to shift the frequency content of test contour III in Appendix A.

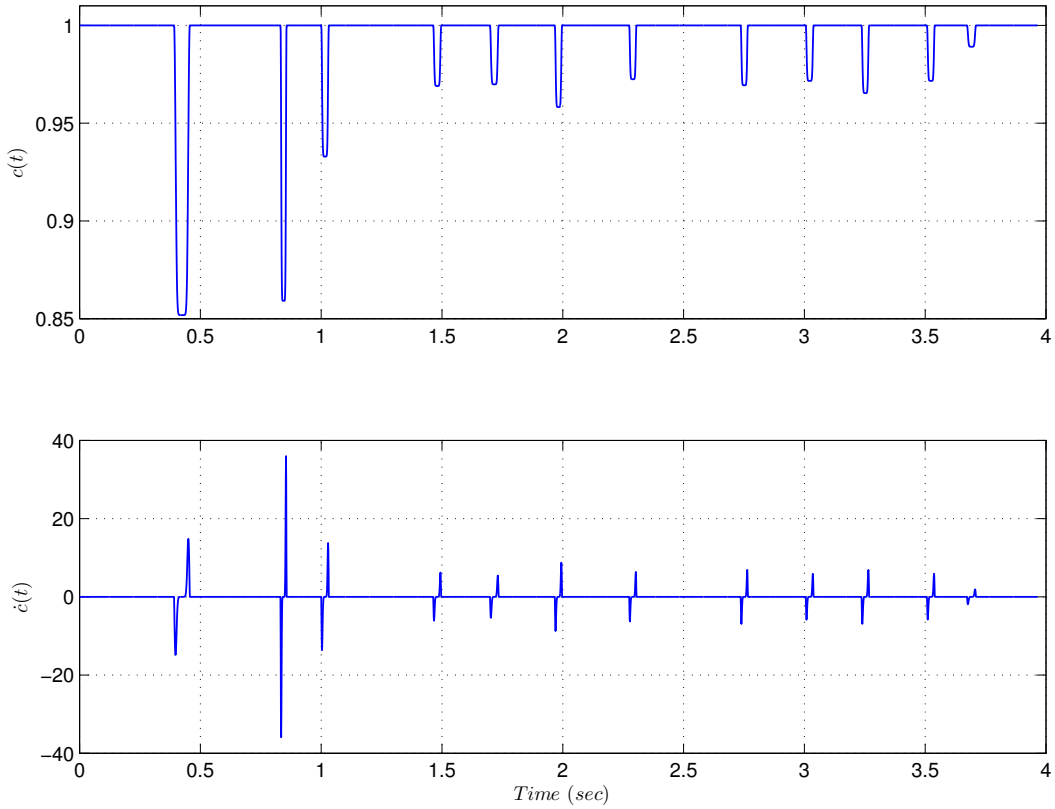


Fig. 5.21: Example of a scaling function built using 12 template functions

5.5 Tuning the Template Function Parameters

The main objective to be fulfilled in optimizing the parameters of the scaling function templates is to minimize the machine oscillations at the targeted region with minimal effect on the time optimality of the original trajectories. Several factors play a role in selecting the type of optimization routine to be used here, the most important of which will be now discussed in detail.

Objective Function: This can be any function that represents a measure for the machine oscillations at the targeted region. In this thesis, the vibrational contour error signal as defined in Chapter 4 is considered as objective function. Usually, evaluating any representative measure for the machine oscillations involves simulating the response of the different machine axes and includes nonlinear operations such as curve fitting, energy and amplitude calculation [39]. Especially for machines with a high degree of freedom, this step is computationally very demanding. Clearly, this objective function is normally a nonlinear function, cannot be defined in a closed form and is very costly to evaluate. This in turn narrows the choices of the optimization method to those who are fast, derivative-free and with a minimal number of objective function evaluations.

Parameter Interaction: The four parameters T_1, T_2, T_3 and c to be optimized are strongly

interrelated. Each change in any of the four parameters will affect the values of the others. This strong interaction requires a simultaneous optimization of the four parameters and thus imposes an additional constraint on the choice of the optimization routine.

Parameter Limits: Several technical limits constrain the applicable range of the four parameters describing the scaling function template. For instance, the constant scale factor c is not allowed to be zero as this results in no motion. The existence of such limits imposes the need for an optimization method which allows inequality and/or equality constraints.

The general problem of finding the proper parameters for each template function that minimize the machine vibrations in the targeted region can be formulated as follows:

$$\min_{\vec{x}\in\mathbb{R}^n} \mathcal{F}(\vec{x}), \qquad \text{subject to} \quad \vec{x}_l \leq \vec{x} \leq \vec{x}_u, \tag{5.47}$$

where

$\mathcal{F}(\vec{x})$ is the optimization cost function, a representative measure for the machine vibration amplitude at the targeted region,

$\vec{x}$ is a vector containing the optimization parameters, in this case $\vec{x} = [T_1, T_2, T_3, c]^T$,

$\vec{x}_l$ and $\vec{x}_u$ are vectors representing the lower and upper limits for the optimization parameters vector $\vec{x}$.

To solve the above problem, a fast, derivative-free optimization method with a small number of cost function evaluations is needed. In this study, the pattern search method with the golden section method as a local optimizer was selected to solve the problem. In addition, a bracketing algorithm based on the Rosenbrock method is developed to bracket the minima for the golden section method. The following subsections discuss in detail each of the mentioned optimization methods.

5.5.1 Pattern Search Method

The pattern search method belongs to the family of derivative–free optimization methods. In the literature there exists a variety of algorithms that address this type of optimization problems. For instant, some approximate the gradient (and possibly the Hessian) using finite differences and then use these approximate gradients within gradient based algorithms. Even though this finite-difference approach is effective in some applications, it cannot be regarded as a general–purpose technique for derivative–free optimization since the number of function evaluations required can be excessive and the approach becomes unreliable in the presence of noise [56]. Other methods construct a model for the function to be optimized based on the function values and optimize such models. These methods can give a good approximation for the minimum of the function, but it is impossible for these methods to insert limits on the optimization parameters. In contrast to the previously mentioned methods, *directional search* methods represent a better solution for the derivative–free optimization problems. In such

methods, search directions are defined and a search algorithm is implemented to search along these directions for a lower function value. Examples of this type of methods include the coordinate search, the pattern search [56] and the Rosenbrock method [57]. All these methods share the same concept, but differ in the way how they define the search direction set and the way of searching along these directions.

The pattern search method is the more general form of the coordinate search method. In such a method a certain set of search directions is defined at each iteration and the cost function $\mathcal{F}$ is evaluated at a given step length along each of these directions. These candidate points form a *frame* or *stencil* around the current iterate. If a point with a significantly lower function value is found, it is adopted as the new iterate, and the center of the frame is shifted to this new point [56]. The method cycles through the set of search directions at each iteration, and the center of the frame jumps from point to point until a satisfactory minimum is found. Different implementations for the pattern search method are available in the literature. The main difference between such implementations is in the definition of the search directions set. For example, two sets of search directions are defined in [56] as

1. Coordinate Set

$$\mathcal{D}_c = \{\pm\vec{e}_1, \pm\vec{e}_2, \ldots, \pm\vec{e}_n\}\,. \tag{5.48}$$

2. Vectors Set

$$\begin{aligned} \mathcal{D}_v &= \left\{\vec{P}_{\mathrm{i}}, \vec{P}_{n+1}\right\}, \qquad \text{for} \quad \mathrm{i} = 1, 2, \ldots, n, \\ \text{with} \qquad \vec{P}_{\mathrm{i}} &= \frac{1}{2n}\left(\vec{e} - \vec{e}_{\mathrm{i}}\right), \qquad \text{and} \qquad \vec{P}_{n+1} = \frac{1}{2n}\vec{e}, \end{aligned}$$

where $\mathcal{D}_c$ and $\mathcal{D}_v$ are the sets of search directions with $\mathbb{R}^{n\times 1} \ni \vec{e} = [1, 1, \ldots, 1]^T$, $\vec{e}_{\mathrm{i}} \in \mathbb{R}^{n\times 1}$ with $\vec{e}_1 = [1, 0, \ldots, 0]^T$, $\vec{e}_2 = [0, 1, \ldots, 0]^T$ and n is the number of optimization parameters.
The implementation as well as the literature [58] showed that both sets of search directions tend to get locked into a zigzag pattern of smaller steps as the optimization approaches the solution. To overcome this problem, Venkataraman proposed an $n + 1$ search directions set in [58], which consists of the *univariate* set, i.e. $D_c = \{\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n\}$, and one additional direction defined as the optimal search direction reached after a complete cycle by the univariate set. In addition, Venkataraman tested both positive and negative directions in the defined set and picked the one with the minimal value of objective function. This set is called the *pattern search set* and is given by

$$\mathcal{D}_p = \left\{\pm\vec{e}_1, \pm\vec{e}_2, \ldots, \pm\vec{e}_n, \pm\vec{P}_{n+1}\right\}, \qquad \text{with} \qquad \vec{P}_{n+1} = \vec{x}_n - \vec{x}_s, \tag{5.49}$$

where $\vec{x}_n$ are the optimal parameters reached after n steps in the univariate directions and $\vec{x}_s$ are the starting values of the parameters.
In this thesis, the pattern search method is customized for optimizing the parameters of the

scaling function template. The pattern search set given by Equation (5.49) is used to define the search directions with the only difference of using a normalized version of the $n+1$st direction

$$\vec{P}_{n+1} = \frac{\vec{x}_n - \vec{x}_s}{\|\vec{x}_n - \vec{x}_s\|}, \tag{5.50}$$

which proved to be more effective than using the unnormalized one. The implemented version of the pattern search method is described by Algorithm 5.1 below. In the algorithm $\vec{x}_0$ is an initial guess for the optimization parameters, N is the maximal number of cycles where a *cycle* is defined as a complete run through all search directions defined by the set of search directions $\mathcal{D}_p$, $\mathcal{F}_c$ and $\vec{x}_c$ are, respectively, the current values of the objective function and optimization parameters, ϵ_1 defines the required tolerance in the objective function and it defines the stopping criteria of the algorithm, $\vec{S}$ is the current search direction with $\vec{S} \in \mathcal{D}_p$ and α is the optimal step length to be taken along the current search direction which is specified via the local optimizer.

Algorithm: *Pattern search method.*

Given $\mathcal{F}(\cdot)$ (Optimization objective function)

Step 1. **Choose** $\vec{x}_0$, N and ϵ_1.
Set $\mathcal{F}_c = \mathcal{F}(\vec{x}_0), \quad \vec{x}_c = \vec{x}_0$

Step 2. **For** $\mathrm{k} = 1, \ldots, N$ (Cycles loop)
For $\mathrm{i} = 1, \ldots, n$ (Directions loop)
$\vec{S} = \vec{e}_{\mathrm{i}}$ (Univariate direction)
$\vec{x} = \vec{x}_c + \alpha\vec{S}$
α is determined via the local minimizer by minimizing $\mathcal{F}(\vec{x})$
If $\mathcal{F}(\vec{x}) \geq \mathcal{F}_c$
$\vec{S} = -\vec{e}_{\mathrm{i}}$ (Reverse direction)
$\vec{x} = \vec{x}_c + \alpha\vec{S}$
End if
$\mathcal{F}_c \leftarrow \mathcal{F}(\vec{x}), \quad \vec{x}_c \leftarrow \vec{x}$
If $|\mathcal{F}_c| \leq \epsilon_1$ **Quit**
End of for loop
$\vec{S} = \frac{\vec{x}_c - \vec{x}_0}{\|\vec{x}_c - \vec{x}_0\|}$ ($n+1$ step, i.e. the pattern step)
$\vec{x} = \vec{x}_c + \alpha\vec{S}$
If $\mathcal{F}(\vec{x}) \geq \mathcal{F}_c$
$\vec{S} = -\vec{S}$ (Reverse direction)
$\vec{x} = \vec{x}_c + \alpha\vec{S}$
End if
$\mathcal{F}_c \leftarrow \mathcal{F}(\vec{x}), \quad \vec{x}_c \leftarrow \vec{x}$

If $|\mathcal{F}(\vec{x}_c)| \leq \epsilon_1$ **Quit**
End of for loop

The last values of $\mathcal{F}_c$ and $\vec{x}_c$ are the optimal values of the objective function and the optimization parameters, respectively.

Algorithm 5.1.: Pattern search method

5.5.2 Golden Section Method

The optimal step length α along the current search direction $\vec{S}$ in the pattern search algorithm can be defined as a sub-optimization problem in the following form

$$\min_{\alpha \in \mathbb{R}} \mathcal{Q}(\alpha), \qquad \text{subject to} \quad \alpha_l \leq \alpha \leq \alpha_u, \tag{5.51}$$

where

$\mathcal{Q}(\alpha)$ is the optimization cost function, in this case $\mathcal{Q}(\alpha) = \mathcal{F}\left(\vec{x}_c + \alpha\vec{S}\right)$,

α is the optimization parameter,

α_l and α_u are the lower and upper limits for the optimization parameter α.

The literature offers a number of well–developed methods for solving such an optimization problem, the simplest of which is the bisection method. Although the bisection method is very simple and relatively fast, the need for evaluating the cost function twice in order to define the next step makes it not practical for real applications especially in high dimensional problems with costly to evaluate objective functions. On the other hand, the golden section method provides a better alternative for this type of problems.
The Golden Section method belongs to the same family of interval reducing methods as the bisection method. As it is the case in any interval reducing method, the golden section method reduces the interval by the same fraction with each iteration. The intervals are derived from the golden ratio, $\varphi = \frac{1+\sqrt{5}}{2}$, [59]. The method is indifferent to the shape and continuity properties of the function being minimized. Most important, the number of iterations to achieve a prescribed tolerance around the minimum $\Delta\alpha$ can be established before the iterations start and in contrast to the bisection method it requires one evaluation of the objective function in each trial [58]. Algorithm 5.2 describes the implemented version of the golden section method with reference to [60]. In the algorithm α_l and α_u are respectively the lower and upper bounds of the step length α, $\vec{S}$ is the optimization direction, $\vec{x}_c$ is the current values of optimization parameters, δ_α is the interval reduction factor which is derived from the golden ratio $\delta_\alpha = \frac{-1+\sqrt{5}}{2}$, N is the number of iterations required to achieve a predefined tolerance interval $\Delta\alpha$ and $\mathcal{Q}(\cdot)$ is the cost function to be minimized.

Algorithm: *Golden section method.*

Given $\mathcal{Q}(\cdot)$ (Optimization objective function)
$\vec{S}$ (Optimization direction)
$\vec{x}_c$ (Current values of optimization parameters)

Step 1. **Define** α_l, α_u (Lower and upper bounds on α)
$\Delta\alpha$ (Tolerance around the minimum)

Step 2. **Evaluate** $\delta_\alpha = \frac{-1+\sqrt{5}}{2} = 0.6180$ (Interval reduction factor)
$N = \frac{\ln\left(\frac{\Delta\alpha}{\alpha_u - \alpha_l}\right)}{\ln(\delta_\alpha)}$ (Number of iterations)
$\alpha_1 = \delta_\alpha \alpha_l + (1-\delta_\alpha)\,\alpha_u, \quad \mathcal{Q}_1 = \mathcal{Q}(\alpha_1) = \mathcal{F}\left(\vec{x}_c + \alpha_1 \vec{S}\right)$
$\alpha_2 = (1-\delta_\alpha)\,\alpha_l + \delta_\alpha \alpha_u, \quad \mathcal{Q}_2 = \mathcal{Q}(\alpha_2) = \mathcal{F}\left(\vec{x}_c + \alpha_2 \vec{S}\right)$

Step 3. **For** $\mathrm{i} = 1, \ldots, N$
If $\mathcal{Q}_1 > \mathcal{Q}_2$
$\alpha_l \leftarrow \alpha_1, \quad \alpha_1 \leftarrow \alpha_2, \quad \mathcal{Q}_1 \leftarrow \mathcal{Q}_2$
$\alpha_2 = (1-\delta_\alpha)\,\alpha_l + \delta_\alpha \alpha_u, \quad \mathcal{Q}_2 = \mathcal{Q}(\alpha_2) = \mathcal{F}\left(\vec{x}_c + \alpha_2 \vec{S}\right)$
Else
$\alpha_u \leftarrow \alpha_2, \quad \alpha_2 \leftarrow \alpha_1, \quad \mathcal{Q}_2 \leftarrow \mathcal{Q}_1$
$\alpha_1 = \delta_\alpha \alpha_l + (1-\delta_\alpha)\,\alpha_u, \quad \mathcal{Q}_1 = \mathcal{Q}(\alpha_1) = \mathcal{F}\left(\vec{x}_c + \alpha_1 \vec{S}\right)$
End if
End For Loop

Algorithm 5.2.: Golden section method algorithm

In the above algorithm, since the number of iterations was know in advance no stopping criterion is needed. Also in each iteration in **Step 3** the objective function is evaluated only once which extensively reduces the number of objective function evaluations.

5.5.3 Bracketing Method

The golden section method, just as any interval reducing method, requires a lower and upper bounds for the optimization parameter within which a single minimum is guaranteed. The process of finding the lower and upper bounds is known as the *bracketing phase* of the optimization process [56]. The lower bound can be simply chosen as $\alpha_l = 0$ since it refers to values at the current golden section iteration. In contrast, there is no easy way available to select the upper bound α_u. In selecting α_u, one must guarantee the existence of a minimum in the selected interval $[\alpha_l, \alpha_u]$. Obviously, what is needed is to step downhill bypassing the minimum, but the question to be answered is how big the step should be. Venkataraman suggested a

generic scanning procedure in [58], in which a simple stepwise search routine is implemented in order to find the upper bound. The routine is based on defining an interval for α. Starting at the lower bound α_l, the interval is doubled until three points are found such that the minimum is bracketed between them. Venkataraman's algorithm is simple, but it suffers from two main drawbacks. First, the algorithm highly depends on the initial interval. Second, the method lacks the ability to impose limits on the optimization parameters which is an essential requirement in our case. To overcome these difficulties, a new bracketing algorithm is developed to find the upper bound of α. The developed algorithm employs some ideas from the *Rosenbrock* optimization algorithm [57]. The implemented algorithm is shown in Algorithm 5.3 below. $\mathcal{F}(\cdot)$ is the optimization objective function, $\vec{x}_c$ contains the current values of the optimization parameters, $\vec{x}_l$ and $\vec{x}_u$ are the lower and upper limits of the optimization parameters vector, $\alpha_{u,0}$ is an initial guess for the step size upper bound, η_s and η_f are respectively the step size factors for the success and fail cases and SF is the success/fail indicator with $SF = -1$ as initial value, $SF = 1$ indicates success and $SF = 0$ indicates failure. Success here is defined to mean that the new value of the objective function after the current trail $\mathcal{F}_{trial}$ is lower than the old value $\mathcal{F}_1$ and the optimization parameters are within their limits.

Algorithm: *Bracketing algorithm.*

Given $\mathcal{F}(\cdot)$ (Optimization objective function)
$\vec{S}$ (Optimization direction)
$\vec{x}_c$ (Current values of optimization parameters))
$\vec{x}_l, \vec{x}_u$ (Lower and upper values for the optimization parameters)

Step 1. **Choose** $\alpha_{u,0} = 0.5$ (Initial guess for the step size upper bound)
$\eta_s = 3$ (Step length factor in success case, $\eta_s > 1$)
$\eta_f = -0.5$ (Step length factor in fail case, $-1 < \eta_f < 0$)

Set $SF = -1$ (Success/fail indicator: $0 =$ success, $1 =$ fail, $-1 =$ start)
$\alpha_u = \alpha_{u,0}$
$\mathcal{F}_1 = \mathcal{F}(\vec{x}_c)$

Step 2. **While** $SF \neq 0$
$\vec{x}_{trial} = \vec{x}_c + \alpha_u \times \vec{S}$
$\mathcal{F}_{trial} = \mathcal{F}(\vec{x}_{trial})$
If $\mathcal{F}_{trial} \leq \mathcal{F}_1$ and $\vec{x}_l < \vec{x}_{trial} < \vec{x}_u$
$\alpha_u = \alpha_u \times \eta_s$
$\vec{x}_c = \vec{x}_{trial}$
$\mathcal{F}_1 = \mathcal{F}_{trial}$
If $SF = -1$
$SF = 1$

End if
Else
$\alpha_u = \alpha_u \times \eta_f$
If $SF = 1$
$\alpha_u = \frac{\alpha_u}{\eta_f}$
$SF = 0$
End if
End if
End While

α_u at the end of the algorithm is the optimal upper bound of the step length.

Algorithm 5.3.: Bracketing algorithm

All methods discussed above were programmed using MATLAB. The test results proved the effectivity and speed of the developed routines in finding the optimal parameters of the template function. Several examples and results are given in Chapter 6.

6

Simulation and Experimental Results

In this chapter, the work in Chapters 4 and 5 will be tested using MATLAB simulations and an actual test rig setup. The tests are made using all test contours mentioned in Appendix A, however, only results from test contour III will be shown in here since this contour contains all different elements which can be found in real applications and the results are much clearer compared to other test contours.

6.1 Simulation Results

For the MATLAB simulations, a two-axis CNC machine tool is considered where each axis is modeled as a two mass-spring-damper system as shown in Figure 2.8. The control structure used to regulate each axis is shown in Figure 2.13. The mechanical system parameters were selected as follow:

Motor inertia: $J_M = 5 \times 10^{-3}\,\mathrm{kg\,m^2}$

Load inertia: $J_L = 5 \times 10^{-3}\,\mathrm{kg\,m^2}$

Torsional spring coefficient: $K = 123.37\,\mathrm{Nm/rad}$

Torsional damping coefficient: $D = 0.063\,\mathrm{Nm\,s/rad}$

The parameters of the control loops tuned according to the method discussed in Chapter 2 and the corresponding sampling times were defined as

IPO sampling time: $T_{sIPO} = 2\,\mathrm{ms}$

Position loop sampling time: $T_{s\theta} = 2\,\mathrm{ms}$

Velocity loop sampling time: $T_{sn} = 125\,\mu\mathrm{s}$

Equivalent delay time of the current control loop: $T_{Ei} = 200\,\mu\mathrm{s}$

Position loop proportional controller gain: $K_v = 1.5\,\frac{\mathrm{m/min}}{\mathrm{mm}}$

Velocity loop proportional controller gain: $K_{pn} = 6\,\mathrm{Nm\,s/rad}$

Velocity loop integration time constant: $T_{nn} = 15\,\mathrm{ms}$

Figure 6.1 shows the frequency response of the simulation model from input torque M_M to motor angular velocity $\dot{\theta}_M$ and Figure 6.2 shows the frequency response of the simulation model mechanics from motor angular velocity $\dot{\theta}_M$ to load angular velocity $\dot{\theta}_L$.

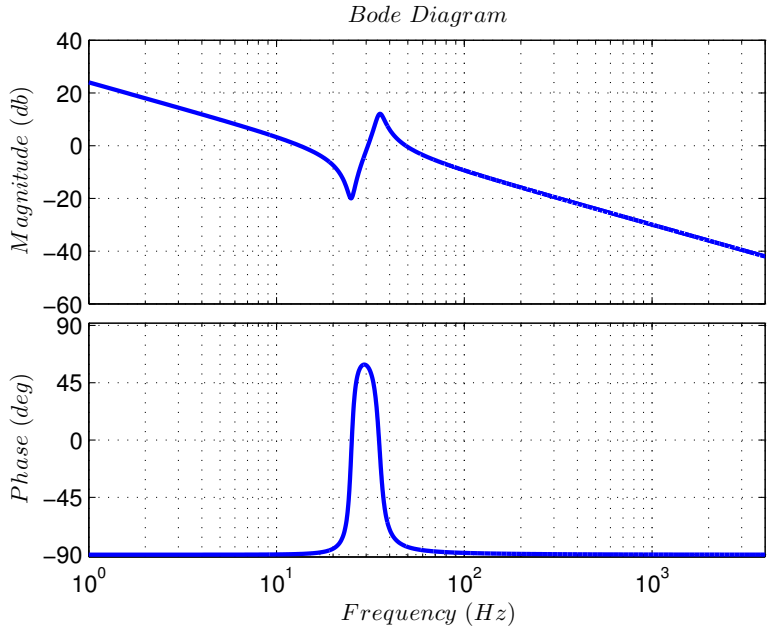


Fig. 6.1: Frequency response of the simulation model

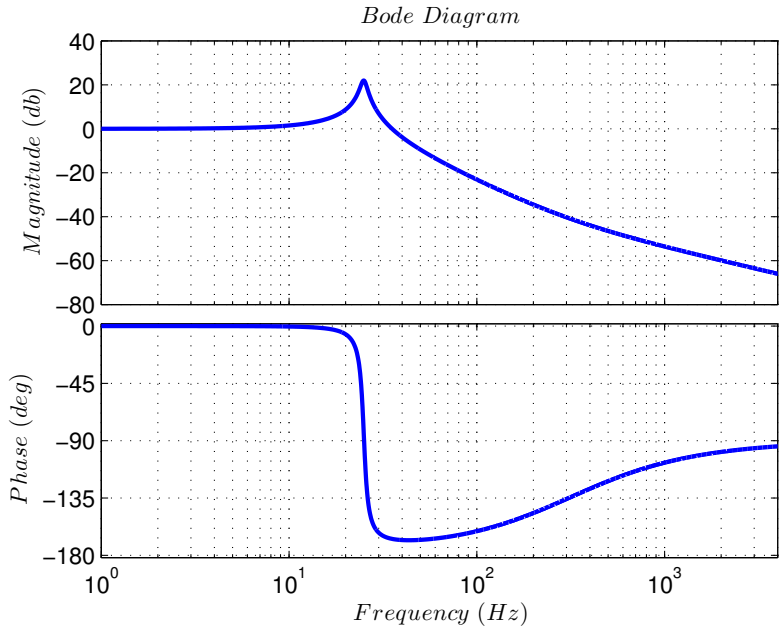


Fig. 6.2: Frequency response of the simulation model mechanics

The simulation model frequency response in Figure 6.1 shows that the system has a single pole and zero with frequencies $F_p = 35.2\,\mathrm{Hz}$ and $F_z = 25\,\mathrm{Hz}$, respectively. The frequency of the system zero is the same as the frequency of the pole in the system mechanics which represents the critical frequency of the machine axis $F_c = F_z = 25\,\mathrm{Hz}$. Also, from the frequency response

of the simulation model mechanics in Figure 6.2, we can identify the damping ratio of the system to be $\zeta = 0.04$.
Figures 6.3 and 6.4 respectively show the frequency response of the velocity and position closed loops of the simulation model.

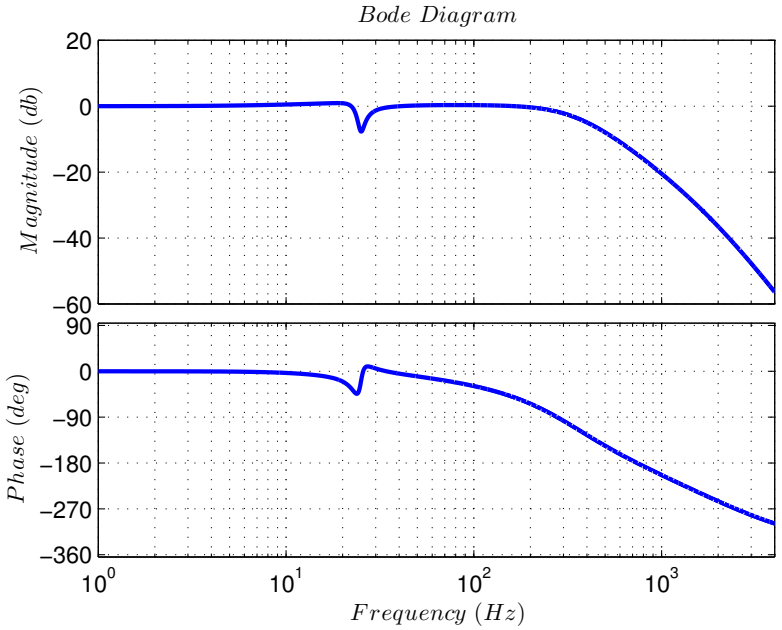


Fig. 6.3: Frequency response of the simulation model velocity closed loop

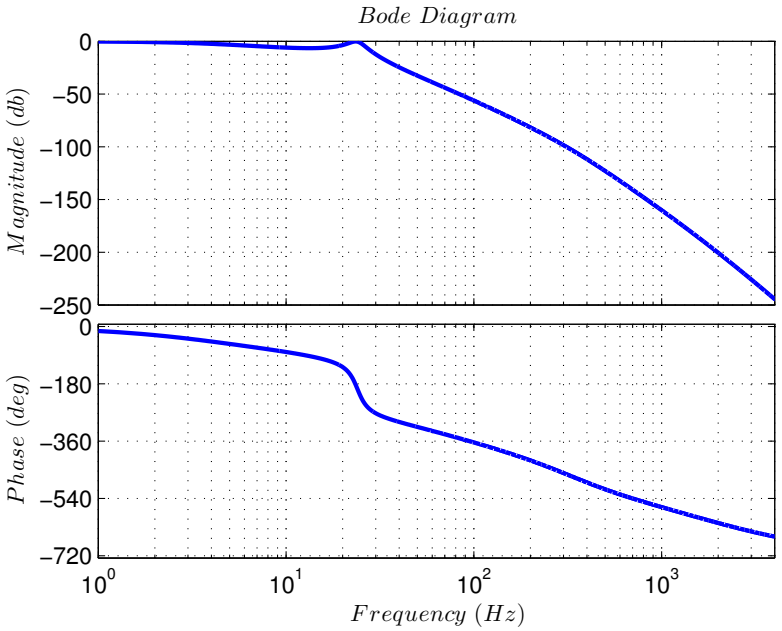


Fig. 6.4: Frequency response of the simulation model position closed loop

A step response from the motor and load side are shown in Figure 6.5. The input signal is a jerk limited step command as shown in Figure A.1.

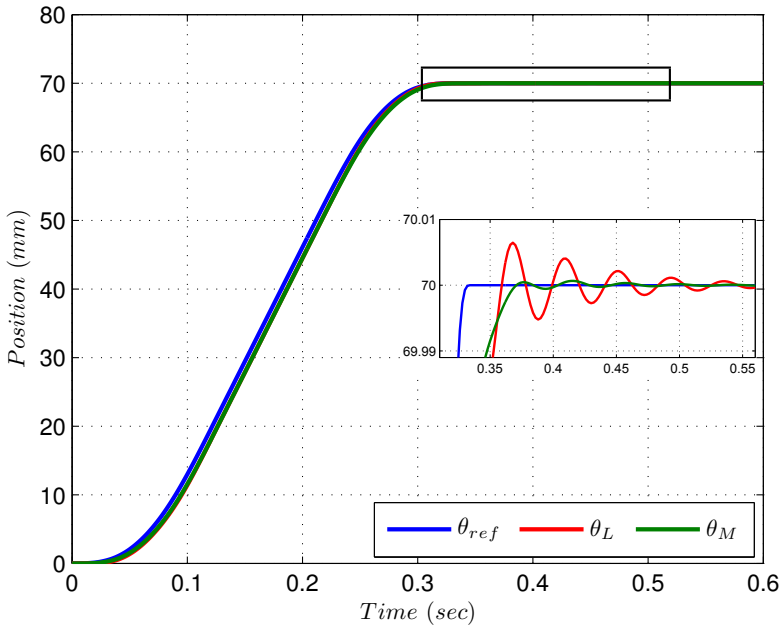


Fig. 6.5: Load and motor response to a jerk limited step command

The response of the simulation model to test contour III is shown in Figure 6.6 where the four marked regions (A,B,C and D) are four selected zoomed views as illustrated in Figure 6.7.

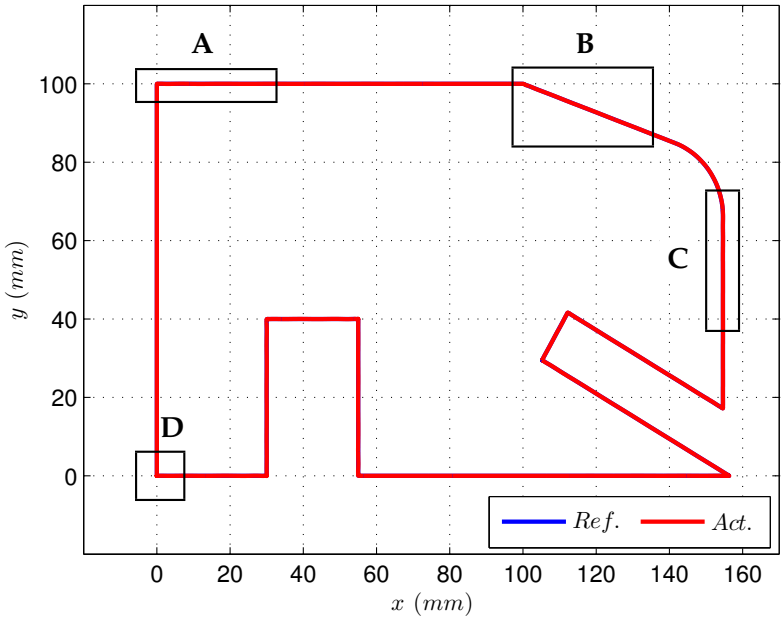


Fig. 6.6: Response of simulation model to test contour III

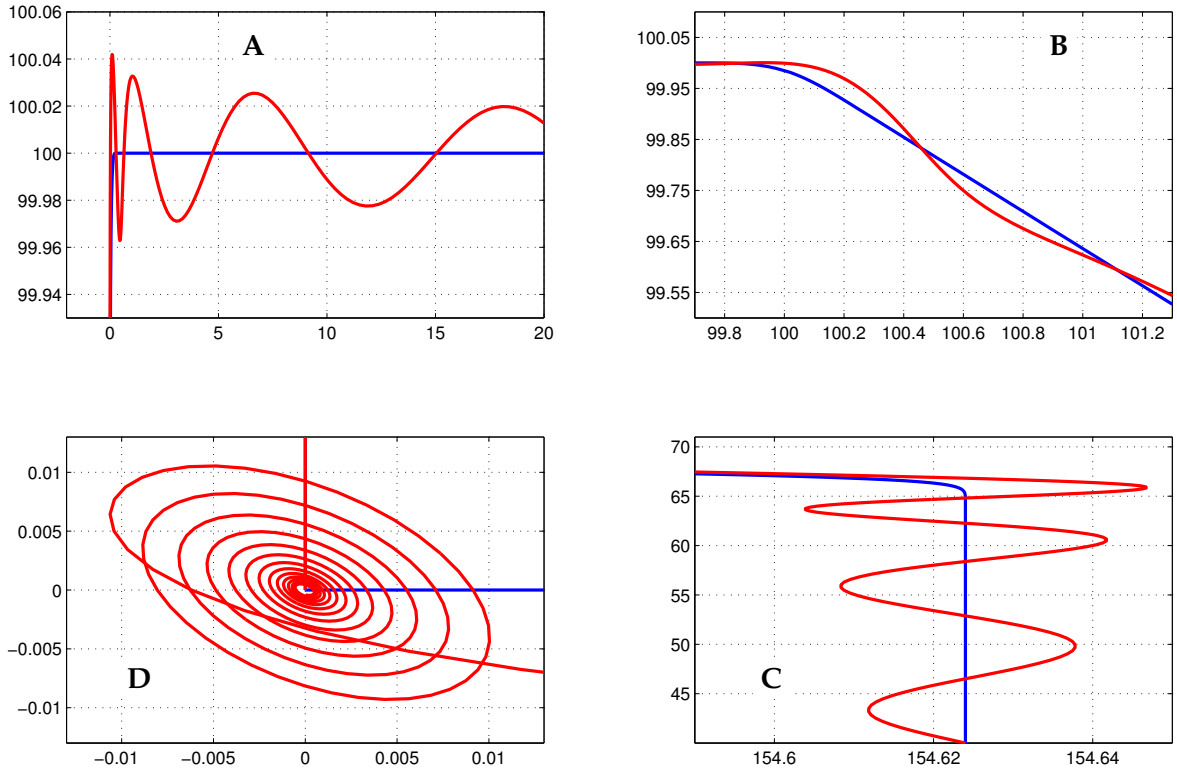


Fig. 6.7: Zoomed views in the response of simulation model to test contour III

The resulting vibrational contour error signal computed as described in Chapter 4 is shown in Figure 6.8

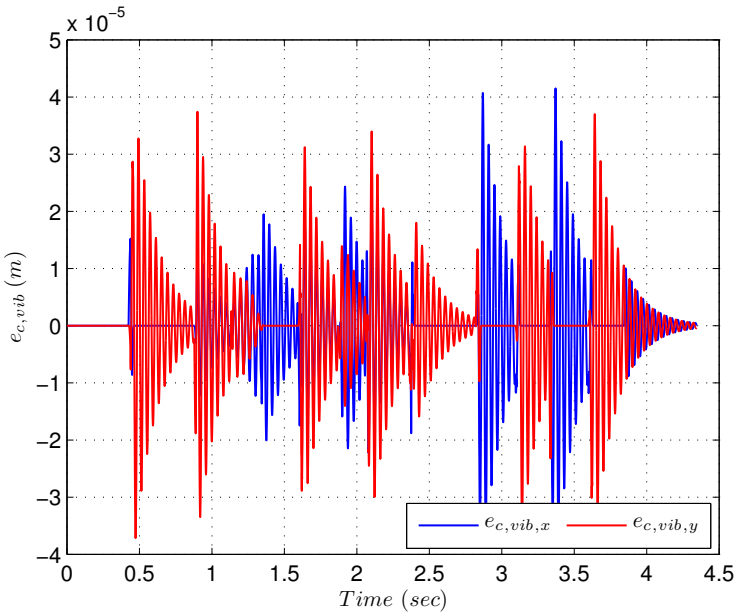


Fig. 6.8: Vibrational contour error signal

Using the wavelet approach defined in Section 5.2, 12 critical oscillation regions were identified as illustrated in Figure 6.9.

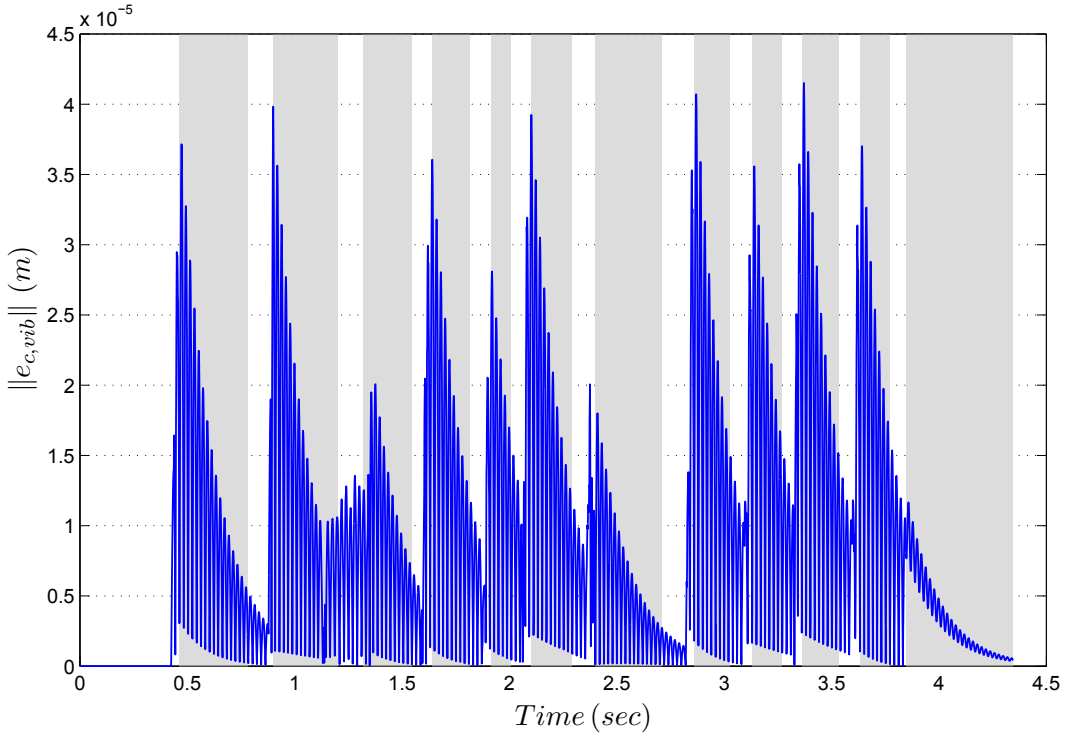


Fig. 6.9: Critical oscillation regions in test contour III

Figure 6.10 shows the resulting scaling function $c\,(t)$ built according to the method discussed in Section 5.4 with scaling function template $g\,(t)$ built with an exponential of power two, i.e. $n = 2$. The parameters of the template functions were tuned as needed by each individual oscillation region using the optimization routines discussed in Section 5.5.

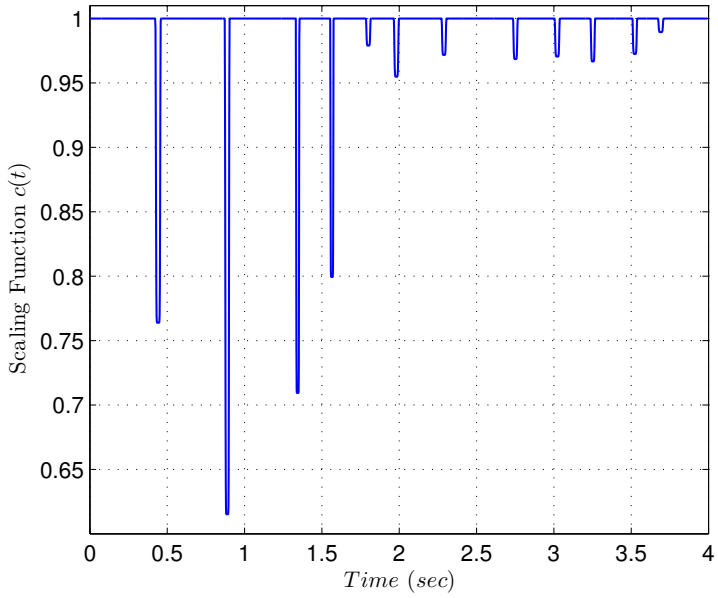


Fig. 6.10: Scaling function used to attenuate the oscillations of test contour III

The simulation model response to test contour III before and after time-scaling is shown in Figure 6.11 with the four zoomed views (A,B,C and D) illustrated in Figure 6.12. The scaling

function was designed to reduce the system oscillations of at most $1\,\mu\mathrm{m}$ which is successfully achieved as illustrated by the results. Additional simulation results using different test contours are shown in Appendix B.

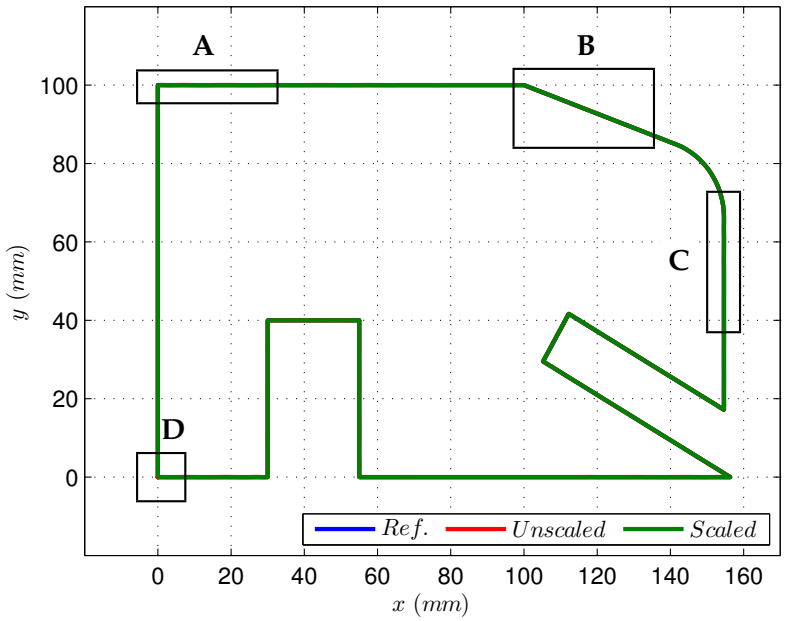


Fig. 6.11: Response of simulation model to test contour III after time-scaling

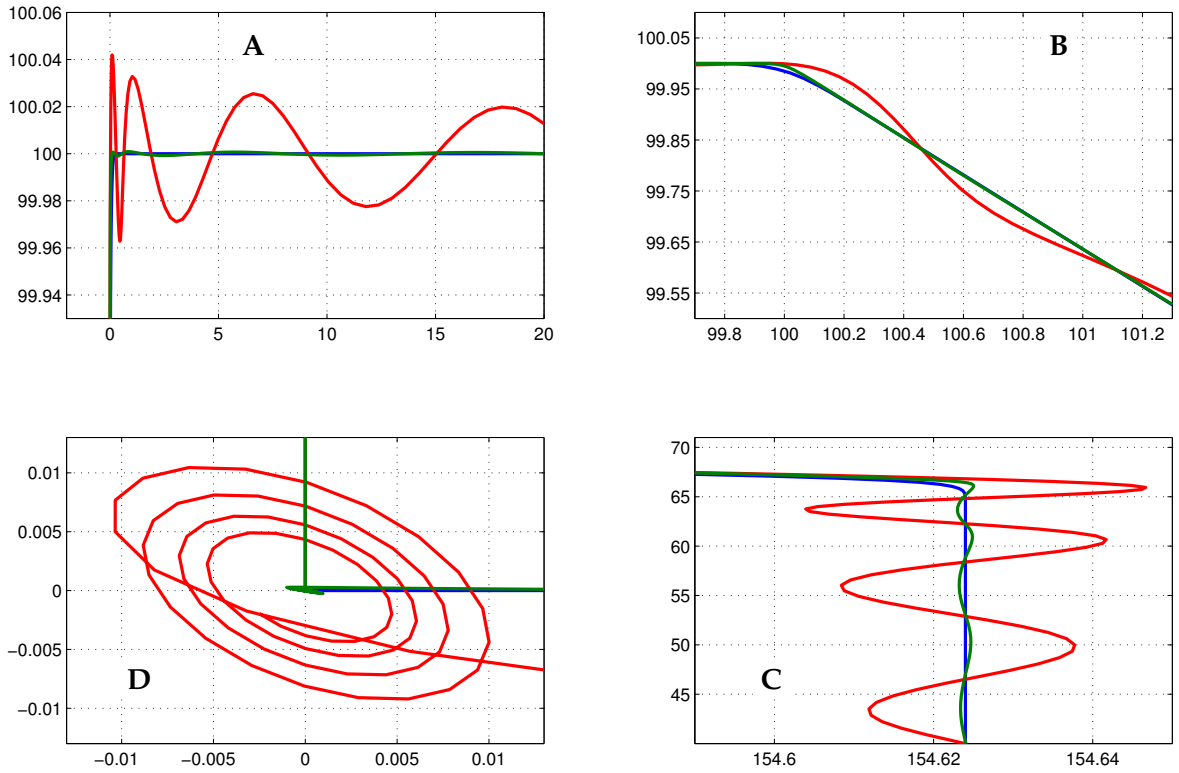


Fig. 6.12: Zoomed views in the response of simulation model to test contour III after time-scaling

6.2 Experimental Results

In addition to the MATLAB simulations, the theoretical findings of this work were tested on a test rig setup that represents the feed drive system of a CNC machine tool. The test rig consists of two AC motors connected by an elastic rod as shown in Figure 6.13.

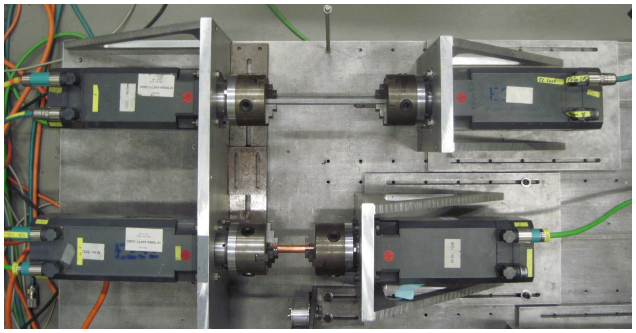


Fig. 6.13: Test rig setup

An open architecture SINUMERIK 840D unit is used to control the two motors in the test rig. The actual position and velocity of the system and the frequency responses of the control loops were obtained via internal traces. The time and frequency optimal trajectories were designed externally using MATLAB and fed directly to the corresponding control loops in the system as reference values. The frequency response of the test rig and its mechanics as well as the corresponding frequency responses of the two mass-spring-damper approximation are shown in Figures 6.14 and 6.15, respectively. The system has one pole at frequency $F_p = 81.5\,\mathrm{Hz}$ and one zero at frequency $F_z = 57.6\,\mathrm{Hz}$.

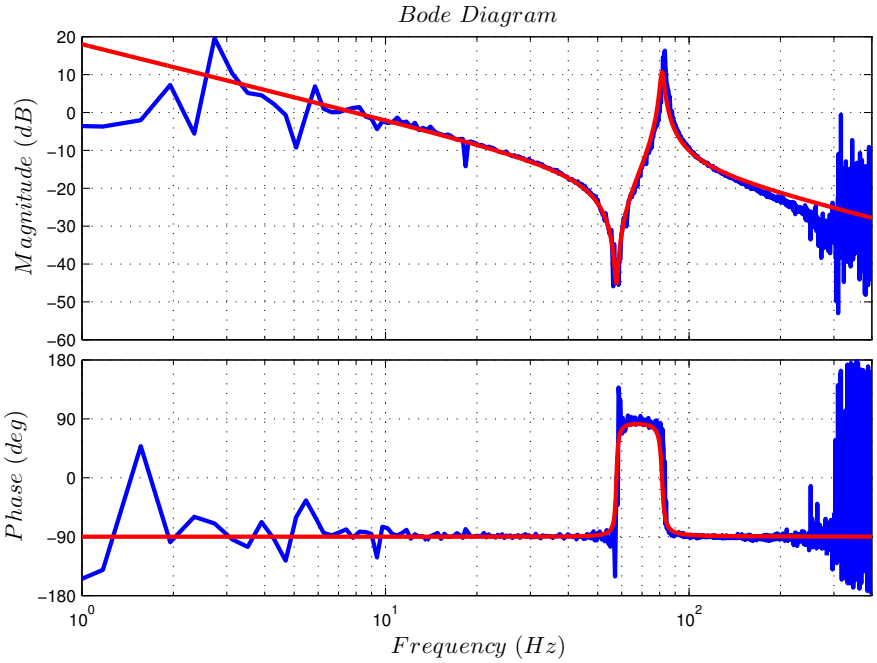


Fig. 6.14: Frequency response of the test rig: Actual measurement (blue), approximation via two mass-spring-damper system (red)

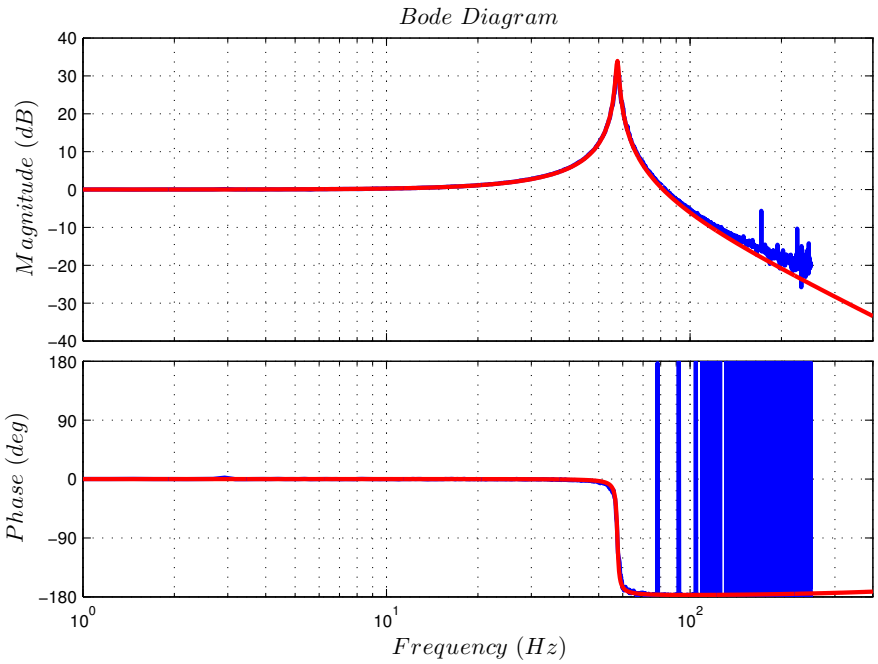


Fig. 6.15: Frequency response of the test rig mechanics: Actual measurement (blue), approximation via two mass-spring-damper system (red)

The frequency response of the velocity and position closed loops of the test rig are shown in Figures 6.16 and 6.17, respectively. As the bode diagrams illustrate, the test rig has been tuned to maximum speed of response which will, indeed, result in high amplitude oscillations in its response in the time domain.

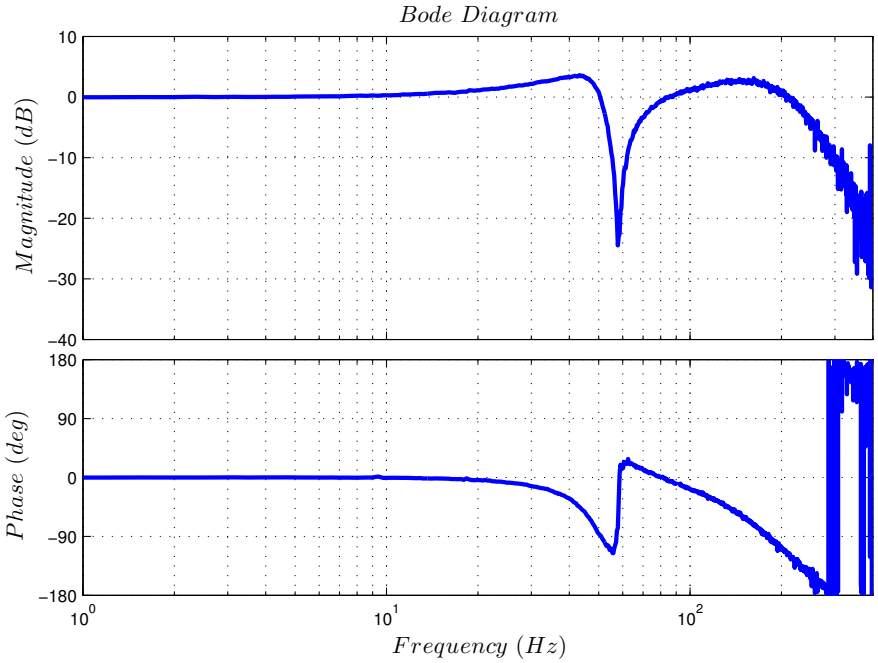


Fig. 6.16: Frequency response of the test rig velocity closed loop

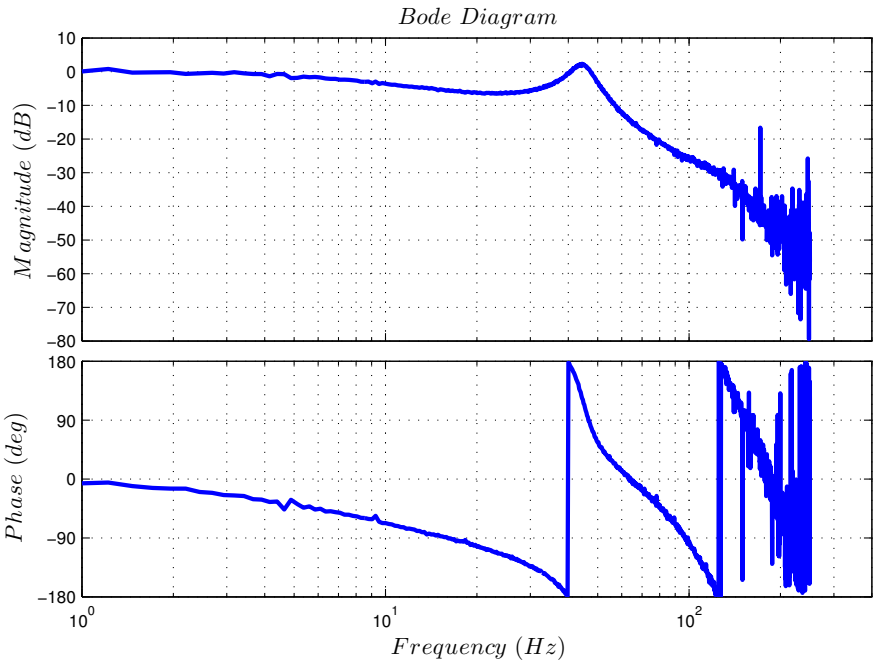


Fig. 6.17: Frequency response of the test rig position closed loop

The mechanical system parameters of the test rig were identified as follow:

Motor inertia: $J_M = 10^{-2}\,\mathrm{kg\,m^2}$

Load inertia: $J_L = 10^{-2}\,\mathrm{kg\,m^2}$

Torsional spring coefficient: $K = 1310\,\mathrm{Nm/rad}$

Torsional damping coefficient: $D = 0.7238\,\mathrm{Nm\,s/rad}$

The parameters of the control loops and the corresponding sampling times were defined as

IPO sampling time: $T_{sIPO} = 4\,\mathrm{ms}$

Position loop sampling time: $T_{s\theta} = 2\,\mathrm{ms}$

Velocity loop sampling time: $T_{sn} = 125\,\mu\mathrm{s}$

Equivalent delay time of the current control loop: $T_{Ei} = 200\,\mu\mathrm{s}$

Position loop proportional controller gain: $K_v = 3\,\frac{\mathrm{m/min}}{\mathrm{mm}}$

Velocity loop proportional controller gain: $K_{pn} = 6\,\mathrm{Nm\,s/rad}$

Velocity loop integration time constant: $T_{nn} = 15\,\mathrm{ms}$

A step response from the test rig is shown in Figure 6.18. The input signal is a jerk limited step command as shown in Figure A.1.

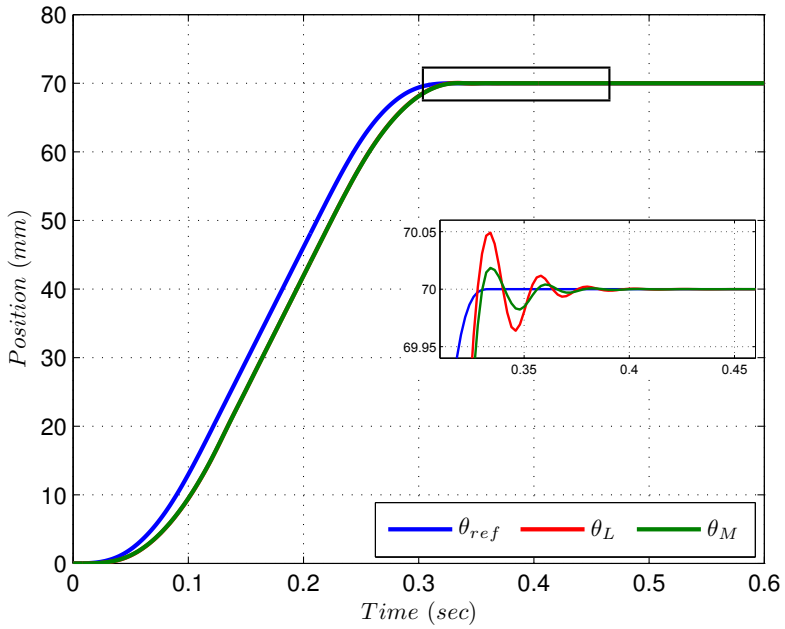


Fig. 6.18: Test rig response to a jerk limited step command

In order to design the time and frequency optimal motion profiles, the system response was approximated with a second order underdamped system with natural frequency $F_n = 40.4\,\text{Hz}$ and damping ratio $\zeta = 0.1$. Figure 6.19 shows the actual measurements of the frequency response of the test rig position closed loop from reference values to load position as well as the frequency response of the second order system.

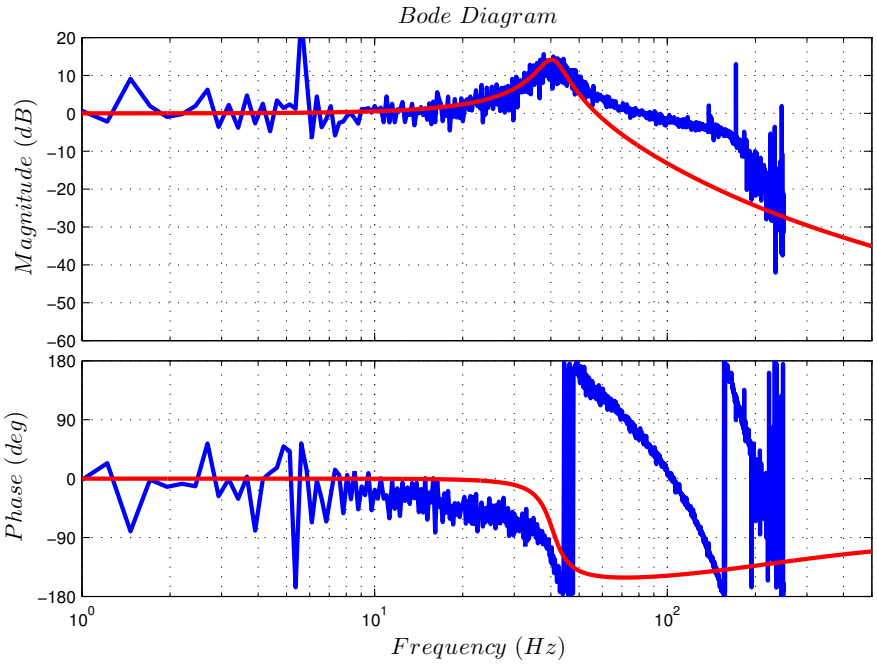


Fig. 6.19: Frequency response of the test rig position closed loop from reference values to load position

For a proof of concept, the test rig was tested using a time and frequency optimal step com-

mand designed using the method described in Chapter 5. Figure 6.20 shows the response of the test rig to a time optimal jerk limited step command and a time and frequency optimal step command. As the results illustrate, our method is capable of driving the test rig with limited or no vibrations without excessively increasing the machining time.

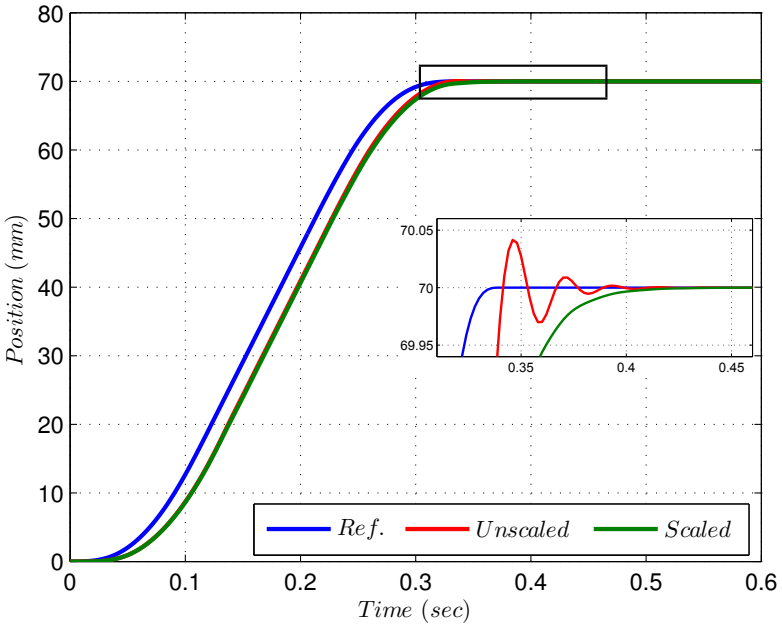


Fig. 6.20: Test rig response to a jerk limited step command before and after time-scaling

7

Conclusion and Future Research Directions

7.1 Conclusion

Realizing machining processes with high speed and high precision is a rising demand for achieving high productivity and thus reducing the production costs and increasing the production quality. The state of the art literature on this subject and specifically in the context of contour following applications is inconclusive on several areas, which this thesis sought to cover. The study was set out to design time and frequency optimal motion profiles for elastic multi-axis CNC machine tools moving along given contours. The time optimality in this context means the design of jerk limited motion profiles which result in the minimal possible machining time under the restrictions of the machine axis capabilities, whereas the frequency optimality means the design of motion profiles which are free of the machine resonance frequencies and thus result in a zero or little mechanical oscillations. In this sense, the time and frequency optimal motion profiles in this thesis refer to those profiles which provide the best compromise between the machining time and the amount of mechanical oscillations.

The thesis satisfied both requirements by starting with time optimal motion profiles which consider the capabilities of the machine axes only and reshaping them in such a way that their frequency content at the machine's critical frequencies is adequately suppressed so that the resulting machine oscillations are below a given limit.

The time optimal motion profiles were designed to be jerk limited and were computed using the state of the art solution provided by the trajectory planning function, the so called *look ahead* function in Siemens controllers.

To suppress the frequency content of the generated time optimal motion profiles at the resonance frequencies of the machine, a well localized shaping method based on time-scaling technique was presented. The shaping process was done on the path trajectories rather than the axis trajectories in order to guarantee that the changes are imposed on all moving axes

simultaneously without altering the original contours. To realize the time-scaling process, first a measure called the *vibrational contour error signal* which quantifies the occurring vibrations in the response of the machine was developed. This measure describes the machine oscillatory behavior on the contour level rather than on the axis level as it is the case in the conventional vibration measures. As a second step, the critical regions in the time optimal trajectories were identified using a specifically designed wavelet-based approach. In this approach a new wavelet called the *balanced impulse response wavelet* was developed which outperforms the existing ones in representing the system behavior. To localize the shaping process, the scaling function which is used to shape the time optimal motion profiles was designed to be highly dynamic. The methodology used in designing the scaling function depends on defining a sufficiently general scaling function template. The template function was parameterized by a finite number of parameters which were optimally tuned via optimization routines as required by the individual critical region. The complete scaling function was then built stepwise by shifting suitably tuned copies of the template function to cover all critical regions in the motion profiles. This methodology has turned out to be modular and flexible.

Results from simulations as well as practical tests have shown that the method is capable of suppressing the machine vibrations to practically any reasonable limit while maintaining time optimal behavior in the designed motion profiles. The method also guarantees that the originally given contours remain unchanged by changing only the dynamics of the motion profiles, i.e. velocity, acceleration and jerks, but not the position.
Compared to the existing state of the art solutions, the presented method outperforms them in terms of maintaining faster machining, perfect contouring and flexibility.

7.2 Future Research Directions

Evaluation on a Real Machine Setup

The research presented in this thesis has focused on developing a solid theoretical base for creating time and frequency optimal motion profiles for CNC machine tools. The implementation and evaluation of the developed methods were done using Matlab and a simple test rig setup for a proof of concept as discussed in Chapter 6. Although the results have shown in principle that the methods are capable of suppressing the machine vibration regardless of the nature of the machine under test, it would be, however, beneficial to implement and evaluate the developed methods on an actual machine setup using 2D or 3D contours.

Online Implementation

In addition to the real machine evaluation, it would be most interesting to extend the presented methods into an online version where motion profiles are designed online to be time and frequency optimal.The extension into an online version provides big challenges, especially with the limited computational time the machines have and the optimization loops in the developed methods.

A

Test Contours

The following motion profiles are designed using the time optimal jerk limited trajectory planning algorithm of Siemens CNC Sinumerik. The axis limitations were defined as

Maximum axes velocity: $\hat{v} = 20\,\mathrm{m/\,min}$

Maximum axes acceleration: $\hat{a} = 4\,\mathrm{m/s^2}$

Maximum axes jerk: $\hat{j} = 100\,\mathrm{m/s^3}$

Step Trajectory:

A conventional jerk limited step trajectory designed for test purposes.

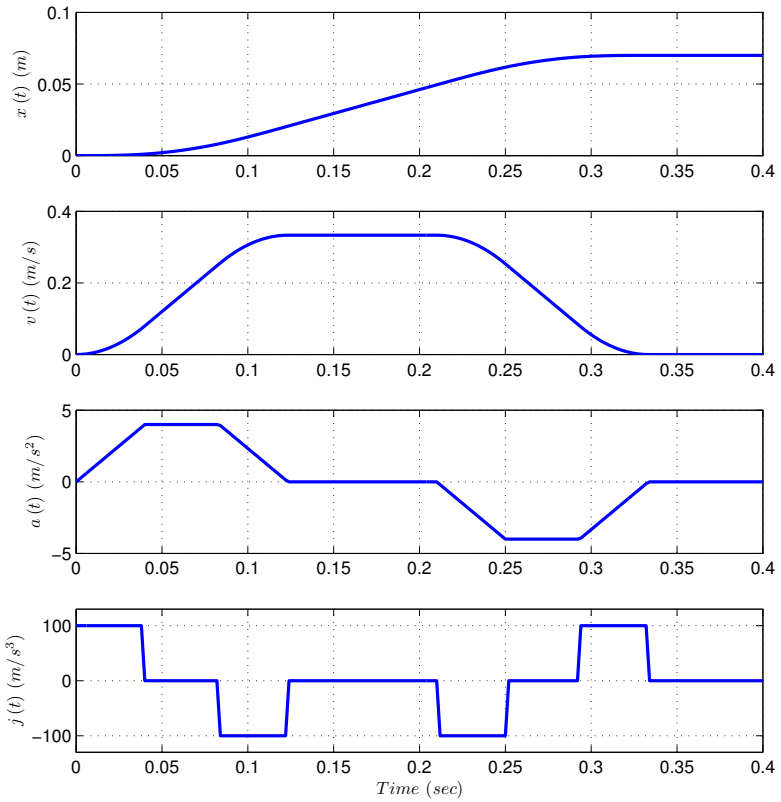


Fig. A.1: Jerk limited step motion profiles

Test Contour I:

2D jerk limited square contour designed for test purposes.

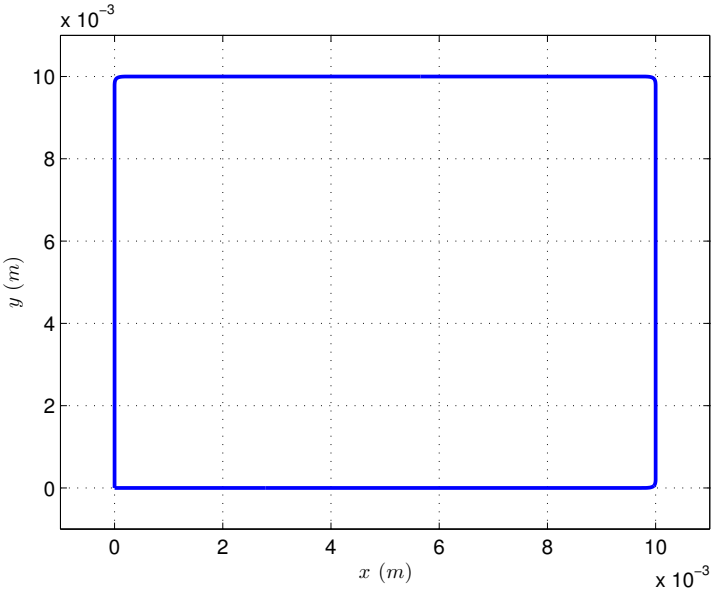


Fig. A.2: Test contour I

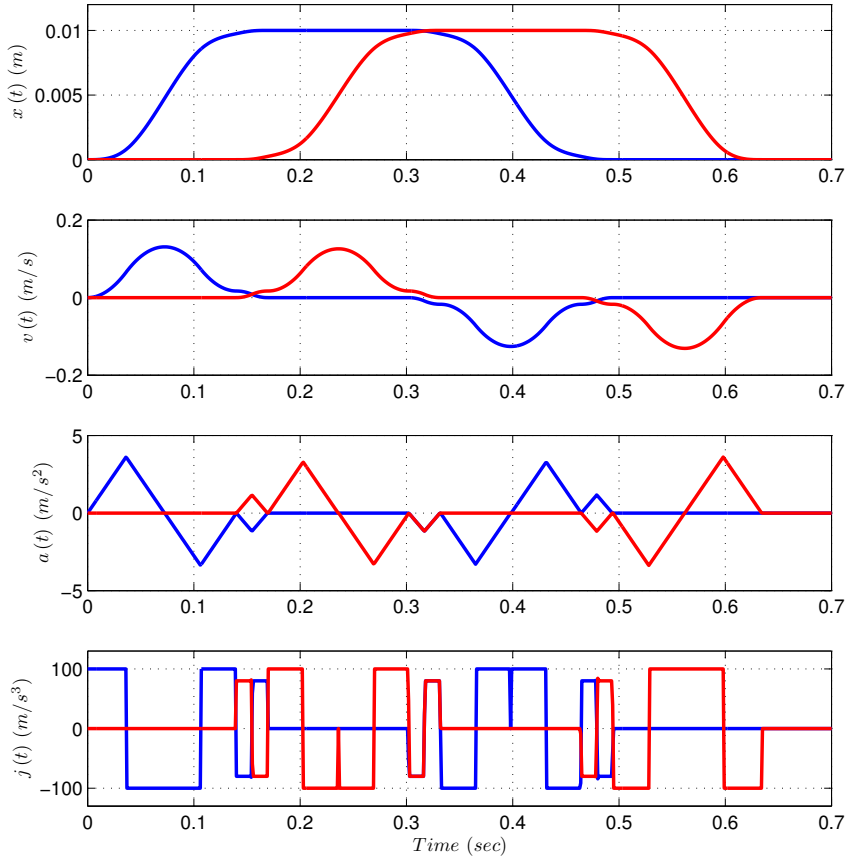


Fig. A.3: Motion profiles for $x - axis$ (blue) and $y - axis$ (red) for test contour I

Test Contour II:

2D jerk limited contour with different slopes designed for test purposes.

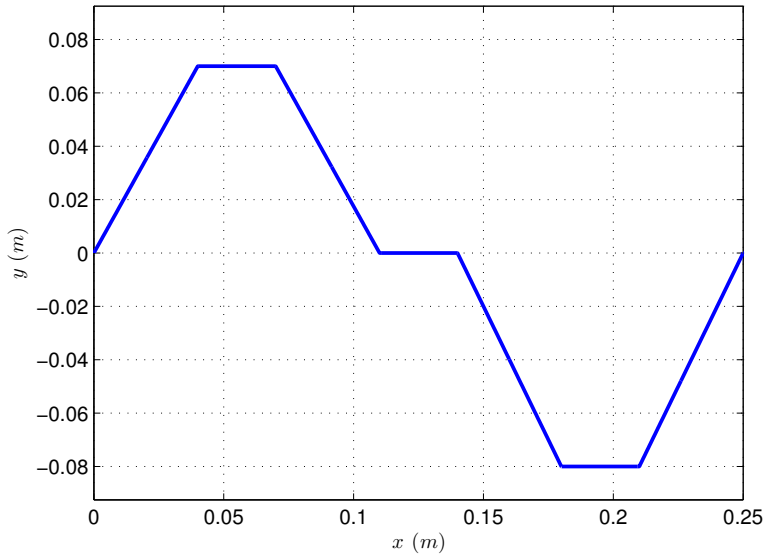


Fig. A.4: Test contour II

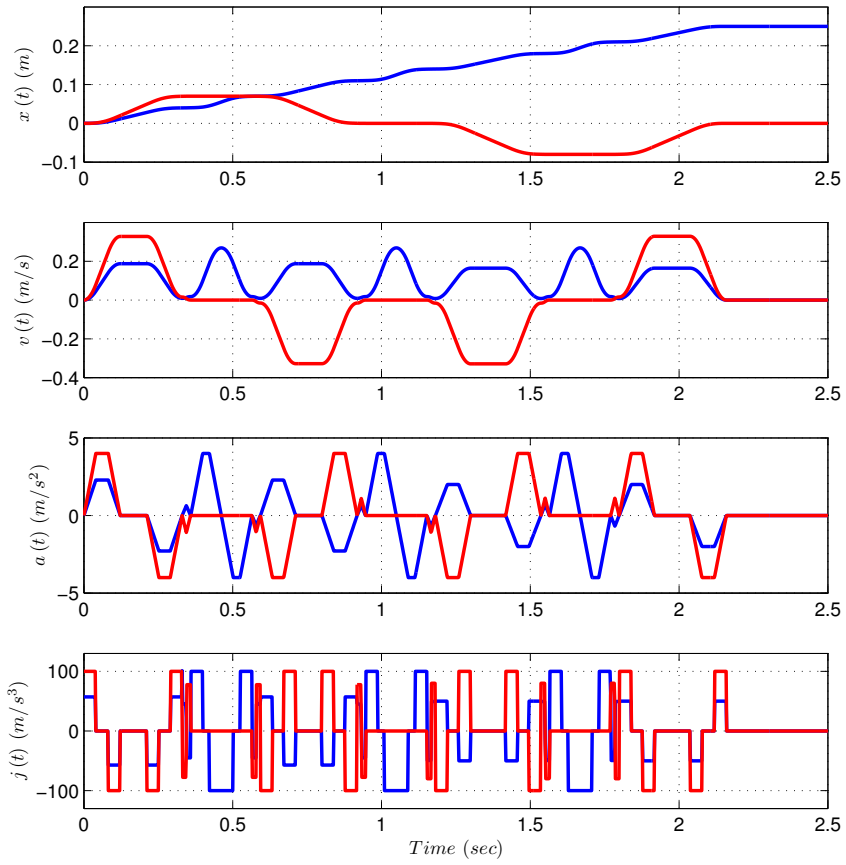


Fig. A.5: Motion profiles for $x - axis$ (blue) and $y - axis$ (red) for test contour II

Test Contour III:

2D jerk limited contour with different contour elements designed for test purposes.

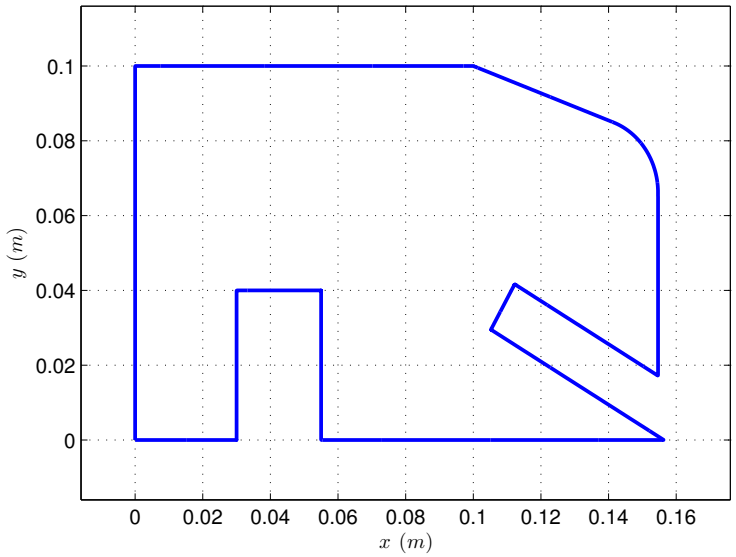


Fig. A.6: Test contour III

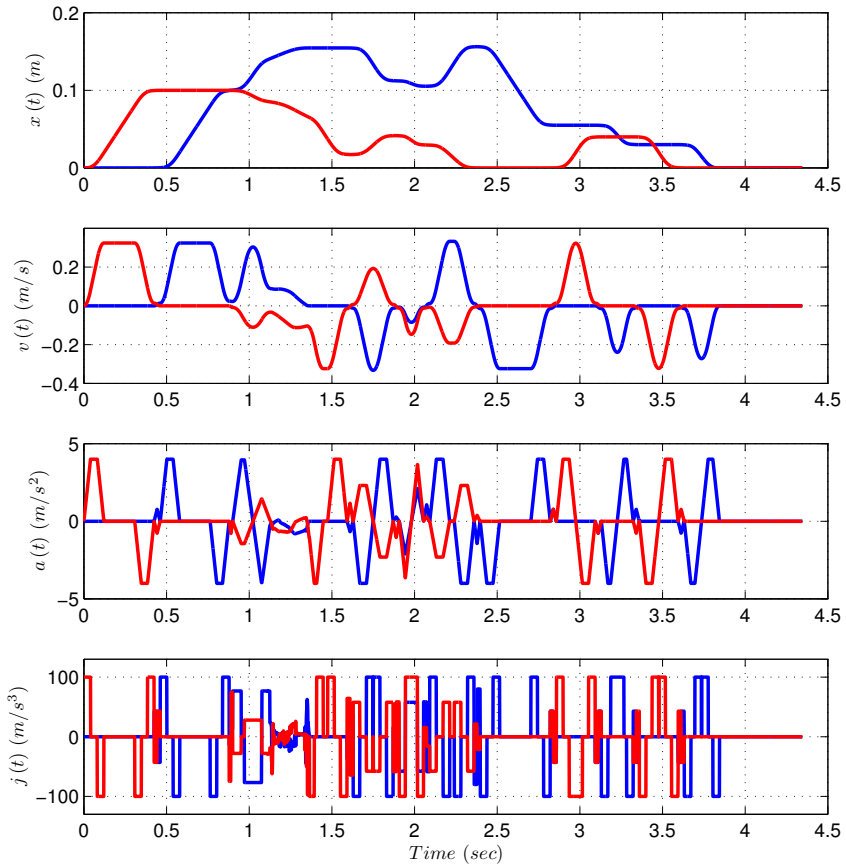


Fig. A.7: Motion profiles for $x - axis$ (blue) and $y - axis$ (red) for the test contour III

Test Contour IV:

A piece of an industrial free form part program.

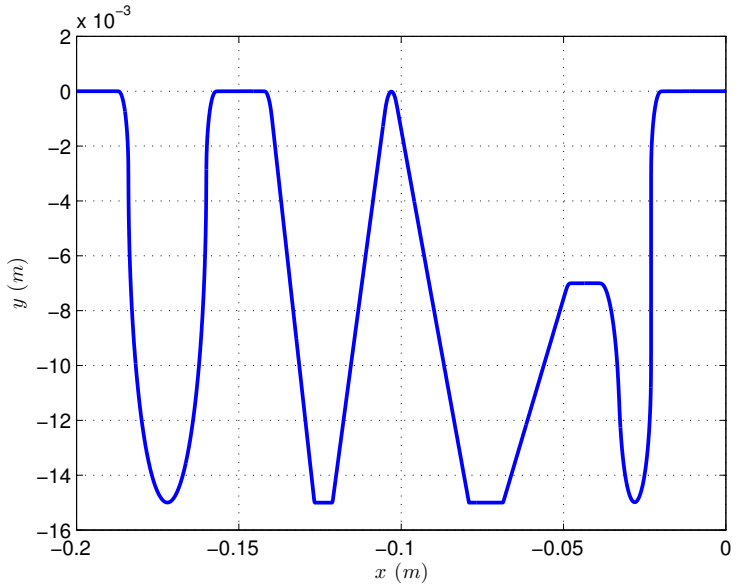


Fig. A.8: Test contour IV

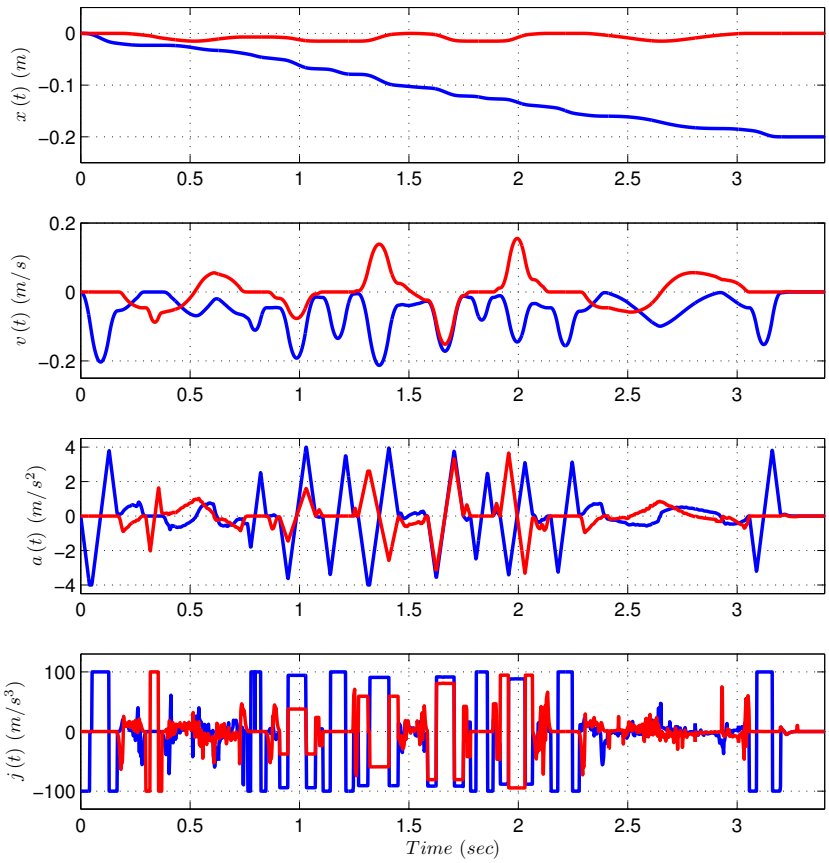


Fig. A.9: Motion profiles for $x - axis$ (blue) and $y - axis$ (red) for the test contour IV

B

Additional Simulation Results

In the following, results from testing our method for designing time and frequency optimal motion profiles using MATLAB simulations will be given. The parameters of the simulation model and the corresponding parameters of the control loops were defined as in Chapter 6.

Results From Step Trajectory

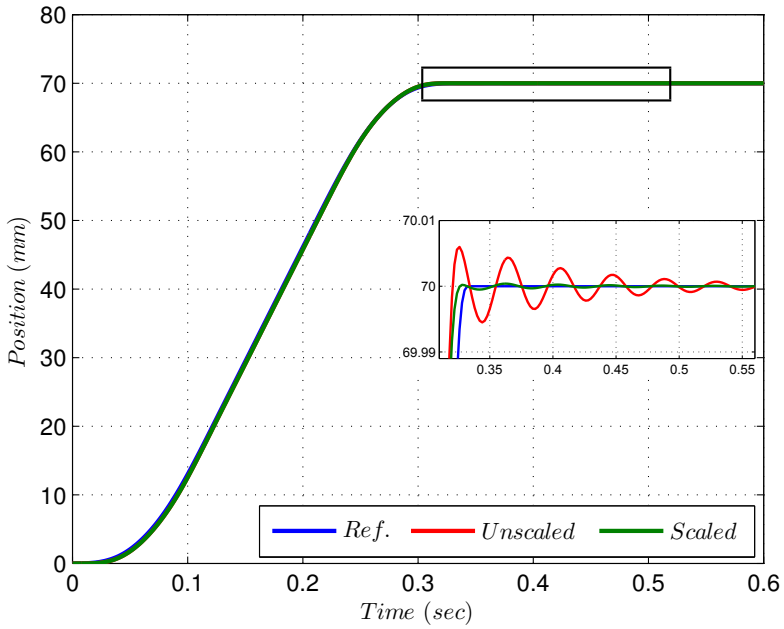


Fig. B.1: Response of simulation model to jerk limited step command before and after time-scaling

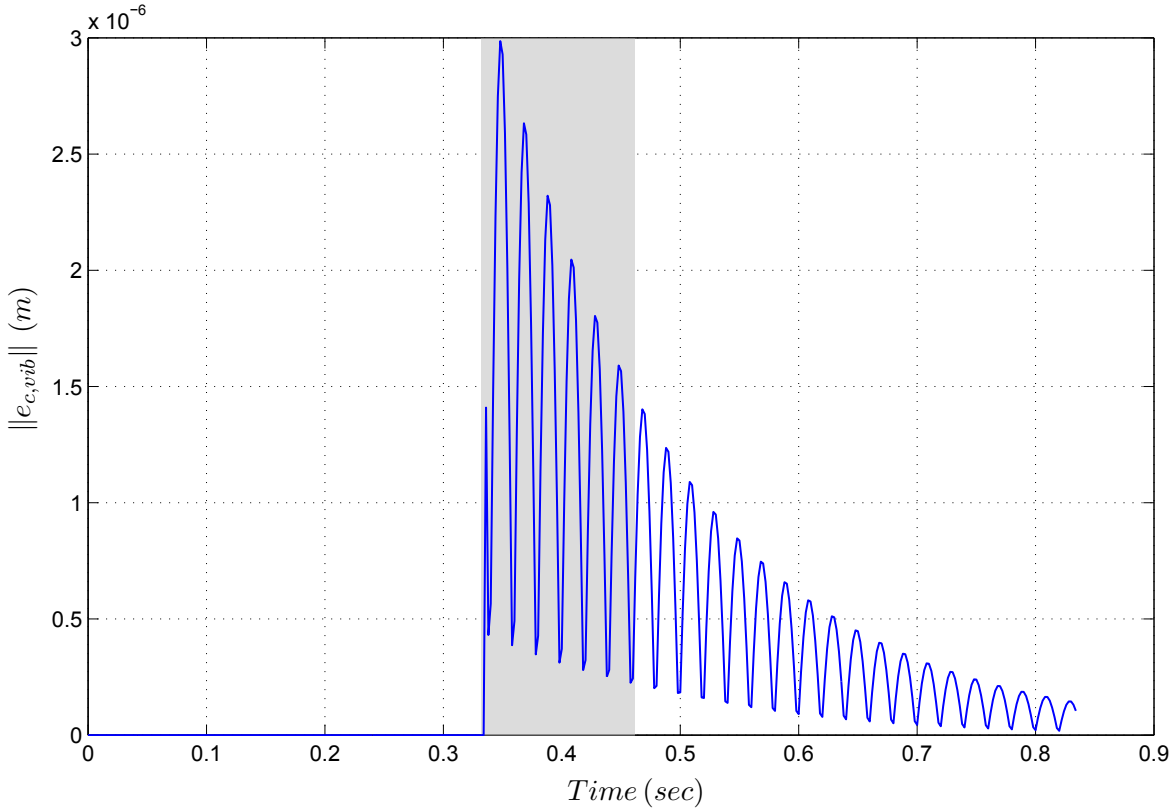


Fig. B.2: Critical oscillation regions for the jerk limited step command

Results From Test Contour I

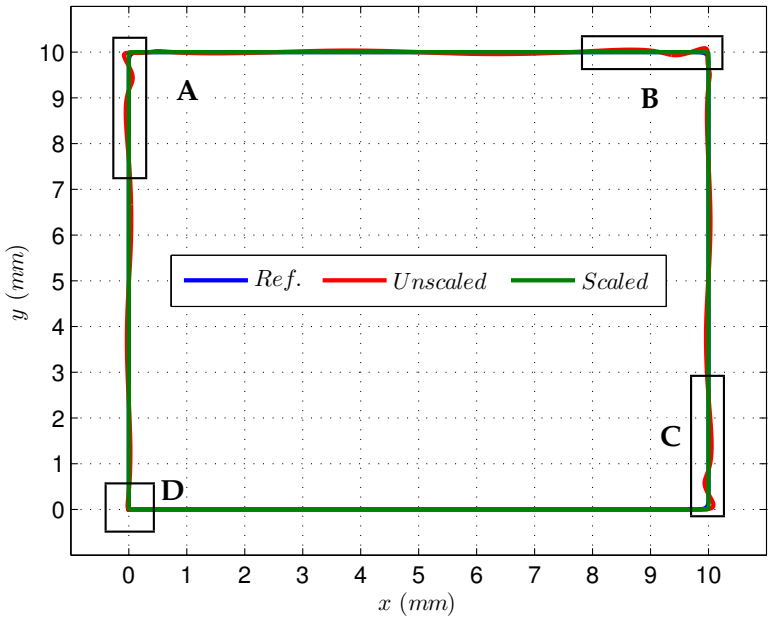


Fig. B.3: Response of simulation model to test contour I before and after time-scaling

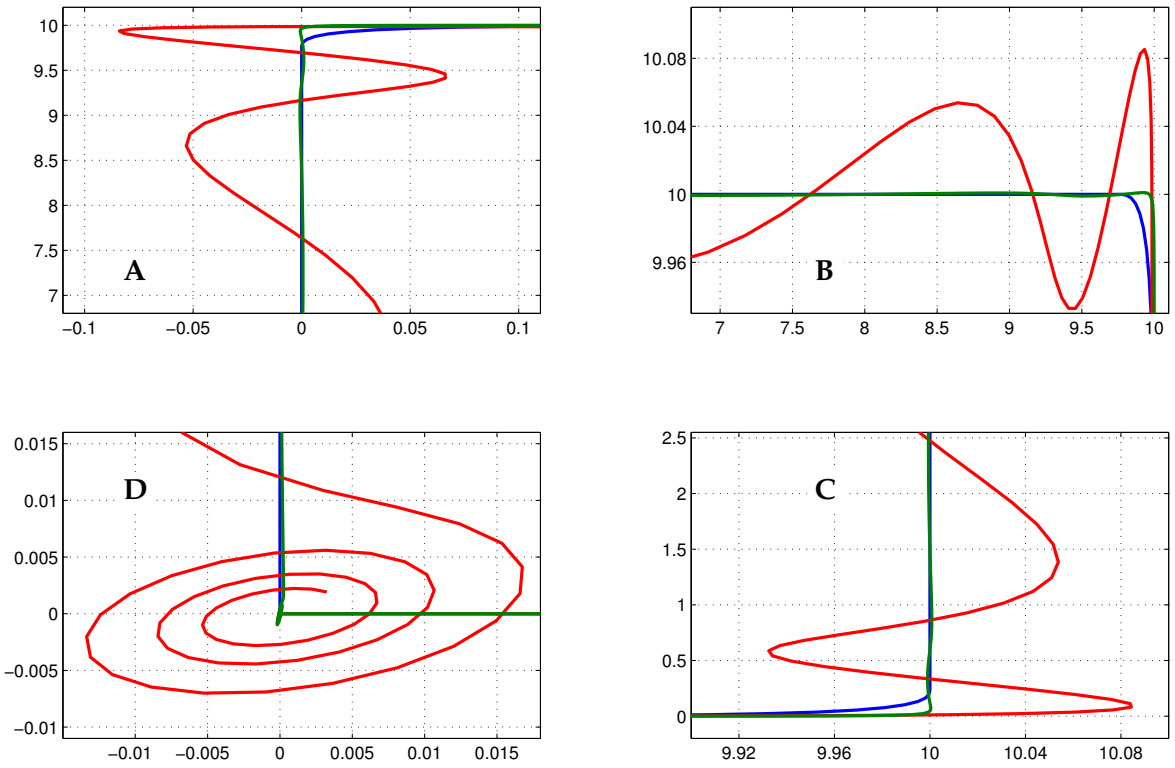


Fig. B.4: Zoomed views in the response of simulation model to test contour I before and after time-scaling

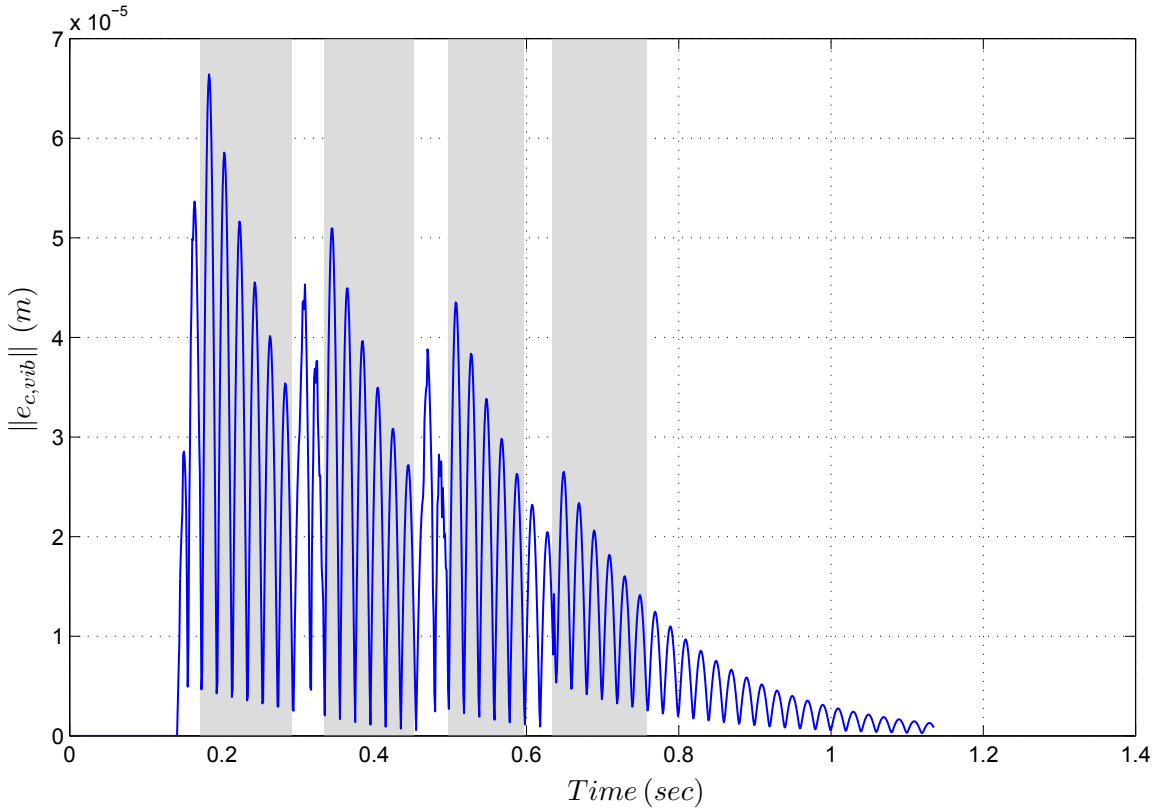


Fig. B.5: Critical oscillation regions for test contour I

Results From Test Contour II

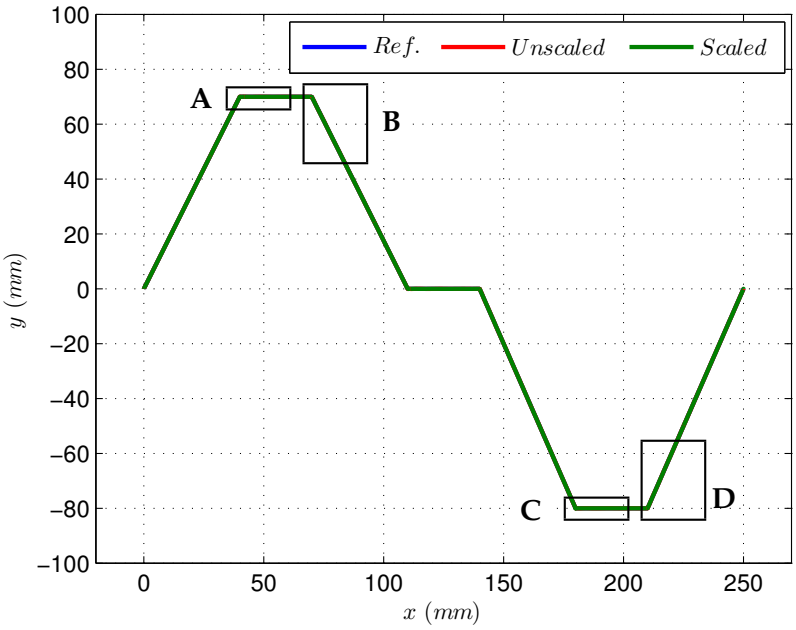


Fig. B.6: Response of simulation model to test contour II before and after time-scaling

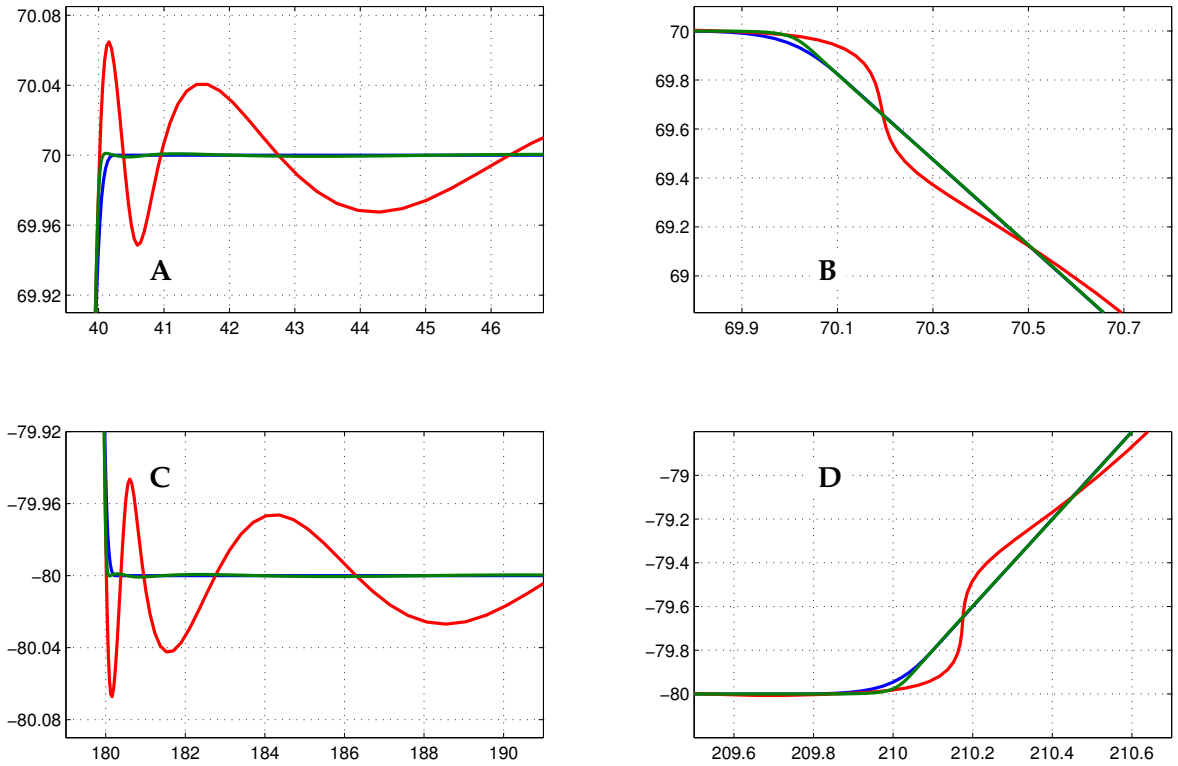


Fig. B.7: Zoomed views in the response of simulation model to test contour II before and after time-scaling

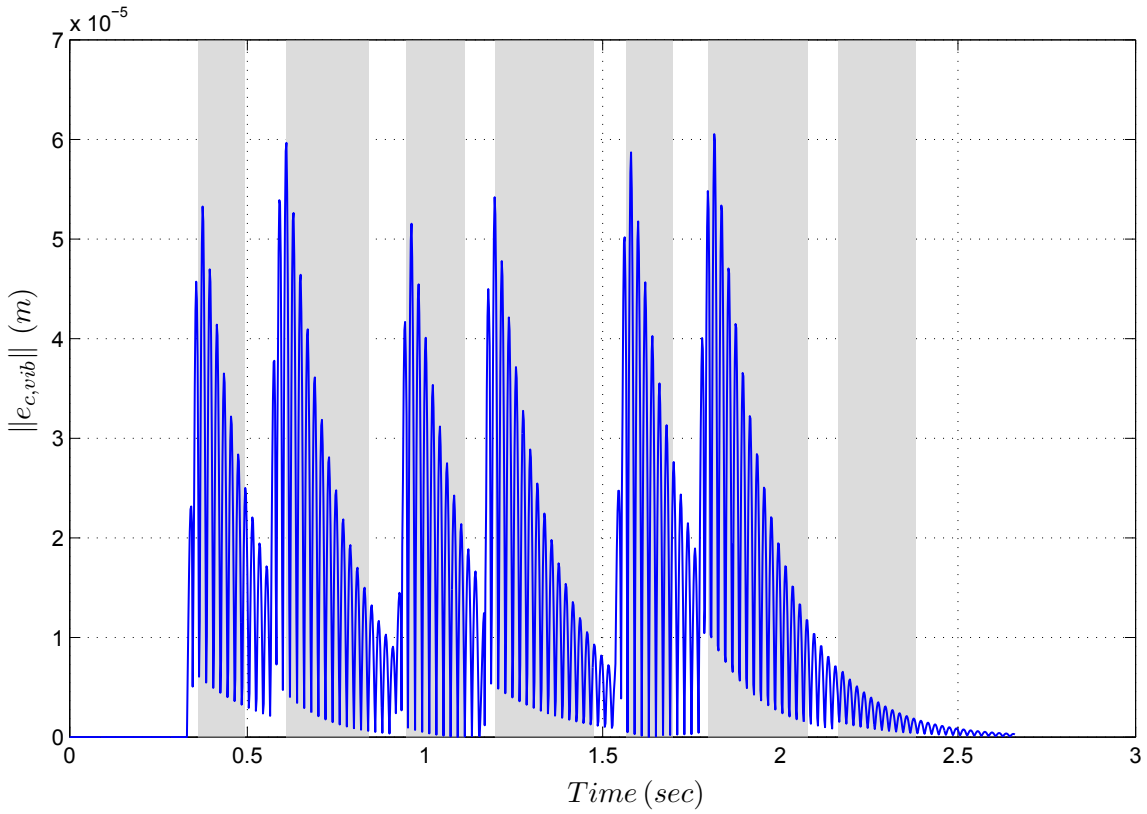


Fig. B.8: Critical oscillation regions for test contour II

Results From Test Contour IV

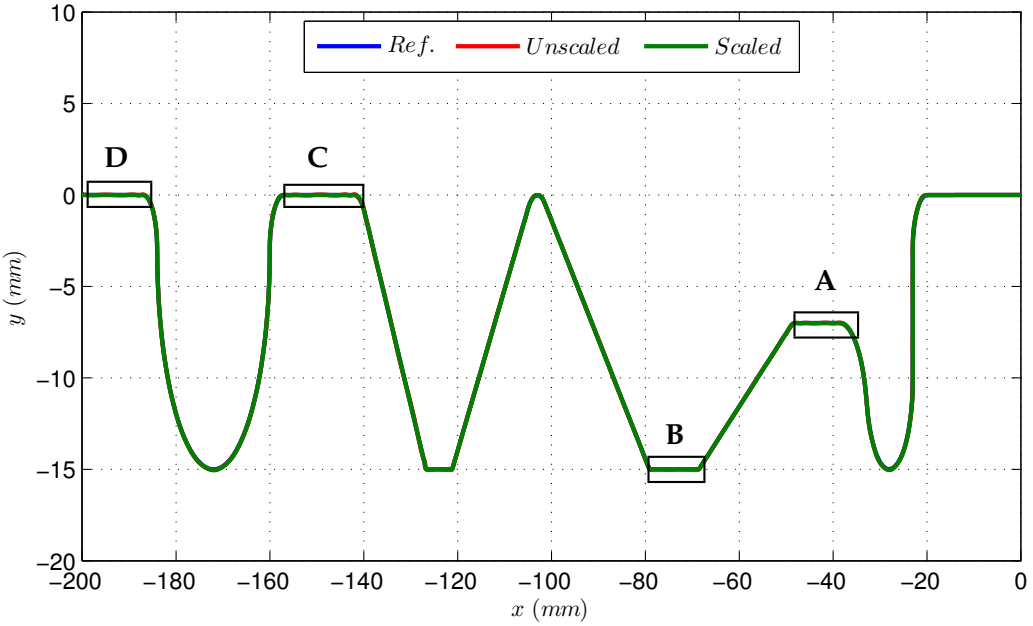


Fig. B.9: Response of simulation model to test contour IV before and after time-scaling

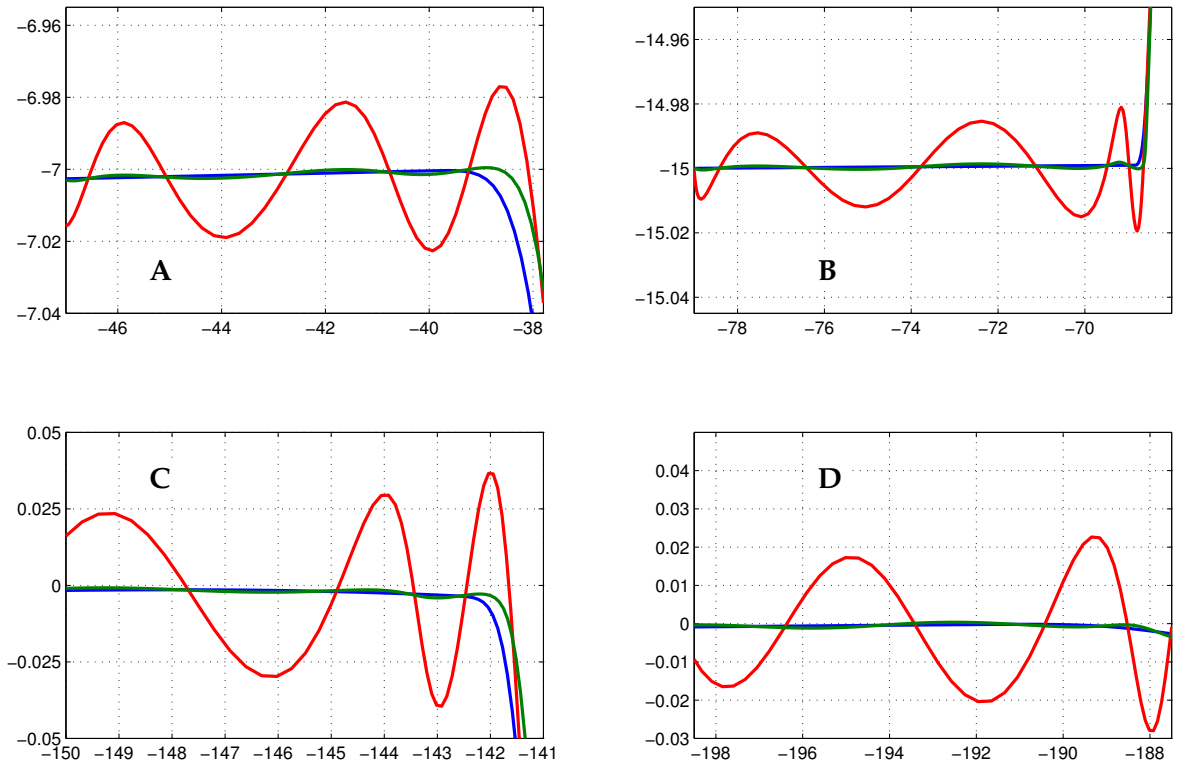


Fig. B.10: Zoomed views in the response of simulation model to test contour IV before and after time-scaling

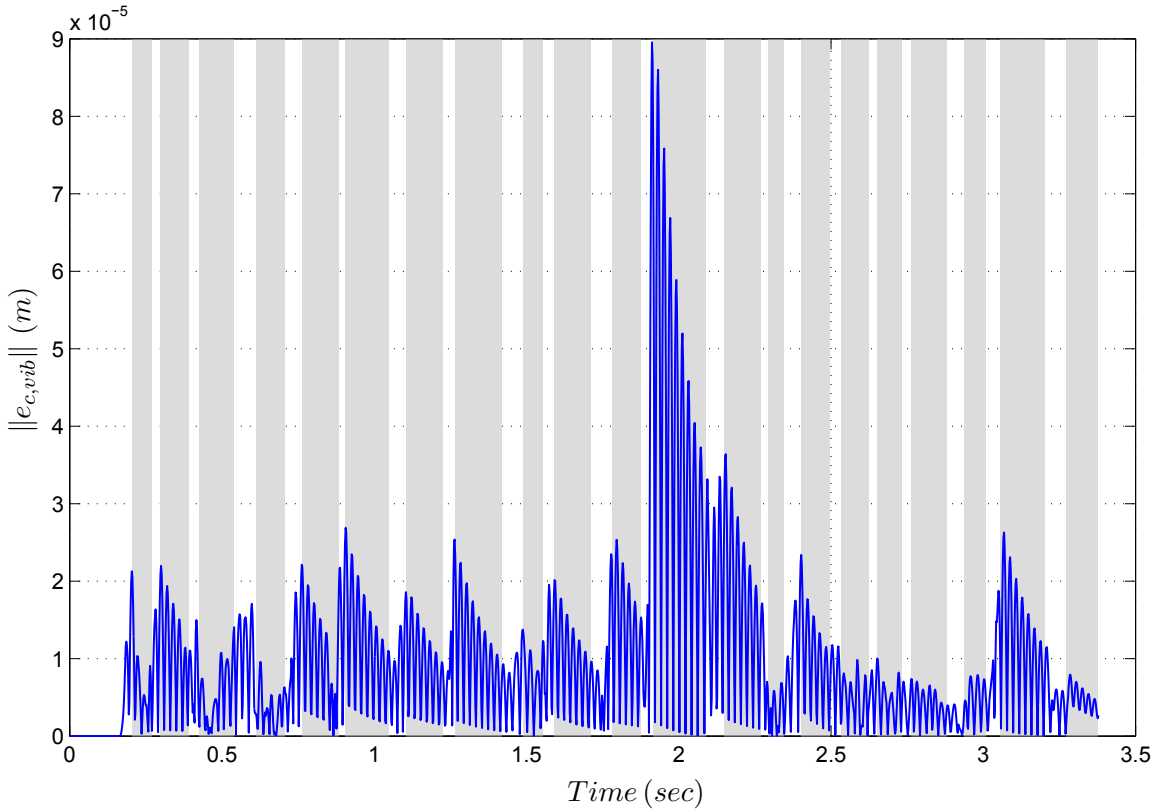


Fig. B.11: Critical oscillation regions for test contour IV

Bibliography

[1] W. Papiernik. *Numerical Controls and Drives for Machine Tools and Industrial Robots lecture notes (Machine Tools IV)*. TUHH. Hamburg, Germany, 2006.

[2] W. Papiernik. "Architecture and Design of Modern CNC/Drive Systems". In: *Proceedings on PCIM-Conference*. Vol. 29. 1996, pp. 271–280.

[3] H. Hermes and J. P. LaSalle. "Functional Analysis and Time Optimal Control". In: *Mathematics in Science and Engineering* 56 (1969), pp. 43–48.

[4] L. Zlajpah. "On Time Optimal Path Control of Manipulators with Bounded Joint Velocities and Torques". In: *IEEE International Conference on Robotics and Automation* (1996), pp. 1572–1577.

[5] D. Constantinescu and E.A. Croft. "Smooth and Time-Optimal Trajectory Planning for Industrial Manipulators Along Specified Paths". In: *Robotic Systems* 17/5 (2000), pp. 233–249.

[6] J. Dong, P.M. Ferreira and J.A. Stori. "Feed-Rate Optimization with Jerk Constraints for Generating Minimum Time Trajectories". In: *International Journal of Machine Tools and Manufacture* 47 (2007), pp. 1941–1955.

[7] S. Macfarlane and E.A. Croft. "Design of Jerk Bounded Trajectories for On-Line Industrial Robot Applications". In: *Proceedings IEEE International Conference on Robotics and Automation*. 1. 2001, pp. 979–984.

[8] S. D. Timar et al. "Algorithms for Time-Optimal Control of CNC Machines Along Curved Tool Paths". In: *Robot Comput-Integr Manuf* 21 (2005), pp. 37–53.

[9] W. Hoffmann, W. Papiernik and T. Sauer. "Verfahren und Einrichtung zur Bewegungsführung eines bewegbaren Maschinenelements einer numerisch gesteurten Maschine". German. Pat. DE 10 2004 059 966 B3. 2006.

[10] R. Ekachaiworasin. "Design and Evaluation of an Algorithm for Jerk–Limited Motion Control". English. MA thesis. Technische Universität Hamburg-Harburg, 2005.

[11] W. Papiernik. "Struktur, Entwurf und Verhalten moderner CNC-Servoantriebe". In: *SPS / IPC / Drives, Stuttgart*. 1996, pp. 397–417.

[12] R. Isermann. *Mechatronic Systems Fundamentals*. Springer, 2005.

[13] R. Isermann. *Fault-Diagnosis Applications, Model-Based Condition Monitoring: Actuators, Drives, Machinery, Plants, Sensors, and Fault-tolerant Systems*. Springer, 2011.

[14] C. Pislaru. "Parameter identification and hybrid mathematical modelling techniques applied to non-linear control systems". PhD thesis. The University of Huddersfield, 2001.

[15] H. Groß, J. Harmann and G. Wiegärtner. *Electrical Feed Drives in Automation*. Ed. by Siemens AG. MCD Corporate Publishing, 2001.

[16] Y. Zhang and W. Zhang. "Modeling of the Feed Drive System Basing on Torsion Multi-oscillator". In: *International Conference on Mechanic Automation and Control Engineering - MACE*. 2011.

[17] B. Siciliano and O. Khatib. *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer-Verlag, 2008, pp. 133–140.

[18] Y. Koren. "Control of Machine Tools". In: *Journal of Manufacturing Science and Engineering* 119 (1997), pp. 749–755.

[19] Z. Pandilov and V. Dukovski. "Analytical Calculation of the CNC Machines Position Loop Gain". In: *Journal of Achievements in Materials and Manufacturing Engineering* 23 (2007), pp. 71–74.

[20] W. J. Book. "Controlled Motion in an Elastic World". In: *ASME, Journal of Dynamic Systems, Measurement, and Control* 115 (1993), pp. 252–261.

[21] Jer-Nan Juang. "Optimal Design of a Passive Vibration Absorber for a Truss Beam". In: *Journal of Guidance and Control* 7.6 (Dec. 1984), pp. 733–739.

[22] T. Bailey and J. E. Hubbard. "Distributed Piezoelectric-Polymer Active Vibration Control of a Cantilever Beam". In: *Journal of Guidance and Control* 3.5 (Sept. 1985), pp. 605–611.

[23] Alberts T. E. et al. "Experiments in Optimal Control of a Flexible Arm with Passive Damping". In: *Fifth VPI&SU/AIAA Symposium on Dynamics and Control of Large Structures*. 1985.

[24] S. G. Tewani, B. L. Walcott and K. E. Rouch. "Active Optimal Vibration Control using Dynamic Absorber". In: *International Conference on Robotics and Automation* (1991), pp. 1182–1187.

[25] D. Schröder. *Elektrische Antriebe - Regelung von Antriebssystemen*. Springer, 2009, pp. 85–114.

[26] N. C. Singer and W. P. Seering. "Using Acausal Shaping Techniques to Reduce Robot Vibration". In: *Proc. IEEE Int. Conference on Robotics and Automation*. Philadelphia, 1988, pp. 1434–1439.

[27] G. H. Tallman and G. H. Smith. "Analog Study of Dead-Beat Posicast Control". In: *IRE Transactions on Automatic Control*. Vol. AC-3. 1958, pp. 14–21.

[28] Tarunraj Singh. *Optimal Reference Shaping for Dynamical Systems: Theory and Applications*. 1st ed. CRC Press, Oct. 2009.

[29] N. C. Singer and W. P. Seering. "Preshaping Command Inputs to Reduce System Vibration". In: *ASME J. of Dynamic Systems, Measurement and Control* 112 (1990), pp. 76–82.

[30] T. Singh and W. Singhose. "Tutorial on Input Shaping/Time Delay Control of Maneuvering Flexible Structures". In: *American Control Conference*. Anchorage, AK, USA, 2002, pp. 1717–1731.

[31] W. Singhose and N. Singer. "Initial Investigations into the Effects of Input Shaping on Trajectory Following". In: *American Control Conference*. Vol. 3. Baltimore, MD, USA, 1994, pp. 2526–2532.

[32] W. Singhose and T. Chuang. "Reducing Deviations from Trajectory Components with Input Shaping". In: *American Control Conference*. Vol. 1. Seattle, WA, USA, 1995, pp. 929–933.

[33] W. Singhose and N. Singer. "Effects of Input Shaping on Two-Dimensional Trajectory Following". In: *IEEE Transactions on Robotics and Automation* 12.6 (Dec. 1996), pp. 881–887.

[34] R. Eloundou and W. Singhose. "Interpretation of Smooth Reference Commands As Input-Shaped Functions". In: *Proceedings of the American Control Conference*. Anchorage, AK, 2002, pp. 4948–4953.

[35] R. Bearee et al. "Influence of a Jerk Controlled Movement Law on the Vibratory Behaviour of High-Dynamics Systems". In: *Journal of Intelligent and Robotic Systems* 42 (2005), pp. 275–293.

[36] H. Z. Li et al. "Motion Profile Design to Reduce Residual Vibration of High-Speed Positioning Stages". In: *IEEE-ASME Transaction on Mechatronics* 14.2 (Apr. 2009), pp. 264–269.

[37] R. Béarée, P.J. Barre and J.P. Hautier. "Vibration Reduction Abilities of Some Jerk Controlled Movement Laws for Industrial Machines". In: *in proc. of the 16th IFAC World Congress* (July 2005).

[38] A. Papoulis. *The Fourier Integral and Its Applications*. McGraw-Hill, 1962.

[39] K. Kozak, W. Singhose and I. Ebert-Uphoff. "Performance Measures For Input Shaping and Command Generation". In: *Journal of Dynamic Systems, Measurement, and Control* 128 (Sept. 2006), pp. 731–736.

[40] L. Y. Pao and W. E. Singhose. "A Comparison of Constant and Variable Amplitude Command Shaping Techniques for Vibration Reduction". In: *Proceedings of IEEE Conference on Control Applications*. Albany, NY, USA, 1995, pp. 875–881.

[41] A. Dhanda and G. Franklin. "Minimum Move-Vibration Control for Flexible Structures". In: *Journal of Dynamic Systems, Measurement, and Control* 130.3 (May 2008), p. 034503.

[42] H. Y. Chuang and C. H. Liu. "A Model-Referenced Adaptive Control Strategy for Improving Contour Accuracy of Multiaxis Machine Tools". In: *IEEE Transactions on Industry Applications* 2 (Oct. 1990), pp. 1539–1544.

[43] M. Y. Cheng and C. C. Lee. "Motion Controller Design for Contour-Following Tasks Based on Real-Time Contour Error Estimation". In: *IEEE Transactions on Industrial Electronics* 54.3 (June 2007), pp. 1686–1695.

[44] S. S. Yeh and P. L. Hsu. "Estimation of the Contouring Error Vector for the Cross-Coupled Control Design". In: *IEEE/ASME Transactions on Mechatronics* 7.1 (Mar. 2002), pp. 44–51.

[45] H. Y. Chuang and C. H. Liu. "Cross-Coupled Adaptive Feedrate Control for Multi-Axis Machine Tools". In: *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME* 113.3 (1991), pp. 451–457.

[46] Y. Koren and Ch. Ch. Lo. "Variable-Gain Cross-Coupling Controller for Contouring". In: *CIRP, Manufacturing Systems* 40.1 (Jan. 1991), pp. 371–374.

[47] K. Erkorkmaz and Y. Altintas. "High Speed Contouring Control Algorithm for CNC Machine Tools". In: *Proceedings of ASME Dynamic Systems and Control Division, IMECE' 98* DSC-64 (1998), pp. 463–469.

[48] M. Sampei and K. Furuta. "On Time Scaling for Nonlinear Systems: Application to Linearization". In: *IEEE Transactions on Automatic Control* AC-31.5 (May 1986), pp. 459–462.

[49] E. Kardoss and B. Kiss. "Tracking Error Based On-Line Trajectory Time Scaling". In: *IEEE Proc. of 10th Int. Conf. on Intelligent Engineering Systems* (2006), pp. 80–85.

[50] J. M. Hollerbach. "Dynamic Scaling of Manipulator Trajectories". In: *Trans. of the ASME, J. of Dynamic Systems, Measurement and Control* 106.106 (1984), pp. 102–106.

[51] E. Oran Brigham. *The Fast Fourier Transform*. Englewood Cliffs, New Jersey, USA: Prentice Hell Inc., 1974, pp. 32–35.

[52] T. Sauer. "Time-frequency analysis, wavelets and why things (can) go wrong". In: *Kognitive Neurophysiologie des Menschen - Human cognitive neurophysiology* 4.1 (2011), pp. 38–62.

[53] S. Mallat. *A Wavelet Tour of Signal Processing The sparse Way*. Academic Press,San Diego, Calif, USA., 2009.

[54] C. Junsheng, Y. Dejie and Y. Yu. "Application of an Impulse Response Wavelet to Fault Diagnosis of Rolling Bearings". In: *Mechanical Systems and Signal Processing* 21 (Nov. 2007), pp. 920–929.

[55] L. Alkafafi, C. Hamm and T. Sauer. "Vibrational Error Extraction Method Based on Wavelet Technique". In: *Lecture Notes in Computer Science*. Mathematical Methods for Curves and Surfaces. Springer, 2013.

[56] J. Nocedal and S. J. Wright. *Numerical Optimization*. New York: Springer, 2006.

[57] H. H. Rosenbrock. "An Automatic Method for finding the Greatest or Least Value of a Function". In: *Computer Journal* 3 (1960), pp. 175–184.

[58] P. Venkataraman. *Applied Optimization with Matlab Programming*. Wiley-Interscience, Dec. 2001.

[59] G. N. Vanderplaats. *Numerical Optimization Techniques for Engineering Design*. New York: McGraw-Hill, 1984.

[60] C. Gerald and P. Wheatley. *Applied Numerical Analysis*. Addison-Wesley, 2004, pp. 412–420.